Méthodes de décomposition et d'échantillonnage pour les problèmes de conception de réseaux à grande échelle dans le cadre des systèmes de transport semi-flexibles

par

Joaquim JUSSEAU

MÉMOIRE PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
COMME EXIGENCE PARTIELLE À L'OBTENTION DE LA MAÎTRISE
AVEC MÉMOIRE EN GÉNIE DES TECHNOLOGIES DE L'INFORMATION
M. Sc. A.

MONTRÉAL, LE 26 JUIN 2026

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

PRÉSENTATION DU JURY

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE:

M. Fausto ERRICO, directeur de mémoire
Département de génie des systèmes, École de technologie supérieure

M. Julien TROCHU, président du jury
Département de génie des systèmes, École de technologie supérieure

M. Okan ARSLAN, jury externe
Département de sciences de la décision, HEC Montréal

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 4 JUIN 2026

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

REMERCIEMENTS

Il me serait difficile de commencer ce mémoire sans remercier tout d'abord mon directeur de recherche, Fausto Errico. Je le remercie particulièrement pour m'avoir fait découvrir l'immense univers de la recherche opérationnelle, univers passionnant dont je ne connaissais rien jusqu'à mon arrivée à Montréal. Je le remercie chaleureusement pour son encadrement tout au long de mon travail, ses conseils avisés et les opportunités qu'il m'a données. Je n'ai aucun doute que les connaissances acquises durant cette maîtrise me seront indispensables pour la suite de mon parcours.

Un grand merci à mes amis qui m'ont accompagné durant ce projet, particulièrement à Alix pour sa présence, son soutien et les nombreux moments de partage qui ont rendu cette période très agréable.

Je remercie également les membres du jury d'avoir accepté de prendre le temps de lire ce mémoire et d'évaluer ces travaux.

Je remercie évidemment mes parents, sans qui rien de tout cela n'aurait été possible.

Je remercie aussi le Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport (CIRRELT) de m'avoir permis de réaliser des expériences numériques sur ses machines.

Méthodes de décomposition et d'échantillonnage pour les problèmes de conception de réseaux à grande échelle dans le cadre des systèmes de transport semi-flexibles

Joaquim JUSSEAU

RÉSUMÉ

Ce mémoire porte sur les problèmes de conception de réseaux à deux niveaux à grande échelle. Dans ce type de problème, un premier niveau de décision consiste à concevoir une infrastructure (un réseau) reliant plusieurs sites entre eux, tandis que le second niveau consiste à router un ensemble de demandes sur le réseau précédemment construit. Ces problèmes apparaissent notamment dans les domaines du transport et de la logistique, et se caractérisent par une forte interaction entre les décisions de conception et de routage. Pour les instances de grande taille, qui impliquent de router un grand nombre de demandes, les méthodes de résolution exactes peinent à converger vers la solution optimale.

L'objectif de ce travail est de développer des approches permettant d'obtenir des solutions de haute qualité, accompagnées de garanties sur la valeur optimale, tout en conservant une bonne capacité de passage à l'échelle. Deux contributions complémentaires sont proposées.

La première renforce les méthodes exactes par l'introduction de nouvelles bornes inférieures exploitant la structure entière des problèmes étudiés. Ces bornes permettent de réduire l'espace de recherche et d'améliorer les performances des algorithmes de type séparation et évaluation.

La seconde introduit un changement de paradigme via une approche par échantillonnage de la demande. L'idée consiste à approximer le problème original à partir d'un sous-ensemble aléatoire de demandes, convenablement remis à l'échelle. Nous montrons que ce type de problème déterministe peut être réinterprété dans un cadre stochastique, ce qui permet de mobiliser les outils de la programmation stochastique afin d'obtenir des estimateurs statistiques de la valeur optimale, ainsi que des bornes inférieures et supérieures valides.

Les approches proposées sont analysées théoriquement et évaluées expérimentalement sur des instances issues de la littérature, démontrant leur efficacité et leur capacité à traiter des instances de grande taille.

Mots-clés: Conception de réseaux, TSP-GL, MUFND, Décomposition de Benders, Méthodes d'échantillonnage

Decomposition and Sampling Methods for Large-Scale Network Design Problems in Semi-Flexible Transportation Systems

Joaquim JUSSEAU

ABSTRACT

This master thesis addresses large-scale two-stage network design problems. In this class of problems, a first-stage decision consists in designing an infrastructure (a network) connecting several sites, while the second-stage decision consists in routing a set of demands over the network previously built. Such problems arise notably in transportation and logistics, and are characterized by a strong interaction between design and routing decisions. For large-scale instances, which involve routing a large number of demands, exact solution methods struggle to converge to the optimal solution.

The objective of this work is to develop approaches that yield high-quality solutions, together with guarantees on the optimal value, while preserving good scalability. Two complementary contributions are proposed.

The first contribution strengthens exact methods by introducing new lower bounds that exploit the integer structure of the problems under study. These bounds reduce the search space and improve the performance of branch-and-bound algorithms.

The second contribution introduces a paradigm shift through a demand sampling approach. The idea consists in approximating the original problem using only a random subset of demands, suitably rescaled. We show that this deterministic problem can be reinterpreted within a stochastic framework, which allows us to leverage the tools of stochastic programming to obtain statistical estimators of the optimal value, as well as valid lower and upper bounds.

The proposed approaches are analyzed theoretically and evaluated experimentally on instances from the literature, demonstrating their effectiveness and their ability to handle large-scale instances.

Keywords: Network Design, TSP-GL, MUFND, Benders Decomposition, Sampling Methods

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

LISTE DES FIGURES

Page

LISTE DES ALGORITHMES

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

BI	Borne inférieure
BS	Borne supérieure
CIRRELT	Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport
ED	Échantillonnage de la Demande
EG	Échantillonnage de graphe (Graph sampling)
ETS	École de Technologie Supérieure
MUFND	Problème de conception de réseau multicommodité sans contrainte de capacité avec coûts fixes (Multicommodity Uncapacitated Fixed-charge Network Design)
PLMNE	Programmation linéaire Mixte en nombres entiers
PM	Problème maître (Master Problem)
SAA	Sample Average Approximation
SPD	Sous-problème dual (Dual Sub-Problem)
TSP	Problème du Voyageur de Commerce (Traveling Salesman Problem)
TSP-GL	Problème du Voyageur de Commerce avec Latence Généralisée (Traveling Salesman Problem with Generalized Latency)

INTRODUCTION

Les problèmes de conception de réseaux occupent une place centrale en recherche opérationnelle, en raison de leur capacité à modéliser des situations décisionnelles complexes issues de nombreux domaines d'application, tels que les systèmes de transport, les chaînes logistiques ou les réseaux de télécommunications.

Dans de nombreux contextes, ces problèmes font intervenir deux niveaux de décision étroitement liés. Dans un premier temps, il s'agit de concevoir une infrastructure, par exemple un réseau de transport ou de distribution en choisissant les liaisons à mettre en place entre différents sites. Dans un second temps, cette infrastructure est exploitée afin de satisfaire un ensemble de demandes, correspondant typiquement à des flux à acheminer entre des points d'origine et de destination. Les décisions de conception conditionnent les possibilités de routage, tandis que la structure des demandes influence en retour les coûts d'exploitation.

La difficulté de ces modèles provient à la fois de ce couplage fort entre décisions et de la taille souvent très importante des instances considérées. En particulier, lorsque le nombre de demandes est élevé, qu'il y a un grand nombre de sites pouvant être desservis et que le volume total de demande à traiter est grand, les formulations mathématiques associées deviennent de très grande dimension, rendant l'utilisation de méthodes exactes difficile, voire impraticable. Bien que des approches permettent de traiter efficacement des instances de taille modérée, leurs performances se dégradent rapidement à mesure que la taille du problème augmente.

Dans ce contexte, ce mémoire s'intéresse à la question suivante : comment obtenir des solutions de haute qualité, accompagnées de garanties sur la valeur optimale, pour des problèmes de conception de réseau à deux niveaux de grande dimension ?

Nous apportons deux contributions complémentaires à cette question.

La première consiste à développer de nouvelles bornes inférieures exploitant la structure entière des solutions obtenues via relaxation linéaire. Ces bornes, étudiées dans le cadre du problème du voyageur de commerce avec latence généralisée (TSP-GL), visent à renforcer les méthodes exactes en réduisant plus efficacement l'espace de recherche. Cette première contribution permet également d'examiner la portée et les limites des stratégies de renforcement des bornes face au passage à l'échelle, et motive ainsi le changement de paradigme adopté dans la seconde contribution.

La seconde contribution, qui constitue l'apport principal de ce mémoire, est une nouvelle méthode destinée à traiter des problèmes de conception de réseau à deux niveaux comportant un grand nombre de demandes. Plutôt que de chercher à renforcer la formulation, cette méthode que nous appelons approche par échantillonnage de la demande (ED) agit directement sur l'espace des demandes en construisant un estimateur repondéré du problème original à partir d'un sous-ensemble aléatoire de demandes. L'originalité de cette démarche repose sur une réinterprétation du problème original : nous montrons qu'un problème purement déterministe peut être interprété dans un contexte stochastique, ce qui nous permet de mobiliser le cadre théorique du *Sample Average Approximation* (SAA) initialement développé pour la programmation stochastique afin de doter notre méthode de garanties statistiques formelles : bornes inférieures et supérieures valides en espérance, intervalles de confiance.

Les méthodes développées font l'objet d'une analyse théorique ainsi que d'une évaluation expérimentale sur deux problèmes structurellement différents, le problème du voyageur de commerce avec latence généralisée (TSP-GL) et le problème de conception de réseau multicommodité sans contraintes de capacité avec coûts fixes (MUFND). Cette évaluation vise à étudier le comportement des méthodes proposées sur des instances de tailles variées, incluant en particulier des instances de grande dimension pour lesquelles les méthodes exactes atteignent leurs limites.

Le mémoire est organisé comme suit. Le Chapitre 1 présente, sous forme d'une revue de la littérature, les concepts et outils nécessaires à la compréhension du travail : algorithmes de séparation et d'évaluation, décomposition de Benders, problèmes TSP-GL et MUFND, et cadre du SAA. Le Chapitre 2 développe les nouvelles bornes inférieures pour le TSP-GL et discute leur portée comme leurs limites. Le Chapitre 3, qui constitue le cœur du mémoire, présente la méthode par ED, ses propriétés théoriques et son application au TSP-GL ainsi qu'au MUFND. Une conclusion générale synthétise les résultats obtenus et trace plusieurs perspectives de recherche.

CHAPITRE 1

REVUE DE LA LITTÉRATURE

Dans ce chapitre, nous présentons une synthèse des concepts et méthodes pertinents pour notre travail. Nous introduisons d'abord les outils algorithmiques fondamentaux que sont la programmation linéaire mixte en nombres entiers et l'algorithme de séparation et d'évaluation, les méthodes de plans coupants, puis le problème du voyageur de commerce, qui a servi de cadre de référence au développement historique de ces techniques. Nous présentons ensuite la décomposition de Benders, dont l'utilisation est centrale dans la suite du mémoire, avant d'introduire les deux problèmes étudiés dans ce travail : le problème du voyageur de commerce avec latence généralisée (TSP-GL) et le problème de conception de réseau multicommodité sans contrainte de capacité avec coûts fixes (MUFND). Nous présentons enfin le cadre des problèmes d'optimisation stochastique à deux étapes ainsi que le principe de la méthode de *Sample Average Approximation* (SAA). Le chapitre se clôt par une discussion des principales limites des approches existantes, en particulier en termes de passage à l'échelle ; ces limitations motivent le développement des méthodes proposées dans la suite du mémoire.

1.1 Programmation Linéaire Mixte en Nombres Entiers

Les problèmes de programmation linéaire mixte en nombres entiers (PLMNE) constituent une classe fondamentale de problèmes en recherche opérationnelle. Ils consistent à optimiser une fonction objectif linéaire sous des contraintes linéaires, avec la particularité que certaines variables sont restreintes à prendre des valeurs entières tandis que d'autres demeurent continues. Cette combinaison de variables discrètes et continues permet de modéliser conjointement des décisions structurelles (choix de routage, d'affectation, de localisation, d'ouverture d'infrastructure) et des décisions d'exploitation à valeurs réelles (quantités produites, flux acheminés, durées).

Un problème de PLMNE peut être formulé de manière générale comme suit :

$$\min \quad c^\top x \tag{1.1a}$$

$$\text{s.c.} \quad Ax \leq b \tag{1.1b}$$

$$x_i \in \mathbb{Z}^+ \quad \forall i \in I \tag{1.1c}$$

$$x_j \in \mathbb{R} \quad \forall j \notin I \tag{1.1d}$$

où $x \in \mathbb{R}^n$ désigne le vecteur des variables de décision, $c \in \mathbb{R}^n$ le vecteur des coefficients de coûts de la fonction objectif, $A \in \mathbb{R}^{m \times n}$ la matrice des coefficients des contraintes et $b \in \mathbb{R}^m$ le vecteur de disponibilités. L'ensemble $I \subseteq \{1, \ldots, n\}$ regroupe les indices des variables soumises à des contraintes d'intégrité, tandis que les variables indexées par le complément $\{1, \ldots, n\} \setminus I$ sont continues. Lorsque $I = \emptyset$, on retrouve un programme linéaire continu ; lorsque $I = \{1, \ldots, n\}$, le problème est purement entier.

Les fondements théoriques de la programmation linéaire ont une histoire qui précède largement sa formalisation moderne. Dès le XIX$^{\text{e}}$ siècle, Joseph Fourier (voir Fourier (1827); Motzkin (1952); Williams (1986)) introduit une méthode d'élimination pour les systèmes d'inégalités linéaires, jetant les bases de la théorie polyédrale. Au XX$^{\text{e}}$ siècle, Kantorovich (1939) propose la première formulation de problèmes de planification de la production sous la forme de programmes linéaires, accompagnée d'une méthode de résolution, ce qui lui vaudra le prix Nobel d'économie en 1975. La percée algorithmique majeure intervient avec Dantzig (1951), qui développe l'algorithme du simplexe, encore aujourd'hui l'une des méthodes de référence pour résoudre les problèmes d'optimisation linéaire continue.

En présence de contraintes d'intégrité, en revanche, ces méthodes ne permettent plus d'obtenir directement une solution optimale. Il devient alors nécessaire de recourir à des approches de résolution spécifiques, adaptées à la nature discrète du problème. Du point de vue de la complexité, les problèmes de PLMNE sont en général NP-difficiles, ce qui signifie qu'aucun algorithme en temps polynomial n'est connu pour les résoudre dans le cas général. En pratique,

leur résolution repose sur des méthodes combinant relaxations linéaires et exploration structurée de l'espace des solutions, permettant de traiter des instances de grande taille.

Aujourd'hui, les solveurs modernes intègrent ces différentes techniques au sein de cadres algorithmiques performants. Les principes de ces méthodes seront détaillés dans la section suivante.

1.2 Algorithme de séparation et d'évaluation

L'algorithme de *Branch-and-Bound*, également appelé algorithme de branchement ou algorithme de séparation et évaluation, est une méthode exacte d'énumération implicite utilisée pour résoudre des problèmes d'optimisation combinatoire et de programmation linéaire en nombres entiers.

Son principe repose sur une exploration intelligente de l'espace des solutions, combinant une décomposition en sous-problèmes et l'utilisation de relaxations pour guider la recherche.

1.2.1 Fondations historiques

Les bases théoriques de l'algorithme de branchement et évaluation ont été posées au début des années 1960. Land & Doig (1960) introduisent une première méthode systématique pour la résolution de problèmes de programmation en nombres entiers, constituant l'un des fondements de cette approche.

Le terme *Branch-and-Bound* est ensuite popularisé par Little, Murty, Sweeney & Karel (1963), qui l'appliquent au problème du voyageur de commerce, mettant en évidence son efficacité pour les problèmes de routage combinatoire.

Enfin, les formulations algorithmiques modernes utilisées dans les solveurs de programmation en nombres entiers ont été structurées par Dakin (1965), qui formalisent l'approche sous forme d'arbre de recherche.

1.2.2 Relaxation linéaire

Dans un problème de programmation linéaire en nombres entiers, la relaxation linéaire consiste à supprimer les contraintes d'intégrité sur les variables entières, c'est-à-dire à autoriser ces variables à prendre des valeurs réelles continues.

Cette relaxation transforme le problème initial en un problème de programmation linéaire classique, beaucoup plus facile à résoudre grâce à l'algorithme du simplexe (voir Dantzig (1951)). La solution obtenue fournit alors une borne inférieure (dans le cas d'un problème de minimisation) du problème entier.

1.2.3 Principe général

L'algorithme de séparation et d'évaluation repose sur deux opérations fondamentales :

- **Branchement :** l'espace des solutions est progressivement partitionné en sous-problèmes plus restreints. À chaque étape, une variable prenant une valeur fractionnaire dans la solution de la relaxation est sélectionnée, et deux sous-problèmes sont créés en imposant respectivement des contraintes de type $x_i \leq \lfloor x_i^* \rfloor$ et $x_i \geq \lceil x_i^* \rceil$.
- **Évaluation :** pour chaque sous-problème, on résout sa relaxation linéaire afin d'obtenir une borne inférieure sur l'ensemble des solutions qu'il contient. Si cette solution est entière, elle constitue une solution réalisable du problème initial et peut améliorer la meilleure solution connue.

Ainsi, l'algorithme construit progressivement un arbre de recherche dont chaque nœud correspond à un sous-problème.

1.2.4 Mécanisme d'élagage

L'efficacité de l'algorithme de séparation et évaluation repose sur sa capacité à éviter l'exploration de certaines branches de l'arbre de recherche grâce à un mécanisme d'élagage.

Dans le cas d'un problème de minimisation, un nœud est élagué dans les cas suivants :

1. La borne inférieure du sous-problème est supérieure ou égale à la meilleure solution réalisable connue. Dans ce cas, aucune solution améliorante ne peut être obtenue dans cette branche.
2. La relaxation linéaire du nœud fournit une solution entière. Cette solution est alors candidate pour devenir la meilleure solution globale si elle améliore la solution courante. Il n'est donc pas nécessaire d'explorer davantage une branche où toutes les variables sont entières.
3. Le sous-problème relaxé est infaisable, ce qui implique que le sous-problème entier associé l'est également.

Ce mécanisme permet de réduire considérablement l'espace de recherche exploré tout en garantissant l'optimalité de la solution finale.

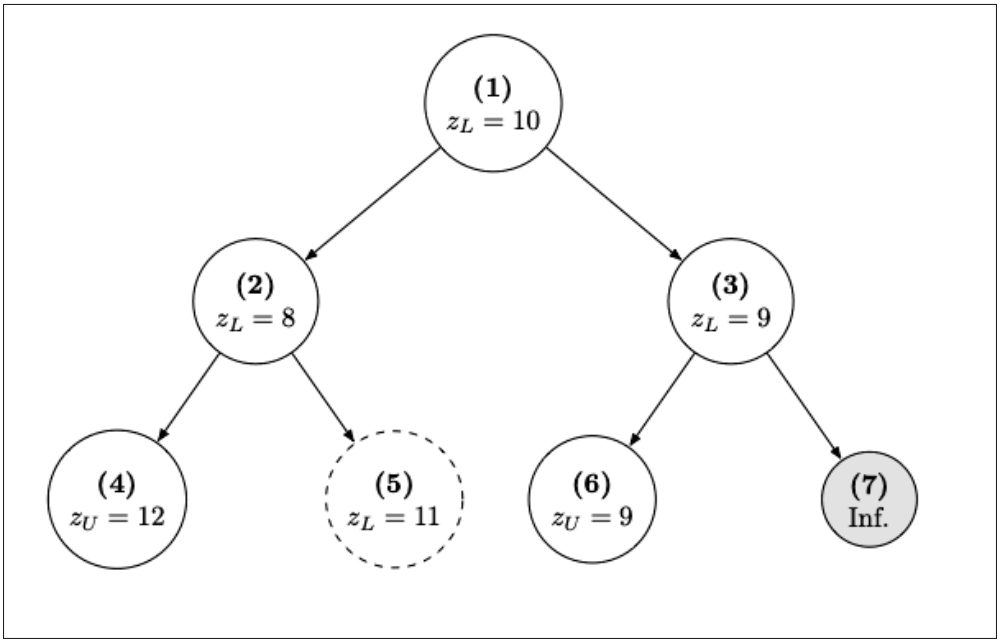

Figure 1.1 Algorithme de séparation et d'évaluation pour un problème de minimisation

La figure 1.1 illustre le déroulement typique d'un algorithme de séparation et évaluation. On suppose ici un parcours en profondeur des nœuds, dans lequel l'algorithme sélectionne

systématiquement l'un des fils du nœud courant et poursuit l'exploration le long de cette branche jusqu'à atteindre un nœud terminal (solution entière, élagage ou infaisabilité), avant d'explorer les branches restantes.

Le nœud racine (1) est d'abord résolu, fournissant une borne inférieure $z_L = 10$. On branche ensuite sur ce nœud, générant les sous-problèmes (2) et (3).

Le nœud (2), de borne inférieure $z_L = 8$, est exploré en priorité. Il est à son tour branché en (4) et (5). Le nœud (4) fournit une solution entière de valeur $z_U = 12$, constituant la première solution réalisable. La meilleure borne supérieure courante est donc $z_U = 12$.

Le nœud (5) possède une borne inférieure $z_L = 11$, inférieure à z_U, mais il peut être élagué si une meilleure solution est trouvée par la suite.

On explore ensuite le nœud (3), avec $z_L = 9$, qui génère les nœuds (6) et (7). Le nœud (6) fournit une solution entière de valeur $z_U = 9$, améliorant la meilleure solution connue. La borne supérieure courante devient alors $z_U = 9$.

À ce stade, tous les nœuds dont la borne inférieure vérifie $z_L \geq 9$ peuvent être élagués. C'est le cas du nœud (5), ainsi que de tout sous-arbre restant. Enfin, le nœud (7) est infaisable et est donc éliminé.

L'algorithme se termine lorsque tous les nœuds ont été explorés ou élagués. La solution optimale est alors donnée par la meilleure solution réalisable trouvée, ici de valeur $z^* = 9$, avec un encadrement final vérifiant $z_L \leq z^* \leq z_U$.

1.3 Plans coupants

Les plans coupants constituent une famille de méthodes de résolution pour les problèmes de programmation linéaire en nombres entiers, indépendante de l'algorithme de séparation et d'évaluation. L'idée centrale consiste à renforcer la relaxation linéaire d'un problème entier en lui adjoignant des inégalités valides, c'est-à-dire des contraintes satisfaites par toutes les

solutions entières réalisables mais violées par la solution courante de la relaxation. En répétant ce procédé, on construit une suite de relaxations de plus en plus serrées, dont l'optimum tend vers celui du problème entier.

1.3.1 Principe général

Soit un problème de PLNE de la forme $\min\{c^\top x : x \in P \cap \mathbb{Z}^n\}$, où $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ désigne le polyèdre associé à sa relaxation linéaire.

Un objet central dans l'analyse des plans coupants est l'enveloppe convexe de l'ensemble des solutions entières réalisables, notée $\text{conv}(P \cap \mathbb{Z}^n)$. Par définition, l'enveloppe convexe d'un ensemble de points $S \subseteq \mathbb{R}^n$ est le plus petit ensemble convexe contenant S, c'est-à-dire l'ensemble de toutes les combinaisons convexes finies de points de S :

$$\text{conv}(S) = \left\{ \sum_{i=1}^{k} \lambda_i x_i \ : \ k \in \mathbb{N},\ x_i \in S,\ \lambda_i \geq 0,\ \sum_{i=1}^{k} \lambda_i = 1 \right\}. \tag{1.2}$$

Lorsque P est un polyèdre borné et que l'on considère ses points entiers, l'ensemble $\text{conv}(P \cap \mathbb{Z}^n)$ est lui-même un polyèdre, dont les sommets sont des solutions entières du problème. Cette propriété est fondamentale : si l'on disposait d'une description complète de $\text{conv}(P \cap \mathbb{Z}^n)$ sous forme d'un système d'inégalités linéaires, le problème entier se résoudrait par l'algorithme du simplexe, puisque l'optimum d'une fonction linéaire sur un polyèdre est nécessairement atteint en l'un de ses sommets. En pratique, une telle description est rarement connue explicitement et comporte généralement un nombre exponentiel d'inégalités, mais elle constitue l'objet de référence vers lequel tendent les méthodes de plans coupants.

Lorsque la solution optimale $\bar{x}$ de la relaxation linéaire n'est pas entière, on a nécessairement $\bar{x} \in P \setminus \text{conv}(P \cap \mathbb{Z}^n)$. L'objectif est alors d'identifier une inégalité $\alpha^\top x \leq \beta$ qui sépare $\bar{x}$ du polyèdre entier, c'est-à-dire vérifiant simultanément $\alpha^\top \bar{x} > \beta$ et $\alpha^\top x \leq \beta$ pour tout $x \in P \cap \mathbb{Z}^n$. Une telle inégalité est appelée plan coupant : elle exclut la solution fractionnaire courante

sans éliminer aucune solution entière réalisable. Elle est ensuite ajoutée à la formulation, et la relaxation linéaire est résolue à nouveau.

L'algorithme des plans coupants, sous sa forme pure, consiste à itérer cette procédure jusqu'à obtenir une solution entière, qui est alors optimale pour le problème initial.

1.3.2 Coupes de Gomory et familles d'inégalités valides

Les premières familles de plans coupants ont été introduites par Gomory (1958), qui propose une méthode permettant de générer, à partir du tableau du simplexe associé à une solution fractionnaire, une inégalité valide coupant cette solution. Ces coupes, appelées coupes de Gomory, fournissent un algorithme de résolution exact dont la convergence en un nombre fini d'itérations est démontrée pour les problèmes de programmation linéaire en nombres entiers purs. L'extension au cas mixte a ensuite été développée par Gomory (1960).

D'autres familles d'inégalités valides ont par la suite été développées, exploitant la structure combinatoire spécifique des problèmes considérés : coupes de couverture, coupes de clique, ou encore les coupes de Chvátal–Gomory (voir Chvátal (1973)).

La séparation, c'est-à-dire l'identification d'une coupe violée par la solution courante, constitue dès lors un problème algorithmique à part entière, dont la difficulté varie selon la famille considérée.

En pratique, l'algorithme des plans coupants pris isolément possède une convergence lente et d'instabilités numériques liées à l'accumulation de coupes successives, ce qui en limite l'usage comme méthode de résolution autonome.

1.3.3 Branch-and-Cut

L'algorithme *Branch-and-Cut* résulte de l'intégration des plans coupants au sein de l'algorithme de séparation et d'évaluation. À chaque nœud de l'arbre de recherche, la relaxation linéaire est résolue, puis enrichie dynamiquement par l'ajout d'inégalités valides séparant la solution

fractionnaire courante. Les coupes ainsi générées renforcent la relaxation localement, voire globalement lorsqu'elles sont valides pour l'ensemble du problème, et améliorent la qualité des bornes inférieures utilisées par le mécanisme d'élagage.

Cette combinaison tire parti des forces complémentaires des deux approches : la capacité du branchement à garantir l'exploration exhaustive de manière implicite de l'espace des solutions, et celle des plans coupants à resserrer la relaxation sans étendre l'arbre. Les travaux fondateurs de Padberg & Rinaldi (1991) ont démontré l'efficacité de cette intégration sur le problème du voyageur de commerce symétrique de grande taille, et ont fixé le cadre algorithmique adopté depuis par l'ensemble des solveurs modernes de programmation en nombres entiers.

1.4 Le problème du voyageur de commerce symétrique

1.4.1 Description du problème

Le problème du voyageur de commerce (TSP pour *Traveling Salesman Problem*) est l'un des problèmes les plus emblématiques en recherche opérationnelle et en optimisation combinatoire. Étant donné un ensemble de villes ainsi que les distances séparant chaque ville les unes des autres, le but de ce problème consiste à trouver le cycle le plus court qui visite chaque ville une et une seule fois. Plus formellement, il s'agit de déterminer un cycle hamiltonien de coût minimum dans un graphe complet non orienté $G = (N, E)$.

L'étude du TSP remonte aux premiers travaux en optimisation combinatoire, avec notamment la contribution fondatrice de Dantzig, Fulkerson & Johnson (1954), qui proposent une première formulation mathématique du problème et introduisent une approche par plans coupants pour sa résolution. Ce travail constitue l'une des premières applications majeures de la programmation linéaire en nombres entiers à un problème combinatoire difficile, et marque le début du développement des méthodes exactes modernes.

Par la suite, le TSP a joué un rôle central dans le développement de l'étude polyédrale des problèmes d'optimisation combinatoire. Ces travaux ont également conduit au développement de

techniques de relaxation avancées, telles que la relaxation lagrangienne proposée par Held & Karp (1970), permettant d'obtenir des bornes inférieures de grande qualité.

Au-delà de son intérêt théorique, le TSP a servi de cadre de référence pour le développement et l'évaluation de nombreuses méthodes algorithmiques en optimisation, notamment sur l'algorithme de séparation et évaluation (voir section 1.2) et ses variantes. Ces techniques ont atteint un haut niveau de maturité, comme en témoigne le solveur de TSP Concorde, décrit en détail par Applegate, Bixby, Chvátal & Cook (2006), capable de résoudre des instances de très grande taille jusqu'à 85 900 villes.

Le TSP intervient également comme sous-problème fondamental dans de nombreux modèles plus complexes, en particulier dans les problèmes de routage de véhicules, comme dans les travaux fondateurs de Dantzig & Ramser (1959). Sa structure simple à énoncer mais difficile à résoudre en fait un terrain d'expérimentation privilégié pour l'étude des méthodes d'optimisation à grande échelle.

1.4.2 Modèle mathématique

On introduit ainsi le modèle d'optimisation du TSP :

Ensembles et paramètres

$N = \{1, \ldots, n\}$ Ensemble des sommets du graphe G.

$E = \{[i, j] : i, j \in N,\ i < j\}$ Ensemble des arêtes du graphe complet non orienté.

$c_{ij} = c_{ji} \geq 0$ Coût associé à l'arête reliant les villes i et j.

$\delta(S)$ Ensemble des arêtes ayant exactement une extrémité dans S.

Le modèle mathématique peut alors se formuler comme un problème de minimisation en nombres entiers :

$$\min \quad \sum_{[i,j]\in E} c_{ij}\, x_{ij} \tag{1.3a}$$

$$\text{s.c.} \quad \sum_{j\in N:\,[i,j]\in E} x_{ij} = 2 \qquad \forall i \in N \quad \text{(contraintes de degré)} \tag{1.3b}$$

$$+ \text{ Contraintes d'élimination des sous-tours} \tag{1.3c}$$

$$x_{ij} \in [0,1] \qquad \forall [i,j] \in E. \tag{1.3d}$$

1.4.3 Élimination des sous-tours

Un sous-tour ou sous-tournée est un cycle formé sur un sous-ensemble strict des sommets $S \subset N$, qui ne visite pas l'ensemble des nœuds du graphe. Autrement dit, il s'agit d'une solution fractionnaire ou entière qui satisfait les contraintes de degré, mais qui se décompose en plusieurs cycles disjoints au lieu de former un unique cycle hamiltonien couvrant tous les sommets.

Pour éviter les sous-tours on ajoute des contraintes d'élimination (*Subtour Elimination Constraints*, SEC) qui sont alors formulées de la manière suivante :

$$x(\delta(S)) \;\geq\; 2 \qquad \forall\, S \subset N,\; 2 \leq |S| \leq |N| - 2, \tag{1.4}$$

où $x(\delta(S)) = \sum_{[i,j]\in\delta(S)} x_{ij}$ désigne la somme des variables associées aux arêtes ayant exactement une extrémité dans S. Introduites par Dantzig *et al.* (1954), ces contraintes imposent que tout sous-ensemble de sommets S soit relié au reste du graphe $N \setminus S$ par au moins deux arêtes, empêchant ainsi la formation de cycles isolés ne couvrant qu'une partie des sommets.

On notera que le nombre de ces contraintes croît de manière exponentielle avec la taille de N, ce qui rend leur intégration explicite impossible en pratique pour des instances de grande taille. Bien que d'autres formulations existent pour éviter ce nombre exponentiel de contraintes, comme celle proposée par Miller, Tucker & Zemlin (1960), cette formulation reste largement utilisée en

résolution exacte. Cette préférence s'explique notamment par leurs propriétés polyédrales : ces contraintes définissent des facettes du polytope du problème du voyageur de commerce, ce qui permet de renforcer significativement la relaxation linéaire.

1.4.4 Génération dynamique de contraintes

1.4.4.1 Cas des solutions entières

Dans le cadre d'un algorithme avec plans coupants, les contraintes d'élimination des sous-tours peuvent être générées dynamiquement à partir des solutions entières rencontrées au cours de l'exploration de l'arbre de recherche.

Plus précisément, à chaque fois qu'une solution entière $\hat{x}$ est obtenue, on identifie les composantes connexes du graphe induit. Si une unique composante couvrant tous les sommets est obtenue, la solution correspond à un cycle hamiltonien et est donc réalisable. En revanche, si plusieurs composantes apparaissent, chacune d'elles définit un sous-tour $S \subset N$ violant des contraintes d'élimination.

Pour chaque sous-tour détecté, on ajoute alors dynamiquement la contrainte :

$$x(\delta(S)) \;\geq\; 2 \qquad \forall\, S \subset N,\; 2 \leq |S| \leq |N| - 2 \tag{1.5}$$

qui est violée par la solution courante. Le solveur poursuit ensuite la recherche en tenant compte de ces nouvelles contraintes.

Ce mécanisme permet de ne générer que les contraintes d'élimination des sous-tours nécessaires, tout en garantissant la correction de l'algorithme.

1.4.4.2 Cas des solutions fractionnaires

Dans l'arbre de branchement, les relaxations linéaires résolues aux nœuds de l'arbre produisent généralement des solutions fractionnaires $\hat{x} \in [0, 1]^{|E|}$. Ces solutions peuvent également violer

les contraintes d'élimination des sous-tours, qu'il convient alors de détecter via un problème de séparation.

Le problème de séparation associé consiste à déterminer s'il existe un sous-ensemble $S \subset N$ tel que :

$$\sum_{i \in S} \sum_{j \in N \setminus S} \hat{x}_{ij} < 2. \tag{1.6}$$

Pour résoudre ce problème, on considère le graphe complet dont les arêtes sont pondérées par les valeurs $\hat{x}_{ij}$. Le problème de séparation se ramène alors à un problème de coupe minimum.

Plus précisément, pour un couple de sommets (s, t) fixé, on détermine une coupe minimum séparant s et t. Autrement dit, on cherche un sous-ensemble de sommets $S \subset N$ contenant s et ne contenant pas t, minimisant la somme des valeurs x_{ij} des arêtes (i, j) reliant S à son complémentaire $N \setminus S$, soit :

$$\sum_{i \in S} \sum_{j \in N \setminus S} x_{ij}. \tag{1.7}$$

Si cette quantité est strictement inférieure à 2, alors la contrainte d'élimination des sous-tours associée à S est violée par $\hat{x}$ et peut être ajoutée au modèle.

Bien que cette procédure puisse être appliquée à tous les couples (s, t), l'utilisation d'un arbre de Gomory-Hu (voir Gomory & Hu (1961)) permet d'identifier une coupe globale minimum en ne résolvant que $n - 1$ problèmes de flot maximum. Cette approche permet ainsi une séparation efficace des contraintes de sous-tours dans les solutions fractionnaires (voir Padberg & Rinaldi (1990)).

1.5 Décomposition de Benders

1.5.1 Présentation

La décomposition de Benders, introduite par Benders (1962), constitue une technique d'optimisation particulièrement adaptée à la résolution de problèmes de conception de réseaux présentant une structure à deux niveaux étudiés dans ce mémoire, dans lesquels les décisions de premier niveau définissent l'infrastructure du réseau, tandis que celles de second niveau correspondent au routage des demandes sur le réseau ainsi construit. La méthode exploite naturellement cette séparation : lorsque le réseau est fixé, il reste à résoudre un ensemble de sous-problèmes de routage qui se révèlent simple étant donné la structure fixée.

Le principe repose sur une approche itérative consistant à résoudre un problème maître restreint, portant sur les seules variables de conception, auquel sont progressivement ajoutées des contraintes appelées coupes, générées à partir de la résolution des sous-problèmes de routage. Ces coupes traduisent dans le problème maître l'information issue du second niveau, permettant ainsi d'obtenir la solution optimale du problème global en enrichissant dynamiquement le modèle avec les contraintes pertinentes, sans avoir à formuler explicitement l'intégralité du problème.

Des synthèses modernes concernant les avantages et les limites de la décomposition de Benders, incluant ses extensions et ses applications récentes, sont proposées par Costa (2005); Rahmaniani, Crainic, Gendreau & Rei (2017). De nombreux travaux ont contribué à améliorer les performances de cette méthode. En particulier, Magnanti & Wong (1981) introduisent de nouvelles stratégies de génération de coupes visant à accélérer la convergence. Plus récemment, Rahmaniani, Ahmed, Crainic, Gendreau & Rei (2020) proposent une approche unifiée combinant décomposition de Benders et relaxation lagrangienne, permettant de renforcer la qualité des coupes générées. Par ailleurs, l'efficacité de la méthode est illustrée dans divers contextes applicatifs, notamment en conception de réseaux et en transport Magnanti & Wong (1984); Errico, Crainic, Malucelli & Nonato (2017); Fontaine & Minner (2018).

1.5.2 Reformulation de Benders

Considérons le problème de programmation linéaire suivant :

$$\min \quad c^\top x + q^\top y \tag{1.8a}$$

$$\text{s.c.} \quad Ax = b, \tag{1.8b}$$

$$Tx + Wy \geqslant h, \tag{1.8c}$$

$$x \geqslant 0, \tag{1.8d}$$

$$y \geqslant 0. \tag{1.8e}$$

Nous faisons ici l'hypothèse que si le vecteur de décision x est fixé, alors le problème consistant à trouver le vecteur optimal y^* est simple. Les variables x définissent le polyèdre

$$\mathcal{A} = \{x : Ax = b,\ x \geqslant 0\}, \tag{1.9}$$

indépendant de y, tandis que les variables y sont contraintes par

$$Wy \geqslant h - Tx, \tag{1.10}$$

ce qui dépend du choix de x.

On suppose dans la suite que $\mathcal{A}$ est borné et que le problème admet une solution optimale.

Pour tout $x \in \mathcal{A}$ fixé, on définit le sous-problème primal (SPP) :

$$f_P(x) = \min\{q^\top y : Wy \geqslant h - Tx,\ y \geqslant 0\}. \tag{1.11}$$

Le problème initial se reformule alors comme :

$$\min \quad c^\top x + f_P(x) \tag{1.12a}$$

$$\text{s.c.} \quad x \in \mathcal{A}. \tag{1.12b}$$

Le dual du SPP, que nous appellerons sous-problème dual (SPD), s'écrit :

$$f_D(x) = \max\{u^\top (h - Tx) : W^\top u \leqslant q,\ u \geqslant 0\}. \tag{1.13}$$

La fonction $f_D(x)$ est convexe et linéaire par morceaux en x, et par dualité forte $f_D(x) = f_P(x)$ dès que le SPP est réalisable.

Le polyèdre admissible du SPD,

$$\mathcal{U} = \{u : W^\top u \leqslant q,\ u \geqslant 0\}, \tag{1.14}$$

ne dépend pas de x : seule la fonction objectif du SPD est affectée par le choix de x. Cette indépendance est la propriété structurelle qui rend possible la décomposition de Benders.

Nous notons :

- $\mathcal{O}$ l'ensemble des points extrêmes de $\mathcal{U}$;
- $\mathcal{F}$ l'ensemble des rayons extrêmes de $\mathcal{U}$, c'est-à-dire les rayons extrêmes de son cône de récession $\mathrm{rec}(\mathcal{U})$.

Ces deux ensembles sont finis, puisque $\mathcal{U}$ est un polyèdre. D'après le théorème de représentation de Minkowski–Weyl (voir Bertsimas & Tsitsiklis (1997)), tout point $u \in \mathcal{U}$ s'écrit comme la somme d'une combinaison convexe de points de $\mathcal{O}$ et d'une combinaison conique de rayons de $\mathcal{F}$. Dans la suite, $\mathcal{O}$ interviendra dans la construction des coupes d'optimalité et $\mathcal{F}$ dans celle des coupes de faisabilité.

Cône de récession de $\mathcal{U}$

Le cône de récession représente les directions le long desquelles on peut se déplacer indéfiniment dans $\mathcal{U}$:

$$\mathrm{rec}(\mathcal{U}) = \{d \in \mathbb{R}^m : W^\top d \leqslant 0,\ d \geqslant 0\}. \tag{1.15}$$

Démonstration. Par définition, $d \in \mathrm{rec}(\mathcal{U})$ si et seulement si $u + \lambda d \in \mathcal{U}$ pour tout $u \in \mathcal{U}$ et tout $\lambda \geqslant 0$, ce qui équivaut à vérifier les deux contraintes de $\mathcal{U}$.

Contrainte $W^\top u \leqslant q$: $W^\top(u + \lambda d) = W^\top u + \lambda W^\top d \leqslant q$ pour tout $\lambda \geqslant 0$ si et seulement si $W^\top d \leqslant 0$.

Contrainte $u \geqslant 0$: $u + \lambda d \geqslant 0$ pour tout $u \geqslant 0$ et tout $\lambda \geqslant 0$ si et seulement si $d \geqslant 0$.

La conjonction donne le résultat. □

Infaisabilité du SPP

Par dualité forte, le SPP est infaisable si et seulement si le SPD est non borné. Le SPD est non borné si et seulement s'il existe une direction $d \in \mathrm{rec}(\mathcal{U})$ le long de laquelle son objectif augmente sans limite.

Démonstration. Soit $u \in \mathcal{U}$ et $d \in \mathrm{rec}(\mathcal{U})$. On a $u + \lambda d \in \mathcal{U}$ pour tout $\lambda \geqslant 0$, et

$$(u + \lambda d)^\top (h - Tx) = u^\top (h - Tx) + \lambda\, d^\top (h - Tx). \tag{1.16}$$

Si $d^\top(h - Tx) > 0$, l'objectif tend vers $+\infty$ lorsque $\lambda \to +\infty$, et le SPD est non borné. Réciproquement, si le SPD est non borné, il existe nécessairement un rayon $d \in \mathrm{rec}(\mathcal{U})$ vérifiant $d^\top(h - Tx) > 0$. □

Autrement dit, tout vecteur $d \in \text{rec}(\mathcal{U})$ tel que $d^\top(h - Tx) > 0$ constitue un certificat d'infaisabilité du SPP pour la solution x courante.

1.5.2.1 Coupes de faisabilité

Pour qu'une solution $x \in \mathcal{A}$ rende le SPP réalisable, il faut donc empêcher l'existence d'un tel certificat, c'est-à-dire imposer :

$$d^\top(h - Tx) \leqslant 0, \quad \forall d \in \text{rec}(\mathcal{U}). \tag{1.17}$$

Le cône $\text{rec}(\mathcal{U})$ étant engendré par ses rayons extrêmes $\mathcal{F}$, il suffit d'imposer sur $\mathcal{F}$:

$$d^\top(h - Tx) \leqslant 0, \quad \forall d \in \mathcal{F}. \tag{1.18}$$

À chaque itération de l'algorithme, lorsque la solution courante $\bar{x}$ rend le SPP infaisable, on identifie un rayon $d \in \mathcal{F}$ violé et l'on ajoute la coupe correspondante au problème maître.

1.5.2.2 Coupes d'optimalité

Supposons que la solution courante $\bar{x}$ rende le SPP réalisable. Par dualité forte, $f_P(\bar{x}) = f_D(\bar{x})$ et cette valeur commune est finie.

Pour intégrer le coût de second niveau dans le problème maître, nous introduisons une variable auxiliaire η destinée à représenter $f_D(x)$. Le problème initial se reformule alors comme :

$$\begin{aligned}
\min \quad & c^\top x + \eta && \text{(1.19a)}\\
\text{s.c.} \quad & Ax = b, && \text{(1.19b)}\\
& \eta \geqslant f_D(x), && \text{(1.19c)}\\
& x \geqslant 0. && \text{(1.19d)}
\end{aligned}$$

La contrainte $\eta \geqslant f_D(x)$ n'est pas directement exploitable : $f_D(x)$ est définie comme la valeur optimale d'un problème d'optimisation dépendant de x, et ne peut donc pas s'écrire telle quelle comme une contrainte linéaire dans le problème maître.

Pour contourner cette difficulté, on utilise le fait que $f_D(x)$ est le maximum d'une fonction linéaire sur le polyèdre $\mathcal{U}$. Cet optimum est nécessairement atteint en un point extrême de $\mathcal{U}$, d'où :

$$f_D(x) = \max_{u \in \mathcal{O}} u^\top (h - Tx). \tag{1.20}$$

Autrement dit, $f_D(x)$ s'écrit comme le maximum d'un nombre fini de fonctions linéaires en x, indexées par les points extrêmes $u \in \mathcal{O}$.

La condition $\eta \geqslant f_D(x)$ équivaut alors à imposer $\eta \geqslant u^{*\top}(h - Tx)$ pour le point extrême u^* qui réalise ce maximum. Comme u^* dépend de x et n'est pas connu a priori, on impose l'inégalité pour tous les points extrêmes :

$$u^\top (h - Tx) \leqslant \eta, \quad \forall u \in \mathcal{O}. \tag{1.21}$$

1.5.2.3 Problème maître restreint

En combinant les éléments précédents, le problème maître complet s'écrit :

$$\begin{aligned}
\min \quad & c^\top x + \eta && \text{(1.22a)}\\
\text{s.c.} \quad & Ax = b, && \text{(1.22b)}\\
& d^\top (h - Tx) \leqslant 0, \quad \forall d \in \mathcal{F}, && \text{(1.22c)}\\
& u^\top (h - Tx) \leqslant \eta, \quad \forall u \in \mathcal{O}, && \text{(1.22d)}\\
& x \geqslant 0. && \text{(1.22e)}
\end{aligned}$$

Les ensembles $\mathcal{F}$ et $\mathcal{O}$ étant typiquement des ensembles impossibles à énumérer en pratique, ces coupes ne sont pas générées a priori : on les génère dynamiquement au fil des itérations à partir des sous-problèmes résolus. Le problème maître restreint, qui ne contient à chaque étape qu'un sous-ensemble de ces coupes, fournit alors une borne inférieure sur la valeur optimale, raffinée à chaque ajout d'une coupe violée jusqu'à convergence.

1.5.3 Algorithme

Initialement, le Problème Maître Restreint est résolu sans aucune coupe de faisabilité ni d'optimalité.

Algorithme 1.1 Décomposition de Benders

Output : Solution optimale du problème

Initialiser le problème maître sans aucune coupe et η libre
while ***true*** **do**
    Résoudre le problème maître restreint et obtenir $(\hat{x}, \hat{\eta})$
    Résoudre le sous-problème dual $f_D(\hat{x})$
    if $f_D(\hat{x})$ *est non borné* **then**
        Extraire un rayon extrême u'
        Ajouter la coupe de faisabilité : $u'^{\top}(h - Tx) \leqslant 0$
    else if $f_D(\hat{x}) > \hat{\eta}$ **then**
        Obtenir le point extrême optimal u^* tel que $f_D(\hat{x}) = u^{*\top}(h - T\hat{x})$
        Ajouter la coupe d'optimalité : $u^{*\top}(h - Tx) \leqslant \eta$
    else
        break
    end if
end while

1.6 Les systèmes de transport semi-flexibles

Le TSP-GL, présenté dans la section suivante, trouve sa motivation dans les systèmes de transport semi-flexibles, que nous décrivons brièvement ici afin de fournir le contexte dans lequel ce problème a été initialement formulé.

Les systèmes de transport traditionnels consistent généralement en une séquence d'arrêts obligatoires à effectuer sous des contraintes de temps strictes. Prenons l'exemple d'un système ferroviaire : une ligne est définie par une gare de départ, une gare d'arrivée et un certain nombre de gares intermédiaires. Le train assurant cette ligne s'arrête à toutes les gares prévues à des horaires prédéfinis, quel que soit le volume de passagers transitant par chacune d'elles. Ce type de modèle est appelé système de transport statique (ou régulier).

À l'autre extrémité du spectre, on trouve des systèmes de transport entièrement guidés par la demande des usagers. Un exemple archétypal est le service de taxis : les véhicules ne se déplacent et ne définissent leur itinéraire qu'en réponse à la requête d'un client. Les arrêts effectués sont donc exclusivement déterminés par les points de prise en charge et de dépose requis. Ce type de modèle est appelé système de transport flexible.

À l'intersection de ces deux paradigmes se trouvent les systèmes de transport semi-flexibles (SFTS pour *Semi-Flexible Transit Systems*). Errico (2008); Errico, Crainic, Malucelli & Nonato (2013) proposent l'une des premières synthèses systématiques de ces systèmes en définissant un cadre méthodologique clair. Contrairement aux systèmes entièrement flexibles, les systèmes semi-flexibles s'appuient sur une ligne de base comportant des arrêts obligatoires (indépendants de la demande journalière) et des horaires de référence. Cependant, ils intègrent également des arrêts optionnels qui ne sont visités que si une demande explicite a été formulée au préalable par un usager, permettant ainsi au véhicule de dévier de sa route principale tout en garantissant le respect des contraintes horaires aux arrêts fixes.

1.7 Le problème du voyageur de commerce avec latence généralisée

1.7.1 Description du problème

Le TSP-GL (*Travelling Salesman Problem with Generalized Latency*) est une extension du problème classique du voyageur de commerce. Dans ce problème introduit par Errico, Crainic, Malucelli & Nonato (2011), l'objectif est de construire un cycle hamiltonien sur un graphe $G = (N, E)$, où N est l'ensemble des sommets et E un ensemble d'arêtes non dirigées, puis sur ce cycle, de router un ensemble de demandes D.

On minimise donc une fonction objectif qui combine :

- Les coûts de conception du circuit hamiltonien.
- Les coûts de routage des demandes.

1.7.2 Ensembles

On note les ensembles suivants :

- N : Ensemble de nœuds.
- E : Ensemble d'arêtes non dirigées.
- A : Ensemble d'arcs dirigés (sans contrainte de capacité).
- D : Ensemble de la demande à router, sous la forme de couple origine-destination.

1.7.3 Paramètres

- $\alpha \in [0; 1]$: ce paramètre permet de définir l'importance accordée à la composante de latence du problème. Plus α est élevé, plus le problème devient compliqué à résoudre.
- d_{hk} $\forall (h, k) \in D$: quantité de demande à router entre le nœud h et k.
- Les coûts c_{ij} $\forall (i, j) \in E$ correspondent à la distance euclidienne entre les nœuds i et j.

- Les coûts $q_{ij}^{hk},\ \forall (i,j) \in A,\ \forall (h,k) \in D$ sont définis par :

$$q_{ij}^{hk} = \frac{d_{hk} \cdot c_{ij}}{\sum_{(h,k)\in D} d_{hk}}. \tag{1.23}$$

$\delta(S) \subseteq E$ désigne l'ensemble des arêtes ayant exactement une extrémité dans S, et $\delta(i)$ est une abréviation pour $\delta(\{i\})$.

De même, $\delta^+(S) \subseteq A$ (resp. $\delta^-(S) \subseteq A$) désigne l'ensemble des arcs dont la queue appartient à S et la tête à $N \setminus S$ (resp. dont la tête appartient à S et la queue à $N \setminus S$). Nous utilisons $\delta^+(i)$ (resp. $\delta^-(i)$) comme abréviation pour $\delta^+(\{i\})$ (resp. $\delta^-(\{i\})$).

Étant donné un sous-ensemble $E^* \subseteq E$ et un sous-ensemble $A^* \subseteq A$, on écrit $x(E^*)$ pour désigner $\sum_{e\in E^*} x_e$ et $f_{hk}(A^*)$ pour désigner $\sum_{(i,j)\in A^*} f_{ij}^{hk}$.

1.7.4 Variables de décision

- $x_e \in \{0,1\}\ \forall e \in E$: variable binaire qui prend la valeur 1 si l'arête $e = [i,j]$ est sélectionnée dans le circuit hamiltonien, sinon la variable prend la valeur 0.
- $f_{ij}^{hk} \geq 0\ \forall\ (i,j) \in A, \forall\ (h,k) \in D$: variable de flot définie pour chaque demande et pour chaque arc orienté. Étant donné un circuit hamiltonien, chaque demande doit être routée sur l'un des deux chemins possibles entre les nœuds h et k selon la direction. L'ensemble des arcs dirigés A ne possède pas de contraintes de capacité ; autrement dit, la demande routée entre les nœuds h et k suivra le plus court chemin entre ces nœuds. En raison de l'absence de contraintes de capacité, la valeur de f_{ij}^{hk} sera alors de 1 si la demande $(h,k) \in D$ passe par l'arc $(i,j) \in A$, et de zéro sinon.

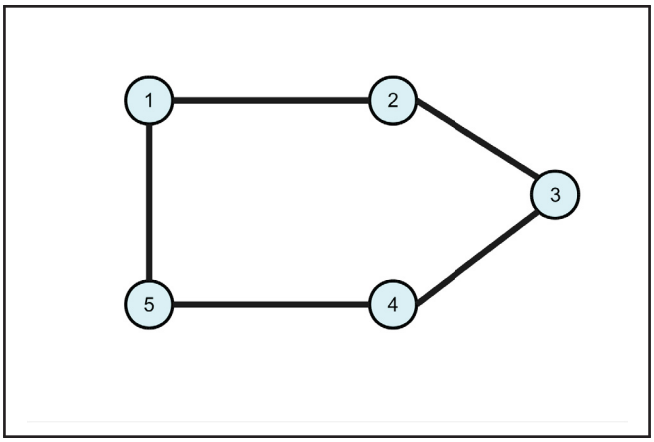

Figure 1.2 Premier niveau de décision du TSP-GL : choix du support de routage

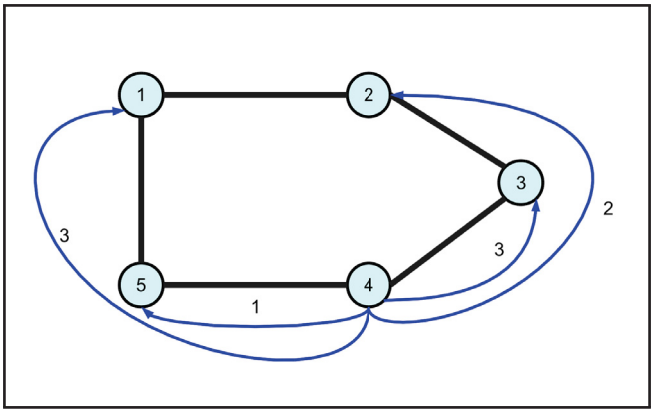

Figure 1.3 Deuxième niveau de décision du TSP-GL : routage des demandes

1.7.5 Modèle mathématique

$$\min \quad (1-\alpha)\sum_{e\in E} c_e x_e + \alpha \sum_{(i,j)\in A}\sum_{(h,k)\in D} q_{ij}^{hk} f_{ij}^{hk} \tag{1.24a}$$

$$x(\delta(i)) = 2 \qquad \forall i \in N \tag{1.24b}$$

$$f^{hk}(\delta^+(i)) - f^{hk}(\delta^-(i)) = \begin{cases} 1 & \text{si } i = h, \\ -1 & \text{si } i = k, \\ 0 & \text{sinon,} \end{cases} \qquad \forall (h,k) \in D,\ \forall i \in N \tag{1.24c}$$

$$f_{ij}^{hk} \geq 0 \qquad \forall (i,j) \in A,\ \forall (h,k) \in D \tag{1.24d}$$

$$x_e - f_{ij}^{hk} \geq 0 \qquad \forall e = [i,j] \in E,\ \forall (h,k) \in D \tag{1.24e}$$

$$x_e - f_{ji}^{hk} \geq 0 \qquad \forall e = [i,j] \in E,\ \forall (h,k) \in D \tag{1.24f}$$

$$x_e \in \{0,1\} \qquad \forall e \in E \tag{1.24g}$$

Comme dit précédemment, la fonction objectif du problème minimise deux composantes ; tout d'abord, la composante de *design* qui calcule les coûts de construction du circuit hamiltonien. La deuxième composante, appelée composante de latence, calcule les coûts de routage de l'ensemble des demandes dans D sur le circuit hamiltonien sélectionné.

La contrainte (1.24b) impose que, pour chaque nœud, il faut sélectionner deux arêtes.

La contrainte (1.24c) assure l'équilibre du flot dans le graphe.

Les contraintes (1.24e) et (1.24f) s'assurent qu'il ne soit pas possible que le flot d'une demande $(h,k) \in D$ passe par une arête $[i,j] \in E$ qui ne soit pas sélectionnée dans le circuit.

Nous pouvons observer que dans cette formulation, le routage de la demande est modélisé par une variable de flot unitaire pour chaque demande. La quantité de demande d_{hk} est prise en compte dans le coût q_{ij}^{hk}.

Enfin, il est essentiel de souligner que cette formulation n'empêche pas l'apparition de sous-tours, cependant, un nombre suffisant de demandes D permet de trouver naturellement une solution sans sous-tours.

1.7.6 Difficultés liées au TSP-GL

Le TSP-GL est un problème fondamentalement complexe à résoudre. Comme le problème du voyageur de commerce classique, le TSP-GL est un problème NP-difficile ; autrement dit, nous ne connaissons pas d'algorithme permettant de le résoudre en temps polynomial. De plus, le TSP-GL présente un nombre élevé de variables et de contraintes. Même la relaxation linéaire du problème, lorsque la taille des instances de données augmente, devient compliquée à résoudre, et la borne inférieure qu'elle fournit ne se révèle pas de bonne qualité. L'augmentation de la valeur du paramètre α rend le problème particulièrement complexe à résoudre. On remarque d'ailleurs que, lorsque $\alpha = 0$, la composante de latence n'est pas prise en compte dans la fonction objectif, et l'on se retrouve dans le cas d'un problème de voyageur de commerce classique.

Cependant, le TSP-GL présente certaines caractéristiques très intéressantes, parmi lesquelles sa similarité avec le problème du voyageur de commerce. En effet, toutes les inégalités valides pour le problème du voyageur de commerce sont également valides pour le TSP-GL.

1.7.7 Décomposition de Benders pour le TSP-GL

La décomposition de Benders s'avère particulièrement efficace pour le TSP-GL. En fixant un circuit hamiltonien défini par les variables $\bar{x}_e$, la composante de latence peut être déduite en résolvant, pour chaque demande $(h, k) \in D$, un problème de plus court chemin sur le graphe induit par $\bar{x}$.

1.7.7.1 Sous-problème dual

Pour chaque demande $(h,k) \in D$, on exprime le sous-problème dual suivant :

$$
\begin{aligned}
z^{hk}(\bar{x}) = \max \quad & \rho_h^{hk} - \rho_k^{hk} - \sum_{e \in E} \bar{\lambda}_e^{hk} \bar{x}_e && && \text{(1.25a)}\\
\text{s.c.} \quad & \rho_i^{hk} - \rho_j^{hk} - \lambda_{ij}^{hk} \leqslant q_{ij}^{hk} && \forall (i,j) \in A && \text{(1.25b)}\\
& \lambda_{ij}^{hk} \geqslant 0 && \forall (i,j) \in A && \text{(1.25c)}\\
& \rho_i^{hk} \in \mathbb{R} && \forall i \in N && \text{(1.25d)}
\end{aligned}
$$

où $\bar{\lambda}_e^{hk} = \lambda_{ij}^{hk} + \lambda_{ji}^{hk}$ pour chaque arête $e = \{i,j\}$. On note :

$$
Q = \left\{ (\lambda, \rho) \in \mathbb{R}^{(|A|+|N|)\,|D|} \;\middle|\; (\lambda^{hk}, \rho^{hk}) \text{ satisfont (1.25b) – (1.25d)},\ \forall (h,k) \in D \right\} \qquad \text{(1.26)}
$$

Le TSP-GL peut alors être reformulé intégralement dans l'espace des variables x et d'une variable auxiliaire η représentant le coût total de latence :

$$
\begin{aligned}
\min \quad & (1-\alpha) \sum_{e \in E} c_e x_e + \alpha \eta && && \text{(1.27a)}\\
\text{s.c.} \quad & \eta \geqslant \sum_{(h,k) \in D} \left(\rho_h^{hk} - \rho_k^{hk} - \sum_{e \in E} \lambda_e^{hk} x_e \right) && \forall (\rho, \lambda) \in \text{extr}(Q) && \text{(1.27b)}\\
& \sum_{(h,k) \in D} \left(\rho_h^{hk} - \rho_k^{hk} - \sum_{e \in E} \lambda_e^{hk} x_e \right) \leqslant 0 && \forall (\rho, \lambda) \in \text{rays}(Q) && \text{(1.27c)}\\
& x(\delta(i)) = 2 && \forall i \in N && \text{(1.27d)}\\
& x_e \in \{0,1\} && \forall e \in E && \text{(1.27e)}
\end{aligned}
$$

On désigne alors comme problème maître restreint la reformulation de Benders sans aucune coupe de faisabilité ni d'optimalité, mais seulement avec les contraintes de degré.

Plutôt que d'utiliser les coupes de faisabilité génériques (1.27c) basées sur les rayons extrêmes de Q, nous exploitons la structure combinatoire du problème. Si une solution $\bar{x}$ du problème maître restreint est infaisable, celle-ci contient par conséquent des sous-tours ; nous ajoutons alors les contraintes d'élimination des sous-tours correspondantes :

$$x(\delta(S)) \geq 2 \qquad \forall\, S \subset N,\ 2 \leq |S| \leq |N| - 2, \tag{1.28}$$

Cette approche est plus robuste car elle utilise les propriétés connues du polyèdre du TSP (voir Gutin & Punnen (2002)).

À l'origine, la décomposition de Benders ne génère des coupes que pour des solutions entières. Cela pose une difficulté naturelle, pour de grandes instances de données, l'obtention de solutions entières dans l'arbre de branchement devient de plus en plus compliquée. Naturellement, un certain nombre de chercheurs se sont penchés sur cette problématique, et McDaniel & Devine (1977) ont proposé une nouvelle approche en montrant que l'on peut utiliser une solution obtenue par relaxation linéaire pour obtenir des coupes de Benders valides. Autrement dit, à chaque nœud fractionnaire, il est possible d'ajouter une coupe de faisabilité ou d'optimalité. Concernant les coupes de faisabilité s'appliquant aux solutions fractionnaires, on utilisera les contraintes présentées dans la section 1.4.4.2.

Algorithme 1.2 Décomposition de Benders appliquée au TSP-GL

Output : Solution optimale du TSP-GL

while ***true*** **do**
  Résoudre le problème maître et obtenir $(\hat{x}, \hat{\eta})$
  $sous\_tour \leftarrow$ **faux**
  foreach $(h, k) \in D$ **do**
    Résoudre le sous-problème dual $z^{hk}(\hat{x})$
    if $z^{hk}(\hat{x})$ *est non borné* **then**
      Ajouter la contrainte d'élimination de sous-tour correspondante
      $sous\_tour \leftarrow$ **vrai**
      break
    else
      Récupérer $(\lambda^{hk}, \rho^{hk})$
    end if
  end foreach
  if *sous_tour* **then**
    continue
  end if
  if $\hat{\eta} < \sum_{(h,k)\in D} \left(\rho_h^{hk} - \rho_k^{hk} - \sum_{e\in E} \bar{\lambda}_e^{hk} \hat{x}_e \right)$ **then**
    Ajouter la coupe d'optimalité :
    $\eta + \sum_{(h,k)\in D} \sum_{e\in E} \bar{\lambda}_e^{hk} x_e \geq \sum_{(h,k)\in D} \left(\rho_h^{hk} - \rho_k^{hk} \right)$
  else
    break
  end if
end while

1.7.7.2 Coupes désagrégées

Il est possible d'introduire une variable η^{hk} pour chaque demande $(h,k) \in D$, telle que $\eta = \sum_{(h,k)\in D} \eta^{hk}$. La fonction objectif devient alors :

$$\min \quad (1-\alpha) \sum_{e\in E} c_e x_e + \alpha \sum_{(h,k)\in D} \eta^{hk} \tag{1.29}$$

Chaque demande génère alors sa propre coupe d'optimalité :

$$\eta^{hk} + \sum_{e\in E} \lambda_e^{hk} x_e \geqslant \rho_h^{hk} - \rho_k^{hk}. \tag{1.30}$$

Bien que cette désagrégation augmente le nombre de contraintes dans le problème maître, elle réduit souvent le nombre total d'itérations nécessaires à la convergence.

1.8 Problème de conception de réseau multicommodité sans contraintes de capacité avec coûts fixes

1.8.1 Description du problème

Le problème de conception de réseau multicommodité sans contraintes de capacité avec coûts fixes (MUFND) est un problème classique d'optimisation de réseaux (Zetina, Contreras & Cordeau (2019); Magnanti & Wong (1984)). Comme le TSP-GL, ce problème implique deux niveaux de décision : un premier niveau de conception déterminant quels arcs installer dans le réseau, et un second niveau de routage déterminant comment router un ensemble de demandes à travers l'infrastructure réseau installée. Dans cette formulation, chaque arc, une fois installé, peut router un volume de flot illimité et la structure de premier niveau n'est pas contrainte à être un cycle.

Le MUFND est défini sur un graphe orienté $G = (N, A)$ où N est l'ensemble des nœuds et A est l'ensemble des arcs. L'objectif est d'installer un sous-ensemble d'arcs et de router toutes les demandes de manière à minimiser les coûts d'installation fixes totaux plus les coûts de routage de chaque demande.

1.8.2 Ensembles

- N : Ensemble des nœuds du réseau.
- A : Ensemble des arcs orientés (sans contrainte de capacité).
- K : Ensemble des demandes.

1.8.3 Paramètres

- $\bar{c}_{ij} \geq 0 \quad \forall (i,j) \in A$: Coût d'installation fixe de l'arc (i,j).
- $q_{ij}^k \geq 0 \quad \forall (i,j) \in A, \forall k \in K$: Coût de routage unitaire pour la demande k sur l'arc (i,j).
- $d_k > 0 \quad \forall k \in K$: Volume de demande k.
- $(s_k, h_k) \quad \forall k \in K$: Nœud d'origine et nœud de destination de la demande k.

1.8.4 Variables de décision

- $x_{ij} \in \{0,1\} \quad \forall (i,j) \in A$: Variable binaire égale à 1 si l'arc (i,j) est installé dans la conception du réseau, 0 sinon.
- $f_{ij}^k \geq 0 \quad \forall (i,j) \in A, \forall k \in K$: Variable de flot représentant la quantité de demande k routée sur l'arc (i,j).

1.8.5 Modèle mathématique

Basée sur la formulation de Zetina *et al.* (2019), nous employons la formulation basée sur les arcs avec des contraintes de liaison désagrégées. Cette approche fournit une meilleure relaxation linéaire que la formulation agrégée et est mieux adaptée à la décomposition de Benders (Magnanti & Wong (1981)). Le MUFND est formulé comme un problème de minimisation :

$$\min \quad \sum_{(i,j)\in A} \bar{c}_{ij} x_{ij} + \sum_{k\in K} \sum_{(i,j)\in A} d_k q_{ij}^k f_{ij}^k \tag{1.31a}$$

$$\text{s.c.} \quad \sum_{j\in N} f_{ji}^k - \sum_{j\in N} f_{ij}^k = \begin{cases} -1 & \text{si } i = s_k \\ 1 & \text{si } i = h_k \\ 0 & \text{sinon} \end{cases} \qquad \forall i \in N, \forall k \in K \tag{1.31b}$$

$$f_{ij}^k \leq x_{ij} \qquad \forall (i,j) \in A, \forall k \in K \tag{1.31c}$$

$$f_{ij}^k \geq 0 \qquad \forall (i,j) \in A, \forall k \in K \tag{1.31d}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{1.31e}$$

La fonction objectif (1.31a) minimise le coût total, combinant les coûts d'installation fixes des arcs et les coûts de routage pondérés par les volumes de demande d_k. Les contraintes de conservation de flot (1.31b) assurent que chaque demande k a un flot sortant d'une unité à son origine s_k et un flot entrant d'une unité à sa destination h_k. Les contraintes de liaison (1.31c) imposent que la demande k ne peut être routée sur l'arc (i, j) que si cet arc est installé dans la conception du réseau.

1.8.6 Décomposition de Benders pour le MUFND

La décomposition de Benders s'avère particulièrement efficace pour le MUFND. En fixant une conception du réseau définie par les variables $\bar{x}_{ij}$, le problème de routage se décompose en $|K|$ sous-problèmes indépendants, chacun correspondant à une demande k.

Soit $\bar{x} \in \mathcal{X} = \{0,1\}^{|A|}$ un réseau fixé. Le sous-problème primal (SPP) décompose en $|K|$ sous-problèmes indépendants, un pour chaque demande $k \in K$:

$$\text{(SPP)} \quad \min \quad \sum_{k \in K} \sum_{(i,j) \in A} d_k q_{ij}^k f_{ij}^k \tag{1.32a}$$

$$\text{s.c.} \quad \sum_{j \in N} f_{ji}^k - \sum_{j \in N} f_{ij}^k = \begin{cases} -1 & \text{si } i = s_k \\ 1 & \text{si } i = h_k \\ 0 & \text{sinon} \end{cases} \qquad \forall i \in N, \forall k \in K \tag{1.32b}$$

$$f_{ij}^k \leq \bar{x}_{ij} \qquad \forall (i,j) \in A, \forall k \in K \tag{1.32c}$$

$$f_{ij}^k \geq 0 \qquad \forall (i,j) \in A, \forall k \in K \tag{1.32d}$$

Par dualité forte, chaque sous-problème peut être exprimé de façon équivalente sous forme duale. Soit λ_i^k les variables duales associées aux contraintes de conservation du flot (1.32b), et μ_{ij}^k les variables duales associées aux contraintes de liaison (1.32c). Le sous-problème dual pour la demande k (SPD^k) est :

$$\text{(SPD}^k\text{)} \quad \max \quad \lambda_{h_k}^k - \lambda_{s_k}^k - \sum_{(i,j) \in A} \mu_{ij}^k \bar{x}_{ij} \tag{1.33a}$$

$$\text{s.c.} \quad \lambda_j^k - \lambda_i^k - \mu_{ij}^k \leq d_k q_{ij}^k \qquad \forall (i,j) \in A \tag{1.33b}$$

$$\mu_{ij}^k \geq 0 \qquad \forall (i,j) \in A \tag{1.33c}$$

$$\lambda_i^k \in \mathbb{R} \qquad \forall i \in N \tag{1.33d}$$

Soit $\mathcal{E}^k$ l'ensemble des rayons extrêmes du SPD^k et $\mathcal{V}^k$ l'ensemble des points extrêmes du SPD^k. En introduisant des variables continues z_k représentant le coût de routage pour chaque

demande $k \in K$, on écrit alors la reformulation de Benders suivante :

$$\text{(PM)} \quad \min \quad \sum_{(i,j)\in A} \bar{c}_{ij} x_{ij} + \sum_{k\in K} z_k \tag{1.34a}$$

$$\text{s.c.} \quad z_k \geq \lambda^k_{h_k} - \lambda^k_{s_k} - \sum_{(i,j)\in A} \mu^k_{ij} x_{ij} \qquad \forall k \in K,\ (\lambda^k, \mu^k) \in \mathcal{V}^k \tag{1.34b}$$

$$0 \geq \bar{\lambda}^k_{h_k} - \bar{\lambda}^k_{s_k} - \sum_{(i,j)\in A} \bar{\mu}^k_{ij} x_{ij} \qquad \forall k \in K,\ (\bar{\lambda}^k, \bar{\mu}^k) \in \mathcal{E}^k \tag{1.34c}$$

$$z_k \in \mathbb{R} \qquad \forall k \in K \tag{1.34d}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in A \tag{1.34e}$$

Les contraintes (1.34b) sont les coupes d'optimalité générées à partir des points extrêmes de chaque dual, tandis que les contraintes (1.34c) sont les coupes de faisabilité générées à partir des rayons extrêmes. Ces coupes se raffinent itérativement jusqu'à convergence vers l'optimum du problème original.

Algorithme 1.3 Décomposition de Benders appliquée au MUFND

Output : Solution optimale du MUFND

while ***true*** **do**
  Résoudre le problème maître et obtenir $(\hat{x}, \{\hat{z}_k\}_{k \in K})$
  ajout_coupe $\leftarrow$ **faux**
  foreach $k \in K$ **do**
    Résoudre SPD$^k(\hat{x})$
    if *SPD*$^k(\hat{x})$ *est non borné* **then**
      Récupérer un rayon extrême $(\bar{\lambda}^k, \bar{\mu}^k)$
      Ajouter la coupe de faisabilité : $0 \geq \bar{\lambda}^k_{h_k} - \bar{\lambda}^k_{s_k} - \sum_{(i,j) \in A} \bar{\mu}^k_{ij} x_{ij}$
      ajout_coupe $\leftarrow$ **vrai**
    else
      Récupérer le point extrême optimal (λ^k, μ^k)
      if $\hat{z}_k < \lambda^k_{h_k} - \lambda^k_{s_k} - \sum_{(i,j) \in A} \mu^k_{ij} \hat{x}_{ij}$ **then**
        Ajouter la coupe d'optimalité : $z_k \geq \lambda^k_{h_k} - \lambda^k_{s_k} - \sum_{(i,j) \in A} \mu^k_{ij} x_{ij}$
        ajout_coupe $\leftarrow$ **vrai**
      end if
    end if
  end foreach
  if $\neg$ *ajout_coupe* **then**
    break
  end if
end while

1.9 Sample Average Approximation

Dans cette section, nous présentons le cadre théorique du *Sample Average Approximation* (SAA), méthode d'approximation pour les problèmes d'optimisation stochastique sur laquelle s'appuie

l'une des contributions de ce mémoire. Nous introduisons d'abord la classe des problèmes d'optimisation stochastique à deux étapes, qui constitue le cadre de référence dans lequel le SAA est usuellement développé. Nous exposons ensuite le principe général de la méthode, qui consiste à approximer l'espérance intervenant dans la fonction objectif par une moyenne empirique calculée sur un échantillon aléatoire. Nous établissons ensuite deux propriétés théoriques fondamentales : l'absence de biais de l'estimateur ainsi obtenu, et la propriété de borne inférieure en espérance sur la valeur optimale du problème original. Nous décrivons enfin comment estimer numériquement ces bornes via des réplications indépendantes, construire des intervalles de confiance associés, et résumons l'ensemble dans une procédure algorithmique.

1.9.1 Problème d'optimisation stochastique à deux étapes

Nous présentons maintenant un nouveau type de problème, formalisé notamment par Dantzig (1955); Dantzig & Madansky (1960); El Agizy (1967), visant à représenter des processus décisionnels séquentiels en présence d'incertitude.

Le principe de ce modèle repose sur une formulation en deux niveaux de décision. Dans un premier temps, une décision de première étape est prise avant la réalisation d'événements aléatoires. Cette décision consiste à choisir une variable $x \in X$, où X désigne l'ensemble des solutions réalisables, et engendre un coût déterministe $c^\top x$.

Dans un second temps, l'incertitude se réalise. Les paramètres aléatoires, modélisés par un vecteur $\xi = (q, T, W, h)$ de loi connue, sont observés. Afin de corriger d'éventuelles inadéquations induites par la décision initiale, une action de recours est introduite sous la forme d'une variable y. Cette seconde étape consiste à résoudre un problème d'optimisation visant à satisfaire les contraintes $Tx + Wy \leqslant h$ tout en minimisant un coût $q^\top y$.

L'objectif est de déterminer une décision de première étape qui anticipe au mieux les réalisations futures de l'aléa. Cela conduit à formuler un problème d'optimisation stochastique tel que :

$$v^* = \min_{x \in X} g(x), \quad \text{où} \quad g(x) = c^\top x + \mathbb{E}_\xi[Q(x, \xi)], \tag{1.35}$$

où $Q(x, \xi) = \min\{q(\xi)^\top y : W(\xi)y \leqslant h(\xi) - T(\xi)x\}$ désigne la valeur optimale du problème de seconde étape pour une réalisation donnée du vecteur aléatoire ξ.

Une considération importante à observer concerne les cas où le problème de seconde étape peut s'avérer infaisable pour certaines réalisations. Si, pour une décision $x \in X$ et un scénario ξ, le système $Tx + Wy \leqslant h$ n'admet aucune solution, la convention standard consiste à poser $Q(x, \xi) = +\infty$, imposant ainsi une pénalité infinie.

On dit que le problème possède la propriété de recours relativement complet (*relatively complete recourse*) (voir Dantzig (1955); Madansky (1960); Birge & Louveaux (2011)) si une telle infaisabilité ne survient jamais, c'est-à-dire si pour tout $x \in X$ et tout scénario ξ, le problème de seconde étape est réalisable.

1.9.2 Principe du SAA

En pratique, le calcul exact de l'espérance $\mathbb{E}_\xi[Q(x, \xi)]$ est rarement possible, notamment lorsque l'ensemble des réalisations du vecteur aléatoire ξ est de grande taille ou infini.

Pour mettre en évidence la structure du problème, on introduit la notation suivante :

$$q(x) := \mathbb{E}_\xi[Q(x, \xi)]. \tag{1.36}$$

Le problème stochastique peut alors s'écrire sous la forme :

$$v^* = \min_{x \in X} \left\{ c^\top x + q(x) \right\}. \tag{1.37}$$

L'idée centrale de la méthode de *Sample Average Approximation* (SAA), développée notamment par Shapiro (1996); Kleywegt, Shapiro & Homem-de Mello (2002), consiste à approximer cette espérance par une moyenne empirique obtenue à partir d'un échantillon aléatoire.

On fait alors l'hypothèse que nous disposons d'un moyen d'échantillonner facilement des réalisations du vecteur aléatoire ξ.

Soit $\xi^1, \xi^2, \ldots, \xi^N$ un échantillon de N réalisations indépendantes et identiquement distribuées du vecteur aléatoire ξ. On définit alors l'approximation de $q(x)$ par :

$$\hat{q}_N(x) := \frac{1}{N} \sum_{j=1}^{N} Q(x, \xi^j). \tag{1.38}$$

Nous obtenons ainsi une approximation du problème original :

$$\hat{v}_N = \min_{x \in X} \left\{ \hat{g}_N(x) := c^\top x + \hat{q}_N(x) \right\} = \min_{x \in X} \left\{ c^\top x + \frac{1}{N} \sum_{j=1}^{N} Q(x, \xi^j) \right\}. \tag{1.39}$$

Autrement dit, la distribution de ξ est remplacée par une distribution discrète où chaque scénario ξ^j est associé à une probabilité uniforme de $1/N$. Ce nouveau problème que nous appellerons problème SAA est donc de même nature qu'un problème stochastique à deux étapes avec un nombre fini de scénarios.

En introduisant explicitement les variables de recours associées à chaque réalisation j, on obtient la formulation déterministe équivalente suivante :

$$\hat{v}_N = \min_{x, y_1, \ldots, y_N} \quad c^\top x + \frac{1}{N} \sum_{j=1}^{N} q_j^\top y_j \tag{1.40a}$$

$$\text{s.c.} \quad x \in X, \tag{1.40b}$$

$$T_j x + W_j y_j \leqslant h_j, \quad \forall j \in \{1, \ldots, N\}. \tag{1.40c}$$

Il est important de souligner que le SAA ne constitue pas un algorithme en tant que tel : il s'agit d'une méthode d'approximation du problème stochastique initial.

En ce sens, le SAA doit être compris comme un cadre méthodologique permettant de transformer un problème stochastique en un problème déterministe de grande taille, dont la structure dépend directement de celle du problème initial. Cette transformation ne fournit pas directement une méthode de résolution, mais définit une famille de problèmes qu'il convient ensuite de résoudre à l'aide de techniques d'optimisation appropriées. Une synthèse moderne des propriétés statistiques et computationnelles du SAA pour des problèmes d'optimisation combinatoire est proposée par de Mello & Bayraksan (2014).

Le choix de la méthode de résolution est donc crucial, en particulier lorsque le nombre de scénarios N est élevé. Dans le cas des problèmes à deux étapes, la structure du problème SAA se prête naturellement à l'utilisation de la décomposition de Benders, qui permet de traiter efficacement les formulations obtenues.

Plusieurs travaux ont ainsi étudié le couplage entre le SAA et la décomposition de Benders notamment Adulyasak, Cordeau & Jans (2015); Taherkhani, Alumur & Hosseini (2020); Özgürbüz, Bektaş & Çağatay Iris (2026).

1.9.3 Propriété de non-biais

L'approximation $\hat{g}_N(x)$ constitue un estimateur non biaisé de $g(x)$.

Démonstration.

$$\mathbb{E}[\hat{g}_N(x)] = \mathbb{E}\left[c^\top x + \frac{1}{N}\sum_{j=1}^{N} Q(x, \xi^j)\right] \tag{1.41}$$

$$= c^\top x + \frac{1}{N}\sum_{j=1}^{N} \mathbb{E}[Q(x, \xi^j)]. \tag{1.42}$$

Puisque les variables aléatoires $\xi^1, \ldots, \xi^N$ sont indépendantes et identiquement distribuées selon la loi de ξ, on a pour tout j :

$$\mathbb{E}[Q(x, \xi^j)] = \mathbb{E}_\xi[Q(x, \xi)]. \tag{1.43}$$

Il en résulte que :

$$\mathbb{E}[\hat{g}_N(x)] = c^\top x + \frac{1}{N} \sum_{j=1}^{N} \mathbb{E}_\xi[Q(x, \xi)] = c^\top x + \mathbb{E}_\xi[Q(x, \xi)] = g(x). \tag{1.44}$$

□

1.9.4 Obtention d'une borne inférieure

Une propriété fondamentale du SAA, établie par Mak, Morton & Wood (1999), démontre que la valeur optimale du problème approximé fournit, en espérance, une borne inférieure valide sur la solution optimale du problème stochastique original :

$$\mathbb{E}[\hat{v}_N] \leqslant v^*, \tag{1.45}$$

où l'espérance est prise sur l'ensemble des échantillons aléatoires $\{\xi^1, \ldots, \xi^N\}$. La convergence a posteriori de cet estimateur a été rigoureusement établie par Shapiro & Homem-de Mello (2000).

Démonstration. Pour toute réalisation de l'échantillon, on a :

$$\hat{g}_N(x) \geqslant \hat{v}_N, \quad \forall x \in X. \tag{1.46}$$

En prenant l'espérance :

$$\mathbb{E}[\hat{g}_N(x)] \geqslant \mathbb{E}[\hat{v}_N]. \tag{1.47}$$

Par non-biais, $\mathbb{E}[\hat{g}_N(x)] = g(x)$, donc :

$$g(x) \geqslant \mathbb{E}[\hat{v}_N], \quad \forall x \in X. \tag{1.48}$$

En particulier pour x^* :

$$v^* \geqslant \mathbb{E}[\hat{v}_N]. \tag{1.49}$$

□

1.9.5 Estimation numérique des bornes

En pratique, on approxime $\mathbb{E}[\hat{v}_N]$ via une procédure basée sur des réplications multiples du problème SAA pour plusieurs échantillons (voir Kleywegt *et al.* (2002); Linderoth, Shapiro & Wright (2006)). On estime donc $\mathbb{E}[\hat{v}_N]$ en résolvant le problème SAA M fois avec des échantillons indépendants.

Soient $\hat{v}_N^1, \ldots, \hat{v}_N^M$ les valeurs optimales obtenues. L'estimateur de la borne inférieure est :

$$\bar{v}_{N,M} := \frac{1}{M} \sum_{m=1}^{M} \hat{v}_N^m. \tag{1.50}$$

Dans notre étude, nous supposons que l'ensemble des scénarios possibles Ξ est fini et peut être énuméré. Sous l'hypothèse de recours relativement complet, la valeur exacte d'une solution

candidate $\hat{x}$ s'obtient par :

$$g(\hat{x}) = c^\top \hat{x} + \sum_{\xi \in \Xi} p_\xi Q(\hat{x}, \xi). \tag{1.51}$$

Chaque résolution du problème SAA fournit une solution de première étape $\hat{x}^m$. Par construction, cette solution est réalisable pour le problème stochastique original, puisqu'elle ne dépend pas d'un scénario particulier et que, sous l'hypothèse de recours relativement complet, une solution de seconde étape existe pour tout $\xi \in \Xi$ et pour tout $\hat{x} \in X$.

On peut ainsi évaluer exactement la qualité de chaque solution $\hat{x}^m$ en calculant sa valeur $g(\hat{x}^m)$. Chacune de ces valeurs constitue une borne supérieure du problème stochastique.

On retient finalement la meilleure de ces bornes :

$$\hat{v}^{\mathrm{UB}} = \min_{m=1,\ldots,M} g(\hat{x}^m). \tag{1.52}$$

1.9.6 Intervalles de confiance sur la borne inférieure

L'estimateur $\bar{v}_{N,M}$ constitue une approximation de l'espérance $\mathbb{E}[\hat{v}_N]$. Afin de quantifier l'incertitude statistique de cet estimateur, il est naturel de construire un intervalle de confiance.

Les valeurs $\hat{v}_N^1, \ldots, \hat{v}_N^M$ étant obtenues à partir d'échantillons indépendants, elles peuvent être considérées comme des réalisations indépendantes d'une variable aléatoire de moyenne $\mathbb{E}[\hat{v}_N]$ et de variance finie. On peut alors estimer la variance par :

$$s_{N,M}^2 = \frac{1}{M-1} \sum_{m=1}^{M} \left(\hat{v}_N^m - \bar{v}_{N,M}\right)^2, \tag{1.53}$$

où $s_{N,M}$ désigne l'écart-type défini par $s_{N,M} = \sqrt{s_{N,M}^2}$.

Dans la pratique, on utilise la loi de Student à $M - 1$ degrés de liberté pour construire un intervalle de confiance $1 - \beta$:

$$\bar{v}_{N,M} \pm t_{1-\beta/2,\, M-1} \cdot \frac{s_{N,M}}{\sqrt{M}}, \tag{1.54}$$

où $t_{1-\beta/2,\, M-1}$ désigne le quantile d'ordre $1 - \beta/2$ de la loi de Student. En particulier, une borne inférieure statistiquement valide du problème stochastique peut être obtenue en considérant la borne inférieure de cet intervalle :

$$\bar{v}_{N,M} - t_{1-\beta/2,\, M-1} \cdot \frac{s_{N,M}}{\sqrt{M}}. \tag{1.55}$$

Cette quantité fournit, avec un niveau de confiance $1 - \beta$, une estimation de $\mathbb{E}[\hat{v}_N]$, et donc, par la propriété fondamentale du SAA, une borne inférieure pour le problème stochastique original.

1.9.7 Procédure algorithmique du SAA

L'algorithme ci-dessous résume la procédure de Sample Average Approximation adaptée à un ensemble de scénarios énumérables.

Algorithme 1.4 Algorithme du SAA

Input : Nombre de réplications M, taille d'échantillon N

Output : Estimateur de la borne inférieure $\bar{v}_{N,M}$, borne supérieure $\hat{v}^{\mathrm{UB}}$

$LB_{sum} \leftarrow 0$

$\hat{v}^{\mathrm{UB}} \leftarrow +\infty$

for $m = 1, \ldots, M$ **do**

  Générer un échantillon i.i.d. $\{\xi_m^1, \ldots, \xi_m^N\}$

  Résoudre le problème SAA et obtenir $\hat{v}_N^m = \min_{x \in X} \hat{g}_N(x)$ et la solution $\hat{x}^m$

  $LB_{sum} \leftarrow LB_{sum} + \hat{v}_N^m$

  Calculer la valeur exacte de la solution $\hat{x}^m$ sur l'ensemble Ξ :

$$g(\hat{x}^m) = c^\top \hat{x}^m + \sum_{\xi \in \Xi} p_\xi \, Q(\hat{x}^m, \xi) \tag{1.56}$$

  if $g(\hat{x}^m) < \hat{v}^{\mathrm{UB}}$ **then**

    $\hat{v}^{\mathrm{UB}} \leftarrow g(\hat{x}^m)$

  end if

end for

$\bar{v}_{N,M} \leftarrow \dfrac{1}{M} LB_{sum}$

1.10 Limites des approches existantes

Malgré les avancées significatives présentées dans la littérature, la résolution des problèmes de conception de réseaux à deux niveaux se heurte à des limites computationnelles majeures lorsque la taille des instances augmente. Cette section discute des principales difficultés rencontrées par les méthodes existantes et motive le développement des approches proposées dans la suite de ce mémoire.

1.10.1 Limites des méthodes exactes classiques

Les méthodes exactes fondées sur les formulations entières mixtes présentées aux sections précédentes, résolues par séparation et évaluation enrichie de plans coupants, atteignent rapidement leurs limites sur les problèmes considérés.

D'abord, la dimension des formulations croît rapidement avec la taille des instances. Les modèles du TSP-GL et du MUFND présentés aux sections 1.7.5 et 1.8.5 font intervenir, pour chaque demande $(h, k) \in D$, un ensemble complet de variables de flot f_{ij}^{hk} indexées sur les arcs du graphe. Le nombre total de variables croît de l'ordre de $|A| \cdot |D|$, ($|A| \cdot |K|$ pour le MUFND). Lorsque le nombre de demandes ou la taille du graphe deviennent importants, la simple résolution de la relaxation linéaire devient elle-même coûteuse, et la borne inférieure qu'elle fournit se révèle souvent de mauvaise qualité pour permettre un élagage efficace.

Bien que la décomposition de Benders offre une formulation naturellement adaptée à cette classe de problèmes de conception de réseaux en exploitant la séparation entre décisions de conception et décisions de routage, sa convergence est souvent lente en pratique. Cette lenteur s'explique d'une part par la difficulté à générer des coupes suffisamment fortes pour la borne inférieure au cours de l'exploration de l'arbre de recherche, et d'autre part par le coût de la résolution répétée des sous-problèmes : à chaque nœud, il est nécessaire de résoudre un sous-problème pour chacune des demandes ; lorsque celles-ci sont nombreuses, cette étape, répétée des milliers de fois, devient très coûteuse.

Les stratégies de renforcement proposées dans la littérature par Magnanti & Wong (1981), apportent des améliorations significatives mais ne suffisent pas, à elles seules, à rendre la méthode pleinement compétitive sur les instances de très grande dimension. Le passage à l'échelle reste limité, en particulier lorsque le nombre de demandes dépasse plusieurs centaines.

1.10.2 Vers de nouvelles approches

Ces limitations mettent en évidence la nécessité d'explorer de nouvelles stratégies, et structurent les deux contributions de ce mémoire selon des approches complémentaires.

Dans un premier temps, afin de pallier le manque d'efficacité des coupes classiques et d'accélérer la convergence des méthodes exactes, nous proposons le développement de nouvelles bornes inférieures valides pour le TSP-GL exploitant la structure entière partielle des solutions issues de la relaxation linéaire (Chapitre 2). Cette première contribution s'inscrit dans la continuité directe des méthodes exactes : elle vise à renforcer la formulation sans en modifier la nature.

Dans un second temps, puisque l'explosion combinatoire provient principalement du nombre élevé de demandes et de leur interaction avec l'infrastructure, nous introduisons un changement de paradigme via une approche par échantillonnage de la demande (Chapitre 3). Plutôt que de chercher à mieux résoudre le problème complet, cette approche consiste à approximer le problème original à partir d'un sous-ensemble aléatoire de demandes, convenablement remis à l'échelle. Nous montrerons que ce problème déterministe peut être réinterpréter de manière stochastique, ce qui, en conséquence, permet de mobiliser les outils développés en programmation stochastique comme le SAA présenté à la section 1.9 et d'obtenir des garanties statistiques formelles sur la valeur optimale du problème original. Cette seconde contribution permet de traiter des instances dont la taille rend les méthodes exactes classiques impraticables.

CHAPITRE 2

DÉVELOPPEMENT DE NOUVELLES BORNES INFÉRIEURES VALIDES POUR LE TSP-GL

2.1 Principes généraux

L'objectif de ce chapitre est d'exploiter la structure du TSP-GL afin de construire de nouvelles coupes visant à accélérer la convergence vers la solution optimale dans un algorithme de séparation et d'évaluation.

Au cours d'un tel algorithme, de nombreuses relaxations linéaires sont résolues, et produisent généralement des solutions fractionnaires. En particulier, les variables de conception $x_e, \forall e \in E$, prennent souvent des valeurs non entières, ce qui induit également des valeurs fractionnaires pour les variables de flot f_{ij}^{hk} via les contraintes de liaison $f_{ij}^{hk} \leq x_e, \forall e = [i,j] \in E, \forall (h,k) \in D$. Nous rappelons ici que le TSP-GL ne contient de contraintes de capacité, un flot illimité peut circuler sur un arc donné.

Cependant, au fur et à mesure de l'ajout de contraintes de branchement et de coupes d'optimalité ou de faisabilité, certaines variables x_e prennent des valeurs entières et définissent ainsi des sous-structures entières au sein des solutions des relaxations linéaires. Une question naturelle est alors la suivante : peut-on exploiter ces sous-structures entières partielles afin de construire des bornes inférieures plus fortes que celles fournies directement par la relaxation linéaire, et les injecter dans le modèle sous forme de nouvelles coupes d'optimalité ?

Des approches répondant à cette question ont été proposées dans la littérature, notamment dans des problèmes de routage de véhicules avec demandes stochastiques. Par exemple, Laporte & Louveaux (1993); Hjorring & Holt (1999) introduisent de nouvelles coupes d'optimalité, activées lorsque certaines structures sont présentes dans la solution courante. De manière analogue, Jabali, Rei, Gendreau & Laporte (2014) exploitent les propriétés structurelles de leur problème pour dériver de nouvelles inégalités valides renforçant les formulations.

Dans cette optique, nous proposons dans ce chapitre deux approches complémentaires pour le TSP-GL. La première exploite la présence simultanée d'arêtes de conception entières et de flots fractionnaires sur ces mêmes arêtes : on génère une disjonction sur la valeur d'un tel flot et l'on en tire une borne inférieure valide. La seconde exploite la structure topologique des composantes fractionnaires du flot dans le graphe : on identifie des chaînes d'arêtes entières et l'on reconstruit, à partir des zones fractionnaires environnantes, des cycles candidats permettant d'obtenir une borne renforcée sur le coût de routage. Dans les deux cas, les bornes obtenues sont injectées dans le modèle sous forme de coupes d'optimalité conditionnelles.

2.2 Première approche

Nous proposons dans cette section une première méthode de renforcement de la borne inférieure pour le TSP-GL, fondée sur l'exploitation de certaines structures présentes dans les solutions de la relaxation linéaire.

L'idée consiste à tirer parti de situations où, malgré des décisions de conception fixées à des valeurs entières, les variables de flot se dispersent de façon fractionnaire. Ce phénomène conduit à une sous-estimation systématique du coût réel de routage dans la relaxation. Pour corriger cet effet, nous considérons les deux cas possibles pour le passage d'un flot par un arc particulier : soit le flot l'emprunte, soit il ne l'emprunte pas et nous évaluons chacun de ces cas par un problème restreint. Le minimum des deux valeurs ainsi obtenues fournit une borne inférieure valide pour toute solution entière respectant une certaine structure préfixée.

Dans la suite, nous introduisons les notations nécessaires, présentons la méthode et la preuve de validité des bornes obtenues, discutons la stratégie de sélection du flot fractionnaire à exploiter, dérivons la coupe d'optimalité conditionnelle correspondante et résumons la procédure dans un algorithme. Nous évaluons enfin son impact à travers une étude expérimentale sur des instances du TSP-GL.

2.2.1 Notations

Le polyèdre P_{LR} correspond à la relaxation linéaire continue de la formulation du TSP-GL, obtenue en relâchant les contraintes d'intégrité (1.24g) en $0 \leq x_e \leq 1$. Il est défini par les contraintes suivantes :

$$x(\delta(i)) = 2 \qquad \forall i \in N \tag{2.1a}$$

$$f^{hk}(\delta^+(i)) - f^{hk}(\delta^-(i)) = \begin{cases} 1 & \text{si } i = h, \\ -1 & \text{si } i = k, \\ 0 & \text{sinon,} \end{cases} \qquad \forall (h,k) \in D,\ \forall i \in N \tag{2.1b}$$

$$x_e - f_{ij}^{hk} \geq 0 \qquad \forall e = [i,j] \in E,\ \forall (h,k) \in D \tag{2.1c}$$

$$x_e - f_{ji}^{hk} \geq 0 \qquad \forall e = [i,j] \in E,\ \forall (h,k) \in D \tag{2.1d}$$

$$f_{ij}^{hk} \geq 0 \qquad \forall (i,j) \in A,\ \forall (h,k) \in D \tag{2.1e}$$

$$0 \leq x_e \leq 1 \qquad \forall e \in E. \tag{2.1f}$$

Nous utilisons les notations suivantes dans la suite de cette section :

- $(\bar{x}, \bar{f})$: solution optimale de la relaxation linéaire du TSP-GL, où $\bar{x}_e$ pour tout $e \in E$ désigne les valeurs des variables de conception et $\bar{f}_{ij}^{hk}$ pour tout $(i,j) \in A$, $(h,k) \in D$ les valeurs des variables de flot.
- $R^1(\bar{x}) = \{e \in E \mid \bar{x}_e = 1\}$: ensemble des arêtes fixées à 1.
- $R^0(\bar{x}) = \{e \in E \mid \bar{x}_e = 0\}$: ensemble des arêtes fixées à 0.
- $F = \left\{(i,j,h,k) \in A \times D \;\middle|\; 0 < \bar{f}_{ij}^{hk} < 1 \text{ et } \bar{x}_{ij} = 1\right\}$: ensemble des flots fractionnaires circulant sur des arcs actifs entiers.

2.2.2 Méthode

Après résolution de la relaxation linéaire du TSP-GL, on obtient une solution $(\bar{x}, \bar{f})$. On sélectionne un quadruplet $(l, s, \hat{h}, \hat{k}) \in F$, correspondant à un flot strictement fractionnaire

circulant sur un arc actif entier. On résout alors les deux relaxations restreintes suivantes du TSP-GL :

$$L^0(\bar{x}) = \min \quad (1-\alpha)\sum_{e\in E} c_e x_e + \alpha \sum_{(i,j)\in A}\sum_{(h,k)\in D} q_{ij}^{hk} f_{ij}^{hk} \tag{2.2a}$$

$$\text{s.c.} \quad x_e = 1, \quad \forall e \in R^1(\bar{x}) \tag{2.2b}$$

$$x_e = 0, \quad \forall e \in R^0(\bar{x}) \tag{2.2c}$$

$$f_{ls}^{\hat{h}\hat{k}} = 0 \tag{2.2d}$$

$$(x, f) \in P_{LR} \tag{2.2e}$$

$$L^1(\bar{x}) = \min \quad (1-\alpha)\sum_{e\in E} c_e x_e + \alpha \sum_{(i,j)\in A}\sum_{(h,k)\in D} q_{ij}^{hk} f_{ij}^{hk} \tag{2.3a}$$

$$\text{s.c.} \quad x_e = 1, \quad \forall e \in R^1(\bar{x}) \tag{2.3b}$$

$$x_e = 0, \quad \forall e \in R^0(\bar{x}) \tag{2.3c}$$

$$f_{ls}^{\hat{h}\hat{k}} = 1 \tag{2.3d}$$

$$(x, f) \in P_{LR} \tag{2.3e}$$

On note P_0 et P_1 les polyèdres associés à ces deux problèmes. Nous montrons ci-dessous que $\min\{L^0(\bar{x}), L^1(\bar{x})\}$ constitue une borne inférieure valide pour toute solution entière respectant les fixations $R^1(\bar{x})$ et $R^0(\bar{x})$.

2.2.3 Preuve de validité de la borne inférieure

Notons $C(\bar{x})$ l'ensemble des solutions entières (x^I, f^I) réalisables pour le TSP-GL et respectant les fixations $R^1(\bar{x})$ et $R^0(\bar{x})$. Cet ensemble correspond aux solutions du sous-arbre de branchement

enraciné au nœud courant. Pour toute solution $(x^I, f^I) \in C(\bar{x})$, on note $z^I(x^I, f^I)$ la valeur de la fonction objectif associée.

En l'absence de contraintes de capacité sur les arcs, le sous-problème de routage de chaque demande (h, k) est un problème de plus court chemin classique. La composante f^I est elle-même entière, c'est-à-dire $f_{ij}^{I,hk} \in \{0, 1\}$ pour tout $(i, j) \in A$ et tout $(h, k) \in D$.

Deux cas se présentent.

Cas 1 : $f_{ls}^{I,\hat{h}\hat{k}} = 0$**.**

La solution (x^I, f^I) appartient au polyèdre P_0. Comme $L^0(\bar{x})$ est la valeur optimale d'une relaxation de ce polyèdre, on a :

$$z^I(x^I, f^I) \geq L^0(\bar{x}). \tag{2.4}$$

Cas 2 : $f_{ls}^{I,\hat{h}\hat{k}} = 1$**.**

De manière analogue, $(x^I, f^I) \in P_1$ et :

$$z^I(x^I, f^I) \geq L^1(\bar{x}). \tag{2.5}$$

Conclusion.

Toute solution de $C(\bar{x})$ appartenant nécessairement à $P_0 \cup P_1$, on en déduit :

$$z^I(x^I, f^I) \geq \min\{L^0(\bar{x}), L^1(\bar{x})\} \qquad \forall (x^I, f^I) \in C(\bar{x}). \tag{2.6}$$

2.2.4 Choix du quadruplet $(l, s, \hat{h}, \hat{k})$

Le nombre de quadruplets candidats dans F peut être élevé. Afin de limiter le nombre de sous-problèmes résolus tout en maximisant l'impact de la borne, nous proposons une heuristique de sélection visant à identifier la variable de flot concentrant simultanément la plus forte fractionnalité et le plus grand impact potentiel sur la fonction objectif.

Pour chaque quadruplet $(i, j, h, k) \in F$, nous définissons sa fractionnalité Δ_{ij}^{hk} comme la proximité de $\bar{f}_{ij}^{hk}$ à la valeur 0.5 :

$$\Delta_{ij}^{hk} = \min\left(\bar{f}_{ij}^{hk}, 1 - \bar{f}_{ij}^{hk}\right). \tag{2.7}$$

Cette quantité est ensuite pondérée par le coût de latence associé, q_{ij}^{hk}, afin d'estimer l'impact potentiel sur la fonction objectif. Le candidat retenu est celui maximisant le score combiné :

$$(l, s, \hat{h}, \hat{k}) \in \underset{(i,j,h,k)\in F}{\arg\max} \left\{q_{ij}^{hk} \cdot \Delta_{ij}^{hk}\right\}. \tag{2.8}$$

2.2.5 Coupe d'optimalité conditionnelle

La borne $LB(\bar{x}) = \min\{L^0(\bar{x}), L^1(\bar{x})\}$ n'est valide que pour les solutions partageant la structure $R^1(\bar{x})$ et $R^0(\bar{x})$. Pour l'injecter sous forme de coupe valide globalement dans l'arbre de recherche, il est nécessaire de la conditionner aux valeurs des variables de fixation.

Notons z la variable représentant la valeur de la fonction objectif totale du modèle, et L une borne inférieure globale connue du problème, par exemple celle issue de la résolution de la relaxation linéaire au nœud racine. Par construction, on a $LB(\bar{x}) \geq L$, puisque $L^0(\bar{x})$ et $L^1(\bar{x})$ sont obtenus en ajoutant des contraintes à un problème dont la relaxation racine a pour valeur L.

On définit le terme :

$$\theta(x) = \sum_{e \in R^1(\bar{x})} x_e - \sum_{e \in R^0(\bar{x})} x_e - |R^1(\bar{x})| + 1. \tag{2.9}$$

La coupe d'optimalité conditionnelle s'écrit alors (voir Laporte & Louveaux (1993)) :

$$(LB(\bar{x}) - L)\,\theta(x) + L \leq z. \tag{2.10}$$

Interprétation.

- Si la solution entière explorée par le solveur respecte la structure ($x_e = 1$ pour toute $e \in R^1(\bar{x})$ et $x_e = 0$ pour toute $e \in R^0(\bar{x})$), alors $\theta(x) = 1$ et la coupe impose $z \geq LB(\bar{x})$.
- Si au moins une variable ne respecte pas sa fixation, alors $\theta(x) \leq 0$. Comme $LB(\bar{x}) - L \geq 0$, le membre de gauche est inférieur ou égal à L, et la coupe se réduit à $z \geq L$, qui est trivialement vérifiée.

L'inégalité est donc valide pour toute solution entière du TSP-GL.

2.2.6 Algorithme

L'algorithme 2.1 résume la procédure complète, depuis la sélection du quadruplet jusqu'au calcul de la borne $LB(\bar{x})$.

Algorithme 2.1 Obtention d'une borne inférieure par disjonction sur un flot fractionnaire

Input : Solution fractionnaire $(\bar{x}, \bar{f})$ de la relaxation linéaire

Output : Borne inférieure $LB(\bar{x})$

Construire les ensembles $R^1(\bar{x})$, $R^0(\bar{x})$ et F

if $F = \emptyset$ **then**

  return L `// aucun flot fractionnaire à exploiter`

end if

Sélectionner le quadruplet $(l, s, \hat{h}, \hat{k}) \in F$ maximisant le score $q_{ij}^{hk} \cdot \Delta_{ij}^{hk}$

Résoudre $L^0(\bar{x})$ avec la contrainte additionnelle $f_{ls}^{\hat{h}\hat{k}} = 0$

Résoudre $L^1(\bar{x})$ avec la contrainte additionnelle $f_{ls}^{\hat{h}\hat{k}} = 1$

$LB(\bar{x}) \leftarrow \min\{L^0(\bar{x}), L^1(\bar{x})\}$

return $LB(\bar{x})$

Remarque.

La procédure peut être itérée sur plusieurs quadruplets en conservant la meilleure borne obtenue : si $LB_1(\bar{x}), \ldots, LB_p(\bar{x})$ sont des bornes valides associées à des quadruplets distincts, alors $\max_{i=1,\ldots,p} LB_i(\bar{x})$ est également valide.

2.3 Seconde approche

Nous proposons une seconde approche de renforcement de la borne inférieure, complémentaire à la précédente. Elle s'appuie directement sur la structure des solutions fractionnaires issues de la relaxation linéaire et, plus précisément, sur les zones du graphe dans lesquelles le flot d'une demande se disperse sur plusieurs chemins. L'idée consiste à exploiter ces structures locales pour reconstruire des configurations plus proches d'une solution entière et obtenir ainsi de meilleures bornes.

Dans la suite, nous décrivons le principe de la méthode, introduisons les outils nécessaires à sa mise en œuvre, puis analysons son impact à travers une étude expérimentale sur des instances du TSP-GL.

2.3.1 Principe général

Soit $(\bar{x}, \bar{f})$ une solution de la relaxation linéaire du TSP-GL. Pour une demande $(h, k) \in D$, on note $Q^{hk}(\bar{x})$ le coût de routage associé dans cette solution, c'est-à-dire le coût d'un plus court chemin entre h et k dans le graphe pondéré par $\bar{x}$.

Les variables de flot f_{ij}^{hk} ne sont pas explicitement contraintes à être binaires dans la formulation du TSP-GL. Cependant, lorsque les variables x sont entières, le routage de chaque demande correspond à un problème de plus court chemin sans contrainte de capacité, et le flot optimal est naturellement entier ($f_{ij}^{hk} \in \{0, 1\}$).

En revanche, dans la relaxation linéaire, les variables $\bar{x}$ peuvent être fractionnaires. Les contraintes $f_{ij}^{hk} \leq \bar{x}_e$ induisent alors des capacités fractionnaires sur les arcs, et le flot $\bar{f}^{hk}$ se répartit sur plusieurs chemins. Cette dispersion conduit à une sous-estimation systématique du coût réel de routage par rapport à toute solution entière.

L'objectif est donc d'exploiter cette structure fractionnaire afin de construire des bornes inférieures renforcées sur le coût de routage, en imposant localement des structures plus proches de solutions entières.

2.3.2 Coupe d'optimalité conditionnelle

On définit une *chaîne entière* $S_p \subseteq E$ comme un ensemble d'arêtes formant un chemin dans le graphe $G = (N, E)$ et tel que toutes ses arêtes prennent la valeur 1 dans la relaxation linéaire :

$$\bar{x}_e = 1 \quad \forall e \in S_p. \tag{2.11}$$

À partir d'une telle chaîne S_p issue de la solution $(\bar{x}, \bar{f})$, nous construisons (voir Laporte & Louveaux (1993); Hjorring & Holt (1999)) la coupe d'optimalité conditionnelle :

$$(\eta_p - L)\left(\sum_{e \in S_p} x_e - (|S_p| - 1)\right) + L \leq \eta, \tag{2.12}$$

où L est une borne inférieure globale issue de la relaxation linéaire au nœud racine, η représente la valeur du sous-problème de routage dans la reformulation de Benders, et η_p est une borne renforcée associée à la structure S_p, dont la construction fait l'objet des sous-sections suivantes.

Interprétation.

- Si toutes les arêtes de S_p sont sélectionnées par le solveur ($\sum_{e \in S_p} x_e = |S_p|$), la coupe impose $\eta \geq \eta_p$ et renforce donc la borne inférieure.
- Dans le cas contraire, la coupe se réduit à $\eta \geq L$ et n'affecte pas la solution.

Cette inégalité permet ainsi d'introduire un renforcement local sans détériorer la validité globale du modèle, à condition que η_p soit construite comme une borne inférieure valide sur le coût de routage de toute solution entière contenant intégralement la chaîne S_p. Les sous-sections qui suivent détaillent cette construction.

2.3.3 Définition d'un blob

Pour une demande $(h, k) \in D$, on définit le sous-graphe des arcs fractionnaires $G^{hk}_{frac} = (N^{hk}_{frac}, A^{hk}_{frac})$, où :

$$A^{hk}_{frac} = \left\{(i, j) \in A \mid 0 < \bar{f}^{hk}_{ij} < 1\right\}, \tag{2.13}$$

et N^{hk}_{frac} est l'ensemble des sommets incidents aux arcs de A^{hk}_{frac}.

On appelle *blob* toute composante connexe $B_q = (N_q, A_q)$ de G^{hk}_{frac}. L'ensemble des blobs associés à la demande (h, k) est noté $\mathcal{B}^{hk} = \{B_1, \ldots, B_m\}$. Intuitivement, un blob correspond à

une zone du réseau dans laquelle le flot associé à la demande (h, k) se disperse sur plusieurs chemins alternatifs.

2.3.4 Construction de la borne η_p : cas favorable

Nous nous plaçons dans le cas où il existe une chaîne entière S_p reliant l'origine h à la destination k. Dans une solution entière du TSP-GL, la conception forme un cycle hamiltonien. Si la chaîne S_p est entièrement sélectionnée, le cycle se décompose alors en deux parties : la chaîne S_p elle-même, et un second chemin reliant k à h et visitant l'ensemble des sommets restants. Or, les sommets restants sont précisément ceux contenus dans les blobs.

La reconstruction du cycle s'opère ainsi en trois étapes.

1. **Identification des points d'attache.** Pour chaque blob B_q, on identifie ses sommets d'intersection avec la chaîne S_p. Ces sommets servent de points d'entrée et de sortie permettant de relier le blob au reste du cycle.
2. **Construction d'un chemin couvrant.** Dans chaque blob B_q, on construit un chemin $\mathcal{P}_q$ reliant son point d'entrée à son point de sortie tout en visitant l'intégralité des sommets de B_q. Parmi tous les chemins couvrants possibles, on retient celui de coût total minimal. Ce chemin est obtenu par résolution d'un sous-problème combinatoire de petite taille, la cardinalité de chaque blob étant typiquement faible en pratique.
3. **Évaluation du routage.** On définit le graphe restreint $\hat{G}^{hk}$ obtenu en juxtaposant la chaîne S_p et les chemins couvrants $\{\mathcal{P}_q\}$: cet ensemble constitue un cycle hamiltonien candidat. Le coût de routage $\hat{Q}^{hk}$ est défini comme le coût du plus court chemin entre h et k sur ce cycle (parmi les deux directions possibles).

Validité de la borne.

Considérons une solution entière respectant la structure S_p, c'est-à-dire telle que $x_e = 1$ pour toute $e \in S_p$. Le cycle hamiltonien associé contient intégralement S_p et complète celle-ci par

un chemin reliant k à h et visitant les sommets restants. Pour chaque blob B_q, ce chemin de complétion traverse les sommets de B_q via une certaine séquence d'arêtes reliant son point d'entrée à son point de sortie : il s'agit donc nécessairement d'un chemin couvrant de B_q. Par définition, le coût de ce chemin est supérieur ou égal à celui du chemin couvrant minimal $\mathcal{P}_q$ retenu dans notre construction. Le coût de routage entre h et k sur le cycle entier est donc supérieur ou égal à $\hat{Q}^{hk}$.

En sommant sur l'ensemble des demandes, on obtient :

$$\eta_p = \sum_{(h,k)\in D} \hat{Q}^{hk}, \tag{2.14}$$

qui constitue par construction une borne inférieure sur le coût total de routage de toute solution entière contenant intégralement la chaîne S_p. La coupe d'optimalité conditionnelle introduite plus haut est donc valide.

Comparaison avec la relaxation linéaire.

La borne $\hat{Q}^{hk}$ est typiquement plus restrictive que la borne fractionnaire $Q^{hk}(\bar{x})$ fournie par la relaxation : en remplaçant la dispersion artificielle du flot par un chemin unique au sein de chaque blob, on supprime l'avantage de mutualisation qui permet à la relaxation linéaire de réduire le coût de routage. Cette propriété, est confirmée par nos expériences numériques (Sous-section 2.4).

2.3.5 Cas pathologique : absence de chaîne entière

Lorsqu'aucune chaîne entière ne relie h à k, le flot est entièrement porté par des arcs fractionnaires, et aucune structure de référence ne peut être extraite. Dans ce cas, on n'apporte aucun renforcement à la borne et l'on conserve :

$$\hat{Q}^{hk} = Q^{hk}(\bar{x}). \tag{2.15}$$

2.3.6 Algorithme

L'algorithme 2.2 résume la procédure d'obtention de la borne renforcée η_p.

Algorithme 2.2 Obtention d'une borne inférieure par exploitation des composantes fractionnaires

Input : Solution fractionnaire $(\bar{x}, \bar{f})$

Output : Borne inférieure renforcée η_p

$\eta_p \leftarrow 0$
for *chaque demande* $(h, k) \in D$ **do**
  Identifier (si elle existe) une chaîne S_p d'arêtes entières reliant h à k, c'est-à-dire un chemin tel que $\bar{x}_e = 1$ pour toute arête $e \in S_p$
  Définir le sous-graphe fractionnaire G^{hk}_{frac} induit par les arcs vérifiant $0 < \bar{f}^{hk}_{ij} < 1$
  Extraire l'ensemble des blobs $\mathcal{B}^{hk}$ comme les composantes connexes de G^{hk}_{frac}
  if *une telle chaîne* S_p *existe* **then**
    Pour chaque blob $B_q \in \mathcal{B}^{hk}$, calculer un chemin couvrant $\mathcal{P}_q$ de coût minimal entre ses points d'attache
    Construire le graphe restreint $\hat{G}^{hk}$ en juxtaposant S_p et les chemins $\{\mathcal{P}_q\}$
    $\hat{Q}^{hk} \leftarrow$ coût du plus court chemin entre h et k sur $\hat{G}^{hk}$
  else
    $\hat{Q}^{hk} \leftarrow Q^{hk}(\bar{x})$
  end if
  $\eta_p \leftarrow \eta_p + \hat{Q}^{hk}$
end for
return η_p

2.4 Résultats expérimentaux

Nous évaluons l'apport des méthodes proposées sur les instances du problème TSP-GL de la famille TSPGL1, issues de la littérature et générées par Errico *et al.* (2017). Ces instances

sont construites sur des réseaux de tailles croissantes et déclinées selon différents scénarios de demande ainsi que plusieurs valeurs du paramètre α, qui modélise le compromis entre le coût de conception du réseau et la latence de routage.

Nous n'exploitons pas l'ensemble complet de la base de tests, mais nous concentrons l'analyse sur un sous-ensemble représentatif d'instances couvrant différentes tailles de réseaux et scénarios de demande. En particulier, nous restreignons l'étude à quelques valeurs de α afin d'évaluer l'impact de ce paramètre sur les performances des méthodes proposées.

Deux scénarios de demande sont considérés. Dans les deux cas, une partie des nœuds est désignée comme pôles, et environ 50% du volume total de la demande circule entre ces pôles, les 50% restants étant répartis entre les autres couples origine-destination.

- **C : Complète et polarisée.** Tous les couples origine-destination portent une demande non nulle, dont les volumes sont générés uniformément sur un intervalle donné.
- **S : Dispersée et polarisée.** Chaque nœud n'a qu'un nombre restreint de destinations possibles, choisies aléatoirement (typiquement quatre par nœud).

Avant de présenter les résultats, il convient de clarifier la nature du protocole expérimental adopté. L'évaluation des deux méthodes proposées poursuit des objectifs distincts, ce qui se traduit par des choix expérimentaux et des formats de présentation différents.

La première approche, fondée sur l'exploitation d'une seule arête fractionnaire, génère une unique coupe d'optimalité par nœud. Son apport étant essentiellement local, nous évaluons son effet directement au nœud racine en comparant la borne inférieure obtenue avec et sans la coupe proposée. Nous testons cette approche sur des instances de 35 nœuds afin d'évaluer son comportement à une taille intéressante ; à cette taille, la résolution complète à l'optimalité n'est pas atteignable en temps raisonnable par décomposition de Benders, c'est pourquoi nous nous restreignons ici à l'analyse au nœud racine.

La seconde approche, fondée sur l'exploitation des composantes fractionnaires génère un ensemble de coupes par nœud et peut être activée tout au long de l'arbre de branchement. Nous souhaitons donc évaluer son impact non seulement au nœud racine mais aussi à l'optimalité globale, afin de mesurer son effet sur la convergence complète de l'algorithme (nombre de nœuds explorés, temps total, nombre de coupes générées). Pour permettre cette résolution à l'optimalité dans un temps raisonnable, nous nous restreignons à des instances de 20 nœuds. Cette restriction sur la taille des instances illustre concrètement les limites de passage à l'échelle des méthodes exactes discutées à la section 1.10, et motive le changement de paradigme proposé au Chapitre 3.

2.4.1 Environnement logiciel et outils de résolution

L'ensemble des méthodes proposées dans ce chapitre a été implémenté en Python (version 3.9.25), un langage de haut niveau largement utilisé en recherche pour sa flexibilité et la richesse de son écosystème scientifique. En particulier, la bibliothèque `networkx` a été utilisée pour la manipulation des structures de graphes.

La résolution des problèmes d'optimisation a été effectuée à l'aide du solveur IBM ILOG CPLEX (version 22.1), qui constitue une référence dans le domaine de l'optimisation et de la recherche opérationnelle. Son intégration avec Python permet le développement et l'expérimentation de nouvelles approches algorithmiques.

Une des fonctionnalités particulièrement intéressantes de CPLEX réside dans la possibilité d'exploiter des mécanismes avancés tels que les *callbacks*, permettant une interaction fine avec l'algorithme de séparation et d'évaluation. Ces mécanismes rendent possible l'ajout dynamique de contraintes, notamment des plans coupants, au cours de la résolution, offrant ainsi un haut degré de personnalisation et un meilleur contrôle des performances de l'algorithme.

Les expériences numériques ont été réalisées sur les machines du CIRRELT, équipées d'un processeur Intel Core i7-7800X et de 124 Go de mémoire vive. Sauf indication contraire, les paramètres par défaut de CPLEX ont été utilisés.

2.4.2 Résultats sur la première approche

Nous évaluons l'apport de la méthode sur des instances du TSP-GL à 35 nœuds, pour trois valeurs de $\alpha \in \{0.4, 0.5, 0.6\}$ et trois instances par valeur. Le Tableau 2.1 rapporte, pour chaque instance, la meilleure borne supérieure connue (BS), la borne inférieure obtenue au nœud racine sans coupe (BI sans CO) et avec la coupe d'optimalité proposée (BI avec CO), ainsi que les écarts relatifs correspondants.

Tableau 2.1 Comparaison des bornes inférieures au nœud racine avec et sans nouvelles coupes d'optimalités

Instance	BS	BI (sans CO)	BI (avec CO)	Ecart sans CO (%)	Ecart avec CO (%)
35 nœuds ($\alpha = 0.4$)	612.7682	597.3824	597.9664	2.52	**2.41**
35 nœuds ($\alpha = 0.4$)	672.6644	655.9422	655.9740	2.49	**2.49**
35 nœuds ($\alpha = 0.4$)	624.0973	594.8639	594.8720	4.68	**4.68**
35 nœuds ($\alpha = 0.5$)	531.5921	510.4103	511.1973	3.98	**3.84**
35 nœuds ($\alpha = 0.5$)	583.0884	554.6069	557.2900	4.88	**4.42**
35 nœuds ($\alpha = 0.5$)	530.3159	513.9650	515.0178	3.08	**2.89**
35 nœuds ($\alpha = 0.6$)	450.4160	423.8708	423.9777	5.90	**5.87**
35 nœuds ($\alpha = 0.6$)	491.3124	460.0877	460.9872	6.36	**6.17**
35 nœuds ($\alpha = 0.6$)	449.5153	429.0687	429.9889	4.55	**4.34**

Les résultats confirment la validité de la méthode : sur l'ensemble des instances testées, la borne inférieure obtenue avec la coupe est toujours au moins aussi bonne que celle obtenue sans, et strictement meilleure dans la majorité des cas. Aucune instance ne présente de détérioration, ce qui est cohérent avec la propriété de validité démontrée plus haut.

Les gains restent modérés en valeur absolue, typiquement de l'ordre de 0.1 à 0.5 point de pourcentage. Ils traduisent néanmoins une amélioration cohérente sur l'ensemble du jeu d'essai, indiquant que la coupe exploite effectivement l'information contenue dans les solutions fractionnaires de la relaxation linéaire, sans introduire d'effet indésirable.

Ces résultats confirment que la dispersion fractionnaire du flot, lorsque les variables de conception sont entières, peut être exploitée pour renforcer la borne inférieure.

2.4.3 Résulats sur la seconde approche

Comme annoncé, la seconde approche fait l'objet d'une évaluation à la fois au nœud racine et à l'optimalité globale. Pour rendre cette résolution à l'optimalité atteignable dans un temps raisonnable, nous nous restreignons à des instances de 20 nœuds. Nous évaluons l'impact de l'approche par blobs sur deux configurations, l'une avec une demande dispersée et l'autre avec une demande complète et polarisée. Les expériences couvrent plusieurs valeurs de α, et chaque ligne représente une moyenne sur trois instances. La notation « NC » désigne l'ajout des nouvelles coupes proposées dans cette section. Aucune borne supérieure n'ayant été trouvée au cours de la résolution du nœud racine, l'écart en pourcentage à ce stade est calculé relativement à la solution optimale, qui est connue pour les instances considérées.

Les résultats sont rapportés sur quatre tableaux, à raison de deux par configuration. Pour chaque configuration, un premier tableau présente les résultats obtenus au nœud racine, et un second les résultats obtenus à l'optimalité globale.

Les Tableaux 2.2 et 2.3 se lisent comme suit. La colonne α identifie la valeur du paramètre de pondération entre coût de conception et latence. Les deux colonnes *sans NC* rapportent, en l'absence des nouvelles coupes, l'*Écart* d'optimalité au nœud racine (calculé par rapport à la solution optimale connue) et le *Temps* de résolution du nœud racine (en secondes). Les trois colonnes *avec NC* reportent, après ajout des coupes proposées dans cette section, l'*Écart* d'optimalité résultant, le nombre moyen de coupes générées (*#Coupes*) et le *Temps* de résolution associé.

Les Tableaux 2.4 et 2.5 se lisent comme suit. La colonne α identifie la valeur du paramètre de pondération. Les deux colonnes *sans NC* rapportent, à la résolution complète à l'optimalité, le nombre de *Nœuds* explorés dans l'arbre de séparation et évaluation et le *Temps* total de résolution (en secondes). Les trois colonnes *avec NC* reportent, après activation des coupes proposées tout au long de l'arbre de branchement, le nombre de *Nœuds* explorés, le nombre total de coupes générées (*#Coupes*) et le *Temps* total de résolution.

Tableau 2.2 TSP-GL, 20 nœuds, demande dispersée — résultats au nœud racine (moyenne sur 3 instances)

α	Sans NC		Avec NC		
	Écart	Temps (s)	Écart	#Coupes	Temps (s)
0.35	2.20%	1.50	2.08%	6.33	1.58
0.40	2.56%	1.83	2.42%	7.67	1.52
0.45	2.80%	2.94	2.76%	8.33	3.17
0.50	3.19%	3.27	2.88%	6.33	3.39
0.55	5.90%	3.21	5.51%	6.67	3.37
0.60	8.14%	2.92	7.73%	7.00	2.96

Tableau 2.3 TSP-GL, 20 nœuds, demande complète — résultats au nœud racine (moyenne sur 3 instances)

α	Sans NC		Avec NC		
	Écart	Temps (s)	Écart	#Coupes	Temps (s)
0.35	2.79%	2.64	2.12%	6.00	2.76
0.40	3.32%	2.85	2.61%	6.00	3.04
0.45	4.37%	3.09	3.74%	5.67	3.12
0.50	6.84%	3.06	6.03%	5.33	2.69
0.55	9.51%	2.54	8.71%	5.33	2.66
0.60	9.42%	2.65	8.11%	6.00	2.76

Les résultats mettent en évidence l'impact de la méthode proposée sur la qualité de la borne inférieure, particulièrement au nœud racine.

Sur l'ensemble des configurations testées, l'ajout des coupes issues de l'exploitation des composantes fractionnaires réduit systématiquement les écarts en pourcentage au nœud racine (Tableaux 2.2 et 2.3). L'amélioration est modérée mais cohérente. Dans le cas de la demande dispersée, l'écart passe ainsi de 2.20% à 2.08% pour $\alpha = 0.35$, et de 8.14% à 7.73% pour $\alpha = 0.60$. Une tendance similaire est observée pour la demande complète, avec des gains plus marqués lorsque α augmente : l'écart passe de 9.42% à 8.11% pour $\alpha = 0.60$.

Tableau 2.4 TSP-GL, 20 nœuds, demande dispersée — résultats à l'optimalité (moyenne sur 3 instances)

α	Sans NC		Avec NC		
	Nœuds	Temps (s)	Nœuds	#Coupes	Temps (s)
0.35	179.33	34.64	179.67	45.67	35.83
0.40	213.33	34.94	213.67	25.67	38.61
0.45	267.00	39.97	263.00	52.67	38.75
0.50	616.00	67.09	618.33	59.00	65.31
0.55	478.67	62.34	479.00	57.33	66.68
0.60	539.00	191.54	479.67	91.33	158.84

Tableau 2.5 TSP-GL, 20 nœuds, demande complète — résultats à l'optimalité (moyenne sur 3 instances)

α	Sans NC		Avec NC		
	Nœuds	Temps (s)	Nœuds	#Coupes	Temps (s)
0.35	50.00	52.81	50.67	38.33	51.72
0.40	79.00	75.03	80.00	48.33	82.34
0.45	153.67	106.46	151.00	56.67	114.54
0.50	235.33	176.27	234.33	74.33	177.48
0.55	419.67	347.15	387.67	85.33	340.83
0.60	496.67	409.33	479.33	112.33	406.39

Cette amélioration confirme que la relaxation linéaire du TSP-GL sous-estime effectivement le coût de routage en raison de la dispersion du flot sur des structures fractionnaires, et que la méthode proposée parvient à corriger partiellement ce phénomène en imposant localement une structure plus proche d'un cycle.

L'effet des coupes est d'autant plus significatif que α est élevé. Cela s'explique par le fait que la composante de latence devient dominante dans la fonction objectif, ce qui accentue l'impact des solutions fractionnaires. Dans ce contexte, les blobs jouent un rôle plus important et leur exploitation génère des inégalités plus pertinentes.

Malgré l'amélioration de la borne au nœud racine, l'effet sur la taille de l'arbre de recherche reste limité pour les petites valeurs de α (Tableaux 2.4 et 2.5) : le nombre de nœuds explorés reste globalement stable, voire légèrement augmenté dans certains cas. En revanche, pour des valeurs plus élevées de α, une réduction du nombre de nœuds est observée. Par exemple, dans le cas de la demande complète avec $\alpha = 0.60$, le nombre moyen de nœuds passe de 496.67 à 479.33.

Le nombre de coupes générées reste modéré (entre 5 et 8 en moyenne au nœud racine), et leur ajout n'entraîne pas de surcoût significatif sur le temps de résolution du problème relaxé, ce qui suggère que la méthode est légère du point de vue computationnel.

L'ensemble de ces observations confirme l'intuition à l'origine de la méthode : les solutions fractionnaires du TSP-GL présentent des structures locales dans lesquelles le flot se répartit sur plusieurs chemins, conduisant à une sous-estimation du coût. La reconstruction d'un cycle imposant une traversée unique de ces composantes permet de restaurer une structure plus proche d'une solution entière et d'obtenir une borne inférieure plus réaliste.

En résumé, la méthode proposée fournit un renforcement léger mais robuste de la borne inférieure, avec un coût computationnel faible. Elle est particulièrement efficace lorsque la composante de latence est prépondérante, et constitue une première étape vers l'exploitation systématique de la structure fractionnaire des solutions du TSP-GL.

2.5 Conclusion

Ce chapitre a présenté deux approches complémentaires pour renforcer les bornes inférieures du TSP-GL dans un algorithme de séparation et d'évaluation, en exploitant la structure entière des solutions de relaxation linéaire.

La première approche, fondée sur une disjonction appliquée à une variable de flot fractionnaire circulant sur un arc activé, fournit une technique systématique et formellement garantie. En contraignant la variable choisie à chacune de ses valeurs extrêmes, on construit deux sous-

problèmes dont la meilleure borne inférieure est valide pour toute solution entière compatible avec les fixations courantes. La coupe d'optimalité conditionnelle associée permet d'injecter cette information dans l'arbre de recherche sans détériorer la validité globale du modèle.

La seconde approche exploite la topologie des composantes fractionnaires du flot, appelées blobs, en présence d'une chaîne d'arêtes entières reliant l'origine et la destination d'une demande. En substituant la dispersion fractionnaire du flot par un chemin couvrant unique au sein de chaque blob, on reconstruit un cycle hamiltonien candidat et l'on en déduit une borne inférieure renforcée sur le coût de routage. La coupe correspondante n'est active que lorsque la chaîne entière est intégralement sélectionnée par le solveur.

Les résultats expérimentaux confirment, pour les deux méthodes, que la borne inférieure est strictement améliorée ou inchangée sur l'ensemble du jeu d'essai, et n'est jamais détériorée, ce qui est cohérent avec les propriétés de validité démontrées. Les gains obtenus restent toutefois modérés en valeur absolue, typiquement de l'ordre de quelques dixièmes de point de pourcentage au nœud racine, et leur impact sur la taille de l'arbre de recherche n'est marqué que lorsque la composante de latence devient prépondérante dans la fonction objectif.

Ces travaux doivent être lus comme une preuve de concept. Ils démontrent qu'il est effectivement possible d'extraire une information exploitable à partir de la dispersion des flots dans la relaxation linéaire pour renforcer les méthodes exactes. Toutefois, l'ampleur modeste des améliorations observées indique que ce levier ne suffit pas, à lui seul, à repousser significativement le passage à l'échelle des méthodes exactes vers des instances de très grande taille. Ce constat motive le changement de paradigme adopté dans le chapitre suivant : plutôt que de chercher à renforcer la relaxation linéaire d'un modèle déterministe, nous proposerons d'agir sur la dimension du problème lui-même par un échantillonnage de l'ensemble des demandes.

CHAPITRE 3

APPROCHE PAR ÉCHANTILLONNAGE DE LA DEMANDE

3.1 Lien avec les approches précédentes

Le chapitre précédent a proposé deux méthodes de renforcement des bornes inférieures pour le TSP-GL, fondées sur l'exploitation de la structure entière des solutions de relaxation linéaire. Bien que valides et conformes aux propriétés démontrées, les améliorations obtenues sont restées modestes.

Notre objectif initial était en effet de traiter les plus grandes instances de Errico *et al.* (2017), sur lesquelles la décomposition de Benders rencontre ses limites : dans notre implémentation en Python, au-delà d'une vingtaine de nœuds, le temps de résolution explose et la convergence vers la solution optimale devient hors d'atteinte en pratique. Les expériences du chapitre précédent ont montré que les nouvelles bornes développées ne permettent pas ce passage à l'échelle.

Un objectif supplémentaire était aussi de trouver de nouvelles approches pouvant être généralisées à une classe générale de problèmes de conception de réseaux, regroupant notamment le TSP-GL et le MUFND.

Cette observation motive le changement de perspective adopté dans ce chapitre. En essayant notamment de trouver de nouvelles approches concurrençant les méthodes exactes.

3.2 Démarche générale

Nous considérons une classe de problèmes de conception de réseaux qui englobe notamment le TSP-GL et le MUFND. Ces problèmes partagent une structure commune à deux niveaux de décision : la construction ou la sélection d'une infrastructure d'une part, et le routage d'un ensemble de demandes sur cette infrastructure d'autre part.

Le coeur de notre démarche se situe dans la recherche d'un moyen d'obtenir une approximation de bonne qualité du problème original en utilisant des problèmes plus simples à résoudre, dérivés du problème original.

Une première idée naïve consisterait à résoudre le problème original sur un sous-ensemble de demande échantillonné. Nous obtenons ainsi un problème restreint qui est de fait plus simple à résoudre. Nous nous posons alors la question suivante :

Le résultat d'une telle approche serait elle représentative du problème original prenant en compte l'ensemble complet de demandes ?

En considérant cette approche naïve celle-ci se heurte immédiatement à une difficulté : en résolvant le problème sur un sous-ensemble restreint de demande, on sous-estime le coût total de routage, et le problème restreint cesse d'être représentatif du problème original.

Trois questions fondamentales structurent alors notre démarche :

- Sous quelles conditions un sous-ensemble de demandes permet-il de construire une approximation représentative du coût de routage du problème complet ?
- Est t'il possible de réinterpréter dans un contexte stochastique les problèmes de conception de réseaux à deux niveaux, notamment en interprétant chaque demande comme une réalisation d'un scénario ?
- Si oui, est t'il possible d'estimer statistiquement la qualité des solutions obtenues via des sous ensembles de demandes en utilisant par exemple le cadre du SAA ?

La réponse à la première question fait apparaître la nécessité d'une mise à l'échelle : l'échantillonnage d'un sous ensemble de demandes suivis d'une résolution directe ne suffit pas, il faut compenser la réduction du nombre de demandes par une mise à l'échelle appropriée pour que le problème restreint approxime fidèlement la valeur de l'objectif original. Cette problématique est traitée dans la sous-section 3.4.1.1.

Pour que la réduction du nombre de demandes soit exploitable, il faut s'assurer que le sous-ensemble traité reste représentatif du problème original. Ici, cela consiste à réinterpréter les demandes du problème déterministe comme les réalisations possibles de scénarios et sélectionner un sous-ensemble revient alors à tirer un échantillon d'une population. Cette réinterprétation ne modifie pas le problème étudié, mais le point de vue adopté sur celui-ci, elle place les problèmes considérés dans un cadre stochastique et ouvre en conséquence l'accès à l'ensemble des outils développés en programmation stochastique. Le *Sample Average Approximation* (SAA) est l'un de ces outils. Cette réinterprétation est détaillée dans la sous section 3.4.1.2.

Une fois cet estimateur correctement construit, la combinaison de plusieurs tirages indépendants permet d'obtenir des bornes inférieures, et l'évaluation a posteriori des solutions échantillonnées sur l'instance complète fournit des bornes supérieures.

Contrairement aux approches existantes, qui agissent sur la structure du réseau en sélectionnant un sous-ensemble d'arêtes ou de nœuds, notre méthode agit sur la dimension des demandes. La différence est importante en pratique : la réduction du réseau restreint l'espace des solutions réalisables et peut donc en exclure la solution optimale du problème d'origine, tandis que notre approche préserve intégralement l'ensemble des décisions de premier niveau et n'intervient que sur l'estimation de leur coût. Nous désignons cette méthode par l'expression **approche par échantillonnage de la demande** (ED).

3.3 Problème de conception de réseaux déterministe à deux niveaux

Cette section est destinée à donner un cadre mathématique général au type de problème traité. Ce cadre mathématique englobe donc les problèmes de conception de réseaux comme le TSP-GL et le MUFND. Nous considérons ainsi la classe de problèmes de conception de réseaux déterministes à deux niveaux suivante :

$$\min_{x \in X} \left\{ c^\top x + \sum_{k \in K} d_k z_k(x) \right\} \tag{3.1}$$

où $x \in X$ représente les décisions de conception du premier niveau, c est un vecteur de coût de conception, K est un ensemble fini de demandes, $d_k \geq 0$ correspond au volume ou au poids de la demande k, et $z_k(x)$ est le coût unitaire de routage optimal pour la demande k étant donné le vecteur de décision x. Le premier niveau sélectionne une infrastructure (arêtes, arcs, tournées), tandis que le second niveau évalue comment les demandes sont routées à travers cette infrastructure.

Pour une conception fixée x, chaque sous-problème définissant $z_k(x)$ est généralement facile à résoudre. La difficulté du problème global ne provient pas de la résolution des sous-problèmes individuels, mais de la nécessité de trouver une conception qui fonctionne bien pour toutes les demandes simultanément. Chaque choix d'infrastructure influence alors un grand nombre de décisions de routage, ce qui rend le problème global difficile à traiter en pratique.

Nous supposons dans la suite de cette section et dans la section 3.4 que le réseau $x \in X$ ne possède pas de contrainte de capacité ; le cas avec contraintes de capacité partagée fera l'objet de la section 3.5.

Le coût $z_k(x)$ correspond alors au coût d'un plus court chemin dans le réseau induit par x. En particulier, le problème de second niveau se décompose en $|K|$ sous-problèmes indépendants, et l'ajout de nouvelles demandes n'affecte pas les routages existants.

3.4 Méthodologie de la méthode d'échantillonnage de la demande

Cette section développe la méthode d'échantillonnage de la demande (ED) pour les problèmes déterministes à deux niveaux. Nous montrons d'abord comment réinterpréter le type de problème étudié de façon stochastique (Sous-section 3.4.1). Nous introduisons ensuite deux approches d'échantillonnage distinctes (Sous-section 3.4.2) et établissons leurs propriétés théoriques (Sous-section 3.4.3). Enfin, nous discutons de l'approche sélectionnée (Sous-section 3.4.4).

3.4.1 Réinterprétation stochastique

3.4.1.1 Remise à l'échelle

Face à la difficulté de traiter l'ensemble exhaustif des demandes K, une idée naturelle consisterait à réduire la taille effective du problème en ne résolvant ce dernier que sur un sous-ensemble $\mathcal{S} \subset K$. Le problème original étant un problème de conception de réseau dont le coût se décompose en une part de conception $c^\top x$ et une part de routage sur l'ensemble des demandes, la stratégie d'échantillonnage consiste à n'évaluer cette part de routage que sur la portion $\mathcal{S}$ des demandes, tout en cherchant à approcher au mieux le coût de routage qui aurait été obtenu sur l'ensemble complet K. L'objet central de cette section est donc la construction d'une approximation fiable du terme $\sum_{k \in K} d_k z_k(x)$ à partir d'un sous-ensemble $\mathcal{S}$.

Une formulation échantillonnée naïve s'écrirait de la forme suivante :

$$g_{\mathcal{S}}^{\text{naive}}(x) = c^\top x + \sum_{k \in \mathcal{S}} d_k z_k(x). \tag{3.2}$$

Cette approche présente un défaut majeur. En ne considérant qu'une fraction des demandes, le coût total de routage est artificiellement réduit. Puisqu'il n'existe aucun mécanisme de pondération, le problème échantillonné sous-estime systématiquement l'objectif déterministe original, ignorant une large part du volume de demande présent dans le modèle complet. Par conséquent, le problème restreint n'est plus représentatif du problème initial.

L'enjeu est donc d'interpréter nos demandes échantillonnées non plus comme un simple sous-ensemble statique, mais comme des observations que l'on peut repondérer via un facteur d'échelle approprié, de manière à reconstituer un estimateur fidèle du coût de routage sur l'ensemble K. Nous cherchons ainsi à construire une fonction objectif de la forme :

$$g_{\mathcal{S}}(x) = c^\top x + \gamma_{\mathcal{S}} \sum_{k \in \mathcal{S}} d_k z_k(x). \tag{3.3}$$

Le facteur multiplicateur $\gamma_{\mathcal{S}}$ est précisément ce facteur de remise à l'échelle : il a pour rôle de compenser les demandes qui n'ont pas été échantillonnées, en attribuant à chaque demande retenue un poids représentatif de la part de la population qu'elle représente. Logiquement, un petit échantillon nécessitera une correction importante, tandis que plus la taille de l'échantillon augmente, plus la compensation diminue. À la limite, lorsque toutes les demandes sont échantillonnées ($\mathcal{S} = K$), aucune correction n'est requise et $\gamma_{\mathcal{S}} = 1$.

3.4.1.2 Définition de l'estimateur

Toutefois, même avec un coefficient $\gamma_{\mathcal{S}}$ adéquat, un unique sous-ensemble aléatoire risque de ne pas capturer toute la diversité de l'ensemble complet. Différents tirages peuvent produire des structures de routage et des valeurs d'objectif très différentes. Pour qu'une telle approximation soit valide, il ne suffit donc pas qu'elle soit valide sur un échantillon particulier, elle doit demeurer représentative en moyenne sur l'ensemble des échantillons possibles.

Plutôt que de considérer K comme un simple ensemble de demandes à traiter, nous proposons de le voir comme une population dont chaque demande est un scénario. Dans ce point de vue, un sous-ensemble $\mathcal{S}$ correspond à un tirage de scénarios selon une loi de probabilité fixée par le procédé d'échantillonnage. Chaque demande échantillonnée est alors une observation tirée de cette population, et la valeur $g_{\mathcal{S}}(x)$ devient une variable aléatoire.

Le problème déterministe initial se trouve ainsi placé dans un cadre stochastique. L'objectif de l'approche par échantillonnage de la demande (ED) devient alors la construction d'un estimateur non biaisé de n scénarios $g_n(x)$ vérifiant :

$$\mathbb{E}[g_n(x)] = g_K(x) := c^\top x + \sum_{k \in K} d_k z_k(x), \quad \text{pour tout } x \in X, \tag{3.4}$$

où l'espérance est calculée sur la loi de l'échantillonnage.

Cette propriété d'absence de biais constitue un parallèle avec l'approche SAA, dans laquelle l'estimateur satisfait $\mathbb{E}[\hat{g}_N(x)] = g(x)$ (voir sous-section 1.9.3). Ce cadre, une fois posé, ouvre

la question de la construction de méthodes d'échantillonnage permettant de satisfaire cette condition d'absence de biais, et de bénéficier ainsi des garanties statistiques associées au SAA.

3.4.2 Méthode d'échantillonnage

Nous proposons deux méthodes distinctes pour construire deux estimateurs. La distinction fondamentale entre ces deux méthodes est que dans la première (Sous-section 3.4.2.1), les demandes avec des volumes plus importants sont échantillonnées plus fréquemment, tandis que dans la seconde (Sous-section 3.4.2.2), chaque demande a une probabilité égale d'être sélectionnée indépendamment de son poids. Les deux approches produisent des estimateurs non biaisés avec des garanties théoriques identiques.

3.4.2.1 Approche 1 : Échantillonnage par unités de demande

Nous définissons le volume total $S_K = \sum_{k \in K} d_k$ comme le volume total des demandes. L'univers d'échantillonnage $\mathcal{U}$ se compose de S_K unités élémentaires, partitionnées par demande : chaque demande k correspond à d_k unités.

Le mécanisme d'échantillonnage procède par tirage de n unités au hasard de $\mathcal{U}$. La probabilité que toute unité échantillonnée appartienne à la demande k est proportionnelle à son volume : $\mathbb{P}(\text{unité échantillonnée appartient à la demande } k) = d_k / S_K$. Par conséquent, les demandes avec des volumes plus importants sont naturellement échantillonnées plus fréquemment.

Toutes les d_k unités associées à la demande k partagent le même coût unitaire de routage $z_k(x)$, qui dépend de la structure de conception x. L'estimateur pour cette première approche prend la forme suivante :

$$g_n^{(1)}(x) = c^\top x + \frac{S_K}{n} \sum_{i=1}^{n} z_{k_i}(x) \tag{3.5}$$

où k_i désigne la demande à laquelle appartient la i-ème unité échantillonnée, et le facteur de mise à l'échelle S_K/n compense le processus d'échantillonnage pour assurer l'absence de biais.

3.4.2.2 Approche 2 : Échantillonnage par scénarios

La seconde approche adopte une stratégie d'échantillonnage fondamentalement différente. Ici, l'univers d'échantillonnage est l'ensemble de demandes K lui-même, et n demandes sont tirées uniformément au hasard, notées $K_1, \ldots, K_n$. Contrairement à la première approche, toutes les demandes ont une probabilité égale d'être sélectionnées indépendamment de leur poids : $\mathbb{P}(K_i = k) = 1/|K|$ pour tout $k \in K$.

Dans cette approche, les volumes de demande apparaissent explicitement dans la fonction objectif pour préserver la représentation correcte du problème original. Lorsque la demande k est échantillonnée, elle contribue son coût complet $d_k z_k(x)$ à l'estimateur. Formellement :

$$g_n^{(2)}(x) = c^\top x + \frac{|K|}{n} \sum_{i=1}^{n} d_{K_i} z_{K_i}(x) \tag{3.6}$$

où d_{K_i} désigne le poids de la demande échantillonnée K_i.

3.4.3 Propriétés théoriques

Nous établissons ici les propriétés théoriques des deux approches d'échantillonnage proposées.

3.4.3.1 Absence de biais

Pour toute décision de premier niveau fixée $x \in X$, les deux estimateurs sont non biaisés. On rappelle que la valeur objectif du problème déterministe complet est notée :

$$g_K(x) := c^\top x + \sum_{k \in K} d_k z_k(x) \tag{3.7}$$

Nous cherchons à établir que :

$$\mathbb{E}[g_n^{(1)}(x)] = \mathbb{E}[g_n^{(2)}(x)] = g_K(x) \tag{3.8}$$

Démonstration pour l'Approche 1. Fixons un $x \in X$ arbitraire. L'estimateur pour l'Approche 1 est :

$$g_n^{(1)}(x) = c^\top x + \frac{S_K}{n} \sum_{i=1}^{n} z_{k_i}(x) \tag{3.9}$$

Puisque $c^\top x$ est déterministe et par linéarité de l'espérance :

$$\mathbb{E}\left[\frac{S_K}{n} \sum_{i=1}^{n} z_{k_i}(x)\right] = \frac{S_K}{n} \sum_{i=1}^{n} \mathbb{E}[z_{k_i}(x)] \tag{3.10}$$

Les unités tirées sont indépendantes et identiquement distribuées selon la distribution uniforme sur $\mathcal{U}$. Par conséquent, pour tout i, l'unité échantillonnée appartient à la demande k avec probabilité d_k / S_K :

$$\mathbb{E}[z_{k_i}(x)] = \sum_{k \in K} \mathbb{P}(\text{unité } i \text{ appartient à la demande } k)\, z_k(x) \tag{3.11}$$

$$= \sum_{k \in K} \frac{d_k}{S_K} z_k(x) \tag{3.12}$$

En substituant :

$$\mathbb{E}[g_n^{(1)}(x)] = c^\top x + \frac{S_K}{n} \sum_{i=1}^{n} \sum_{k \in K} \frac{d_k}{S_K} z_k(x) \tag{3.13}$$

$$= c^\top x + \frac{S_K}{n} \cdot n \cdot \sum_{k \in K} \frac{d_k}{S_K} z_k(x) \tag{3.14}$$

$$= c^\top x + \sum_{k \in K} d_k z_k(x) = g_K(x) \tag{3.15}$$

□

Démonstration pour l'Approche 2. Fixons un $x \in X$ arbitraire. L'estimateur pour l'Approche 2 est :

$$g_n^{(2)}(x) = c^\top x + \frac{|K|}{n} \sum_{i=1}^{n} d_{K_i} z_{K_i}(x) \tag{3.16}$$

Par linéarité de l'espérance :

$$\mathbb{E}\left[\frac{|K|}{n} \sum_{i=1}^{n} d_{K_i} z_{K_i}(x)\right] = \frac{|K|}{n} \sum_{i=1}^{n} \mathbb{E}[d_{K_i} z_{K_i}(x)] \tag{3.17}$$

Les demandes sont tirées indépendamment et identiquement de la distribution uniforme sur K. Pour tout i :

$$\mathbb{E}[d_{K_i} z_{K_i}(x)] = \sum_{k \in K} \mathbb{P}(K_i = k)\, d_k z_k(x) \tag{3.18}$$

$$= \sum_{k \in K} \frac{1}{|K|} d_k z_k(x) \tag{3.19}$$

En substituant :

$$\mathbb{E}[g_n^{(2)}(x)] = c^\top x + \frac{|K|}{n} \sum_{i=1}^{n} \sum_{k \in K} \frac{1}{|K|} d_k z_k(x) \tag{3.20}$$

$$= c^\top x + \frac{|K|}{n} \cdot n \cdot \sum_{k \in K} \frac{1}{|K|} d_k z_k(x) \tag{3.21}$$

$$= c^\top x + \sum_{k \in K} d_k z_k(x) = g_K(x) \tag{3.22}$$

□

3.4.3.2 Propriété de borne inférieure

Dans la suite de cette sous-section, g_n désigne indifféremment $g_n^{(1)}$ ou $g_n^{(2)}$: le raisonnement ne fait intervenir que la propriété d'absence de biais, qui a été établie pour les deux estimateurs.

La propriété d'absence de biais établie ci-dessus a une conséquence immédiate et importante : la valeur optimale espérée de l'estimateur fournit une borne inférieure valide sur l'optimum du problème complet. Désignons :

$$v^* := \min_{x \in X} g_K(x) \quad \text{et} \quad v_n^* := \min_{x \in X} g_n(x) \tag{3.23}$$

les valeurs optimales des problèmes déterministe et approché, respectivement. Alors :

$$\mathbb{E}[v_n^*] \leq v^* \tag{3.24}$$

Démonstration. Soit $x^* = \arg\min_{x \in X} g_K(x)$, de sorte que $v^* = g_K(x^*)$. Par définition du minimum, pour toute réalisation de l'échantillon :

$$v_n^* = \min_{x \in X} g_n(x) \leq g_n(x^*). \tag{3.25}$$

En prenant l'espérance :

$$\mathbb{E}[v_n^*] \leq \mathbb{E}[g_n(x^*)] = g_K(x^*) = v^*, \tag{3.26}$$

où l'égalité découle de la propriété d'absence de biais appliquée à la solution fixée x^*. □

3.4.4 Sélection de l'approche d'échantillonnage

Bien que les deux approches fournissent des estimateurs non biaisés avec des garanties théoriques identiques, nous sélectionnons l'Approche 2 (échantillonnage par scénarios) pour notre étude, sur la base de considérations pratiques de mise en œuvre. L'avantage principal de l'Approche 2 est le contrôle direct sur la taille du problème : l'échantillonnage de n demandes de K réduit immédiatement le nombre de variables du second niveau, ce qui facilite la gestion du coût computationnel de chaque réplication. Dans toute la suite de ce chapitre, nous notons donc simplement $g_n(x) := g_n^{(2)}(x)$.

3.4.5 Calcul des bornes via la méthode d'échantillonnage de la demande

3.4.5.1 Construction de la borne inférieure

D'après la propriété établie en Sous-section 3.4.3.2, on a $\mathbb{E}[v_n^*] \leq v^*$. L'espérance étant prise sur l'ensemble des échantillons possibles, son calcul exact est en pratique impossible puisqu'il faudrait énumérer tous les sous-ensembles de demandes de taille n possibles dans K. Nous l'approchons donc par une moyenne empirique calculée sur un nombre fini de réplications indépendantes.

Soit M le nombre de réplications. Pour chaque réplication $m = 1, \ldots, M$, nous échantillonnons indépendamment un sous-ensemble de demandes $K_m \subseteq K$ de taille n, ce qui définit l'estimateur de la m-ième réplication :

$$\hat{g}_{n,m}(x) = c^\top x + \frac{|K|}{n} \sum_{k \in K_m} d_k z_k(x). \tag{3.27}$$

Nous résolvons ensuite le problème correspondant :

$$v_{n,m}^* = \min_{x \in X} \hat{g}_{n,m}(x). \tag{3.28}$$

Chaque réplication produit une valeur optimale $v_{n,m}^*$ et une solution associée $x_m^* \in X$. La moyenne empirique de ces valeurs optimales fournit alors une approximation de $\mathbb{E}[v_n^*]$, et constitue donc, en espérance, une borne inférieure sur v^* :

$$\bar{v}_{n,M} = \frac{1}{M} \sum_{m=1}^{M} v_{n,m}^*. \tag{3.29}$$

En appliquant le cadre rappelé en Sous-section 1.9.6, nous construisons à partir des M réplications un intervalle de confiance de niveau $1 - \beta$ pour $\mathbb{E}[v_n^*]$, et nous retenons la borne inférieure de

cet intervalle :

$$\mathrm{BI}_{n,M} = \bar{v}_{n,M} - t_{1-\beta/2,\,M-1} \cdot \frac{s_{n,M}}{\sqrt{M}}, \tag{3.30}$$

où $s_{n,M}$ désigne l'écart-type empirique des $v^*_{n,1}, \ldots, v^*_{n,M}$, et $t_{1-\beta/2,\,M-1}$ le quantile d'ordre $1-\beta/2$ de la loi de Student à $M-1$ degrés de liberté.

3.4.5.2 Construction de la borne supérieure

Pour obtenir une borne supérieure valide, nous évaluons les solutions candidates $\{x^*_1, \ldots, x^*_M\}$ par rapport à l'ensemble complet de demandes K.

Une solution x^*_m obtenue à partir d'un échantillon K_m peut être infaisable pour le problème complet, c'est-à-dire qu'elle ne fournit pas l'infrastructure nécessaire pour router toutes les demandes $k \in K \setminus K_m$. Dans de tels cas, nous définissons $\hat{x}_m$ comme la version réparée de x^*_m, obtenue en ajoutant l'infrastructure minimale requise pour rendre x^*_m réalisable pour l'ensemble complet K. Si x^*_m est déjà réalisable pour tout K, alors $\hat{x}_m = x^*_m$.

Nous évaluons le coût exact de chaque solution réparée :

$$g_K(\hat{x}_m) = c^\top \hat{x}_m + \sum_{k \in K} d_k z_k(\hat{x}_m) \tag{3.31}$$

La meilleure solution $\bar{x}$ et la borne supérieure résultante BS_M sont données par :

$$\mathrm{BS}_M = g_K(\bar{x}) = \min_{m=1,\ldots,M} g_K(\hat{x}_m) \tag{3.32}$$

Puisque $\bar{x}$ est réalisable pour le problème déterministe original, $\mathrm{BS}_M \geqslant v^*$ est une borne supérieure valide.

3.4.5.3 Estimation de l'écart d'optimalité

La qualité de la meilleure solution $\bar{x}$ est estimée via l'écart d'optimalité, calculé à partir de la borne supérieure BS_M et de la borne inférieure $\mathrm{BI}_{n,M}$ obtenue avec un niveau de confiance de $1-\beta$:

$$\text{Écart} = \frac{\mathrm{BS}_M - \mathrm{BI}_{n,M}}{\mathrm{BS}_M} \times 100\%. \tag{3.33}$$

Lorsque $n \to |K|$ et $M \to \infty$, les bornes $\mathrm{BI}_{n,M}$ et BS_M convergent vers v^* (voir Mak *et al.* (1999); Shapiro, Dentcheva & Ruszczynski (2021)).

3.5 Extension aux problèmes de conception de réseau avec capacités

Il est naturel de se demander si la nouvelle méthode développée s'étend aux problèmes de conception de réseau intégrant des contraintes de capacité partagée sur les arcs. Dans un tel contexte, le sous-problème de routage couple l'ensemble des demandes via les capacités ; les demandes ne sont donc plus routées indépendamment les unes des autres.

3.5.1 Notations

Soit $Q(x, \mathcal{S})$ le coût optimal du routage pour un sous-ensemble donné de demandes $\mathcal{S} \subseteq K$, sur x. Le coût total, évalué sur l'ensemble complet des demandes K, est défini par :

$$g_K^{\text{cap}}(x) := c^\top x + Q(x, K) \tag{3.34}$$

Afin de relier ce coût global de routage aux demandes individuelles, notons $z_k^*(x)$ le coût de la demande k au sein de la solution globale optimale pour l'ensemble complet K. Le coût réel du second niveau peut ainsi être décomposé :

$$Q(x, K) = \sum_{k \in K} d_k z_k^*(x) \tag{3.35}$$

En suivant la méthodologie d'échantillonnage par scénarios (voir Sous-section 3.4.2.2), nous tirons n demandes uniformément au hasard. Soit $\mathcal{S}_n = \{K_1, \ldots, K_n\}$ cet échantillon aléatoire. L'estimateur pour ce problème avec contraintes de capacité prend alors la forme :

$$g_n^{\text{cap}}(x) = c^\top x + \frac{|K|}{n} Q(x, \mathcal{S}_n) \tag{3.36}$$

3.5.2 Analyse du biais de l'estimateur

Nous démontrons à présent que cet estimateur présente un biais négatif par rapport au coût du problème global. Ce résultat est important : il montre que la propriété d'absence de biais, centrale dans le cas sans capacité, ne se transfère pas directement au cas avec capacité.

Démonstration. Puisque Q modélise un problème de minimisation, le coût optimal sur l'échantillon restreint est naturellement majoré par le coût qu'aurait le flot optimal global évalué sur ce même échantillon :

$$Q(x, \mathcal{S}_n) \leq \sum_{i=1}^{n} d_{K_i} z^*_{K_i}(x) \tag{3.37}$$

En passant à l'espérance sur l'estimateur (3.36), on obtient :

$$\mathbb{E}[g_n^{\text{cap}}(x)] = c^\top x + \frac{|K|}{n} \mathbb{E}[Q(x, \mathcal{S}_n)] \tag{3.38}$$

En y substituant notre majorant, on obtient l'inégalité :

$$\mathbb{E}[g_n^{\text{cap}}(x)] \leq c^\top x + \frac{|K|}{n} \mathbb{E}\left[\sum_{i=1}^{n} d_{K_i} z^*_{K_i}(x)\right] \tag{3.39}$$

Par linéarité de l'espérance, et sachant que les demandes sont tirées de manière indépendante selon une loi uniforme ($\mathbb{P}(K_i = k) = \frac{1}{|K|}$) :

$$\mathbb{E}\left[\sum_{i=1}^{n} d_{K_i} z^*_{K_i}(x)\right] = \sum_{i=1}^{n} \mathbb{E}[d_{K_i} z^*_{K_i}(x)] \tag{3.40}$$

$$= \sum_{i=1}^{n} \sum_{k \in K} \mathbb{P}(K_i = k) \cdot d_k z^*_k(x) \tag{3.41}$$

$$= \sum_{i=1}^{n} \frac{1}{|K|} \sum_{k \in K} d_k z^*_k(x) \tag{3.42}$$

$$= \frac{n}{|K|} \sum_{k \in K} d_k z^*_k(x) \tag{3.43}$$

$$= \frac{n}{|K|} Q(x, K) \tag{3.44}$$

En réinjectant ce résultat dans l'inégalité 3.39, nous obtenons :

$$\mathbb{E}[g^{\text{cap}}_n(x)] \leq c^\top x + \frac{|K|}{n} \cdot \frac{n}{|K|} Q(x, K) \tag{3.45}$$

$$= c^\top x + Q(x, K) = g^{\text{cap}}_K(x). \tag{3.46}$$

Ainsi, l'espérance de l'estimateur avec capacité est inférieure ou égale à la valeur réelle de la fonction objectif. Cela démontre que cet estimateur sous-évalue systématiquement le coût réel, introduisant de fait un biais négatif. □

3.5.3 Conséquences

Le biais étant négatif, la propriété de borne inférieure reste valide concernant les cas avec capacités : en reprenant le raisonnement de la Sous-section 3.4.3.2, on obtient

$$\mathbb{E}[\min_{x \in X} g^{\text{cap}}_n(x)] \;\leq\; \min_{x \in X} g^{\text{cap}}_K(x). \tag{3.47}$$

La méthode par ED reste donc applicable et fournit une borne inférieure valide. Toutefois, contrairement au cas sans capacité où l'estimateur est centré sur la valeur cible, l'estimateur sous-évalue ici systématiquement le coût réel. La borne inférieure obtenue est par conséquent de moins bonne qualité en comparaison avec le cas sans contraintes de capacité, et la qualité de l'encadrement dégradée.

3.6 Application de la méthode par échantillonnage de la demande au TSP-GL et au MUFND

Pour justifier l'application de notre nouvelle méthode décrite dans la section précédente au TSP-GL et au MUFND, il est nécessaire de montrer que ces deux problèmes admettent une structure à deux niveaux. Cette décomposition est centrale : elle sépare les décisions de conception (premier niveau) des décisions de routage (deuxième niveau), permettant une réduction efficace du problème via échantillonnage des demandes.

3.6.1 Applicabilité au TSP-GL

Pour le TSP-GL, la variable de premier niveau x représente l'ensemble des arêtes sélectionnées pour former un cycle hamiltonien sur l'ensemble des nœuds. Une fois ce cycle construit, chaque demande $k \in K$ peut être routée indépendamment des autres : il suffit d'emprunter le plus court des deux chemins reliant son origine à sa destination le long du cycle.

Cette structure satisfait les hypothèses requises par la méthode par ED :

- Le coût de premier niveau $c^\top x$ correspond au coût de construction du cycle et est indépendant des demandes.
- Pour chaque demande k, le coût de routage $z_k(x)$ est défini comme la longueur du plus court chemin entre l'origine et la destination de k le long du cycle x.
- La structure de premier niveau étant indépendante des demandes considérées, le réseau obtenu en résolvant le problème sur un sous-ensemble $\mathcal{S} \subset K$ reste un cycle hamiltonien sur

l'ensemble des nœuds. Ce cycle est donc capable de router non seulement les demandes de $\mathcal{S}$, mais également l'ensemble complet des demandes K.

Par conséquent, le TSP-GL respecte bien la structure à deux niveaux décrite dans la section 3.3. Lors de l'application de l'approche par échantillonnage, on peut échantillonner un sous-ensemble de taille n, $K_m \subset K$ de demandes pour la réplication m. La solution x^*_m obtenue en résolvant le problème approché restreint à K_m reste réalisable pour l'ensemble complet K sans modification. Cela signifie qu'au moment de l'évaluation (Sous-section 3.4.5.2), $\hat{x}_m = x^*_m$: aucune réparation n'est nécessaire.

En résumé, puisque chaque solution x^*_m est réalisable pour le problème complet, les bornes inférieures et supérieures sont calculées directement :

- **Borne inférieure** : $\mathrm{BI}_{n,M}$ obtenue à partir des $v^*_{n,1}, \ldots, v^*_{n,M}$ (Éq. 3.30).
- **Borne supérieure** : $\mathrm{BS}_M = \min_{m=1,\ldots,M} g_K(x^*_m)$ (Éq. 3.32), où $\hat{x}_m = x^*_m$.

3.6.2 Applicabilité au MUFND

Pour le MUFND, la situation est différente. La variable de premier niveau x représente l'ensemble des arcs définissant le réseau sur lequel les demandes $k \in K$ seront routées.

Contrairement au TSP-GL où un cycle couvre tout routage possible, le MUFND ne contraint pas le réseau représenté par x à avoir une structure particulière. Il est alors possible qu'une demande $k \in K \setminus K_m$ (non présente dans l'échantillon) ne puisse pas être routée dans le réseau défini par x^*_m. En particulier, le réseau induit par x^*_m peut ne pas contenir de chemin reliant l'origine et la destination de cette demande, ce qui rend x^*_m infaisable pour le problème complet.

On notera que cette infaisabilité n'est pas un défaut de l'estimateur lui-même : la propriété de non-biais établie à la sous-section 3.4.3.1 reste valide. La difficulté étant que la solution x^*_m obtenu peut ne pas être réalisable pour le problème complet, ce qui empêche son évaluation directe pour produire une borne supérieure.

Pour éviter ces problèmes d'infaisabilité, une phase de réparation est nécessaire. Pour chaque solution x_m^* issue d'une réplication par ED, on définit $\hat{x}_m$ comme suit :

$$\hat{x}_m = x_m^* \cup x_{\text{repair},m} \tag{3.48}$$

où $x_{\text{repair},m}$ est l'ensemble d'arcs additionnels nécessaires pour rendre la solution réalisable vis-à-vis de l'ensemble complet de demandes K. Formellement, cet ensemble est obtenu en résolvant le problème suivant :

$$\min \quad \sum_{(i,j)\in A} \bar{c}_{ij} x_{ij} + \sum_{k\in K} \sum_{(i,j)\in A} d_k q_{ij}^k f_{ij}^k \tag{3.49a}$$

$$\text{s.c.} \quad \sum_{j\in N} f_{ji}^k - \sum_{j\in N} f_{ij}^k = \begin{cases} -1 & \text{si } i = s_k \\ 1 & \text{si } i = h_k \\ 0 & \text{sinon} \end{cases} \qquad \forall i \in N, \forall k \in K \tag{3.49b}$$

$$f_{ij}^k \leq x_{ij} \qquad \forall (i,j) \in A, \forall k \in K \tag{3.49c}$$

$$f_{ij}^k \geq 0 \qquad \forall (i,j) \in A, \forall k \in K \tag{3.49d}$$

$$x_{ij} \geq (x_m^*)_{ij} \qquad \forall (i,j) \in A \tag{3.49e}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{3.49f}$$

En pratique, cela revient à résoudre le MUFND sur le graphe initial, en imposant que tous les arcs sélectionnés dans x_m^* soient conservés, tout en laissant le solveur libre de sélectionner des arcs additionnels afin de restaurer la faisabilité vis-à-vis de l'ensemble des demandes.

Pour le MUFND, la procédure est par conséquent légèrement modifiée :

- **Borne inférieure** : $\text{BI}_{n,M}$ obtenue à partir des $v_{n,1}^*, \ldots, v_{n,M}^*$ (Éq. 3.30).
- **Borne supérieure** : BS_M est obtenue en évaluant les solutions réparées $\hat{x}_m = x_m^* \cup x_{\text{repair},m}$ (Éq. 3.32).

La solution réparée $\hat{x}_m$ est réalisable pour le problème complet, ce qui garantit que BS_M est une borne supérieure valide sur v^*.

Concernant le coût computationnel de cette phase de réparation, il est légitime de s'interroger puisque le problème d'optimisation (3.49) requiert la résolution d'un MUFND sur l'ensemble complet des demandes. Cependant, la fixation des variables de conception déjà sélectionnées par le problème approché ($x_{ij} \geq (x^*_m)_{ij}$) réduit drastiquement l'espace de recherche. En pratique, sur nos expériences, cette phase de réparation ne représente qu'une fraction marginale du temps total de l'approche ED (généralement moins de 5 à 10% du temps de résolution global), l'essentiel de l'effort computationnel demeurant alloué à la résolution des problèmes restreints.

3.7 Résultats expérimentaux

Dans cette section, nous présentons les résultats expérimentaux obtenus afin d'évaluer les performances de la méthode proposée. Nous décrivons tout d'abord les choix de paramètres utilisés dans le cadre de notre nouvelle méthode d'échantillonnage, en particulier le nombre de réplications et la taille des échantillons. Nous analysons ensuite les résultats obtenus sur nos deux problèmes : le TSP-GL, qui constitue un cadre de référence issu de la littérature, puis le MUFND, pour lequel nous considérons à la fois des instances classiques et de nouvelles instances de grande taille conçues pour mettre en évidence les limites des méthodes exactes. Enfin, nous comparons systématiquement la méthode par ED avec des méthodes existantes, incluant la décomposition de Benders exacte et l'approche par échantillonnage de graphe (EG) décrite dans la section 3.7.2.

3.7.1 Environnement logiciel et outils de résolution

Comme dans le chapitre 2, l'ensemble des méthodes proposées dans ce chapitre a été développé en Python (version 3.9.25).

Les expériences numériques ont été réalisées sur les machines du CIRRELT, équipées d'un processeur Intel Core i7-7800X et de 124 Go de mémoire vive. Sauf indication contraire, les paramètres par défaut de CPLEX ont été utilisés.

3.7.2 Approches existantes basées sur la réduction du réseau

Soit $G = (N, E)$ le graphe initial sur lequel sont définies les variables de décision du premier niveau $x \in X$. Les travaux de Ma & Zavala (2022) proposent de réduire la complexité du problème en agissant directement sur ce graphe, via des techniques d'échantillonnage de graphe et d'agrégation de nœuds.

Nous nous concentrons ici sur l'approche par échantillonnage de graphe (EG). L'idée consiste à construire un sous-graphe $G' = (N, E')$ en échantillonnant un sous-ensemble d'arêtes $E' \subseteq E$, avec $|E'| \ll |E|$. On définit alors un ensemble de décisions restreint $X' \subseteq X$, correspondant aux solutions réalisables sur le sous-graphe G'.

On résout ensuite le problème restreint :

$$\min_{x \in X'} \left\{ c^\top x + \sum_{k \in K} d_k z_k(x) \right\}. \tag{3.50}$$

Toute solution $x \in X'$ réalisable pour ce problème restreint est également réalisable pour le problème original. Par conséquent, la valeur objective associée fournit naturellement une borne supérieure du problème initial.

En pratique, cette procédure est répétée plusieurs fois à partir de différents sous-graphes échantillonnés. Soit R le nombre de réplications, et $E'_1, \ldots, E'_R$ les sous-ensembles d'arêtes générés. On obtient ainsi une collection de solutions réalisables $x^{(1)}, \ldots, x^{(R)}$, et la meilleure borne supérieure est donnée par :

$$\min_{r=1,\ldots,R} \left\{ c^\top x^{(r)} + \sum_{k \in K} d_k z_k\left(x^{(r)}\right) \right\}. \tag{3.51}$$

Cette approche constitue une base de comparaison naturelle avec la méthode ED, qui repose sur une réduction de la dimension des demandes plutôt que sur celle du réseau.

3.7.3 Valeurs des paramètres d'échantillonnage

Comme souligné dans la littérature fondatrice sur le SAA (Shapiro, A. (2003)), des valeurs de réplication relativement faibles telles que $M = 5$ ou $M = 10$ sont souvent suffisantes en pratique pour obtenir des estimations fiables de la valeur optimale espérée.

En complément de ces considérations théoriques, nous avons mené des expériences préliminaires avec différentes valeurs de M afin d'évaluer le compromis entre la qualité des solutions et le coût de calcul. Ces expériences montrent que l'augmentation de M n'apporte que des améliorations marginales en termes de stabilité de l'estimateur, tout en augmentant significativement le temps de calcul, chaque réplication nécessitant la résolution d'un problème d'optimisation NP-difficile complet à optimalité.

Sur la base de ces observations, nous fixons $M = 5$ comme compromis entre précision et temps de calcul. Ce choix est cohérent avec les pratiques usuelles dans les études numériques basées sur le SAA et adapté aux instances de grande taille considérées dans ce travail.

Dans nos expériences numériques, pour la méthode par ED, nous utilisons un échantillonnage représentant 10% de l'ensemble des demandes. Pour l'approche EG, nous considérons également un sous-ensemble contenant 10% des arêtes du graphe initial et effectuons $R = 5$ réplications, en sélectionnant la meilleure borne supérieure obtenue.

3.7.4 Résultats expérimentaux sur le TSP-GL

Pour évaluer la performance de la méthode par ED sur le TSP-GL, nous utilisons un ensemble d'instances développé par Errico *et al.* (2017) appelées TSPGL2. Ces instances s'appuient sur la célèbre bibliothèque de référence TSPLIB. Toutes les instances du TSP-GL sont résolues avec $\alpha = 0.5$, équilibrant les coûts de conception et de latence.

Les résultats sont rapportés sur deux tableaux. Le Tableau 3.1 rassemble les bornes inférieures et supérieures brutes obtenues par chaque méthode, tandis que le Tableau 3.2 en synthétise les écarts d'optimalité et les temps de calcul associés.

Le Tableau 3.1 se lit comme suit. La colonne *Instance* identifie l'instance considérée, le suffixe -S correspondant à une demande dispersée et le suffixe -C à une demande complète et polarisée (voir section 2.4). Les quatre colonnes suivantes concernent la borne inférieure : *ED* donne la valeur de la borne inférieure produite par la méthode par échantillonnage de la demande ; σ en est l'écart-type ; *IC* reporte la borne prenant en compte l'intervalle de confiance à 95 % associé ; *Référence* indique la borne inférieure obtenue par décomposition de Benders exacte dans Errico *et al.* (2017).

Les trois colonnes suivantes concernent la borne supérieure : *ED* est la meilleure borne supérieure trouvée par notre méthode, ; *Référence* est celle obtenue dans l'approche développée par Errico *et al.* (2017), *EG* est la meilleure valeur obtenue par EG sur cinq sous-graphes (voir section 3.7.2).

Le Tableau 3.2 se lit comme suit. La colonne *Écart ED* reporte l'écart d'optimalité obtenu par la méthode par ED, calculé entre la borne supérieure ED et la borne inférieure ED, exprimé en pourcentage. La colonne *Écart IC* est calculée de manière analogue, mais en utilisant la borne inférieure prenant en compte l'intervalle de confiance : il s'agit donc d'un écart intégrant l'incertitude statistique due à l'échantillonnage. La colonne *Écart Référence* reprend l'écart rapporté par Errico *et al.* (2017), obtenu à la limite de temps imposée à la décomposition de Benders exacte. Les deux dernières colonnes donnent les temps de calcul (en secondes) de la méthode par ED et de la méthode par échantillonnage de graphe (EG) ; les temps de la méthode exacte ne sont pas reportés car celle-ci atteint systématiquement la limite de temps (5 heures) sur ces instances. La dernière ligne reporte la moyenne sur l'ensemble des instances.

Les Tableaux 3.1 et 3.2 comparent les performances de la méthode par ED avec celles de la décomposition de Benders exacte rapportées dans Errico *et al.* (2017), ainsi qu'avec l'approche EG expliquée dans la section 3.7.2. Pour toutes les instances considérées, la méthode exacte atteint la limite de temps sans résoudre complètement le problème, fournissant ainsi un encadrement

Tableau 3.1 TSPGL2 — Bornes inférieures et supérieures obtenues par chaque méthode

Instance	**Borne inférieure**				**Borne supérieure**		
	ED	σ	IC	Référence	ED	Référence	EG
gr48-S	5432.17	34.03	5389.92	5383.99	5503.53	5618.11	21437
gr48-C	5612.84	36.55	5567.46	5431.39	5767.99	5767.99	20721
eil51-S	459.62	4.56	453.96	452.61	465.30	465.30	1438
eil51-C	463.11	5.53	456.24	454.78	474.62	483.96	1356
eil76-S	578.39	7.45	569.14	566.36	593.09	603.18	2064
eil76-C	572.73	7.41	563.53	564.71	594.43	595.33	2165
eil101-S	669.28	8.44	658.80	655.32	714.53	714.53	2163
eil101-C	669.91	9.30	658.36	655.00	714.81	714.81	2249
kroA100-S	22909.45	91.79	22795.48	22623.10	23582.00	23595.20	159800
kroA100-C	23006.62	101.83	22880.18	22630.90	23315.00	23315.00	160354
kroB100-S	24267.18	97.89	24145.63	23707.00	24815.80	24815.80	157859
kroB100-C	23698.77	108.75	23563.74	23247.90	24417.00	24654.10	156365
kroC100-S	22610.53	82.67	22507.88	22394.10	23263.50	23263.50	175693
kroC100-C	23113.09	95.73	22994.23	22528.10	23688.50	23688.50	173643
kroD100-S	23359.66	111.47	23221.25	23040.20	24027.00	24069.20	169372
kroD100-C	23445.31	96.74	23325.19	22841.20	23896.50	23896.50	168942
gr96-S	60404.22	148.26	60220.13	59045.00	62215.90	62215.90	207843
gr96-C	59738.88	157.28	59543.59	58651.60	62656.20	62656.20	221397
att48-S	11462.47	68.77	11377.08	11347.50	11792.60	11792.60	37218
att48-C	11471.63	72.15	11382.04	11339.40	11633.70	11633.70	34193
berlin52-S	8269.95	55.42	8201.14	8200.22	8271.37	8271.37	26184
berlin52-C	8442.19	69.88	8355.42	8314.09	8582.92	8582.92	26842
st70-S	734.48	8.12	724.40	721.63	748.37	748.37	2288
st70-C	738.92	8.95	727.81	722.44	756.65	756.65	2520
pr76-S	113428.36	612.85	112667.41	112037.00	116485.00	116485.00	352918
pr76-C	115214.78	745.12	114289.59	113962.00	122162.00	122162.00	391274
rat99-S	1298.51	14.88	1280.03	1271.88	1380.47	1380.47	4339
rat99-C	1291.88	12.95	1275.80	1275.85	1327.03	1327.03	3987
rd100-S	8612.74	78.43	8515.36	8438.23	8855.01	8855.01	51744
rd100-C	8594.36	84.92	8488.92	8413.56	8877.60	8877.60	55918

de la solution optimale que nous exprimons sous la forme d'un écart en pourcentage entre borne inférieure et borne supérieure.

Tableau 3.2 TSPGL2 — Écarts d'optimalité et temps de calcul

Instance	**Écart (%)**			**Temps (s)**	
	ED	IC	Référence	ED	EG
gr48-S	**1.30**	2.06	4.17	526	2072
gr48-C	**2.69**	3.48	5.84	522	1985
eil51-S	**1.22**	2.44	2.73	447	2143
eil51-C	**2.43**	3.87	6.03	518	2217
eil76-S	**2.48**	4.04	6.10	2405	2682
eil76-C	**3.65**	5.20	5.14	2673	2795
eil101-S	**6.33**	7.80	8.29	2738	7450
eil101-C	**6.28**	7.90	8.37	2694	7624
kroA100-S	**2.85**	3.34	4.12	2762	10189
kroA100-C	**1.32**	1.86	2.93	2728	10346
kroB100-S	**2.21**	2.70	4.47	2790	9875
kroB100-C	**2.94**	3.50	5.70	2713	9956
kroC100-S	**2.81**	3.25	3.74	2799	10621
kroC100-C	**2.43**	2.93	4.90	2681	10482
kroD100-S	**2.78**	3.36	4.28	2756	10096
kroD100-C	**1.89**	2.39	4.42	2700	10234
gr96-S	**2.91**	3.21	5.10	2853	7357
gr96-C	**4.66**	4.97	6.39	2942	7586
att48-S	**2.80**	3.52	3.77	498	2126
att48-C	**1.39**	2.16	2.53	512	2051
berlin52-S	**0.02**	0.85	0.86	534	2281
berlin52-C	**1.64**	2.65	3.13	548	2325
st70-S	**1.86**	3.20	3.57	2210	2958
st70-C	**2.34**	3.81	4.52	2385	3121
pr76-S	**2.62**	3.28	3.82	2520	3185
pr76-C	**5.69**	6.44	6.71	2668	3274
rat99-S	**5.94**	7.28	7.87	3735	10325
rat99-C	**2.65**	3.86	3.86	3690	10155
rd100-S	**2.74**	3.84	4.71	2815	10781
rd100-C	**3.19**	4.38	5.23	2768	10724
Moyenne	**2.87**	**3.79**	4.78	**2171**	6300

La méthode par ED fournit systématiquement des écarts d'optimalité plus faibles, avec une moyenne de 2.87%, contre 4.78% pour les bornes issues de la décomposition de Benders exacte. Cela confirme la capacité de la méthode par ED à produire des solutions de haute qualité dans

un temps de calcul raisonnable. L'ajout de l'écart-type σ et de l'intervalle de confiance (IC) à 95% permet d'évaluer la stabilité statistique de l'estimation de la borne inférieure. On observe que les écarts-types restent globalement modérés par rapport aux valeurs des bornes, ce qui indique une variance contrôlée de l'estimateur.

Les écarts associés à l'intervalle de confiance (colonne IC) sont logiquement plus élevés que ceux de la méthode par ED seule, puisqu'ils intègrent l'incertitude statistique liée à l'échantillonnage. Néanmoins, ils restent nettement inférieurs aux écarts rapportés dans la littérature, ce qui montre que même en tenant compte de cette incertitude, la méthode par ED demeure compétitive.

On observe également que les bornes supérieures obtenues par EG sont très largement dégradées par rapport à celles de la méthode par ED, avec des écarts pouvant devenir très importants sur certaines instances. Cela indique que la réduction du réseau par échantillonnage d'arêtes n'est pas adaptée dans ce contexte, car elle détériore fortement la qualité des solutions réalisables.

Le temps de calcul de la méthode par ED augmente naturellement avec la taille des instances : les petites instances (48–51 nœuds) sont traitées en environ 500 secondes, tandis que les instances de taille intermédiaire (76–101 nœuds) nécessitent entre 2300 et 2900 secondes, avec une augmentation plus marquée pour certaines instances (jusqu'à environ 3700 secondes). Enfin, la méthode par ED produit des bornes supérieures systématiquement égales ou meilleures que celles de la littérature, en exploitant le fait que tout cycle hamiltonien constitue une solution réalisable. Elle reste stable, avec des écarts inférieurs à 7% même pour les instances les plus grandes, tout en offrant une garantie statistique sur la qualité des bornes inférieures.

3.7.5 Résultats expérimentaux sur le MUFND

Pour les expériences computationnelles, nous avons testé la méthode par ED sur les célèbres instances Canad (voir Crainic, Frangioni & Gendron (2001)).

Nous effectuons tout d'abord une analyse de sensibilité sur les deux instances les plus difficiles de la classe IV de Zetina *et al.* (2019). Les tableaux 3.3 et 3.4 présentent les résultats de la

Tableau 3.3 Analyse de sensibilité : performance de la méthode par ED en fonction de la taille d'échantillon sur l'instance Classe IV (50,1500,1000)

Taille d'échantillon n	**BI (ED)**	**BS (ED)**	**Écart ED (%)**	**Temps (s)**
200	179 228	197 949	10.45	1 377
400	185 914	195 901	5.37	2 117
600	189 932	194 577	2.45	3 094
700	190 741	192 769	1.06	3 870
800	190 941	192 473	0.80	5 611
900	191 161	192 254	0.57	7 505
Décomposition de Benders exacte (10800s) : BI=189 325, BS=191 581, Écart=1.19%				

Tableau 3.4 Analyse de sensibilité : performance de la méthode par ED en fonction de la taille d'échantillon sur l'instance Classe IV (50,1500,1500)

Taille d'échantillon n	**BI (ED)**	**BS (ED)**	**Écart ED (%)**	**Temps (s)**
200	259 648	287 625	10.77	1 449
400	267 911	281 485	5.06	1 803
600	271 039	280 271	3.44	2 462
700	272 384	280 483	2.97	2 885
1000	274 919	278 250	1.21	5 450
1100	275 860	277 701	0.67	6 950
Décomposition de Benders exacte (10800s) : BI=275 122, BS=277 437, Écart=0.83%				

méthode par ED pour différentes tailles d'échantillon n (de 200 à 1100 demandes, représentant 13% à 73% de l'ensemble total des demandes).

Les Tableaux 3.3 et 3.4 se lisent comme suit. La colonne *Taille d'échantillon n* indique le nombre de demandes tirées à chaque réplication. Les deux colonnes suivantes rapportent les bornes produites par la méthode par ED : la borne inférieure *BI (ED)*, obtenue à partir des valeurs optimales des problèmes échantillonnés, et la borne supérieure *BS (ED)*, obtenue après réparation des solutions sur l'ensemble complet des demandes. La colonne *Écart ED (%)* donne l'écart relatif entre ces deux bornes, exprimé en pourcentage. La colonne *Temps (s)* reporte le temps total de calcul, en secondes, incluant la résolution des réplications et la phase de réparation. La ligne finale rappelle, pour comparaison, la borne inférieure, la borne supérieure et l'écart obtenus par la décomposition de Benders exacte à l'issue d'une limite de temps de trois heures.

Comme attendu, l'augmentation de la taille d'échantillon améliore constamment la qualité de la solution : l'écart d'optimalité diminue d'approximativement 10% à $n = 200$ à moins de 1% à $n = 900$–1100. Cependant, ces instances se sont avérées insuffisamment difficiles pour démontrer les avantages computationnels de la méthode par ED par rapport aux méthodes exactes. La décomposition de Benders exacte atteint des écarts de 1.19% et 0.83% respectivement dans la limite de 3 heures, suggérant que ces instances restent traitables pour les solveurs à l'état de l'art.

Cette observation nous a motivés à introduire une nouvelle classe d'instances, que nous désignons comme Classe V, spécifiquement conçue pour modéliser des problèmes significativement plus difficiles. Puisque de telles instances ne sont pas disponibles dans la littérature existante, nous les générons à l'aide du générateur Mulgen (voir Crainic *et al.* (2001)). Ces instances comportent jusqu'à 140 nœuds, 4500 arcs et 5000 demandes. Elles visent explicitement à repousser les limites des méthodes exactes comme la décomposition de Benders, lesquelles peinent à produire des solutions réalisables dans des délais raisonnables à cette échelle.

Les résultats sont rapportés sur deux tableaux. Le Tableau 3.5 rassemble les bornes inférieures et supérieures brutes obtenues par chaque méthode, et le Tableau 3.6 en synthétise les écarts d'optimalité, les temps de calcul et l'amélioration relative de la borne inférieure apportée par la méthode par ED.

Le Tableau 3.5 se lit comme suit. La colonne *Instance* identifie chaque instance par le triplet $(n, |A|, |K|)$ donnant respectivement le nombre de nœuds, d'arcs et de demandes. Toutes les bornes sont exprimées en multiples de 10^9. Les quatre colonnes de la borne inférieure sont : *ED*, valeur produite par la méthode par échantillonnage de la demande ; σ, son écart-type ; *IC*, la borne inférieure prenant en compte l'intervalle de confiance à 95 % associé ; et *Benders*, la borne inférieure obtenue par la décomposition de Benders exacte, suivie d'un * lorsque l'optimum global est atteint et d'un † lorsque la limite de temps est atteinte. Les trois colonnes de la borne supérieure rapportent la meilleure solution réalisable produite par la méthode par *ED*, par échantillonnage de graphe (*EG*), et par décomposition de *Benders* (notée – lorsque celle-ci n'a trouvé aucune solution réalisable dans le temps imparti).

Tableau 3.5 Classe V — Bornes inférieures et supérieures obtenues par chaque méthode

Instance	**Borne inférieure** ($\times 10^9$)				**Borne supérieure** ($\times 10^9$)		
$(n, \lvert A\rvert, \lvert K\rvert)$	ED	σ	IC	Benders	ED	EG	Benders
80, 2000, 2500	17.59	1.07	16.26	18.26*	18.26	30.56	18.26*
90, 2500, 2500	23.73	1.14	22.31	24.75*	24.75	46.94	24.75*
100, 2500, 3000	**23.80**	1.22	22.28	22.63†	24.95	58.62	–†
120, 3500, 4000	**31.66**	1.47	29.83	24.77†	33.73	78.53	–†
140, 4500, 5000	**41.05**	1.57	39.10	25.15†	47.14	98.52	–†

* Instance résolue à l'optimalité par la décomposition de Benders.
† Limite de temps atteinte ; Benders n'a produit aucune solution réalisable.

Le Tableau 3.6 se lit comme suit. Les deux colonnes d'*Écart* reportent l'écart d'optimalité de la méthode par ED, calculé entre la borne supérieure ED et, respectivement, la moyenne de l'estimateur (*ED*) et la borne infériure avec l'intervalle de confiance à 95 % (*IC*). Cette seconde variante constitue une estimation intégrant l'incertitude statistique liée à l'échantillonnage. Les trois colonnes de *Temps* donnent le temps de calcul (en secondes) des trois méthodes ; la notation >10800 signale les instances pour lesquelles la décomposition de Benders a atteint la limite de temps de trois heures. La dernière colonne, *Amélioration BI*, rapporte l'amélioration relative de la borne inférieure ED par rapport à celle de Benders : une valeur positive signifie que la borne ED est meilleure, tandis qu'une valeur négative correspond aux cas où Benders a atteint l'optimum exact, situations dans lesquelles l'estimateur reste, par construction, légèrement en inférieur.

Tableau 3.6 Classe V — Écarts d'optimalité, temps de calcul et amélioration de la borne inférieure

Instance	**Écart (%)**		**Temps (s)**			**Amélioration BI**
$(n, \lvert A\rvert, \lvert K\rvert)$	ED	IC	ED	EG	Benders	(%)
80, 2000, 2500	3.68	10.95	**849**	2849	2706	−3.67
90, 2500, 2500	4.12	9.86	**922**	3189	3208	−4.12
100, 2500, 3000	4.63	10.70	**1264**	3745	>10800	5.17
120, 3500, 4000	6.12	11.56	**1823**	5685	>10800	27.82
140, 4500, 5000	12.91	17.06	**2251**	9814	>10800	63.22

Les Tableaux 3.5 et 3.6 présentent les résultats obtenus sur les instances de la Classe V. On distingue deux cas.

Pour les instances de petite taille (80–90 nœuds), la décomposition de Benders exacte résout le problème à l'optimalité. Dans ce cas, la méthode par ED trouve également la solution optimale tout en réduisant significativement le temps de calcul (de 2706 à 849 secondes pour 80 nœuds, et de 3208 à 922 secondes pour 90 nœuds).

Il est important de préciser la signification des valeurs négatives (−3.67% et −4.12%) observées dans la colonne Amélioration BI pour ces deux premières instances. Ces valeurs s'expliquent par le fait que Benders parvient à résoudre ces instances spécifiques à l'optimalité. La borne inférieure de la méthode par ED étant issue d'une approximation par échantillonnage, elle est naturellement légèrement inférieure à l'optimum exact trouvé par Benders. En d'autres termes, l'amélioration est ici négative non pas parce que la méthode par ED est moins performante sur le fond, mais parce que la comparaison se fait face à une valeur exacte.

Pour les instances plus grandes (100–140 nœuds), la méthode exacte atteint la limite de temps sans produire de solution réalisable, fournissant uniquement des bornes inférieures. En revanche, la méthode par ED fournit systématiquement des bornes supérieures réalisables ainsi que des bornes inférieures améliorées par rapport à la méthode exacte. Par exemple, pour l'instance à 100 nœuds, la méthode par ED améliore la borne inférieure de Benders de 5.17% et obtient un écart d'optimalité de 4.63% basé sur l'estimateur. En prenant en compte l'intervalle de confiance, cet écart peut atteindre 10.70%, ce qui reflète l'incertitude statistique associée à l'estimation par échantillonnage. Cet effet se retrouve sur les instances de plus grande taille, avec des améliorations des bornes inférieures de 27.82% et 63.22% pour les instances à 120 et 140 nœuds respectivement. Les intervalles de confiance associés montrent également une augmentation des écarts possibles (11.56% et 17.06%), ce qui traduit une variabilité plus forte de l'estimateur lorsque la taille du problème augmente.

Par ailleurs, la comparaison avec l'approche EG met en évidence une dégradation significative de la qualité des bornes supérieures. Pour toutes les instances, les solutions obtenues par EG

sont nettement moins bonnes que celles issues de la méthode par ED (58.62 contre 24.95 pour 100 nœuds, et 98.52 contre 47.14 pour 140 nœuds). Ce comportement confirme les observations faites sur le TSP-GL, où l'échantillonnage direct sur le graphe tend également à produire des solutions de qualité inférieure, une limite qui s'accentue à mesure que la taille des instances augmente.

Dans l'ensemble, ces résultats montrent que la méthode par ED reste performante sur des instances de grande taille, pour lesquelles les méthodes exactes peinent à produire des solutions réalisables dans le temps imparti. En particulier, elle permet d'obtenir des solutions réalisables de bonne qualité ainsi que des bornes informatives, constituant ainsi une alternative pratique lorsque les approches exactes deviennent difficiles à appliquer.

3.8 Conclusion

L'objectif de ce chapitre était de proposer une approche permettant de traiter des problèmes de conception de réseau à deux niveaux comportant un grand nombre de demandes, en s'affranchissant des limites de passage à l'échelle des méthodes exactes. Plutôt que de modifier la structure du réseau, la démarche retenue consiste à agir directement sur l'ensemble des demandes, en introduisant un mécanisme d'échantillonnage repondéré de l'ensemble des demandes.

Cette approche repose sur l'idée que la complexité des problèmes considérés ne provient pas uniquement du grand nombre de demandes, mais surtout de l'interaction entre les décisions d'infrastructure et le routage de ces demandes. Le nombre élevé de demandes amplifie cette difficulté en augmentant fortement la taille et la complexité du problème de routage associé. En réduisant cette dimension via un échantillonnage, il devient alors possible de préserver la structure décisionnelle du problème tout en diminuant significativement sa taille effective.

3.8.1 Interprétation des résultats

Les résultats obtenus mettent en évidence deux comportements distincts selon la taille et la structure des instances considérées.

Dans le cas du TSP-GL, la méthode par ED permet systématiquement d'obtenir des solutions de bonne qualité, avec des encadrements des solutions optimales inférieurs à ceux fournis par la décomposition de Benders exacte dans le temps imparti. La méthode proposée permet donc non seulement de réduire les temps de calcul, mais aussi d'améliorer la qualité des bornes disponibles dans un cadre de temps limité. Il est important de préciser que le TSP-GL est un problème en général plus difficile à résoudre en comparaison avec le MUFND, dû notamment à la contrainte forçant le réseau à être un cycle hamiltonien.

Dans le cas du MUFND, les résultats mettent en évidence une transition nette entre des instances de taille modérée, pour lesquelles les méthodes exactes restent compétitives, et des instances de grande taille, pour lesquelles elles deviennent inopérantes. Sur les instances les plus grandes, la décomposition de Benders ne produit aucune solution faisable, tandis que la méthode par ED est capable de fournir simultanément des bornes inférieures et supérieures, permettant ainsi de certifier un écart à l'optimalité. Ce point est central : l'intérêt de la méthode par ED ne réside pas uniquement dans l'amélioration des performances, mais dans sa capacité à produire de l'information exploitable là où les méthodes exactes échouent complètement.

Par ailleurs, l'analyse de sensibilité sur la taille des échantillons montre un comportement attendu mais important : l'augmentation de la taille de l'échantillon améliore progressivement la qualité des solutions, au prix d'un coût de calcul plus élevé. Ce compromis met en évidence le rôle de la taille d'échantillon comme paramètre de contrôle direct entre précision et temps de calcul.

L'échec de l'approche EG s'explique par une raison fondamentale qui tient à la nature même de la restriction qu'elle impose. En échantillonnant un sous-ensemble d'arêtes $E' \subseteq E$, cette méthode réduit l'espace des solutions réalisables à $X' \subseteq X$, excluant ainsi potentiellement la solution optimale x^* du problème original. En effet, si x^* utilise des arêtes absentes de E', elle devient irréalisable dans le sous-graphe considéré, et aucune réplication ne pourra la retrouver. La probabilité d'éliminer x^* croît avec le taux de sous-échantillonnage : à 10% des arêtes, la quasi-totalité des solutions de bonne qualité sont structurellement inaccessibles. En revanche, la méthode par ED ne restreint jamais l'espace des décisions de premier niveau : pour

toute réplication m, l'ensemble X reste intact, et toute solution $x \in X$ (y compris x^*) demeure réalisable. C'est cette propriété qui garantit la validité théorique des bornes obtenues et explique la supériorité empirique systématiquement observée de la méthode par ED sur l'approche EG, tant sur le TSP-GL que sur le MUFND.

3.8.2 Discussion des hypothèses

L'hypothèse principale sous-jacente à la méthode par ED est que, bien que le problème résulte d'un couplage entre décisions d'infrastructure et de routage, l'information pertinente pour ces décisions peut être capturée à partir d'un sous-ensemble de demandes. Autrement dit, il est supposé qu'un échantillon de taille réduite permet d'approcher de manière fidèle les caractéristiques essentielles influençant la structure de la solution. Les résultats obtenus confirment largement cette hypothèse dans les cas étudiés.

Cependant, cette hypothèse implique également que l'échantillon soit suffisamment représentatif de l'ensemble des demandes. Si cette condition est vérifiée empiriquement dans les instances testées, elle n'est pas garantie en général. En particulier, des demandes ayant un impact structurel important sur la solution pourraient ne pas être sélectionnées dans un échantillon de taille réduite, ce qui pourrait dégrader la qualité des solutions obtenues.

Ce point est partiellement atténué par le cadre SAA sous-jacent, qui repose sur des réplications indépendantes et permet de construire des bornes statistiques. Néanmoins, il souligne l'importance du schéma d'échantillonnage et de ses propriétés.

3.8.3 Apport de l'approche

L'un des apports principaux de ce travail est de proposer une alternative conceptuelle aux approches existantes. Alors que les méthodes existantes cherchent à simplifier le réseau ou à renforcer les formulations, la méthode par ED consiste à conserver le modèle intact et à réduire la taille effective du problème via l'échantillonnage des demandes.

Cette stratégie présente un avantage important : elle est générique et s'applique à toute classe de problèmes présentant la structure en deux niveaux décrite précédemment. De plus, elle s'intègre naturellement dans des schémas existants, notamment les approches de type décomposition, sans nécessiter de modification profonde des formulations.

Enfin, le recours au cadre SAA comme fondement théorique permet d'associer à la méthode par ED des garanties formelles, en particulier en termes de bornes inférieures et supérieures et de convergence, ce qui la distingue de nombreuses heuristiques.

3.8.4 Limites et pistes de réflexion

Plusieurs limites doivent néanmoins être soulignées. Tout d'abord, les performances de la méthode par ED dépendent directement du choix des paramètres, en particulier la taille de l'échantillon et le nombre de réplications.

Ensuite, bien que le cadre SAA fournisse des garanties en espérance, les résultats obtenus restent dépendants des réalisations de l'échantillonnage. Par ailleurs, la méthode par ED repose sur un schéma d'échantillonnage uniforme des demandes. Si ce choix permet de préserver la simplicité et les propriétés théoriques du modèle, il peut s'avérer sous-optimal dans des contextes où certaines demandes jouent un rôle plus structurant que d'autres.

Enfin, dans le cas du MUFND, la nécessité de résoudre un problème de réparation pour obtenir une solution faisable sur l'ensemble complet des demandes constitue un coût supplémentaire, qui peut limiter le gain global de temps de calcul.

Une première piste de réflexion prometteuse consisterait à développer des approches algorithmiques capables d'identifier et d'ajuster dynamiquement les paramètres d'échantillonnage pour une instance donnée. Au lieu d'imposer une taille d'échantillon statique, il serait pertinent de s'inspirer des méthodes d'échantillonnage séquentiel. Par exemple, les travaux de Bayraksan & Morton (2011) démontrent l'efficacité des procédures adaptatives où la taille de l'échantillon croît progressivement, suivant l'évolution des bornes obtenues à chaque itération.

En complément de cette dimension purement séquentielle, l'identification de ces paramètres doit intégrer une gestion intelligente de l'effort de résolution. Il serait ainsi possible d'exploiter la littérature traitant de l'allocation optimale du budget de calcul (voir par exemple Royset & Szechtman (2013)), afin de trouver le meilleur compromis entre la taille des échantillons (n) et le nombre de réplications indépendantes (M). Enfin, pour maximiser l'impact de chaque scénario généré, ces procédures algorithmiques gagneraient à être couplées avec des techniques d'amélioration qualitative de l'échantillon, telles que le regroupement intelligent (clustering) (voir Emelogu, Chowdhury, Marufuzzaman, Bian & Eksioglu (2016)), formant ainsi un cadre par ED auto-adaptatif, robuste et performant.

Dans une perspective plus récente, il serait également pertinent d'explorer des approches d'échantillonnage guidées par les données (*data-driven sampling*). L'idée serait d'apprendre, à partir des solutions obtenues lors des premières réplications, quelles demandes ont le plus d'impact sur la structure de la solution, et d'adapter dynamiquement la distribution d'échantillonnage en conséquence.

Ce type d'approche s'inscrit dans le cadre plus large de l'optimisation assistée par apprentissage, aussi appelé (*learning-augmented optimization*), qui connaît un développement rapide (voir Bengio, Lodi & Prouvost (2021)).

CONCLUSION ET RECOMMANDATIONS

Les problèmes d'optimisation combinatoire à deux niveaux, qui modélisent simultanément la conception d'une infrastructure et le routage d'un grand nombre de demandes, constituent un défi majeur en recherche opérationnelle. La forte interaction entre les décisions de conception et celles de routage engendre une croissance rapide de la complexité avec la taille des instances et met en échec les méthodes exactes traditionnelles dès que le nombre de demandes devient important. Ce mémoire s'est attaché à apporter des éléments de réponse à ce défi en explorant deux voies de résolution distinctes mais complémentaires.

La première voie a consisté à enrichir les méthodes exactes en exploitant la structure entière des solutions de relaxation linéaire. À travers l'étude du problème du voyageur de commerce avec latence généralisée (TSP-GL), nous avons proposé deux familles de bornes inférieures s'appuyant sur la dispersion du flot dans la relaxation. Si les gains expérimentaux obtenus sont modérés, ces travaux valident l'idée selon laquelle la dispersion fractionnaire du flot peut être contrainte intelligemment sans exclure de solutions optimales, et fournissent un outil théorique additionnel pour renforcer les méthodes exactes sur des instances de taille modérée.

Toutefois, ces techniques de renforcement n'atteignent pas l'objectif initial : traiter des instances de grande taille. La barrière du passage à l'échelle nous a conduits à un changement de paradigme. Plutôt que d'agir sur la topologie du réseau, approche qui s'est avérée inefficace dans notre cas, nous avons proposé de réduire la dimension effective du problème en agissant directement sur l'ensemble des demandes. Cette idée constitue la contribution principale du mémoire et donne naissance à une nouvelle méthode, que nous avons nommée approche par échantillonnage de la demande (ED).

L'originalité de cette méthode tient à son positionnement conceptuel. Partant de problèmes purement déterministes, nous avons montré qu'il était possible de les réinterpréter comme des problèmes stochastiques en interprétant les demandes comme des réalisations de scénarios

aléatoires. Cette réinterprétation permet l'utilisation des outils théoriques de la programmation stochastique. Nous avons en particulier mobilisé le cadre du *Sample Average Approximation* (SAA), pour doter notre approche de garanties statistiques formelles, à savoir des bornes inférieures et supérieures valides. La méthode par ED ne se réduit donc pas à une simple application du SAA ; elle constitue un pont entre l'optimisation combinatoire déterministe et la programmation stochastique, exploitant les propriétés du SAA.

L'approche a été appliquée sur deux problèmes, le TSP-GL et le MUFND, et évaluée sur des instances de la littérature ainsi que sur une nouvelle classe d'instances de grande taille spécifiquement construite pour mettre en difficulté les méthodes exactes. Les résultats numériques confirment l'intérêt pratique de la méthode : sur des instances MUFND atteignant 140 nœuds, 4500 arcs et 5000 demandes, là où la décomposition de Benders exacte échoue à produire la moindre solution réalisable dans le temps imparti, la méthode par ED fournit simultanément des bornes inférieures statistiquement fiables et des bornes supérieures de bonne qualité, permettant ainsi d'obtenir des écarts d'optimalité.

Les perspectives ouvertes par ce travail sont nombreuses. À court terme, une direction naturelle consiste à hybrider les deux axes du mémoire : les techniques de renforcement de bornes inférieures développées dans la première partie pourraient être intégrées au sein des sous-problèmes résolus à chaque réplication de la méthode par ED, afin d'améliorer la qualité des relaxations sous-jacentes et de réduire le temps de convergence de chaque réplication. Une telle hybridation tirerait parti des forces complémentaires des deux approches : la capacité de la méthode par ED à produire rapidement des bornes statistiques sur des instances de grande taille, et les mécanismes de renforcement issus des méthodes exactes pour accélérer l'obtention des résultats.

À plus long terme, une extension naturelle consiste à doter la méthode par ED de schémas d'échantillonnage adaptatifs, dans lesquels la taille des échantillons et la distribution sous-

jacente seraient ajustées dynamiquement en fonction de l'information accumulée au fil des réplications. Une voie particulièrement prometteuse consiste à coupler ce cadre avec des techniques d'apprentissage automatique afin d'identifier, au sein de l'ensemble des demandes, celles qui sont les plus structurantes vis-à-vis de la décision de premier niveau, et de concentrer l'effort de calcul sur ces dernières. Ces directions s'inscrivent dans le développement plus large de l'optimisation assistée par apprentissage et offrent un horizon de recherche riche pour les travaux à venir.

LISTE DE RÉFÉRENCES

Adulyasak, Y., Cordeau, J.-F. & Jans, R. (2015). Benders Decomposition for Production Routing Under Demand Uncertainty. *Operations Research*, 63(4), 851-867. doi : 10.1287/opre.2015.1401. [43]

Applegate, D. L., Bixby, R. E., Chvátal, V. & Cook, W. J. (2006). *The Traveling Salesman Problem : A Computational Study*. Princeton, NJ : Princeton University Press. doi : 10.5860/choice.45-0928. [14]

Bayraksan, G. & Morton, D. P. (2011). A Sequential Sampling Procedure for Stochastic Programming. *Operations Research*, 59(4), 898-913. doi : 10.1287/opre.1110.0926. [106]

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1), 238–252. doi : 10.1007/BF01386316. [18]

Bengio, Y., Lodi, A. & Prouvost, A. (2021). Machine learning for combinatorial optimization : A methodological tour d'horizon. *European Journal of Operational Research*, 290(2), 405-421. doi : https://doi.org/10.1016/j.ejor.2020.07.063. [107]

Bertsimas, D. & Tsitsiklis, J. N. (1997). *Introduction to Linear Optimization*. Belmont, MA : Athena Scientific. [20]

Birge, J. R. & Louveaux, F. (2011). *Introduction to Stochastic Programming* (éd. 2). Springer. doi : 10.1007/978-1-4614-0237-4. [41]

Chvátal, V. (1973). Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete Mathematics*, 4(4), 305-337. doi : https://doi.org/10.1016/0012-365X(73)90167-2. [12]

Costa, A. M. (2005). A survey on benders decomposition applied to fixed-charge network design problems. *Computers Operations Research*, 32(6), 1429-1450. doi : https://doi.org/10.1016/j.cor.2003.11.012. [18]

Crainic, T. G., Frangioni, A. & Gendron, B. (2001). Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics*, 112(1), 73-99. doi : https://doi.org/10.1016/S0166-218X(00)00310-3. Combinatorial Optimization Symposium, Selected Papers. [98, 100]

Dakin, R. J. (1965). A tree-search algorithm for mixed integer programming problems. *The Computer Journal*, 8(3), 250-255. doi : 10.1093/comjnl/8.3.250. [7]

Dantzig, G. B. & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, 6(1), 80–91. doi : 10.1287/mnsc.6.1.80. [14]

Dantzig, G., Fulkerson, R. & Johnson, S. (1954). Solution of a Large-Scale Traveling-Salesman Problem. *Journal of the Operations Research Society of America*, 2(4), 393–410. doi : 10.1287/opre.2.4.393. [13, 15]

Dantzig, G. B. (1951). Maximization of a Linear Function of Variables Subject to Linear Inequalities. Dans Koopmans, T. C. (Éd.), *Activity Analysis of Production and Allocation* (pp. 339–347). New York and London : Wiley and Chapman & Hall. [6, 8]

Dantzig, G. B. (1955). Linear Programming under Uncertainty. *Management Science*, 1(3/4), 197–206. Repéré à http://www.jstor.org/stable/2627159. [40, 41]

Dantzig, G. B. & Madansky, A. (1960). On the solution of two-stage linear programs under uncertainty. *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1 : Contributions to the Theory of Statistics*, pp. 165–176. [40]

de Mello, T. H. & Bayraksan, G. (2014). Monte Carlo sampling-based methods for stochastic optimization. *Surveys in Operations Research and Management Science*, 19(1), 56-85. doi : https://doi.org/10.1016/j.sorms.2014.05.001. [43]

El Agizy, M. (1967). Two-Stage Programming under Uncertainty with Discrete Distribution Function. *Operations Research*, 15(1), 55-70. doi : 10.1287/opre.15.1.55. [40]

Emelogu, A., Chowdhury, S., Marufuzzaman, M., Bian, L. & Eksioglu, B. (2016). An enhanced sample average approximation method for stochastic optimization. *International Journal of Production Economics*, 182, 230-252. doi : https://doi.org/10.1016/j.ijpe.2016.08.032. [107]

Errico, F. (2008). *The Design of Flexible Transit Systems : Models and Solution Methods*. (Thèse de doctorat, Politecnico di Milano, Milano, Italy). [25]

Errico, F., Crainic, T. G., Malucelli, F. & Nonato, M. (2011). *The design problem for single-line demand adaptive transit systems* (Rapport n°CIRRELT-2011-65). Montréal, Canada. [26]

Errico, F., Crainic, T. G., Malucelli, F. & Nonato, M. (2013). A survey on planning semi-flexible transit systems : Methodological issues and a unifying framework. *Transportation Research Part C : Emerging Technologies*, 36, 324–338. doi : 10.1016/j.trc.2013.08.010. [25]

Errico, F., Crainic, T. G., Malucelli, F. & Nonato, M. (2017). A Benders Decomposition Approach for the Symmetric TSP with Generalized Latency Arising in the Design of Semiflexible Transit Systems. *Transportation Science*, 51(2), 706–722. doi : 10.1287/trsc.2015.0636. [18, 63, 73, 94, 95]

Fontaine, P. & Minner, S. (2018). Benders decomposition for the Hazmat Transport Network Design Problem. *European Journal of Operational Research*, 267(3), 996-1002. doi : 10.1016/j.ejor.2017.12.042. [18]

Fourier, J. (1827). Mémoires de l'Académie des sciences de l'Institut de France. *Mémoires de l'Académie des sciences de l'Institut de France*, 7, xlvi–lv. [6]

Gomory, R. E. (1958). Outline of an Algorithm for Integer Solutions to Linear Programs. *Bulletin of the American Mathematical Society*, 64(5), 275–278. [12]

Gomory, R. E. (1960). *An Algorithm for the Mixed Integer Problem*. Santa Monica, CA. [12]

Gomory, R. E. & Hu, T. (1961). Multi-terminal network flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4), 551–570. doi : 10.1137/0109047. [17]

Gutin, G. & Punnen, A. P. (2002). *The Traveling Salesman Problem and Its Variations*. New York : Springer Science and Business Media. [32]

Held, M. & Karp, R. M. (1970). The Traveling-Salesman Problem and Minimum Spanning Trees. *Operations Research*, 18(6), 1138-1162. doi : 10.1287/opre.18.6.1138. [14]

Hjorring, C. & Holt, J. (1999). New optimality cuts for a single-vehicle stochastic routing problem. *Annals of Operations Research*, 86, 569–584. doi : 10.1023/A:1018995927636. [51, 60]

Jabali, O., Rei, W., Gendreau, M. & Laporte, G. (2014). Partial-route inequalities for the multi-vehicle routing problem with stochastic demands. *Discrete Applied Mathematics*, 177, 121–136. doi : 10.1016/j.dam.2014.05.040. [51]

Kantorovich, L. V. (1939). The Mathematical Method of Production Planning and Organization. *Management Science*, 6(4), 363–422. [6]

Kleywegt, A. J., Shapiro, A. & Homem-de Mello, T. (2002). The Sample Average Approximation Method for Stochastic Discrete Optimization. *SIAM Journal on Optimization*, 12(2), 479-502. doi : 10.1137/S1052623499363220. [41, 45]

Land, A. H. & Doig, A. G. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, 28(3), 497–520. doi : 10.2307/1910129. [7]

Laporte, G. & Louveaux, F. V. (1993). The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3), 133-142. doi : https://doi.org/10.1016/0167-6377(93)90002-X. [51, 57, 60]

Linderoth, J., Shapiro, A. & Wright, S. J. (2006). The Empirical Behavior of Sampling Methods for Stochastic Programming. *Annals of Operations Research*, 142(1), 215–241. doi : 10.1007/s10479-006-6169-8. [45]

Little, J. D. C., Murty, K. G., Sweeney, D. W. & Karel, C. (1963). An Algorithm for the Traveling Salesman Problem. *Operations Research*, 11(6), 972–989. [7]

Ma, J. & Zavala, V. M. (2022). Solution of large-scale supply chain models using graph sampling & coarsening. 163, 107832. doi : 10.1016/j.compchemeng.2022.107832. [93]

Madansky, A. (1960). Inequalities for Stochastic Linear Programming Problems. *Management Science*, 6(2), 197-204. doi : 10.1287/mnsc.6.2.197. [41]

Magnanti, T. L. & Wong, R. T. (1981). Accelerating Benders Decomposition : Algorithmic Enhancement and Model Selection Criteria. *Operations Research*, 29(3), 464-484. doi : 10.1287/opre.29.3.464. [18, 35, 49]

Magnanti, T. L. & Wong, R. T. (1984). Network Design and Transportation Planning : Models and Algorithms. *Transportation Science*, 18(1), 1-55. doi : 10.1287/trsc.18.1.1. [18, 34]

Mak, W.-K., Morton, D. P. & Wood, R. (1999). Monte Carlo bounding techniques for determining solution quality in stochastic programs. *Operations Research Letters*, 24(1), 47-56. doi : https://doi.org/10.1016/S0167-6377(98)00054-6. [44, 86]

McDaniel, D. & Devine, M. (1977). A Modified Benders' Partition Algorithm for Mixed Integer Programming. *Management Science*, 24(3), 312–319. doi : 10.1287/mnsc.24.3.312. [32]

Miller, C. E., Tucker, A. W. & Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4), 326–329. doi : 10.1145/321043.321046. [15]

Motzkin, T. S. (1952). *The Theory of Linear Inequalities*. Rand Corporation. [6]

Padberg, M. & Rinaldi, G. (1990). An efficient algorithm for the minimum capacity cut problem. *Mathematical Programming*, 47(1-3), 19–36. doi : 10.1007/bf01580850. [17]

Padberg, M. & Rinaldi, G. (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review*, 33(1), 60–100. doi : 10.1137/1033004. [13]

Rahmaniani, R., Crainic, T. G., Gendreau, M. & Rei, W. (2017). The Benders decomposition algorithm : A literature review. *European Journal of Operational Research*, 259(3), 801-817. doi : https://doi.org/10.1016/j.ejor.2016.12.005. [18]

Rahmaniani, R., Ahmed, S., Crainic, T. G., Gendreau, M. & Rei, W. (2020). The Benders Dual Decomposition Method. *Operations Research*, 68(3), 878-895. doi : 10.1287/opre.2019.1892. [18]

Royset, J. O. & Szechtman, R. (2013). Optimal Budget Allocation for Sample Average Approximation. *Operations Research*, 61(3), 762-776. doi : 10.1287/opre.2013.1163. [107]

Shapiro, A. (1996). Simulation-based optimization—convergence analysis and statistical inference. *Communications in Statistics. Stochastic Models*, 12(3), 425–454. doi : 10.1080/15326349608807393. [41]

Shapiro, A. & Homem-de Mello, T. (2000). On the Rate of Convergence of Optimal Solutions of Monte Carlo Approximations of Stochastic Programs. *SIAM Journal on Optimization*, 11(1), 70-86. doi : 10.1137/S1052623498349541. [44]

Shapiro, A., Dentcheva, D. & Ruszczynski, A. (2021). *Lectures on Stochastic Programming : Modeling and Theory, Third Edition*. Philadelphia, PA : Society for Industrial and Applied Mathematics. doi : 10.1137/1.9781611976595. [86]

Shapiro, A. (2003). Monte Carlo sampling approach to stochastic programming. *ESAIM : Proc.*, 13, 65-73. doi : 10.1051/proc:2003003. [94]

Taherkhani, G., Alumur, S. A. & Hosseini, M. (2020). Benders Decomposition for the Profit Maximizing Capacitated Hub Location Problem with Multiple Demand Classes. *Transportation Science*, 54(6), 1446-1470. doi : 10.1287/trsc.2020.1003. [43]

Williams, H. P. (1986). Fourier's Method of Linear Programming and its Dual. *The American Mathematical Monthly*, 93(9), 681–695. doi : 10.2307/2322281. [6]

Zetina, C. A., Contreras, I. & Cordeau, J.-F. (2019). Exact algorithms based on Benders decomposition for multicommodity uncapacitated fixed-charge network design. 111, 311–324. doi : 10.1016/j.cor.2019.07.007. [34, 35, 98]

Özgürbüz, E., Bektaş, T. & Çağatay Iris. (2026). Tailored Benders decomposition for two-stage distributionally robust combinatorial optimisation. *European Journal of Operational Research*, 333(1), 53-66. doi : https://doi.org/10.1016/j.ejor.2026.01.048. [43]