Titre: Title:	Génération de colonnes pour le balayage printanier des rues
Auteur: Author:	Fairouz Mansouri
Date:	2025
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Mansouri, F. (2025). Génération de colonnes pour le balayage printanier des rues [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/65785/

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Génération de colonnes pour le balayage printanier des rues

FAIROUZ MANSOURI
Département de mathématiques et de génie industriel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Mathématiques appliquées

Avril 2025

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Génération de colonnes pour le balayage printanier des rues

présenté par **Fairouz MANSOURI**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Antoine LEGRAIN, président
Issmaïl EL HALLAOUI, membre et directeur de recherche
Amina LAMGHARI, membre et codirectrice de recherche
Chahid AHABCHANE, membre externe

DÉDICACE

À la mémoire de mon grand-père bien-aimé, Rachid AZIRI, qui nous a quittés, mais dont la sagesse, la bonté et la présence silencieuse m'accompagnent encore à chaque pas. Ce travail est aussi un hommage à ce qu'il m'a transmis.

À mes grands-parents encore en vie, source d'amour, de courage et de repères précieux.

À mes parents, pour qui les kilomètres n'ont jamais été une barrière. Même à distance, vous ne m'avez jamais laissée seule. Votre présence constante, vos appels, vos prières et votre foi en moi m'ont portée jusqu'ici. J'ai étudié loin de vous, mais je vous ai sentis proches à chaque étape. Merci pour tout.

À toute ma famille, MANSOURI et AZIRI, pour leur soutien, leur affection et leur fierté partagée.

À Monsieur le Professeur YAGOUNI Mohammd, pour tout ce qu'il m'a transmis, pour avoir cru en moi, et pour m'avoir donné la force et l'envie d'aller toujours plus loin. Je lui dois une grande partie de ce travail.

Et enfin, à Christine et Marie-Carline, pour leur gentillesse, leur écoute et leur soutien sincère durant ce parcours.

REMERCIEMENTS

Après une période difficile où j'avais presque perdu l'espoir d'une seconde chance de réussir, la lumière est venue grâce à Madame Marie-Carline LAJEUNESSE, Amina LAMGHARI, ma codirectrice, et Issmaïl EL HALLAOUI, mon directeur de recherche, qui ont su m'ouvrir à nouveau la voie du succès. Ma première pensée de gratitude leur est dédiée, et aucune expression ne saurait suffire pour exprimer ma reconnaissance infinie.

Un remerciement particulier à mon directeur Issmaïl EL HALLAOUI et à ma codirectrice Amina LAMGHARI, pour leur confiance en mes compétences, leur encouragement constant, et leur accompagnement généreux. Leur soutien indéfectible et leur aide précieuse, avec tous les moyens possibles, ont été des piliers essentiels de mon parcours.

Je tiens également à exprimer ma profonde reconnaissance à Adil Tahir, pour sa disponibilité, sa patience et ses conseils éclairés, ainsi qu'à Monsieur Ahmed Beljadid pour son aide précieuse et son appui.

Je remercie Antoine LEGRAIN, pour avoir accepté la présidence de mon jury, et Chahid AHABCHANE, qui a accepté d'en être membre.

Mes sincères remerciements vont également à toute l'équipe du laboratoire GERAD pour leur soutien et leur présence bienveillante, ainsi qu'à toute l'équipe administrative du Département de mathématiques et de génie industriel (MAGI) pour leur accompagnement et leur disponibilité tout au long de ce parcours.

Enfin, et surtout, je rends hommage aux personnes les plus chères à mon cœur : mon père MANSOURI Farid, ma mère, mon frère et mes sœurs pour leur encouragement inlassable, leur amour et leur foi en moi. Merci pour ta lumière et ta présence, même à distance, Zoubir. Une mention spéciale à Christine, qui a été à mes côtés à chaque instant, dans les moments de joie comme dans ceux de doute.

RÉSUMÉ

La fin de la saison hivernale marque le lancement du grand ménage printanier, durant lequel des milliers de kilomètres de routes doivent être nettoyées pour éliminer les résidus laissés par l'hiver, notamment les abrasifs et le sable. Ces opérations sont essentielles pour garantir la sécurité et la propreté des infrastructures routières. Réalisées dans un laps de temps très court, elles représentent un défi logistique important pour les gestionnaires chargés de leur planification. Face à ce défi, se pose un problème crucial : comment organiser le balayage des rues à l'aide d'une flotte de véhicules et de plusieurs dépôts, tout en minimisant les coûts et les déplacements à vide ? Ce problème, connu sous le nom de **problème de tournée de véhicules pour le balayage printanier**, vise à planifier des itinéraires permettant de couvrir l'ensemble du réseau routier dans un temps total minimal, en respectant un ensemble de contraintes : la durée maximale des quarts de travail et la continuité entre les quarts.
Deux modélisations ont été proposées dans la littérature : celle par arcs, détaillée mais très coûteuse même sur de petites instances, et celle par nœuds, plus compacte, mais avec une relaxation continue de faible qualité. Cette étude cherche à améliorer cette relaxation pour obtenir de bonnes solutions en un temps raisonnable. Pour cela, une formulation mathématique basée sur les chemins a été développée, où chaque variable représente une route.
Le problème est résolu à l'aide d'une approche de génération de colonnes, reposant sur une décomposition de Dantzig-Wolfe : la formulation par chemins constitue le problème maître, et les sous-problèmes sont modélisés comme des problèmes de plus court chemin avec contraintes de ressources. Des routes initiales, générées par une heuristique inspirée de GRASP (*Greedy Randomized Adaptive Search Procedure*), permettent de stabiliser les variables duales et d'accélérer la convergence. Une heuristique de branchement en profondeur (*Diving Heuristic*) est ensuite appliquée pour obtenir une solution entière. Pour les grandes instances, une stratégie d'agrégation a été utilisée afin de réduire la complexité du problème. Les expérimentations ont été menées sur un ensemble varié d'instances, incluant un cas réel de 174 tâches et 4 dépôts à Victoriaville, Québec, Canada. Les résultats confirment l'efficacité de l'approche, notamment sur une instance de 60 tâches où l'écart d'intégralité a été réduit de 97% à 21% en quelques secondes, et une solution optimale a été atteinte pour une instance de 40 tâches. En moyenne, la borne inférieure obtenue est 20 fois meilleure que celle générée par CPLEX, avec un gain significatif en temps de calcul : 406 secondes, contre 2 heures pour CPLEX.

ABSTRACT

The end of the winter season marks the beginning of the spring street sweeping operations, during which thousands of kilometers of roads must be cleaned to remove winter residues such as abrasives and sand. These operations are essential to ensure the safety and cleanliness of road infrastructure. Carried out in a very short space of time, they represent a major logistical challenge for the managers in charge of planning them. This raises a crucial question: how can street sweeping be efficiently organized using a fleet of vehicles and multiple depots, while minimizing costs and empty trips? This problem, known as the Vehicle Routing Problem for Spring Street Sweeping, aims to design routes that cover the entire road network within a minimal total duration, while satisfying a set of constraints, such as the maximum duration of work shifts and the continuity between shifts. Two modeling formulations have been proposed in the literature: arc-based formulations, which are detailed but computationally expensive even on small instances, and node-based formulations, which are more compact but suffer from weak continuous relaxations. This study seeks to improve the relaxation to obtain high-quality solutions within reasonable computation times. To this end, a path-based mathematical formulation was developed, where each variable represents a route.

The problem is solved using a column generation approach based on a Dantzig-Wolfe decomposition: the path-based formulation defines the master problem, while the subproblems are modeled as resource-constrained shortest path problems (RCSPP). Initial routes, generated by a heuristic inspired by GRASP(*Greedy Randomized Adaptive Search Procedure*) , are used to stabilize the dual variables and accelerate the convergence. A diving heuristic is then applied to obtain an integer solution. For large instances, an aggregation strategy was implemented to reduce the problem complexity.

The experiments were conducted on a diverse set of instances, including a real-world case with 174 tasks and 4 depots in Victoriaville, Québec, Canada. The results confirm the effectiveness of the proposed approach, notably on a 60-task instance where the integrality gap was reduced from 97% to 21% in a matter of seconds, and an optimal solution was obtained for a 40-task instance. On average, the lower bound computed is 20 times more accurate than that generated by CPLEX, with a significant reduction in computation time: 406 seconds compared to 2 hours for CPLEX.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

CHAPITRE 1 INTRODUCTION

Avant d'explorer les différentes branches de notre étude, ce chapitre constitue une entrée essentielle vers la compréhension globale de ce travail en présentant le problème étudié, les motivations qui justifient cette recherche et les objectifs visés. Nous terminerons en esquissant le plan général de ce mémoire.

1.1 Problème étudié et contexte

Chaque hiver, dans les pays froids et enneigés, les conditions climatiques difficiles rendent les routes dangereuses en raison du verglas et de la neige accumulée. Pour assurer la sécurité des conducteurs, des piétons et des autres usagers de la route, il est essentiel de mettre en œuvre des opérations de maintenance hivernale, telles que le déneigement et le déglaçage. Ces interventions doivent être réalisées rapidement pour éviter l'accumulation de neige et de glace sur la chaussée, et ainsi limiter les risques d'accidents. Parmi les pratiques courantes, l'épandage de sable, de gravier ou de sel améliore l'adhérence et renforce la sécurité routière. Cependant, une fois l'hiver terminé, une partie de ces matériaux abrasifs demeure sur les rues. Leur accumulation pose non seulement des risques pour la sécurité des cyclistes et des piétons, mais peut également entraîner des impacts environnementaux tels que la pollution de l'air et de l'eau, ainsi que des effets néfastes sur les écosystèmes locaux et la santé humaine [2]. Pour faire face à ces risques potentiels, les municipalités mettent en place une opération de nettoyage : le balayage printanier des rues. Cette intervention ne vise pas seulement à nettoyer les rues, mais joue aussi un rôle important dans la sécurité routière et la protection de l'environnement. Elle implique des coûts importants et nécessite une organisation rigoureuse. La gestion de cette opération est généralement assurée par les services municipaux, qui doivent planifier le balayage. Avant le début des travaux, plusieurs éléments doivent être planifiés : la négociation des contrats, la détermination des exigences techniques et administratives, et l'élaboration des plans de balayage en fonction des priorités territoriales [1].

- **Processus de balayage :**

  Le balayage printanier est souvent confié à des entreprises privées, responsables d'intervenir sur des rues bien définis, incluant à la fois des routes municipales et des autoroutes situées dans la zone du contrat.

Ce processus repose sur l'utilisation d'une flotte d'équipements organisée comme présenté dans la figure 1.1. Cette flotte comprend en général, deux balayeuses, deux camions-bennes et un véhicule de sécurité qui suit les balayeuses et les camions pour signaler leur présence aux véhicules arrivant par l'arrière.

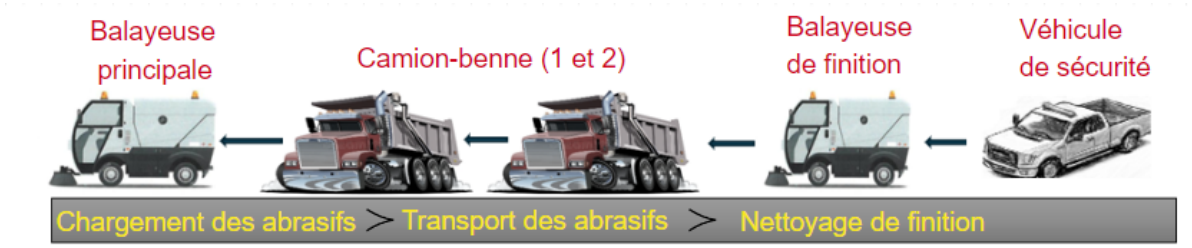


FIGURE 1.1 Organisation de la flotte pour le balayage printanier des rues

La première balayeuse, qui est la balayeuse principale, continue de collecter en continu les abrasifs et les transfère dans un camion-benne. Lorsque celui-ci est plein, il se rend au site de déchargement pour vider son contenu, tandis qu'un second camion prend immédiatement sa place afin de maintenir les opérations sans interruption. La deuxième balayeuse (balayeuse de finition) prend le relais, ramassant les résidus laissés par la première balayeuse, bien que son réservoir se remplisse à un rythme plus lent. L'utilisation simultanée de deux camions garantit que la balayeuse principale reste opérationnelle en permanence, permettant ainsi à la flotte de fonctionner de manière continue.

- **Contraintes opérationnelles du balayage :**

Lors de la planification du plan de balayage, deux contraintes opérationnelles principales doivent être respectées.

1. **Organisation en quarts :**
   Le travail est réparti en quarts (plages horaires) afin de respecter les limites de temps de travail des équipes. Chaque équipe commence et termine son quart dans un dépôt, où les véhicules sont aussi entretenus et stationnés. À la fin de chaque tournée, la flotte est reprise par l'équipe suivante.

2. **Restrictions sur les autoroutes :**

   Les autoroutes ne peuvent être balayées que la nuit pour minimiser l'impact sur la circulation, tandis que les routes secondaires et les rues urbaines peuvent être nettoyées de jour comme de nuit. De plus, certaines portions de route ne peuvent pas être balayées en raison de contraintes géographiques ou administratives. La flotte doit donc parfois circuler sur ces segments sans intervenir, ce qui crée des trajets à vide.

- **Problématique de tournées pour le balayage printanier :**

  Le problème de tournées pour le balayage printanier consiste à planifier les itinéraires de la flotte afin de couvrir l'ensemble des segments à nettoyer dans un contrat donné, tout en respectant les contraintes opérationnelles et en minimisant le coût total de l'opération.

  Ce coût total se compose de deux éléments principaux : les coûts fixes et les coûts variables. Les coûts fixes incluent les dépenses liées à l'équipement, aux salaires des équipes et à la location des dépôts. Les coûts variables, quant à eux, dépendent de la distance totale parcourue par la flotte, y compris les trajets à vide.

  Ainsi, l'objectif est de minimiser le coût total, tout en réduisant le temps nécessaire pour balayer l'ensemble des segments concernés.

1.2 Motivation de recherche

La saison de balayage printanier est assez courte, ne durant généralement que quelques semaines, ce qui rend indispensable l'élaboration d'un plan de routage efficace des véhicules. De plus, en raison de faibles marges bénéficiaires, un souci majeur pour chaque entrepreneur est de concevoir un plan minimisant le temps d'exécution, et donc les coûts. Malgré l'intérêt manifesté jusqu'à présent par l'industrie pour ce sujet, pratiquement peu de travaux académiques se sont penchés sur son optimisation à grande échelle. La résolution de problèmes issus de l'industrie, notamment ceux liés au balayage printanier, représente une motivation importante pour la conception de nouvelles approches en optimisation combinatoire.
Les approches existantes, basées sur des formulations mathématiques classiques, comme celles par arcs ou par nœuds, présentent des limites notables. La formulation par arcs génère un modèle de taille considérable, ce qui complique sa résolution même pour des instances de

petite taille en raison d'une explosion du nombre de variables et de contraintes. Quant à la formulation par nœuds, elle souffre d'une relaxation continue de mauvaise qualité, avec des écarts très élevés entre les bornes inférieure et supérieure (jusqu'à 97 % pour une instance de 50 tâches dans un temps de calcul d'une heure). À cela s'ajoute un phénomène de dégénérescence, qui ralentit la convergence des algorithmes de résolution. En outre, ces formulations montrent une incapacité à résoudre des instances de moyenne et grande taille dans des délais raisonnables. Sous une limite de deux heures, CPLEX n'a pu résoudre que de petites instances comportant jusqu'à 24 segments de route pour une formulation par arcs et 60 segments pour une formulation par nœuds. Ces limites mettent en évidence le caractère computationnellement difficile du problème et confirment la nécessité de développer des méthodes plus performantes, capables de surmonter les défis liés à la taille du modèle, à la qualité de la relaxation et à la dégénérescence.

1.3 Objectifs de projet

Notre travail vise à repousser les limites des approches existantes en proposant une nouvelle méthodologie pour résoudre le problème de tournées de balayage printanier.
Les objectifs de cette recherche sont les suivants :

— Élaborer une formulation par chemins, où chaque variable de décision représente une tournée (route) d'un dépôt à un autre sans excéder la durée maximale. Cette formulation s'appuie sur un ensemble de routes réalisables, générées en amont grâce à une adaptation de la phase de construction de la métaheuristique GRASP (*Greedy Randomized Adaptive Search Procedure*). L'ensemble des routes ainsi obtenu sert à stabiliser les variables duales et fournir une borne supérieure de bonne qualité.

— Développer une méthode de résolution heuristique hybride, combinant la génération de colonnes, basée sur les routes générées par GRASP et une heuristique de branchement en profondeur (*Diving Heuristic*) pour améliorer la qualité de la relaxation continue, en fournissant des solutions de meilleure qualité et en réduisant significativement l'écart entre les bornes inférieure et supérieure, tout en permettant de résoudre rapidement et efficacement des instances de grande taille.

— Étendre l'horizon de planification (à une semaine, voire un mois ou plus). Pour atteindre cet objectif, une stratégie d'agrégation des tâches est proposée, permettant de transformer des instances complexes et de grande taille en instances de taille moyenne, plus faciles à résoudre avec les deux approches proposées. Cette approche offre une

solution pragmatique pour répondre aux besoins industriels, en offrant aux entrepreneurs des outils pratiques pour minimiser les coûts et relever les défis opérationnels sur des horizons temporels élargis.

1.4 Plan de mémoire

Ce mémoire est structuré en six chapitres pour présenter les travaux réalisés. Le chapitre 2 est consacré à une revue de littérature qui analyse les différentes méthodes existantes pour résoudre le problème de balayage ainsi que des problèmes similaires, permettant de situer notre contribution dans le champ des recherches actuelles. Dans le chapitre 3, nous présentons les deux formulations mathématiques classiques, respectivement basées sur les arcs et les nœuds, utilisées pour formuler le problème. Nous proposons ensuite notre nouvelle formulation basée sur les chemins. Le chapitre 4 décrit la méthodologie proposée, articulée autour de la phase de construction GRASP, de l'approche DH et de la décomposition de Dantzig-Wolfe. Ensuite, le chapitre 5 expose les expériences numériques réalisées pour valider l'approche et analyse en profondeur les résultats obtenus. Enfin, le chapitre 6 conclut ce mémoire en récapitulant la méthode proposée et en suggérant des pistes d'amélioration ainsi que des perspectives pour de futures recherches.

CHAPITRE 2 REVUE DE LITTÉRATURE

Ce chapitre présente une revue de littérature des principales contributions scientifiques relatives à l'optimisation de tournées de véhicules, en particulier dans les services municipaux tels que la collecte de résidus, l'entretien hivernal, le balayage des rues et plus spécifiquement le balayage printanier. Dans un premier temps, nous présentons les approches proposées dans la littérature pour la résolution des problèmes de tournées de véhicules dans divers contextes opérationnels, en distinguant les problèmes de tournées sur nœuds (*Vehicle Routing Problem, VRP*) et les problèmes de tournées sur arcs (*Arc Routing Problem, ARP*), abordés dans la section 2.1. Nous poursuivons ensuite avec la méthode exacte *Branch and Price*, largement utilisée pour la résolution des problèmes de tournées. Ses principes fondamentaux ainsi que sa pertinence dans le cadre de la génération de colonnes sont décrits en section 2.2. Nous clôturons ce chapitre par une analyse critique et une synthèse des approches présentées (section 2.3).

2.1 Optimisation de tournées de véhicules pour les services municipaux

Les services municipaux, tels que la collecte des résidus, le déneigement, le balayage des rues et le balayage printanier, jouent un rôle crucial dans le maintien de la qualité de vie en milieu urbain. L'efficacité de ces services dépend en grande partie de la planification optimale des itinéraires de véhicules, afin de minimiser les coûts opérationnels et l'empreinte environnementale tout en respectant les contraintes spécifiques à chaque tâche. Dans la littérature, ces problèmes de planification sont souvent modélisés comme des problèmes de tournées de véhicules ou des problèmes de tournées sur arcs. Le choix entre ces deux modélisations dépend de la nature spécifique du service considéré. Par exemple :

- Problèmes de tournées de véhicules (VRP) : utilisés lorsque les points de service sont des nœuds du réseau, comme dans la collecte de déchets où chaque domicile représente un point de collecte.
- Problèmes de tournées sur arcs (ARP) : appropriés lorsque le service doit être effectué le long des arêtes du réseau, comme le déneigement ou le balayage des rues, où l'ensemble de la rue nécessite une intervention.

Plusieurs contributions majeures ont façonné la recherche sur les problèmes de tournées sur arcs (ARP). On peut notamment citer l'ouvrage de Corberán et Laporte [3], qui constitue une synthèse exhaustive des variantes classiques telles que le problème du postier chinois,

le problème rural du postier, ainsi que des extensions plus complexes comme le *capacitated arc routing problem (CARP)* et le *windy rural postman problem (WRPP)*. Les auteurs y présentent également différentes approches de résolution, aussi bien exactes qu'heuristiques, offrant ainsi une vue d'ensemble structurée des défis méthodologiques associés aux ARP. Dans le même esprit, la thèse de Wøhlk [4] apporte une contribution en explorant en profondeur le *capacitated arc routing problem (CARP)*. Elle y propose notamment de nouvelles bornes inférieures spécifiques à ce problème, ainsi que des algorithmes exacts et des heuristiques efficaces permettant de résoudre des instances de grande taille. Par ailleurs, Vidal et al. [5] proposent une approche algorithmique générique capable de s'adapter à différents types de problèmes de tournées, qu'ils soient définis sur des nœuds, des arêtes ou des arcs. Cette flexibilité permet de traiter divers contextes opérationnels dans un cadre méthodologique commun. Enfin, la revue de Corberán et al. [6] retrace l'évolution des ARP, en mettant en évidence les avancées méthodologiques, les problématiques actuelles ainsi que les perspectives de recherche. Les auteurs insistent également sur les nombreuses applications pratiques, notamment dans les services municipaux tels que le déneigement et la collecte des déchets.

D'autres travaux se sont penchés sur la transformation des problèmes de tournées sur arcs en problèmes de tournées sur nœuds, dans le but de bénéficier des méthodes avancées développées pour le VRP. À ce titre, Foulds et al. [7] proposent une méthode compacte de transformation, remplaçant chaque arc par un nœud, tout en réduisant la taille du graphe. Enfin, dans le contexte des réseaux multi-dépôts, Montoya et al. [8] présentent une revue approfondie des approches de modélisation et de résolution du VRP à dépôts multiples *(Multiple Depot Vehicle Routing Problem (MDVRP))*, soulignant son importance pour les services municipaux. En lien direct avec les applications municipales, certains auteurs ont proposé des modèles et des approches spécifiques adaptés à des contextes opérationnels concrets. On peut citer l'étude de Lopes et al. [9] sur le problème de localisation et de tournées sur arcs (*Location-Arc Routing Problem, LARP*) proposant des approches heuristiques et fournissant des instances de test pertinentes pour des applications telles que la collecte de déchets et l'épandage d'abrasifs en hiver. De même, Usberti et al. [10] ont développé une méthode GRASP avec reliaison de chemins évolutifs (*evolutionary path-relinking*) pour résoudre le problème de tournées sur arcs avec capacité, démontrant son efficacité dans des contextes tels que le balayage des rues et la collecte de déchets. Martínez et al. [11] ont introduit un algorithme BRKGA (*Biased Random-Key Genetic Algorithms*) pour le CARP, contribuant ainsi à l'amélioration des solutions pour les problèmes de tournées sur arcs. Dans ce qui suit, nous présenterons quelques travaux réalisés sur ces problèmes, en explorant différentes approches adaptées aux services municipaux.

2.1.1 Collecte de résidus

Dans le domaine de la collecte de résidus urbains, plusieurs études récentes ont exploré des approches innovantes pour optimiser les tournées de véhicules. Par exemple, Liang et al. [12] ont proposé une mini-revue des métaheuristiques appliquées au problème de tournées pour la collecte des déchets *(Waste Collection Routing Problem, WCRP)*, en comparant des méthodes telles que l'optimisation par colonies de fourmis, le recuit simulé, et la procédure GRASP. Cette étude montre que l'utilisation des systèmes d'information géographique (SIG) peut améliorer la planification des itinéraires tout en réduisant les coûts opérationnels. De même, dans le cadre d'une approche plus durable, l'étude menée par Zhou et al. [13] à Shanghai introduit un modèle multi-objectifs basé sur l'algorithme NSGA-III (*Non-dominated Sorting Genetic Algorithm III*) et le recuit simulé, permettant de maximiser les bénéfices économiques tout en équilibrant la charge de travail et en minimisant les fuites de déchets durant le transport. Contrairement à ces méthodes basées sur des métaheuristiques, l'algorithme hybride NN-SOS (*Nearest Neighbourhood-Symbiotic Organisms Search*) proposé par Umam et al. [14] combine l'algorithme du plus proche voisin avec la recherche par organismes symbiotiques. Le rôle de NN est de générer des itinéraires initiaux en fonction de la proximité des points de collecte. Quant à SOS, il s'agit d'une métaheuristique qui améliore les itinéraires initiaux par une recherche locale, inspirée de phénomènes naturels tels que le mutualisme, le commensalisme et le parasitisme, dans le but d'atteindre un optimum global. Cette méthode offre des résultats rapides (11,9 secondes de calcul) et une réduction d'environ 10% de la distance totale parcourue (de 68 km à 61,34 km).

2.1.2 Entretien hivernal

L'entretien hivernal, incluant le déneigement et l'épandage de sel, présente des défis logistiques importants en raison des contraintes de temps, des conditions météorologiques changeantes et de l'obligation de couvrir l'ensemble du réseau routier. La littérature propose divers modèles d'optimisation pour résoudre ces problématiques, en intégrant notamment les capacités limitées des véhicules et les fenêtres de temps.

Perrier et al. [15] ont proposé un modèle de programmation linéaire en nombres entiers pour optimiser les tournées de déneigement en milieu urbain. Ce modèle repose sur un réseau de flux multi-commodités intégrant des contraintes spécifiques, telles que les restrictions de virage, l'équilibrage des charges et le service en tandem sur certaines voies. Deux heuristiques ont été développées pour résoudre ce problème : une approche parallèle, qui décompose le problème global en sous-problèmes plus simples résolus indépendamment, et une approche

séquentielle, où les sous-problèmes sont traités un à un dans un ordre prédéfini. Dans les deux cas, les sous-problèmes sont formulés sous forme de modèles mathématiques et résolus avec CPLEX. L'étude a été menée sur le réseau de la ville de Dieppe, au Nouveau-Brunswick, Canada, composé de 462 sommets et 1234 arcs. La flotte utilisée comprend huit véhicules hétérogènes (un niveleur, deux chasse-neige et cinq chargeurs), utilisés à partir d'un seul dépôt. Les résultats montrent que l'approche parallèle permet de réduire d'environ 60% la durée totale des opérations de déneigement par rapport au plan de déneigement existant.

De leur côté, Salazar-Aguilar et al. [16] ont développé une métaheuristique ALNS (*Adaptive Large Neighborhood Search*) pour résoudre un problème de routage d'arcs synchronisé, permettant la coordination simultanée des véhicules pour le déneigement de routes à plusieurs voies. Leur approche a été testée sur des instances générées aléatoirement (jusqu'à 300 sommets et 795 arcs) ainsi qu'une étude de cas réelle a été menée sur la ville de Dieppe, une banlieue de Moncton, au Nouveau-Brunswick, comprenant 430 sommets et 1056 arcs, tous à une ou deux voies. Leur algorithme a démontré une performance remarquable, générant un ensemble d'itinéraires optimisés en seulement 538 secondes, avec un makespan (durée totale) de 3 heures et 28 minutes.

Une étude menée par Liu et al. [17] à Edmonton propose une approche basée sur l'utilisation d'un modèle basé sur le CARP (*Capacitated Arc Routing Problem*) pour minimiser la distance totale parcourue lors des opérations de déneigement urbain. L'étude s'est concentrée sur un sous-réseau du sud de la ville, comprenant 91 liens routiers et 55 nœuds. Cette approche intègre des analyses de sensibilité permettant d'évaluer l'impact des paramètres clés tels que l'emplacement des dépôts et le nombre d'itinéraires nécessaires sur les coûts opérationnels. Les résultats obtenus montrent que le choix stratégique du dépôt peut réduire la distance de déplacement à vide, passant de 4 % à 1 %, en fonction de la localisation choisie. En outre, l'emplacement optimal du dépôt évolue en fonction du nombre d'itinéraires, se rapprochant du centre du réseau pour minimiser le temps de service requis. Avec six itinéraires de déneigement, le temps nécessaire pour compléter le service varie de 2 à 3 heures en fonction de l'emplacement du dépôt.

Contrairement aux approches classiques de déneigement, qui traitent séparément les opérations de déneigement et d'épandage, Quirion-Blais et al. [18] ont développé une méthode innovante intégrant la double fonctionnalité des véhicules, capables à la fois de déneiger et d'épandre simultanément, améliorant ainsi l'efficacité opérationnelle. Leur algorithme ALNS

permet de créer des itinéraires optimisés, en prenant en compte la hiérarchie du réseau, les restrictions de virage, ainsi que la compatibilité entre les rues et les véhicules, en particulier pour les routes locales, incluant les zones résidentielles et commerciales. L'étude de cas menée dans une ville du Nord du Québec montre que cette approche multifonctionnelle réduit le temps de service de manière significative, passant de 12000 secondes avec un seul niveleur à 4200 secondes avec trois niveleurs.

En plus de ces travaux spécifiques sur le déneigement, il convient de noter que la littérature sur la maintenance hivernale des routes a également bénéficié de contributions de Perrier et al. [19]. Dans leurs revues exhaustives, les auteurs offrent une vue d'ensemble sur les différentes variantes du problème de tournées pour le déneigement (*Snow plow Routing Problem*) et sur les techniques d'optimisation développées pour répondre aux défis opérationnels hivernaux. La Partie III [20] de leur travail se concentre sur le problème de tournées de véhicules et la localisation des dépôts pour l'épandage, tandis que la Partie IV [19] aborde le problème de tournées et le dimensionnement des flottes pour le déneigement et l'élimination de la neige.

2.1.3 Balayage des rues

Plusieurs chercheurs se sont intéressés à l'optimisation des tournées de balayage, proposant des modèles mathématiques et des algorithmes avancés pour améliorer l'efficacité des services municipaux. Les approches développées varient du problème de tournées de véhicules (VRP) aux problèmes de tournées sur arcs (ARP), en passant par des méthodes heuristiques et des métaheuristiques permettant de s'adapter aux contraintes urbaines et opérationnelles.

La gestion des contraintes opérationnelles, telles que les horaires de service, la capacité des balayeuses et les restrictions propres aux zones de balayage, constitue un élément clé des travaux de Bodin et Kursh [21]. Face aux défis posés par la complexité du réseau urbain et les limitations liées à la circulation et au stationnement, les auteurs ont élaboré une approche pratique pour optimiser la planification des itinéraires de balayeuses à New York et Washington DC. Leur méthode repose sur la modélisation du problème sous forme d'un problème de tournées de véhicules (VRP), intégrant à la fois des méthodes heuristiques et des techniques de programmation linéaire.

Eglese et Murdock [22] se sont penchés sur l'optimisation du balayage de rues dans le nord-ouest de l'Angleterre, en développant un algorithme heuristique visant principalement à minimiser les distances parcourues par les balayeuses tout en respectant les contraintes de capacité

et de temps. Leur approche se distingue par sa flexibilité, permettant une adaptation rapide aux changements opérationnels, tels que l'ajout de nouveaux véhicules ou la modification des sites de décharge. Contrairement aux environnements urbains où les restrictions de stationnement compliquent la planification, leur étude s'est concentrée sur les zones rurales où ces contraintes n'existent pas et où les routes à double sens offrent plus de souplesse. En effet, traiter les routes comme des voies à double sens a permis de surmonter une limitation clé de la méthode de Bodin et Kursh [21], qui ne permettait pas de balayer les deux côtés d'une rue le même jour.

L'étude de Blazquez et al. [23] propose une méthode innovante pour optimiser les itinéraires de balayage des rues en se concentrant sur une zone spécifique du nord-est de la municipalité de Santiago, au Chili, composée de 9 rues à sens unique et 2 rues à double sens. Contrairement aux approches classiques du problème du postier chinois et du problème du postier rural, où chaque rue n'est visitée qu'une fois, les auteurs introduisent une nouvelle contrainte : chaque rue doit être balayée autant de fois qu'elle possède de côtés, en raison de la capacité des balayeuses à ne nettoyer qu'un seul côté à la fois. Pour simplifier le modèle, deux hypothèses sont posées : la plage horaire de 11h à 6h est suffisante pour balayer l'ensemble de la zone, et la capacité du véhicule permet de couvrir toutes les rues sans nécessiter de vidange au dépôt. Pour résoudre ce problème, les auteurs ont transformé le problème de routage sur arcs en un problème de routage sur nœuds. En construisant un graphe spécifique représentant le réseau de rues, permettant d'utiliser des méthodes éprouvées de résolution du problème du voyageur de commerce *(Travelling Salesman Problem, TSP*), notamment l'algorithme du plus proche voisin. Cette approche a permis de réduire jusqu'à 37% la distance parcourue par les balayeuses.

Quant à Cerrone et al. [24], ils ont proposé une nouvelle variante du problème en tenant compte des contraintes de stationnement multi-périodes, où la disponibilité des côtés de rue pour le stationnement varie chaque jour. Les auteurs ont développé un algorithme génétique (AG) permettant de déterminer à la fois les jours de stationnement autorisé et les itinéraires des balayeuses, en minimisant la distance parcourue. Comparé à une implémentation CPLEX et à une heuristique de recherche locale (LS), l'algorithme s'est avéré plus performant, notamment sur de grandes instances (jusqu'à 225 nœuds), avec un temps de calcul raisonnable.

Dans leur contribution au problème de tournées sur arcs, Golden et al. [25] ont introduit une technique de zigzag, qui consiste à permettre le balayage de deux segments opposés

d'une rue dans une même tournée en alternant les directions, lorsque les créneaux horaires le permettent. Cette stratégie est particulièrement utile pour modéliser des contraintes de disponibilité temporelle des segments, activables pendant des créneaux horaires spécifiques. Cette adaptation est particulièrement pertinente pour des applications telles que la collecte de déchets et la livraison de journaux, ainsi que pour le balayage des rues ou l'épandage d'abrasifs , où elle peut réduire de façon significative les coûts et les temps de service. Le modèle qu'ils proposent transforme le problème de tournées sur arcs en un problème de tournées sur nœuds, avec une formulation mathématique solvable par des solveurs standards tels que CPLEX et Gurobi. Cette étude illustre l'impact des options de zigzag et des fenêtres de temps sur la réduction des coûts et l'efficacité du service dans des contextes urbains variés.

Les travaux de Yurtseven et Gökçe [26] sur les véhicules électriques se distinguent comme les premiers à développer un modèle mathématique utilisant la programmation linéaire en nombres entiers mixtes pour optimiser le routage des balayeuses de rue électriques. Ce modèle, qui aborde des contraintes opérationnelles telles que la recharge, l'élimination des déchets, et les pauses obligatoires, gère une flotte hétérogène adaptée aux besoins en consommation d'énergie de chaque itinéraire. Une étude de cas appliquée à la ville d'Izmir en Turquie, où le réseau de rues comprend 14 nœuds et un dépôt, utilise des données réelles, illustrant l'efficacité du modèle. Le problème a été résolu à l'aide de CPLEX, permettant d'analyser l'impact des ajustements des fenêtres de temps et des variations des demandes de nettoyage sur le temps de résolution.

Dans la continuité de ces travaux, Parsons et al. [27] ont proposé une méthode en deux étapes pour optimiser le balayage des rues, en se concentrant uniquement sur les routes artérielles et collectrices (AC) ainsi que sur les routes résidentielles (RES). Leur approche commence par diviser le réseau routier en plusieurs zones de balayage distinctes, puis génère des itinéraires optimisés pour chaque zone. Les opérations de balayage se déroulent au printemps, en été et en automne. L'étude de cas menée à Oshawa, au Canada, a couvert environ 693 km de routes résidentielles et 445 km de routes artérielles, en divisant le réseau en 17 zones opérationnelles (5 pour les AC et 12 pour les RES). Pour assurer une couverture complète des bordures, les routes à sens unique sont balayées deux fois dans le même sens, tandis que les routes à double sens le sont dans les deux directions. L'approche utilise un algorithme de clustering en deux étapes pour la génération des zones de balayage, suivi d'un algorithme combinant Hierholzer (méthode utilisée pour construire des circuits couvrant toutes les arêtes d'un graphe), la recherche tabou, et une version modifiée de l'optimisation par colonie de fourmis, qui permet de générer des détours optimaux afin de supprimer les demi-tours interdits dans les itinéraires.

2.1.4 Balayage printanier

Le balayage printanier constitue une variante du problème de balayage des rues, effectué à la fin de l'hiver afin d'éliminer les abrasifs, tels que le sable ou le gravier, répandus durant la saison froide pour améliorer l'adhérence sur les surfaces enneigées ou verglacées. Cette variante se caractérise par une durée d'exécution courte et un grand nombre de segments à traiter dans un laps de temps restreint. Bien que le balayage des rues ait fait l'objet de nombreuses recherches, peu d'études se sont spécifiquement intéressées à cette variante.

En commençant par les travaux de Kraiem et al. [28], qui ont proposé un modèle de programmation linéaire en nombres entiers mixtes (MILP) pour aborder le problème de balayage printanier sous la forme d'un problème de routage sur arcs multi-dépôts avec assignation flexible du dépôt final (*Multi-Depot Arc Routing Problem with Flexible assignment of End Depot, MDARPFED*). Ce modèle vise à minimiser le temps total de trajet tout en respectant des contraintes opérationnelles spécifiques, telles que la réalisation obligatoire du balayage des arcs d'autoroutes uniquement pendant les quarts de nuit, tandis que les autres arcs peuvent être balayés de jour comme de nuit. Une contrainte supplémentaire impose que chaque quart débute au dépôt de fin du quart précédent, assurant ainsi une continuité opérationnelle fluide. Les expérimentations sur de petites instances (jusqu'à 18 nœuds) ont montré que le modèle MDARPFED permet de réaliser des économies de temps de trajet comprises entre 2,97 % et 12,09 % par rapport au modèle classique à dépôt unique. Le modèle a été résolu à l'aide de CPLEX, qui a montré des performances prometteuses sur les petites instances. Cependant, en raison de la complexité exponentielle du problème, classé NP-difficile, CPLEX n'a pu trouver des solutions optimales que pour de petites instances dans des délais raisonnables.

Pour surmonter les limites identifiées dans leur première étude, notamment la difficulté de résoudre des instances de grande taille avec le modèle MILP, Kraiem et al. [29] ont développé une approche métaheuristique basée sur la recherche adaptative à grand voisinage (ALNS). Cette méthode vise à améliorer l'efficacité computationnelle et à fournir des solutions de bonne qualité dans des délais raisonnables pour le MDARPFED, en prenant en compte la flexibilité des dépôts ainsi que la gestion spécifique des types d'arcs (30 % d'autoroutes et 70 % d'autres routes). Contrairement à CPLEX, limité aux petites instances (avec 24 nœuds et 2 dépôts) et nécessitant 7200 secondes de calcul, ALNS permet de résoudre efficacement des instances de taille beaucoup plus importante, allant jusqu'à 228 nœuds avec 4 dépôts, tout en maintenant un temps de calcul compétitif.

Nous concluons cette section avec l'étude de Lamghari et al. [1], qui propose un modèle mathématique basé sur les nœuds pour résoudre le problème de balayage printanier. Cette étude transforme le modèle de tournées sur arcs présenté par Kraiem et al. [28] en un modèle de tournées sur nœuds, où chaque segment de rue est représenté comme une tâche spécifique à effectuer. Cette reformulation permet d'obtenir une modélisation plus compacte. Cependant, malgré cette reformulation, le modèle n'a pas pu résoudre les grandes instances dans un délai raisonnable avec CPLEX, révélant les limites de cette approche exacte pour les instances de grande taille. Pour contourner cette limite, les auteurs ont développé trois stratégies de décomposition. Parmi elles, la méthode PBD (*Proximity-Based Decomposition*) s'est révélée la plus efficace. Elle consiste à diviser les tâches en groupes géographiquement proches, de manière à former des sous-problèmes plus petits et plus faciles à résoudre. Chaque groupe est ensuite associé à un ensemble de quarts, permettant une résolution progressive du problème complet. Cette approche a été testée sur un cas réel à Victoriaville, Québec, Canada, comprenant 172 tâches. Elle permet d'obtenir des solutions de bonne qualité dans un temps limité à 10800 secondes, y compris pour des instances de grande taille (jusqu'à 180 tâches et 4 dépôts). Elle surpasse CPLEX, qui présente des limites pour résoudre ces instances dans un délai raisonnable.

2.2 Branch and Price

La méthode *Branch-and-Price* repose sur deux piliers fondamentaux en optimisation combinatoire : *Branch-and-Bound* et la génération de colonnes. Afin de mieux comprendre cette approche, il est essentiel de présenter d'abord la méthode *Branch-and-Bound*.

Branch-and-Bound est une méthode exacte utilisée pour aborder des problèmes d'optimisation en nombres entiers en explorant de manière exhaustive un arbre de recherche. Chaque nœud de cet arbre représente un sous-problème, défini comme un ensemble de solutions possibles. À chaque itération, un sous-problème est sélectionné pour être exploré en appliquant une règle de séparation, qui consiste à diviser l'espace des solutions en sous-ensembles plus faciles à résoudre. Ce processus de branchements se poursuit jusqu'à l'obtention d'une solution entière. Parallèlement, des règles d'élagage sont appliquées pour éliminer les sous-problèmes qui ne peuvent pas mener à une meilleure solution que celle déjà trouvée, permettant ainsi de réduire considérablement l'espace de recherche. Cette méthode est décrite en détail dans l'article de Morrison et al. [30], qui présente également des recherches récentes sur ses divers mécanismes.

Continuons avec *Branch-and-Price* une méthode largement utilisée pour résoudre des problèmes de programmation linéaire en nombres entiers de très grande taille. Elle combine la génération de colonnes (GC) avec le cadre du *Branch-and-Bound*, en tirant parti de la décomposition de Dantzig-Wolfe [31], permettant de décomposer le problème en un problème maître restreint (PMR) et un ou plusieurs sous-problèmes (SP_s). À chaque itération, la relaxation linéaire du PMR est résolue, puis le SP est activé afin de générer de nouvelles colonnes rentables. Ce processus de génération continue jusqu'à ce qu'aucune colonne supplémentaire ne soit générée, marquant ainsi la convergence du processus de GC. Lorsque la solution obtenue est fractionnaire, des stratégies de branchement sont appliquées pour garantir l'intégralité de la solution. Chaque nouvelle application de *Branch-and-Bound* entraîne une mise à jour du PMR, nécessitant une nouvelle génération de colonnes à chaque nœud de branchement. Cette méthode se révèle particulièrement efficace pour gérer les défis posés par la taille et la complexité des problèmes industriels.

Barnhart et al. [32] ont approfondi cette méthode en mettant en avant les avantages et les défis liés à son application. Ils expliquent comment la génération de colonnes permet d'améliorer la qualité des relaxations linéaires et de traiter efficacement les problèmes comportant un très grand nombre de variables. Les auteurs soulignent également l'importance des formulations enrichies en variables, qui fournissent une approximation améliorée de l'enveloppe convexe des solutions réalisables. Cependant, l'intégration de la génération de colonnes dans un cadre de programmation en nombres entiers pose plusieurs défis. Parmi ces défis, on trouve la complexité accrue du sous-problème, qui peut nécessiter des approches d'approximation pour être résolu efficacement, rendant la génération de nouvelles colonnes plus problématique. De plus, les techniques de branchement standards ne garantissent pas toujours une amélioration significative des bornes, la scalabilité de la méthode peut diminuer pour des problèmes de grande taille, soulignant ainsi la nécessité d'un compromis entre précision et rapidité dans le processus. Barnhart et al. [32] illustrent ces concepts à travers des applications concrètes, notamment en planification de tournées de véhicules, où *Branch-and-Price* démontre son efficacité sur des instances de grande taille. Les travaux présentés par Barnhart et al. [32] constituent ainsi une référence majeure sur les méthodes et les défis liés à l'utilisation de la génération de colonnes en programmation linéaire en nombres entiers. De plus, ils proposent une description détaillée des algorithmes standards de *Branch-and-Price*, mettant en avant leur structure et leur mise en œuvre.

2.3 Analyse critique et synthèse

Ce chapitre a présenté un aperçu structuré des travaux de recherche visant à résoudre les problèmes de tournées de véhicules dans divers contextes municipaux, tels que la collecte de déchets, l'entretien hivernal et le balayage des rues. Les méthodes recensées couvrent un éventail de stratégies allant des modèles exacts (MILP) à des métaheuristiques, en passant par des heuristiques de construction ou encore des techniques de décomposition. Le balayage printanier, en tant que variante saisonnière du balayage des rues, a été abordé plus récemment à travers trois études principales, qui ont proposé des modèles dédiés et adapté certaines méthodes existantes aux contraintes propres à ce service, notamment la planification des opérations dans un laps de temps restreint, la répartition des types de routes (autoroutes et autres rues) entre les quarts de jour et de nuit, ainsi que la continuité entre les quarts à travers les dépôts utilisés.

Bien que la littérature soit riche en contributions, tant en termes de modélisation que de méthodes de résolution, plusieurs constats critiques peuvent être formulés. D'une part, on observe un usage varié de métaheuristiques (ALNS, colonies de fourmis, SOS) pour surmonter les limites des méthodes exactes, notamment sur les grandes instances. D'autres études ont opté pour des heuristiques de construction ou des stratégies de décomposition (par zones, par proximité) afin de simplifier la résolution. Certaines approches ont même transformé les problèmes sur arcs en problèmes sur nœuds pour bénéficier d'algorithmes standards comme le TSP. Mais ces approches ne fournissent aucune garantie sur la qualité des solutions obtenues. Les modèles développés pour le balayage printanier présentent encore des limitations : les approches exactes proposées ne permettent pas de résoudre efficacement les grandes instances (au-delà de 24 nœuds), tandis que les reformulations basées sur les nœuds souffrent d'une mauvaise qualité de la relaxation continue. En outre, aucune reformulation du problème de balayage printanier sous forme de chemin n'a été proposée. Par ailleurs, la résolution de grandes instances (jusqu'à 180 tâches) a nécessité des temps de calcul très élevés (jusqu'à 10800 secondes), sans garantie d'optimalité. Aucune des études abordées ne repose sur une méthode exacte robuste telle que *Branch and Price*, pourtant largement reconnue pour son potentiel à résoudre des problèmes de grande taille dans le domaine des tournées de véhicules et à fournir des solutions de meilleure qualité. C'est dans ce contexte que notre travail s'inscrit, en s'appuyant sur la méthode *Branch and Price*, spécifiquement adaptée à ce type de problème, dans le but de combler cette lacune dans les approches existantes. Afin de rendre cette méthode applicable aux cas réalistes, nous proposons également une stratégie d'agrégation permettant de traiter des instances de grande taille dans un temps raisonnable.

CHAPITRE 3 MODÈLES MATHÉMATIQUES EXISTANTS ET PROPOSITION D'UNE NOUVELLE FORMULATION MATHÉMATIQUE

Dans ce chapitre, nous présentons les modèles mathématiques existants et proposons une nouvelle formulation adaptée au problème étudié. Nous débutons par une formulation classique basée sur les tournées sur arcs, en introduisant les notations, variables de décision, paramètres, ainsi que la description du modèle associé (3.1). Ensuite, nous explorons une formulation centrée sur les tournées sur nœuds, en adoptant la même structure de présentation (3.2). Enfin, pour surmonter les limitations des modèles précédents, nous proposons une nouvelle formulation basée sur des chemins. Cette approche est décrite à travers des notations spécifiques, une représentation graphique explicative, et un modèle mathématique, nommé MIP3 (3.3).

3.1 Formulation en problème de tournées sur arcs

Kraiem et al. [29] ont proposé une formulation du problème de balayage printanier comme un problème de tournées sur arcs, une approche pertinente, puisque la demande de balayage est localisée sur les arcs d'un réseau routier. Dans ce cadre, ils ont conceptualisé ce problème sous forme d'un problème de tournées sur arcs à dépôts multiples avec assignation flexible du dépôt de fin (*Multi-Depot Arc Routing Problem with Flexible Assignment of End Depot, MDARPFED*), ce qui offre la flexibilité aux véhicules de ne pas nécessairement retourner au même dépôt de départ. Pour aborder cette complexité, ils ont développé un modèle de programmation linéaire en nombres entiers mixtes, conçu pour optimiser les tournées dans des configurations avec différents types d'arcs. Nous présentons dans ce qui suit la formulation détaillée de cette approche.

3.1.1 Formulation et modèle mathématique

Le modèle MDARPFED avec différents types d'arcs est défini sur un graphe orienté $G_1 = (V_1, A_1)$, où $V_1 = \{1, \ldots, n, n+1, \ldots, n+m\}$ représente l'ensemble des sommets et $A_1 = \{(i,j)|i,j \in V; i \neq j\}$ représente l'ensemble des arcs. $K = \{0, \ldots, e\}$ est l'ensemble des positions des arcs où $k \in K$ et e est le nombre maximal d'arcs dans tous les quarts. Noter bien que l'ensemble des arcs d'autoroutes H nécessite un service uniquement pendant un quart de nuit. Chaque arc doit être entretenu deux fois car les deux côtés de l'autoroute doivent être balayés et l'ensemble des arcs restants R peut être entretenu une fois.

Catégorie	Description
Ensembles	V_1 : Ensemble des sommets $V = \{1, \ldots, n, n+1, \ldots, n+m\}$ A_1 : Ensemble des arcs O : Ensemble des arcs de type $Autre - route$ H : Ensemble des arcs de type $Autoroutes$ D : Ensemble des dépôts $D = \{n+1, \ldots, n+m\}$ Q : Ensemble des quarts $Q = \{1, \ldots, c\}$ Q_p : Ensemble des quarts de nuit $Q_p = \{2p; p \in \mathbb{N}^*\}$ K : Ensemble des positions des arcs $K = \{1, \ldots, e\}$
Paramètres	n : Nombre total de nœuds à l'exception des dépôts m : Nombre total de dépôts c : Nombre maximal de quarts e : Nombre maximal d'arcs dans tous les quarts Γ : Durée maximale de quart q $\in Q$ M : Grand nombre
Variables	x^k_{ijq} : 1 si l'arc (i,j) est traversé sans service dans le quart q à la position k, 0 sinon y^k_{ijq} : 1 si l'arc (i,j) est servi dans le quart q à la position k, 0 sinon w_q : 1 si le quart de travail q est utilisé, 0 sinon u^k_{ijq} : Heure de début du service ou de traversée de l'arc (i,j) dans le quart q à la position k f_q : Heure de fin du quart q
Autres paramètres et variables	S_{ij} : Temps de service de l'arc (i,j) t_{ij} : Temps de traversée sans service de l'arc (i,j) M1 : Valeur maximale entre S_{ij} et t_{ij} (égale à 12), utilisée pour désactiver certaines contraintes.

Tableau 3.1 Notations et définitions du modèle MDARPFED.

Le modèle mathématique du MDARPFED est formulé comme un programme linéaire en nombres entiers mixtes de la manière suivante :

$$
(P_1) = \begin{cases}
\min \displaystyle\sum_{q \in Q} (12 q w_q + f_q) & & (1) \\
\text{s.c :} & & \\
\displaystyle\sum_{q \in Q} \sum_{k \in K} y^k_{ijq} = 1 & \forall (i,j) \in R & (2) \\
\displaystyle\sum_{q \in Q_p} \sum_{k \in K} y^k_{ijq} = 1 & \forall (i,j) \in H & (3) \\
y^k_{ijq} = 0 & \forall (i,j) \in A_1 \setminus \{R, H\}, \forall k \in K, \forall q \in Q & (4) \\
\displaystyle\sum_{(i,j) \in A_1} (y^k_{ijq} + x^k_{ijq}) \leq 1 & \forall k \in K, \forall q \in Q & (5) \\
\displaystyle\sum_{i \in V_1} (y^k_{ijq} + x^k_{ijq}) \geq 1 - M \left[1 - \sum_{h \in V_1} (y^{k+1}_{jhq} + x^{k+1}_{jhq}) \right] & \forall j \in VD,\ \forall k \in K \setminus \{e-1\},\ \forall q \in Q & (6) \\
\displaystyle\sum_{i \in V_1} (y^k_{ijq} + x^k_{ijq}) \leq \sum_{h \in V_1} (y^{k+1}_{jhq} + x^{k+1}_{jhq}) & \forall j \in VD,\ \forall k \in K \setminus \{e-1\},\ \forall q \in Q & (7) \\
u^0_{ijq} = 0 & \forall i,j \in V_1, \forall q \in Q & (8) \\
w_q \geq y^k_{ijq} + x^k_{ijq} & \forall i,j \in V_1,\ \forall q \in Q,\ \forall k \in K & (9)
\end{cases}
$$

Suite page suivante

$$
(P_1) = \begin{cases}
\displaystyle\sum_{i\in D}\sum_{j\in V_1}(y^0_{ijq}+x^0_{ijq}) = w_q & \forall q\in Q & (10)\\
(u^k_{ijq}+S_{ij}) - \displaystyle\sum_{h\in V_1} u^{k+1}_{jhq} \le M1\left(1-y^k_{ijq}\right) & \forall i\in V_1,\ \forall j\in VD,\ \forall k\in K\setminus\{e{-}1\},\ \forall q\in Q & (11)\\
(u^k_{ijq}+t_{ij}) - \displaystyle\sum_{h\in V-2} u^{k+1}_{jhq} \le M1\left(1-x^k_{ijq}\right) & \forall i\in V_1,\ \forall j\in VD,\ \forall k\in K\setminus\{e{-}1\},\ \forall q\in Q & (12)\\
\Gamma(y^k_{ijq}+x^k_{ijq}) \ge u^k_{ijq} & \forall i\in V_1,\ \forall j\in V_1,\ \forall k\in K,\ \forall q\in Q & (13)\\
u^k_{ijq} \le \Gamma - S_{ij}\, y^k_{ijq} & \forall i\in V_1,\ \forall j\in D,\ \forall k\in K,\ \forall q\in Q & (14)\\
u^k_{ijq} \le \Gamma - t_{ij}\, x^k_{ijq} & \forall i\in V_1,\ \forall j\in D,\ \forall k\in K,\ \forall q\in Q & (15)\\
\displaystyle\sum_{k\in K}\sum_{i\in V_1}(x^k_{ijq}+y^k_{ijq}) \ge \sum_{h\in V_1}(x^0_{jhq+1}+y^0_{jhq+1}) & \forall j\in D,\ \forall q\in Q & (16)\\
\displaystyle\sum_{i\in V_1}\sum_{j\in D}\sum_{k\in K}(y^k_{ijq}+x^k_{ijq}) \le w_q & \forall q\in Q & (17)\\
f_q \ge u^k_{ijq} + t_{ij}\, x^k_{ijq} & \forall i\in V_1,\ \forall j\in D,\ \forall k\in K,\ \forall q\in Q & (18)\\
f_q \ge u^k_{ijq} + S_{ij}\, y^k_{ijq} & \forall i\in V_1,\ \forall j\in D,\ \forall k\in K,\ \forall q\in Q & (19)\\
\displaystyle\sum_{i\in V_1}(x^k_{ijq}+y^k_{ijq}) - \sum_{h\in V_1}(x^{k+1}_{jhq}+y^{k+1}_{jhq}) \le \sum_{d\in D}\sum_{i\in V_1}(x^k_{idq}+y^k_{idq}) & \forall j\in V_1,\ \forall k\in K\setminus\{e{-}1\},\ \forall q\in Q & (20)\\
\displaystyle\sum_{h\in V_1}(x^{e-1}_{jhq}+y^{e-1}_{jhq}) \le \sum_{i\in V_1}\sum_{d\in D}(x^{e-1}_{idq}+y^{e-1}_{idq}) & \forall j\in V_1,\ \forall q\in Q & (21)\\
x^k_{ijq}\in\{0,1\} & \forall (i,j)\in A_1, \forall k\in K, \forall q\in Q & (22)\\
y^k_{ijq}\in\{0,1\} & \forall (i,j)\in A, \forall k\in K, \forall q\in Q & (23)\\
u^k_{ijq}\ge 0 & \forall (i,j)\in A_1, \forall k\in K, \forall q\in Q & (24)\\
f_q\ge 0 & \forall q\in Q & (25)\\
w_q\ge 0 & \forall q\in Q & (26)
\end{cases}
$$

La fonction objectif (1) vise à minimiser le nombre total de quarts utilisés ainsi que le temps total de déplacement, le coefficient 12 correspondant à la durée fixe (en heures) d'un quart. Les contraintes (2) garantissent que chaque arc de type R doit être servi exactement une fois, indépendamment du quart. Les contraintes (3) imposent que chaque arc de type H soit également servi une seule fois, mais uniquement pendant un quart de nuit. Les contraintes (4) assurent que seuls les arcs de type R et H peuvent être servis. Les contraintes (5) veillent à ce qu'il n'y ait qu'un seul arc actif par position dans chaque quart, et que cet arc soit parcouru (sans service), soit servi. Les contraintes (6) et (7) assurent la conservation du flux. Les contraintes (8) définissent que l'heure de départ du premier arc de chaque quart est nulle, marquant le début de l'activité à temps zéro. Les contraintes (9) spécifient qu'un arc n'existe que si le quart dans lequel il figure est actif. Les contraintes (10) imposent qu'un quart actif doit commencer par un arc sortant d'un dépôt. Les contraintes (11) et (12) assurent que le début du service ou du parcours de l'arc suivant correspond au temps courant augmenté du temps de service ou de parcours. Les contraintes (13) indiquent que si un arc n'est pas servi ni parcouru, son heure de début est égale à zéro. Les contraintes (14) et (15) limitent le temps total d'un arc menant à un dépôt (temps de départ + temps de service ou de traversée), de sorte qu'il ne dépasse pas la durée maximale autorisée Γ. Les contraintes (16) établissent

que le quart suivant doit débuter au dépôt où s'est terminé le quart précédent, assurant une transition cohérente entre les quarts. Les contraintes (17) imposent qu'un quart, s'il est actif, doit se terminer dans un seul dépôt. Les contraintes (18) et (19) définissent le temps d'arrivée à la fin de chaque quart, en tenant compte du dernier arc parcouru ou servi. Les contraintes (20) et (21) assurent que le dernier nœud visité dans un quart est un dépôt, garantissant la faisabilité opérationnelle. Enfin, les contraintes (22) à (26) imposent des conditions sur les variables de décision.

3.2 Formulation en problème de tournées sur nœuds

Le modèle (P_1) présente une complexité marquée due à l'abondance de variables binaires et continues, rendant son utilisation ardue même pour des instances de petite taille, comme le soulignent Kraiem et al. [29] L'inconvénient majeur de cette approche réside dans la taille considérable du modèle, incluant $|Q| + 2|A||Q||K|$ variables binaires et $|S| + |A||Q||K|$ variables continues, $|A|$ et $|K|$ représentant respectivement le nombre d'arcs et les positions possibles. En réponse, Lamghari et al. [33] mettent en avant la nécessité d'une reformulation vers un problème de tournées sur nœuds plus compact.

3.2.1 Formulation et modèle mathématique

Le problème a été reformulé en un problème de tournées sur nœuds. Chaque segment à balayer est considéré comme une tâche qui représente un nœud dans un graphe $G_2 = (V_2, A_2)$, où V_2 est l'ensemble des nœuds et A_2 l'ensemble des arcs. Le tableau suivant présente les principaux composants de la formulation mathématique proposée.

Catégorie	**Description**
Ensembles	— V_2 : Ensemble de nœuds, partitionné en trois sous-ensembles : $V_2 = D \cup N \cup \{0, l\}$. — D : Ensemble des dépôts. — N : Ensemble des tâches à réaliser. — $\{0, l\}$: Nœuds fictifs représentant le début et la fin du réseau. — S : Ensemble des quarts, subdivisé en S_{d} pour les quarts de jour et S_{n} pour les quarts de nuit où $S = S_d \cup S_n$ et $S_d \cap S_n = \emptyset$. — A_2 : Ensemble d'arcs subdivisé en : — $A_{0D} = \{(0, d) : d \in D\}$: Arcs du point de départ vers les dépôts. — $A_R = \{(i, j) : i \in D, j \in N\} \cup \{(i, j) : i, j \in N, i \neq j\} \cup \{(i, j) : i \in N, j \in D\}$. — $A_{Dl} = \{(d, l) : d \in D\}$.

Suite page suivante

Catégorie	Description
Paramètres	— c_{ij} : Coûts associés à chaque arc (i, j). — τ_{ij} : Temps de trajet pour chaque arc (i, j). — $\sigma_i = \delta_i$: Temps nécessaire pour réaliser la tâche i, égal à zéro si $i \in D \cup \{0, l\}$. — Γ_s : Durée maximale d'un quart s $\in S$
Variables de décision	— $z_{ij}^s \in \{0, 1\}$: 1 si j est visité après i dans le quart s, 0 sinon. — $g_i^s \geq 0$: le moment du balayage au nœud i dans le quart s, valide seulement si i est visité pendant le quart s.

TABLEAU 3.2 Notations et structure du modèle de tournées sur nœuds

Le deuxième modèle mathématique proposé est présenté ci-dessous :

$$
(P_2) = \begin{cases}
\min Z_2 = \displaystyle\sum_{s\in S}\sum_{(i,j)\in A_2} c_{ij} z_{ij}^s & & (24)\\
\text{s.c.} \displaystyle\sum_{s\in S}\sum_{i\in D\cup N} z_{ij}^s = 1 & \forall j \in O & (25)\\
\displaystyle\sum_{s\in S_n}\sum_{i\in D\cup N} z_{ij}^s = 1 & \forall j \in H & (26)\\
\displaystyle\sum_{d\in D} z_{0d}^s \leq 1 & \forall s \in S & (27)\\
\displaystyle\sum_{d\in D} z_{0d}^s = \sum_{d\in D} z_{dl}^s & \forall s \in S & (28)\\
z_{0d}^{s+1} \leq z_{dl}^s & \forall d \in D, \forall s \in S & (29)\\
\displaystyle\sum_{i:(i,k)\in A_{0D}\cup A_R} z_{ik}^s = \sum_{j:(k,j)\in A_R\cup A_{Dl}} z_{kj}^s & \forall k \in D\cup N, \forall s \in S & (30)\\
g_0^s = 0 & \forall s \in S & (31)\\
g_i^s \leq \Gamma_s \displaystyle\sum_{(i,j)\in A_R} z_{ij}^s & \forall i \in D\cup N, \forall s \in S & (32)\\
g_l^s \leq \Gamma_s & s \in S & (33)\\
g_i^s + \sigma_i + \tau_{ij} - g_j^s \leq (\Gamma_s + \sigma_i + \tau_{ij})(1 - z_{ij}^s) & \forall (i,j) \in A_2, \forall s \in S & (34)\\
z_{ij}^s = 0 & \forall (i,j) \in A_2, \forall s \in S : \sigma_i + \tau_{ij} > \Gamma_s & (35)\\
z_{ij}^s \in \{0, 1\} & \forall (i,j) \in A_2, \forall s \in S & (36)\\
g_i^s \geq 0 & \forall i \in V_2, \forall s \in S. & (37)
\end{cases}
$$

La fonction objectif (24) minimise la somme des coûts fixes associés aux quarts de travail ainsi que les coûts variables entre les tâches. D'une part, les contraintes (25) et (26) garantissent que chaque tâche est effectuée exactement une fois durant un quart approprié. D'autre part,

les contraintes (27) et (28) stipulent qu'un quart est utilisé uniquement si la flotte de balayage quitte le terminal de départ pour aller à un dépôt et doit entrer au terminal de fin depuis un dépôt à la fin du quart. Par ailleurs, la contrainte (29) capture l'interdépendance entre les quarts en imposant que le dépôt où un quart se termine soit le même que celui où le quart suivant commence. En outre, la contrainte (30) assure que des chemins consécutifs sont formés pour chaque quart. De plus, les contraintes (31) à (34) garantissent que les variables g_i^s suivent le temps écoulé depuis le début d'un quart et que la durée maximale de tout quart ne soit jamais dépassée. Elles permettent également d'éliminer les sous-tours. Enfin, les contraintes (35) à (37) définissent les domaines des variables de décision.

3.3 Proposition d'une formulation par chemin

Dans cette section, nous introduisons une reformulation mathématique du problème de balayage, fondée sur la formulation par route, aussi appelée formulation basée sur les chemins. C'est une approche où chaque variable de décision représente une route entière, c'est-à-dire une séquence de tâches à réaliser qui débute dans un dépôt et se termine, éventuellement, dans un autre dépôt. Cette flexibilité permet de mieux modéliser les transitions entre les quarts de travail. Ce type de formulation présente l'avantage de mieux gérer la complexité des modèles traditionnels en décomposant le problème original en problème maître et sous-problème faciles à résoudre [31].

3.3.1 Notations

Catégorie	**Description**
Ensembles	— R : Ensemble de toutes les routes possibles. — $r \in$ R : Une route r correspond à une séquence de tâches débutant et se terminant au dépôt, assignée à un quart de travail spécifique. — D=$D_J \cup D_N$: Ensemble des dépôts où D_J l'ensemble des dépôts de jour et D_N est l'ensemble des dépôts de nuit, i.e., les routes commencent le soir (6.PM par exemple). — $\{s, t\}$: Nœuds fictifs (s : source, t : puits) représentant respectivement le début et la fin du réseau. — A_3 : Ensemble d'arcs subdivisé en : — $A_{sD} = \{(s, d) : d \in D\}$: Arcs de la source s aux dépôts. — $A_D = \{(d, d') : d, d' \in D\}$: Arcs entre les dépôts (s'il existe une route directe de d à d'). — $A_{Dt} = \{(d, t) : d \in D\}$. — T : Ensemble de toutes les tâches, divisé en T_{AH} pour les tâches de type autoroute AH et T_{AR} pour les taches de type autres routes AR.

Suite page suivante

Catégorie	Description
Paramètres	— a_{ir} : Prend la valeur 1 si la route r exécute la tâche i, 0 sinon. — $b^r_{dd'}$: Vaut 1 si la route r commence au dépôt d et termine au dépôt d', 0 sinon. — C_r : Coût de la route r incluant le coût fixe de l'utilisation d'un quart et le coût variable entre les tâches.
Variables de décision	— θ_r : 1 si la route r est choisie, 0 sinon. — χ_{ij} : 1 si l'arc (i, j) est choisi, 0 sinon. $\forall (i, j) \in A_{sD} \cup A_{Dt}$.

TABLEAU 3.3 Notations

3.3.2 Représentation graphique

Suite à notre reformulation mathématique qui présente chaque variable de décision comme une route distincte, nous proposons une visualisation sous forme de méta-graphe $G' = (V', A')$, où nous redéfinissons la structure classique du problème de tournées, les dépôts sont définis comme des nœuds (V' est l'ensemble des dépôts), et les routes comme des arcs qui relient ces nœuds (A' est l'ensemble des arcs). La figure 3.1 illustre un exemple de méta-graphe pour notre formulation par chemin.

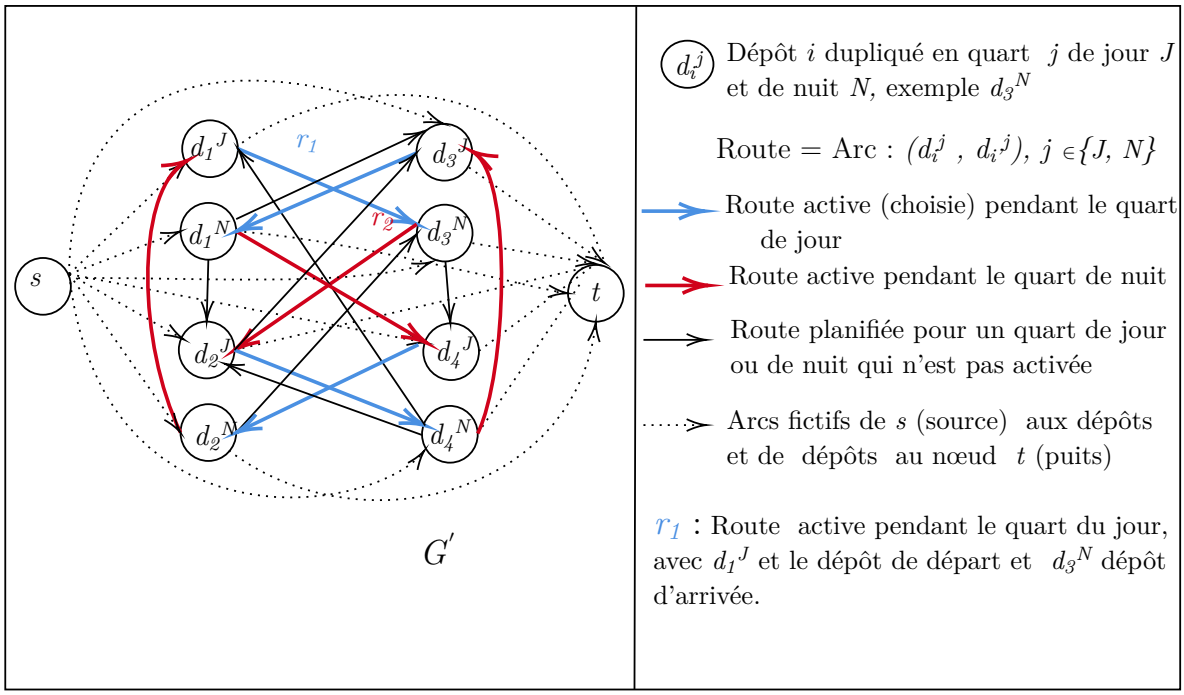


FIGURE 3.1 Exemple de méta-graphe G' avec 4 dépôts et un ensemble de routes

Cette figure 3.1 illustre un méta-graphe du système de routage pour le balayage des rues, où les dépôts d_i^J et d_i^N sont représentés comme des nœuds, chaque dépôt est dupliqué pour distinguer les activités diurnes et nocturnes. Les routes, notées r_j, servent d'arcs et sont colorées en bleu pour les quarts de jour et en rouge pour les quarts de nuit, illustrant l'activation séquentielle des routes. Par exemple, l'arc r_1 montre que la route r_1 est active du dépôt d_1^J à d_2^N durant le quart de jour. La transition entre les quarts est visualisée par le passage d'un arc bleu à un rouge, soulignant la continuité opérationnelle. Les arcs noirs représentent les routes planifiées mais non sélectionnées. Les connexions entre les dépôts de jour uniquement ou de nuit uniquement sont supprimées, interdisant ainsi la succession de deux quarts identiques.

3.3.3 Modèle mathématique MIP3

Le modèle mathématique proposé est présenté ci-dessous :

$$
(P_3) = \begin{cases}
\min \displaystyle\sum_{r \in R} C_r \theta_r & & (38) \\
\text{s.c.} \quad \displaystyle\sum_{r \in R} a_{ir} \theta_r = 1 & \forall i \in T & (39) \\
\displaystyle\sum_{d' : (d,d') \in A_D} \sum_{r \in R} b^r_{dd'} \theta_r + \chi_{dt} - \sum_{d' : (d',d) \in A_D} \sum_{r \in R} b^r_{d'd} \theta_r - \chi_{sd} = 0 & \forall d \in D & (40) \\
\displaystyle\sum_{d : (s,d) \in A_{sD}} \chi_{sd} = 1 & & (41) \\
\displaystyle\sum_{d : (d,t) \in A_{Dt}} \chi_{dt} = 1 & & (42) \\
\theta_r \in \{0,1\} & \forall r \in R & (43) \\
\chi_{ij} \in \{0,1\} & \forall (i,j) \in A_{sD} \cup A_{Dt} & (44)
\end{cases}
$$

La fonction objectif (38) minimise le coût total des routes sélectionnées. La contrainte (39) assure que chaque tâche est exécutée exactement une fois, c'est-à-dire assignée à une seule route au cours d'un quart. La contrainte (40) garantit que pour chaque dépôt, le flux entrant est égal au flux sortant, et garantit qu'à chaque dépôt, le nombre de routes qui se terminent est égal au nombre de routes qui en partent. Cette contrainte a été introduite pour assurer la continuité opérationnelle entre les quarts de travail : la fin d'un quart doit coïncider avec le début du suivant, ce qui est modélisé par le lien direct entre les dépôts d'arrivée et de départ. De même, les contraintes (41) et (42) veillent à ce que le flux total partant de la source vers les dépôts de jour soit égal à 1, et que le flux total allant des dépôts vers le puits soit également 1, assurant ainsi un unique flux actif de début à fin. Enfin, les contraintes (43) et (44) définissent le domaine des variables de décision.
Cette formulation par chemin sert de base à la méthodologie de résolution proposée, qui sera détaillée dans le chapitre suivant.

CHAPITRE 4 Méthodologie de résolution

Dans ce chapitre, nous présentons la méthodologie adoptée pour atteindre les objectifs de recherche décrits dans la section 1.3. Cette méthodologie repose sur une combinaison d'approches exactes et heuristiques adaptées à la résolution du problème étudié. Notre travail constitue avant tout une preuve de concept, visant principalement à l'amélioration de la borne inférieure, un aspect crucial puisque les bornes obtenues avec CPLEX sont trop faibles. Les métaheuristiques comme GRASP fournissent rapidement une bonne solution, mais sans aucune garantie sur leur qualité, ce qui justifie l'utilisation d'une approche complémentaire inspirée du principe de *Branch-and-Price*, combinant la génération de colonnes et une heuristique de branchement, pour renforcer la borne inférieure et mieux encadrer la solution optimale. Nous avons choisi d'utiliser GRASP et l'heuristique de branchement en profondeur (DH : *Diving Heuristic*) car elles sont plus faciles à implémenter dans le cadre de cette preuve de concept et permettent de montrer le potentiel de *Branch-and-Price*.
Le reste du chapitre est structuré comme suit. Dans la section 4.1, nous décrivons la phase de construction de GRASP, qui génère des solutions initiales réalisables. La section 4.2 introduit la décomposition de Dantzig-Wolfe, en détaillant à la fois le Problème Maître (PM) et les Sous-Problèmes (SP_s), y compris les variantes indexées par type de quart (jour/nuit). Ensuite, la section 4.3 présente le processus de génération de colonnes, ainsi que l'heuristique de branchement utilisée pour obtenir des solutions entières. Enfin, la section 4.4 présente une phase de post-optimisation permettant de déterminer les horaires sur l'horizon de planification.

4.1 Phase de Construction GRASP

Pour générer un ensemble de routes initiales, nous nous sommes inspirés de la phase de construction de la métaheuristique GRASP (*Greedy Randomized Adaptive Search Procedure*), telle que présentée par Resende et al. [34].

4.1.1 Principe de la phase de construction GRASP

La phase de construction de GRASP repose sur un aspect glouton et un aspect aléatoire pour générer une solution initiale. Dans le cadre du problème étudié, chaque solution correspond à un ensemble de routes réalisables. À chaque itération, une liste de candidats est constituée, contenant les éléments (dans notre cas, des tâches) pouvant être ajoutés à une tournée sans violer les contraintes du problème. Ces éléments sont évalués selon une fonction gloutonne, et les meilleurs sont regroupés dans une liste restreinte de candidats (*RCL : Restricted Candidate List*). Un élément est ensuite sélectionné aléatoirement dans la *RCL* afin d'introduire de la diversité dans la construction et ajouté à la solution courante. Après l'ajout de chaque élément, la liste des candidats est mise à jour, et les

critères d'évaluation des éléments restants sont recalculés. Ces critères permettent de déterminer quels éléments sont les plus intéressants à ajouter ensuite, en tenant compte des aspects comme le coût. Ce processus est répété jusqu'à ce qu'aucun nouvel élément ne puisse être ajouté. Si la solution obtenue n'est pas réalisable, une procédure de réparation est appliquée pour garantir sa faisabilité. L'algorithme 1 illustre le fonctionnement de cette phase de construction, adapté de [34].

Algorithm 1 Phase de construction de GRASP

1: **Entrée :** Ensemble des éléments candidats
2: **Sortie :** Solution construite
3: Initialiser la solution $\mathcal{S} \leftarrow \emptyset$
4: Initialiser l'ensemble des candidats $\mathcal{L}$
5: Évaluer les coûts incrémentaux des éléments de $\mathcal{L}$
6: **while** $\mathcal{L} \neq \emptyset$ **do**
7: Construire RCL
8: Sélectionner un élément s aléatoirement dans RCL
9: Ajouter s à la solution $\mathcal{S}$
10: Mettre à jour l'ensemble des candidats $\mathcal{L}$
11: Mettre à jour l'évaluation des coûts incrémentaux
12: **end while**
13: **Retourner** $\mathcal{S}$

La phase de construction de GRASP commence par une initialisation de la solution vide et de l'ensemble des candidats (lignes 3 à 5 de l'Algorithme 1). Ensuite, les coûts d'ajout des éléments candidats sont évalués (ligne 5), ce qui permet de construire une liste restreinte de candidats (RCL) contenant les meilleurs éléments(ligne 7). Un élément est ensuite sélectionné aléatoirement dans la RCL pour favoriser la diversité (ligne 8), puis ajouté à la solution courante et immédiatement retiré de la liste des candidats (ligne 9). À chaque itération, l'ensemble des candidats est mis à jour (ligne 10) et les coûts d'ajout des tâches restantes sont recalculés pour refléter l'impact des choix effectués (ligne 11). Ce processus se poursuit jusqu'à ce qu'il ne reste plus de candidats valides (ligne 6). Enfin, la solution construite est retournée (ligne 13).

4.1.2 Adaptation au problème étudié

En s'appuyant sur l'algorithme 1, nous avons adapté la phase de construction de GRASP au problème de balayage printanier. L'objectif est de générer un ensemble de routes, chacune correspondant à un quart de travail. Une route commence à un dépôt, effectue une série de tâches (les rues à balayer), et retourne à un dépôt, en respectant la durée maximale Γ. Les éléments candidats sont les tâches, sélectionnées de manière séquentielle tout en respectant les contraintes de durée et de connectivité avec les dépôts. À chaque étape, une liste restreinte de candidats (RCL) est construite selon un critère de coût, puis une tâche est choisie aléatoirement dans cette liste et insérée dans la route si la contrainte de durée est respectée. Le pseudo-code implémenté est présenté dans l'algorithme 2.

Algorithm 2 Phase de construction de GRASP pour le problème de balayage printanier

Entrée :
    T - Ensemble des tâches
    D - Ensemble des dépôts
    Γ - Durée maximale d'un quart de \\ Durée maximale autorisée pour une route
    k - Taille de RCL
    nbR - Nombre maximum de routes par tâche \\ Critère d'arrêt
Sortie : R - Ensemble des routes réalisables \\ Liste des routes générées
Initialiser $R \leftarrow \emptyset$
for chaque dépôt $d \in D$ **do** \\ Compteur des routes générées pour ce dépôt
    $T_{\text{restant}} \leftarrow \emptyset$
    for chaque tâche $t \in T$ **do**
        total_routes $\leftarrow 0$ \\ Nombre de routes générées pour cette tâche
        continuer_génération $\leftarrow$ **vrai**
        while *continuer_génération* et *total_routes* $< nbR$ **do**
            Initialiser une nouvelle route $r \leftarrow \{d, t\}$ \\ Démarrer avec le dépôt d et t
            Calculer la durée et le coût de r
            Construire $T_{\text{restant}} \leftarrow T \setminus \{t\}$ \\ Mise à jour des tâches restantes
            while durée(r) $< \Gamma$ et $T_{\text{restant}} \neq \emptyset$ **do** \\ Ajout de nouvelles tâches
                Construire une liste de candidats faisables C
                $RCL \leftarrow$ Sélectionner les k meilleures tâches de C selon $f(t)$
                Sélectionner aléatoirement $t_{\text{suiv}} \in RCL$
                if durée(r) + durée(t_{suiv}) + durée retour dépôt $\leq \Gamma$ **then**
                    Ajouter t_{suiv} à r \\ Ajout de la tâche suivante t_{suiv} à la route r
                    Mettre à jour $T_{\text{restant}} \leftarrow T_{\text{restant}} \setminus \{t_{\text{suiv}}\}$
                else
                    Quitter la boucle \\ Arrêt si aucune tâche ne peut être ajoutée
                end if
            end while
            if aucune tâche ne peut être ajoutée à cause de Γ **then**
                Terminer la route en ajoutant le dépôt le plus proche
                Ajouter r à R
                Mettre à jour *total_routes*
            end if
            Phase complémentaire de génération \\ Voir explication 4.1.2
            if *total_routes* $\geq nbR$ **then**
                continuer_génération $\leftarrow$ **faux**
            end if
        end while
    end for
end for
Retourner R \\ Retourne toutes les routes générées

L'Algorithme 2 décrit la phase de construction de GRASP pour le problème étudié. Il commence par l'initialisation de l'ensemble vide des routes R (ligne 7). Ensuite, pour chaque dépôt d (ligne 9) et pour chaque tâche $t \in T$ (ligne 11), plusieurs routes sont générées tant que le nombre maximal de routes par tâche nbR n'est pas atteint (ligne 14). La durée de la route r est mise à jour en tenant compte du temps de service de t et du temps de déplacement depuis le dépôt (ligne 16). Tant que la durée (r) ne dépasse pas Γ et qu'il reste des tâches à assigner, une liste de candidats C est constituée avec les tâches non attribuées pouvant être ajoutées sans violer la contrainte de durée maximale. Chaque tâche candidate t' est évaluée selon une fonction de coût $f(t')$, définie comme la somme du coût de déplacement entre la tâche t et la tâche t', additionnée au temps de service de t' (lignes 19-20). (RCL) est ensuite formée en sélectionnant les k meilleures tâches selon $f(t')$, c'est-à-dire celles ayant le coût minimal parmi les candidats. Parmi ces tâches, une tâche t_{suiv} est choisie aléatoirement dans la RCL (ligne 20-21). Avant son ajout, la faisabilité de la route est vérifiée en considérant non seulement son impact sur la durée(r), mais aussi le temps nécessaire pour retourner au dépôt le plus proche (ligne 22). Si l'ajout de t_{suiv} suivi du retour au dépôt dépasse Γ, la tâche est rejetée et la route est complétée en ajoutant le dépôt de retour le plus proche (cela permet d'éviter la nécessité d'une procédure de réparation). Sinon, la tâche est ajoutée à la route et la liste des tâches restantes est mise à jour (lignes 23-27). Si aucune tâche ne peut être ajoutée à cause de la contrainte de durée, la route est complétée en ajoutant le dépôt le plus proche. Cette route est ensuite ajoutée à R. Ensuite, commence la phase complémentaire de génération (ligne 34). Cette étape vise à assurer la continuité des quarts de travail et à garantir que l'ensemble des tâches soit couvert. À partir des tâches restantes non encore attribuées, de nouvelles routes sont générées en redémarrant la construction depuis le dépôt de fin de la route précédente. Une tâche de départ est choisie aléatoirement parmi les tâches restantes, puis une route est construite selon le même principe que dans la phase principale (lignes 16-31). Ce processus est répété tant qu'il y a des tâches non assignées. Le compteur nbR n'est pas mis à jour dans cette phase, car il est utilisé uniquement pour limiter le nombre de routes générées à partir d'une même tâche initiale dans la phase principale. Si le nombre total de routes générées atteint nbR pour une tâche, la génération s'arrête pour cette tâche et passe à la suivante (35-37). Une fois les routes générées pour tous les dépôts, l'ensemble des routes réalisables R est retourné (ligne 41).

4.1.3 Approche GRASP-MIP3 : Résolution basée sur les routes générées

À l'issue de la phase de construction de GRASP, les routes générées ont servi à peupler le modèle MIP3 (P_3), en fournissant une solution réalisable au problème (borne supérieure) en sélectionnant les meilleures routes, tout en permettant d'estimer la borne inférieure via la relaxation linéaire. De plus, le nombre de quarts trouvés correspond au nombre de quarts utilisés dans le modèle de CPLEX (P_2) présenté à la section 3.2.1. Cette méthode hybride permet d'évaluer la pertinence des routes construites par GRASP dans un cadre pratique. Elle est particulièrement utile dans les cas où la génération de colonnes est trop coûteuse ou lente à converger.

4.2 Décomposition de Dantzig-Wolfe

4.2.1 Problème Maître *(PM)*

Le problème maître modélise l'ensemble de notre problème, incluant toutes les variables et contraintes nécessaires pour obtenir une solution optimale de la relaxation continue. Ces variables représentent des chemins partant d'un dépôt et se terminant à un autre, chacun correspondant à une route dans le réseau. Le modèle mathématique (P_3) présenté dans la section 3.3.3 correspond à cette formulation et sert de base pour notre analyse et résolution du problème. Pour faciliter la lecture de ce chapitre, nous recopions ci-dessous le modèle mathématique : $(PM) = (P_3)$

$$
(PM) = \begin{cases}
\min \displaystyle\sum_{r \in R} C_r \theta_r & & (38) \\
\text{s.t.} \quad \displaystyle\sum_{r \in R} a_{ir} \theta_r = 1 & \forall i \in T & (39) \\
\displaystyle\sum_{d':(d,d') \in A_D} \sum_{r \in R} b^r_{dd'} \theta_r + \chi_{dt} - \sum_{d':(d',d) \in A} \sum_{r \in R} b^r_{d'd} \theta_r - \chi_{sd} = 0 & \forall d \in D & (40) \\
\displaystyle\sum_{d:(s,d) \in A_{sD}} \chi_{sd} = 1 & & (41) \\
\displaystyle\sum_{d:(d,t) \in A_{Dt}} \chi_{dt} = 1 & & (42) \\
\theta_r \in \{0,1\} & \forall r \in R & (43) \\
\chi_{ij} \in \{0,1\} & \forall (i,j) \in A_{sD} \cup A_{Dt} & (44)
\end{cases}
$$

Le Problème Maître Restreint (PMR) est une version simplifiée du problème maître. Alors que le Problème Maître inclut initialement toutes les variables, le PMR commence avec un sous-ensemble limité de ces variables. Cette approche rend le problème original facile à résoudre. À chaque itération du processus, de nouvelles variables, de coût réduit négatif, sont introduites dans le PMR. Ces variables sont générées par les sous-problèmes, permettant ainsi une amélioration progressive de la solution. L'objectif est d'augmenter de manière contrôlée le PMR jusqu'à convergence, où aucune variable de coût réduit négatif ne peut être ajoutée, indiquant ainsi que la solution optimale est atteinte.

4.2.2 Sous-Problème *(SP)*

Dans notre approche, nous considérons deux sous-problèmes distincts : un sous-problème (SP_J) pour le quart de jour et un sous-problème (SP_N) pour le quart de nuit. Cette distinction est nécessaire en raison des contraintes spécifiques associées aux types de tâches : les tâches de type AH (autoroute) doivent être réalisées la nuit, tandis que les tâches de type AR (autres routes) peuvent

être réalisées le jour ou la nuit. Ainsi, le SP_N inclut les deux types de tâches, alors que celui de jour est un sous-ensemble du SP_N qui ne contient que les tâches de type AR. Pour modéliser cette distinction, nous définissons un sous-problème (SP_q), indexé par le type de quart q, où $q \in \{J, N\}$. Le graphe associé au SP_J est un graphe induit du SP_N, obtenu en ne conservant que les nœuds (tâches) de type AR. Nous présentons ci-après les graphes associés aux deux sous-problèmes ainsi que deux exemples illustrant cette distinction (figures 4.1 et 4.2).

Graphe de jour $G_J = (V_J, A_J)$

- $V_J = T_O \cup D \cup \{s, t\}$.
- $A_J = \{(i, j) : i, j \in V_J, i \neq j\}$.
- Ensemble de tâches $T_J = T_O$.

Graphe de nuit $G_N = (V_N, A_N)$

- $V_N = T_O \cup D \cup T_H \cup \{s, t\}$.
- $A_N = \{\ (i, j) : i, j \in V_N, i \neq j\}$.
- Ensemble de tâches $T_N = T_O \cup T_H$.

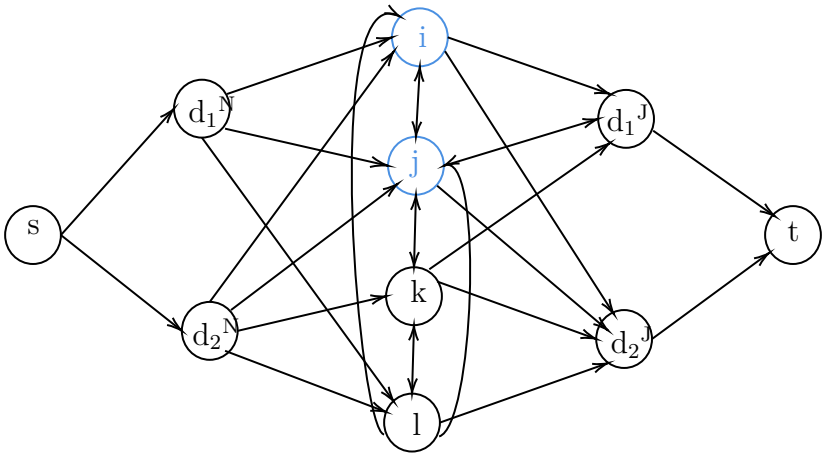


FIGURE 4.1 Un petit exemple de graphe de nuit avec deux dépôts et quatre tâches : $\{i, j\}$ de jour et $\{k, l\}$ de nuit

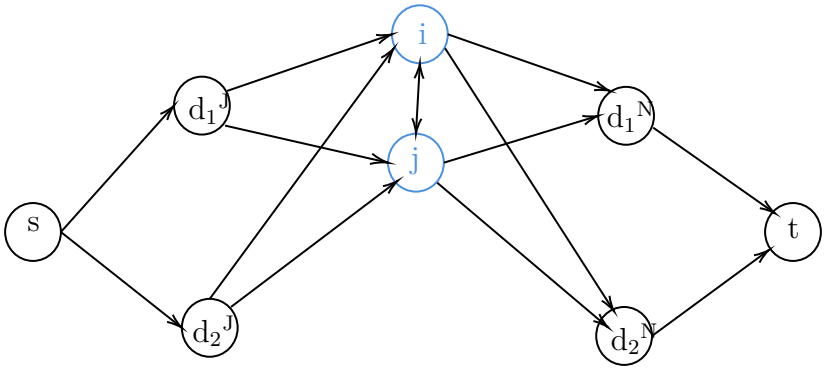


FIGURE 4.2 Un petit exemple de graphe de jour avec deux dépôts et deux tâches : $\{i, j\}$

Les figures 4.1 et 4.2 illustrent deux graphes représentant des activités de jour et de nuit respectivement. Le graphe de jour est un sous-ensemble du graphe de nuit car les tâches de jour sont aussi incluses dans le graphe de nuit, chaque graphe comportant le même nombre de dépôts. La différence principale réside dans les tâches à exécuter : dans le graphe de jour, les tâches $\{i, j\}$ de type AR sont représentées en bleu et correspondent aux autres routes à emprunter à tout moment du quart (jour ou nuit). Le graphe de nuit inclut à la fois les tâches $\{i, j\}$ de type AR et les tâches $\{k, l\}$ de type AH empruntées pendant le quart de nuit. Les nœuds s et t sont fictifs, représentent les nœuds source et puits.

Nous formulons le SP_q, où $q \in \{J, N\}$, sous forme d'un programme linéaire en nombres entiers, défini avec des variables représentant les arcs du réseau et s'appuie sur les valeurs duales obtenues à partir du problème maître restreint PMR. Nous obtenons les valeurs duales π_i associées aux contraintes de partitionnement (39) et ϕ_d aux contraintes de conservation de flot (40) en résolvant PMR. Ainsi, le coût réduit d'une variable θ_r est :

$$\bar{C}_r = C_r - \sum_{i \in T} a_{ir}\pi_i - \sum_{d \in D} \left(\sum_{d' : (d,d') \in A_D} b^r_{dd'} - \sum_{d' : (d',d) \in A_D} b^r_{d'd} \right) \phi_d$$

.

Comme une route r est un ensemble d'arcs $(i, j) \in A$, on peut donc introduire la variable x_{ij} égale à 1 si l'arc (i, j) est dans la route r, 0 sinon. Ainsi, on peut écrire le coût réduit de l'arc (i, j) comme suit :

$$\bar{C}_{ij} = \begin{cases} c_{ij} - \pi_i - \pi_j & \text{si } (i,j) \in A_T \\ c_{ij} - \phi_i & \text{si } (i,j) \in A_{DI} \\ c_{ij} + \phi_j & \text{si } (i,j) \in A_{ID} \end{cases}$$

Où T est l'ensemble des tâches, D l'ensemble des dépôts et A l'ensemble des arcs, subdivisé en :

- $A_T = \{(i, j) : i, j \in T, i \neq j\}$
- $A_{DI} = \{(i, j) : i \in D, j \in T\}$
- $A_{ID} = \{(i, j) : i \in T, j \in D\}$

Pour illustrer ce calcul de coût réduit, la figure 4.3 présente un exemple simple avec deux tâches et deux dépôts.

Nous présentons ensuite le modèle mathématique du sous-problème (SP_q), où $q \in \{J, N\}$.
Les ensembles V_q, D_q et A_q, où $q \in \{J, N\}$, sont définis dans la représentation de chaque graphe de jour G_J ou de nuit G_N.

Variables de Décision

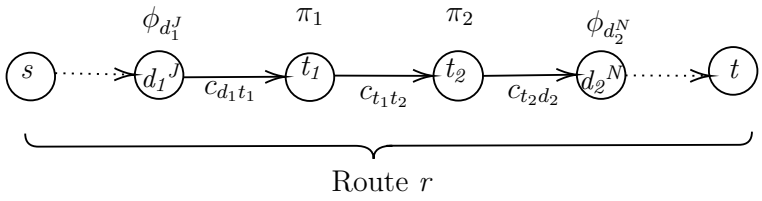
— x_{ij} : Variable binaire égale à 1 si la tâche j est visitée immédiatement après le tâche i. 0 sinon.

Le modèle du SP_q, où $q \in \{J, N\}$, est défini comme suit :

$$
(SP_q) = \begin{cases}
\text{Min} \quad \displaystyle\sum_{(i,j)\in A_q} \bar{C}_{ij} \quad x_{ij} & & (45)\\
\text{s.t.} \quad \displaystyle\sum_{j:(i,j)\in A_q} x_{ij} - \sum_{j:(j,i)\in A_q} x_{ji} = 0, & \forall i \in V_q \setminus D_q & (46)\\
\displaystyle\sum_{(i,j)\in A_q} (\sigma_i + \tau_{ij}) \quad x_{ij} \leq \Gamma & & (47)\\
\displaystyle\sum_{i:(s,i)\in A_q} x_{si} = 1 & & (48)\\
\displaystyle\sum_{i:(i,t)\in A_q} x_{it} = 1 & & (49)\\
\displaystyle\sum_{i}\sum_{j} x_{ij} \leq |S| - 1 & q \subset V_q, 2 \leq |S| \leq |V_q| - 2 & (50)\\
x_{ij} \in \{0, 1\}, & \forall (i,j) \in A_q & (51)
\end{cases}
$$

La contrainte (46) garantit que, pour chaque nœud (à l'exception des dépôts de départ et d'arrivée), le nombre d'entrées est égal au nombre de sorties, ce qui permet de former des routes continues tout en éliminant les sous-tours. La contrainte (47) assure que la durée totale pour effectuer les tâches, incluant les temps de déplacement et les temps de service des tâches, ne dépasse pas la durée maximale Γ autorisée pour un quart de travail. La contrainte (48) impose qu'un seul arc sort du nœud fictif source (s), tandis que la contrainte (49) garantit qu'un seul arc entre dans le nœud fictif puits (t). Cela permet de définir le début et la fin du chemin. La contrainte (50) permet d'éliminer les sous-tours en assurant que les arcs sélectionnés ne forment pas de cycles fermés sur un sous-ensemble de nœuds (S). Enfin, la contrainte (51) définit les variables de décision comme étant binaires.
Il y a deux sous-problèmes à résoudre, chacun visant à trouver une nouvelle route de coût réduit négatif. Ils sont modélisés comme des problèmes de plus court chemin avec contraintes de ressources (PCCCR). La ressource considérée dans notre cas est la longueur du quart de travail (contrainte (47)). La résolution repose sur un algorithme d'étiquetage, basé sur la programmation dynamique. À chaque nœud, plusieurs étiquettes peuvent être générées, représentant des chemins partiels, conte-

nant des informations telles que le coût, le temps consommé et les tâches visitées. Une règle de dominance est appliquée pour éliminer les étiquettes dominées, c'est-à-dire celles qui sont moins efficaces que d'autres sur les mêmes critères. Toutefois, la difficulté majeure de l'algorithme réside dans le fait que la présence de cycles ou de chemins longs peut engendrer une explosion du nombre d'étiquettes à gérer [35].



$$\begin{aligned}\tilde{C}_r = {} & C_r - \textit{contribution duale (de chaque tâche } t_i) - \textit{contribution duale (de chaque dep\^ot } d) \\ & = (c_{d_1t_1} + c_{t_1t_2} + c_{t_2d_2}) - \pi_1 - \pi_2 - [\phi_{d_1^J} - \phi_{d_2^N}] \\ & = (c_{t_1t_2} - \pi_1 - \pi_2) + (c_{d_1t_1} - \phi_{d_1^J}) + (c_{t_2d_2} + \phi_{d_2^N})\end{aligned}$$

t_i *Tâche i*	$c_{t_1t_2}$ Coût de l'arc (t_1,t_2)
$d_1{}^J$ Depôt où commence la route r	π_i Variable duale associée à la tâche i
$d_1{}^N$ Depôt où termine la route r	$\phi_{d_j^Q}$ Variable duale associée au depôt d_j^Q, $Q \in \{J, N\}$
	$\tilde{C}_r$ Coût réduit de la route r

FIGURE 4.3 Exemple de calcul du coût réduit pour une route avec deux tâches et deux dépôts

4.3 Génération de colonnes et l'heuristique de branchement

4.3.1 Génération de colonnes

L'algorithme de génération de colonnes présenté ici s'appuie sur les principes décrits dans les sections précédentes pour résoudre le problème étudié de manière itérative. Les bases théoriques de cet algorithme, notamment la définition du Problème Maître Restreint (PMR) et des Sous-Problèmes (SP), sont présentées en détail dans la section 4.2. De même, le calcul des coûts réduits et leur importance dans l'ajout de nouvelles colonnes sont expliqués dans la section 4.2.2. En début de processus, le (PMR) est initialisé avec un sous-ensemble limité de colonnes $R_r \subset R$. Ces colonnes initiales sont issues de la phase de construction de GRASP et représentent des routes réalisables ; elles sont utilisées pour stabiliser les variables duales et réduire le nombre d'appels aux sous-problèmes, ce qui diminue le nombre d'itérations de génération de colonnes. Ce PMR initial est résolu pour définir les variables duales associées aux contraintes de couverture et de conservation de flot (39)-(40), notamment π_t, ϕ_d, qui sont ensuite utilisées pour guider la génération de nouvelles colonnes.

Deux sous-problèmes, explicités dans la section 4.2.2, sont ensuite résolus. Le sous-problème de jour (SP_J) génère des routes de jour incluant uniquement des tâches de type AR, tandis que le sous-problème nuit (SP_N) génère les routes de nuit combinant des tâches de type AH et de type AR. Chaque sous-problème est modélisé comme un problème de plus court chemin avec contraintes de ressources sur un graphe adapté, où les coûts des arcs sont ajustés en fonction des valeurs duales, comme définies dans 4.2.2. Les colonnes ainsi générées, pour lesquelles le coût réduit $\bar{C}_r$ est strictement négatif, sont rajoutées au PMR, qui est ensuite résolu avec ce nouvel ensemble enrichi de colonnes. Ce processus itératif se poursuit jusqu'à ce qu'aucune colonne avec un coût réduit négatif ne puisse être identifiée, garantissant ainsi la convergence vers une solution optimale. L'organigramme qui résume ce processus est présenté dans la figure 4.6.

4.3.2 Heuristique de branchement (Diving Heuristic, DH)

L'approche développée DH est organisée sous la forme d'un arbre de branchement. La résolution commence par la génération de colonnes classiques : le problème maître restreint (PMR) est résolu de manière itérative, en ajoutant de nouvelles colonnes tant que des colonnes avec un coût réduit négatif existent. Une fois la relaxation continue du PMR résolue à l'optimalité (c'est-à-dire lorsque tous les coûts réduits sont supérieurs ou égaux à zéro), on procède au branchement.
Le branchement est effectué sur la variable présentant la plus grande valeur fractionnaire dans la solution courante. Cela crée un nouveau nœud dans l'arbre, où certaines variables sont fixées à 1. Fixer une variable à 1 signifie que les tâches couvertes par cette colonne sont considérées comme affectées. Ces tâches sont alors éliminées dans les sous-problèmes, ce qui permet de réduire la taille des sous-problèmes résolus à chaque nœud et d'éviter la génération de colonnes redondantes. Chaque nœud de branchement est ensuite résolu à l'optimalité en relançant la génération de colonnes avec plusieurs (SP_s) réduits. Ce processus se poursuit jusqu'à atteindre une solution entière pour l'ensemble du problème. La stratégie d'accélération repose principalement sur deux points : l'élimination des tâches déjà couvertes par les colonnes fixées dans le nœud courant et la suppression des arcs associés aux tâches déjà assignées, ce qui simplifie considérablement les sous-problèmes de génération de colonnes. Ainsi, cette approche réduit la complexité des sous-problèmes. La figure 4.4 ci-dessous présente une vue d'ensemble du processus de résolution utilisé, combinant la génération de colonnes et *Diving Heuristic*.

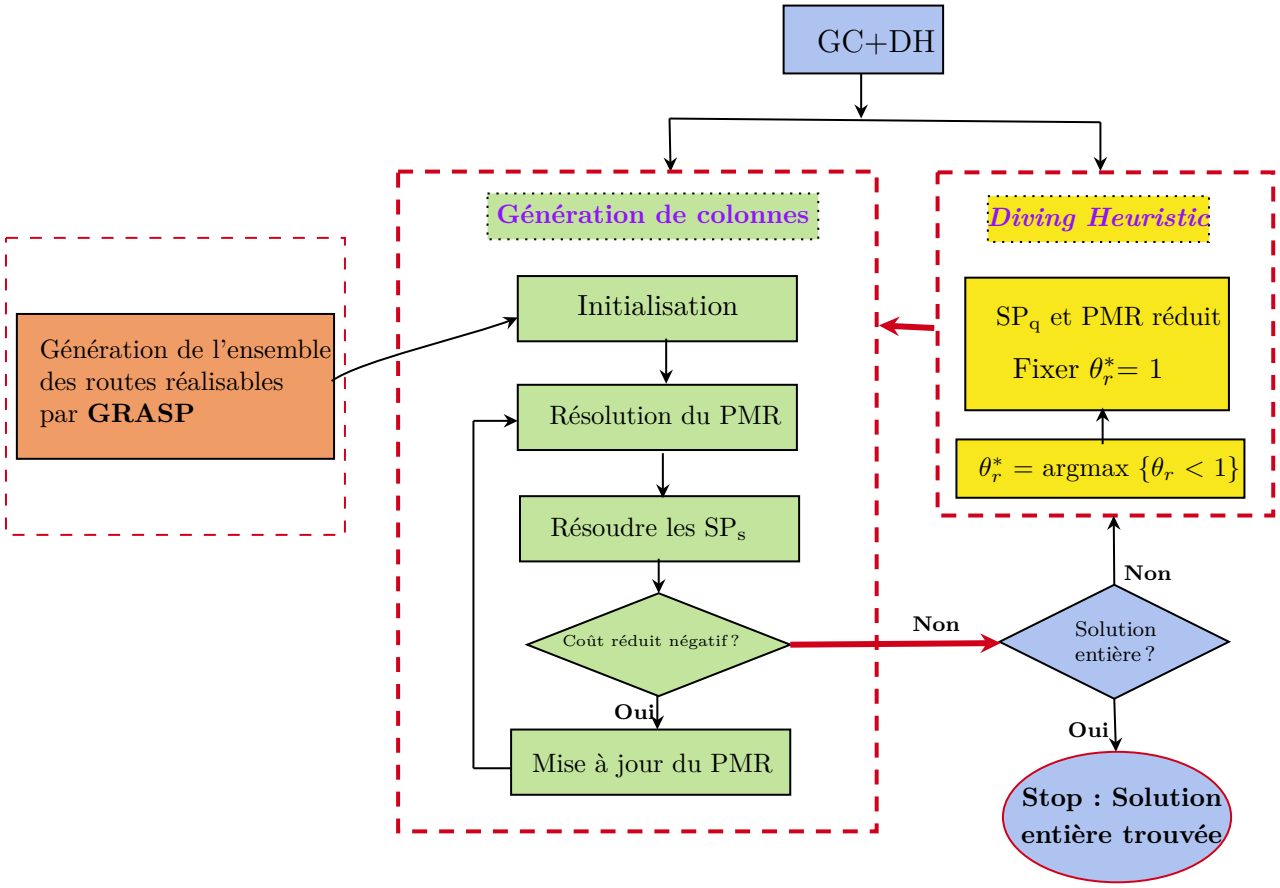


FIGURE 4.4 Architecture de l'approche : génération de colonnes et *diving heuristic (GC+DH)*

4.4 Post-optimisation des horaires sur l'horizon

Les horaires sur l'horizon de planification peuvent être déterminés lors d'une phase de post-optimisation. Après la génération des routes, correspondant aux quarts de travail nécessaires pour couvrir le réseau de balayage, les horaires précis ne sont pas définis à l'issue de la phase de résolution du problème. La post-optimisation consiste à affecter ces quarts sur un horizon de plusieurs jours, en attribuant à chacun une plage horaire fixe. Cette planification vise à éviter les conflits d'assignation et à assurer une transition fluide entre les quarts successifs. L'objectif de cette post-optimisation est :

- de faire en sorte que chaque route soit exécutée dans le créneau horaire correspondant (quart de jour ou de nuit).
- de garantir la continuité des opérations sur plusieurs jours, en veillant à ce que le dépôt de fin d'un quart soit le dépôt de départ du quart suivant.
- de planifier chaque quart dans un créneau fixe, en tenant compte des horaires définis lors de la post-optimisation.

L'affectation des quarts de travail sur l'horizon temporel peut être visualisée à l'aide du méta-graphe illustré en figure 3.1, introduit dans la section 3.3.3, et repris en figure 4.5, afin de montrer l'alternance entre les quarts ainsi que la structuration de leur répartition sur plusieurs jours.

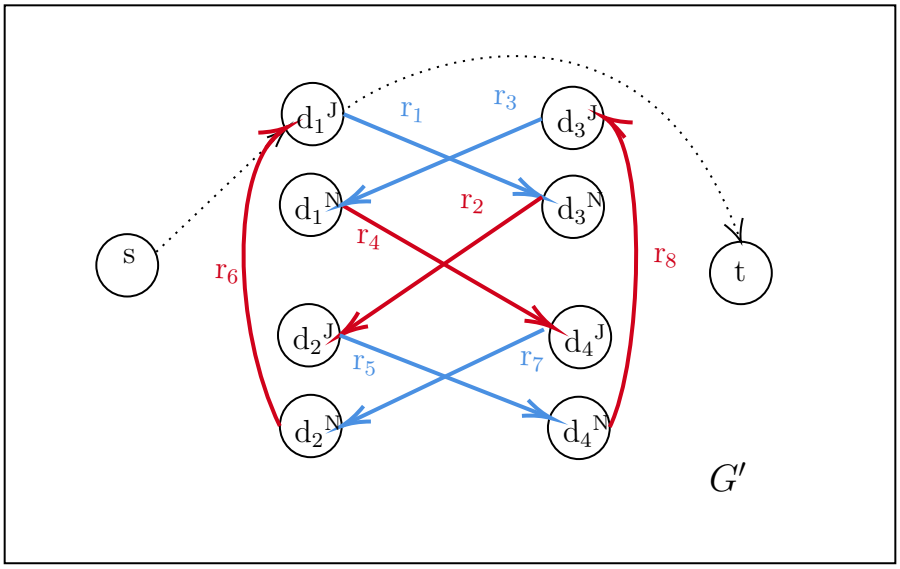


FIGURE 4.5 Méta-graphe $G^{'}$ illustrant les routes obtenues après la phase de résolution

Dans cette représentation, rappelons que chaque dépôt est dupliqué : un nœud représente le dépôt pour le quart de jour, l'autre pour le quart de nuit. Cette distinction permet de visualiser clairement la transition entre les quarts successifs. Les arcs rouges correspondent aux routes affectées aux quarts de nuit. Les arcs bleus correspondent aux routes affectées aux quarts de jour. Ainsi, la solution obtenue est constituée de 8 routes notées $(r_1, ..., r_8)$, et chaque route est associée à un quart de travail unique et effectue un ensemble de tâches et respecte la durée maximale autorisée pour un quart de travail. L'information obtenue sur les routes permet d'assigner chaque tâche à son quart d'exécution correspondant. Puisque chaque route correspond à un quart de travail, nous définissons l'ensemble des quarts comme suit : Quarts $= \{q_1, q_2, ..., q_8\}$, avec les quarts impairs (q_1, q_3, q_5, q_7) correspondant aux quarts de jour et les quarts pairs (q_2, q_4, q_6, q_8) correspondant aux quarts de nuit.

Une fois les quarts définis, il est nécessaire de déterminer leur séquence d'exécution sur l'horizon temporel. Pour cela, nous utilisons une phase de post-optimisation basée sur un problème de plus court chemin de s à t, pour déterminer la séquence optimale des routes à exécuter, où chaque route est représentée comme un nœud, et un arc entre deux routes est défini si elles sont consécutives, garantissant ainsi une transition fluide entre quarts de jour et quarts de nuit. Une fois cette séquence obtenue, les quarts de travail sont affectés en alternance. Enfin, chaque paire de quarts successifs

constitue une journée complète, permettant ainsi d'établir un *planning* structuré sur l'horizon temporel. En plus de l'affectation des routes aux quarts de travail, il est essentiel de structurer leur exécution sur l'horizon temporel. Chaque quart de travail dure 10 heures, suivi d'une pause opérationnelle de 2 heures pour la maintenance et la pause des équipes avant le début du quart suivant. Ainsi, l'horaire des quarts est défini en respectant ces intervalles. Ainsi, le tableau 4.1 présente la planification finale, en détaillant l'affectation des quarts sur plusieurs jours, en indiquant la route exécutée, le dépôt de départ et d'arrivée, les tâches couvertes et la plage horaire de chaque quart.

Jour	**Quart**	**Route assignée**	**Dépôt de départ**	**Tâches assignées**	**Dépôt d'arrivée**	**Plage horaire**
Jour 1	q_1	r_1	d_1^J	$\{t_1, t_2, t_3\}$	d_3^N	06h00–16h00
	q_2	r_2	d_3^N	$\{t_4, t_5, t_6\}$	d_2^J	18h00–04h00
Jour 2	q_3	r_5	d_2^J	$\{t_7, t_8, t_9\}$	d_4^N	06h00–16h00
	q_4	r_8	d_4^N	$\{t_{10}, t_{11}\}$	d_3^J	18h00–04h00
Jour 3	q_5	r_3	d_3^J	$\{t_{12}, t_{13}, t_{14}\}$	d_1^N	06h00–16h00
	q_6	r_4	d_1^N	$\{t_{15}, t_{16}\}$	d_4^J	18h00–04h00
Jour 4	q_7	r_7	d_4^J	$\{t_{17}, t_{18}, t_{19}\}$	d_2^N	06h00–16h00
	q_8	r_6	d_2^N	$\{t_{20}, t_{21}\}$	d_1^J	18h00–04h00

TABLEAU 4.1 Post-optimisation : Affectation des quarts de travail sur l'horizon temporel

Le tableau 4.1 permet de visualiser clairement comment les quarts de travail sont répartis sur quatre jours et comment ils s'enchaînent pour couvrir l'ensemble des tâches planifiées. Chaque quart est associé à une plage horaire bien définie, tout en garantissant une transition fluide entre les quarts successifs. La post-optimisation finalise ainsi le calendrier d'exécution des tâches (i.e., les rues à balayer), en assurant une couverture complète des tâches sur l'horizon planifié.

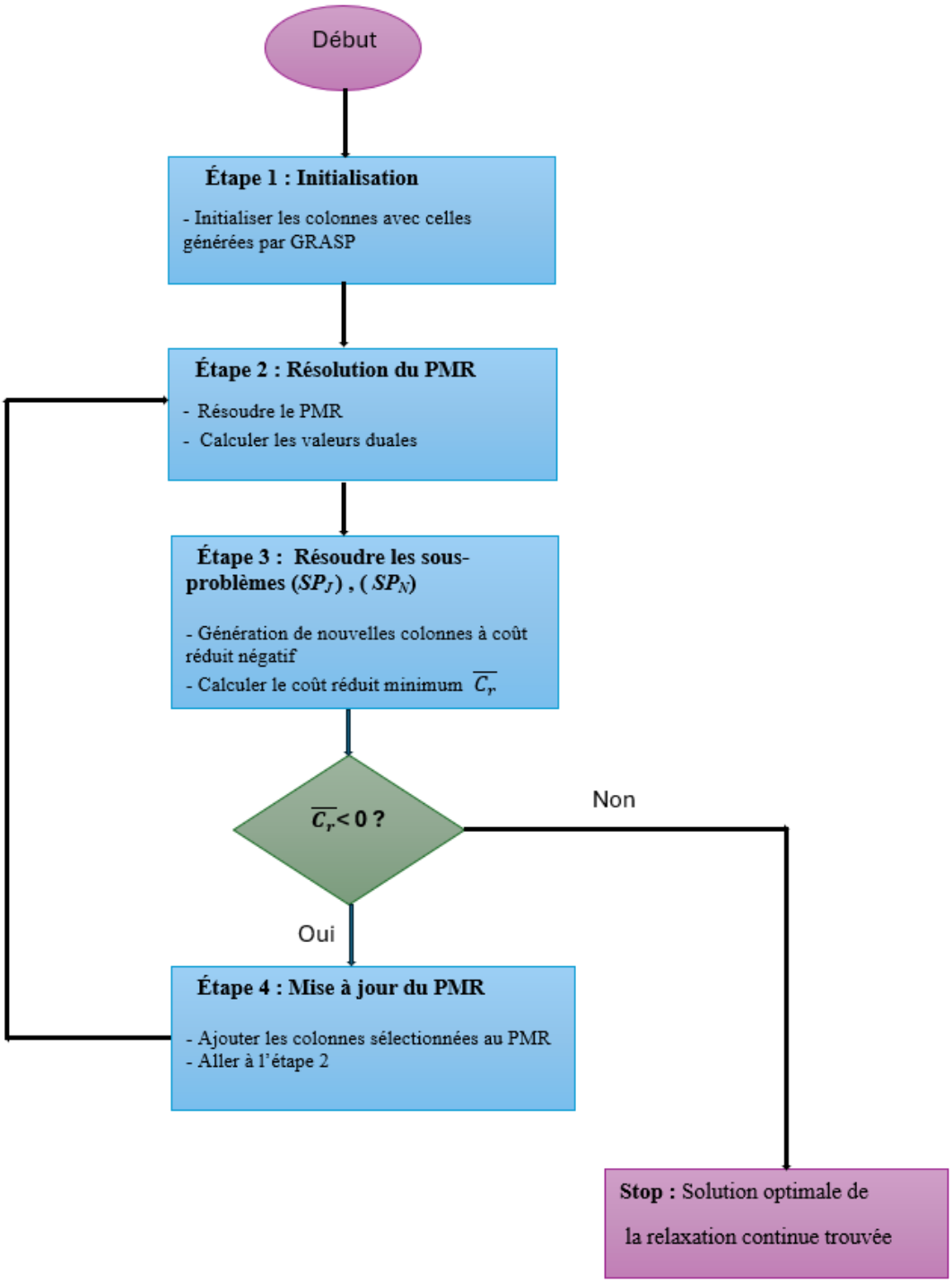


FIGURE 4.6 Organigramme du processus itératif de génération de colonnes

CHAPITRE 5 Résultats

Dans ce chapitre, nous présentons les résultats obtenus en appliquant les différentes approches proposées. Nous commençons par détailler la méthode adoptée pour générer les instances supplémentaires à partir des données de base dans la section 5.1, puis nous décrivons les caractéristiques de ces instances dans la section 5.2. Ensuite, nous analysons les performances des différentes approches en comparant d'abord les résultats obtenus avec CPLEX (5.3), puis ceux obtenus avec GRASP-MIP3 (5.4). Nous poursuivons avec l'analyse des performances de l'approche DH (5.5). Une comparaison des différentes méthodes est ensuite réalisée dans la section 5.6. Nous évaluons également la performance des approches en testant nos approches sur un ensemble de 30 instances générées (5.7), avant d'appliquer nos méthodes à une instance réelle de Victoriaville, présentée dans la section 5.8.

L'ensemble des programmes a été implémenté en C++ et exécuté sur une machine sous Linux (noyau `5.14.0-503.14.1.el9_5.x86_64`). La machine est équipée d'un processeur Intel Core i7-12700 comprenant 12 cœurs physiques et 20 threads, cadencé à une fréquence maximale de 4.90 GHz, avec 62 Go de mémoire vive (RAM). La résolution des modèles MIP a été effectuée à l'aide du solveur commercial IBM CPLEX (version 22.1.1). Concernant la génération de colonnes et DH, plusieurs sous-problèmes de génération de colonnes, formulés comme des SPPRC (*Shortest Path Problem with Resource Constraints*), ont été résolus par programmation dynamique en utilisant la bibliothèque Boost, version 1.54.

5.1 Génération d'instances

Pour tester et valider notre approche, nous avons utilisé des instances issues du travail de Lamghari et al. [1], qui se basent sur l'instance DI-NEARP-n422-Q2k-TP.dat de Vidal [5], pour générer 48 instances. Lamghari et al. [1] ont structuré le réseau routier en reliant une série de nœuds Nord-Sud et Est-Ouest pour former des autoroutes (AH), tandis que les autres nœuds étaient connectés par des routes secondaires (autres routes (AR)). Les temps de service et de déplacement ont été calculés à partir des distances euclidiennes et des vitesses de balayage et de déplacement fournies par un partenaire industriel.

Dans notre travail, nous avons sélectionné un sous-ensemble de ces instances, comprenant 40, 50 et 60 tâches sans apporter de modifications aux données. À partir de ces sous-ensembles, nous avons généré de nouvelles instances aléatoires en modifiant les temps de service et les temps de déplacement. Les modifications appliquées à chaque sous-ensemble sont détaillées ci-dessous. Nous avons également testé notre approche sur une instance basée sur des données réelles de Victoriaville, Québec, Canada, de 172 tâches où 46 tâches sont de type AH et 128 sont de type AR , étudiée dans [1], afin d'évaluer notre approche dans un contexte réaliste.

Les instances sont générées comme suit :

- Instances de 40 tâches : Nous avons généré 10 instances en multipliant les temps de service et de déplacement par un facteur aléatoire choisi dans l'intervalle $[3, 5]$. Chaque instance a été générée en utilisant un *seed* différent, allant de 1 à 10.

- Instances de 50 tâches : Le même procédé a été appliqué, avec cette fois un facteur aléatoire pris dans l'intervalle $[2.5, 3.5]$, et toujours 10 *seeds* différents pour produire 10 instances.

- Instances de 60 tâches : Pour ces instances, nous avons simulé 10 instances supplémentaires en multipliant les temps de service et de déplacement par un facteur aléatoire dans l'intervalle $[2, 4]$, chaque instance utilisant un *seed* distinct.

Les instances générées ont pour objectif de fournir un cadre pertinent pour tester notre méthodologie et de permettre la résolution d'instances de grande taille. En fait, en augmentant les temps de service et de déplacement, on arrive à ce qu'on pourrait obtenir si on applique des stratégies d'agrégation. Ces stratégies consistent à ramener des instances contenant plus de 100 tâches à des tailles plus réduites, comme des instances de 40 tâches, tout en conservant les caractéristiques essentielles du problème. Notre travail s'inscrit dans un horizon de planification plus large (généralement un mois), correspondant à un contrat de balayage mensuel. Dans ce cadre, les instances générées permettent de modéliser des *plannings* réalistes en répartissant les tâches sur plusieurs quarts de travail successifs. Cette organisation, qui peut s'étendre jusqu'à 7 quarts de travail (planification hebdomadaire), simule des *plannings* réalistes pour des périodes prolongées.

5.2 Description d'instances

Chaque instance comprend un ensemble de tâches ainsi que **4 dépôts** ($|\boldsymbol{D}| = 4$), qui servent de points de départ et/ou d'arrivée pour les routes générées. Ces instances ont été générées à partir des données décrites dans la section précédente, en tenant compte de la diversité des types de tâches : les tâches d'autoroute ($\boldsymbol{AH}$) et les tâches d'autres routes ($\boldsymbol{AR}$). Le nombre total de tâches est noté ($|\boldsymbol{N}|$) et correspond à la somme des tâches d'autoroute ($|\boldsymbol{AH}|$) et des tâches d'autres routes ($|\boldsymbol{AR}|$).

Les instances utilisées dans cette étude sont divisées en trois catégories :

1. Les 3 instances principales (**A__40**, **A__50**, **A__60**), qui regroupent respectivement 40, 50 et 60 tâches. Le tableau 5.1 présente leurs caractéristiques principales, notamment le nombre total de tâches et la répartition entre les deux types de tâches, ainsi que deux indicateurs de temps : le **temps de service moyen** ($\boldsymbol{TSM}$), et le **temps de déplacement moyen** ($\boldsymbol{TDM}$), tous deux exprimés en minutes (min) et calculés respectivement comme la moyenne des temps de service des tâches et la moyenne des temps de déplacement entre les tâches.

2. Les caractéristiques des 30 instances générées à partir des trois instances principales, désignées par des identifiants allant de **A__40__1** à **A__60__10**, sont détaillées dans le tableau 5.2. Ces variations sont obtenues suite à l'application des facteurs aléatoires aux temps de service et de déplacement, comme décrits dans la section 5.1.
3. L'instance **A__174** basée sur un cas réel provenant de la ville de **Victoriaville**, qui représente une application concrète du problème étudié avec **174 tâches**. Le tableau 5.3 détaille ses caractéristiques.

ID de l'instance	$\|D\|$	$\|N\|$	$\|AH\|$	$\|AR\|$	$TSM_{(min)}$	$TDM_{(min)}$
A_40	4	40	10	30	12.3	14.1
A_50	4	50	12	38	16.8	18.2
A_60	4	60	15	45	13.5	18.4

Tableau 5.1 Caractéristiques des 3 instances principales.

ID de l'instance	$\|D\|$	$\|N\|$	$\|AH\|$	$\|AR\|$	$TSM_{(min)}$	$TDM_{(min)}$
A_40_1	4	40	10	30	49.5	55.9
A_40_2	4	40	10	30	48.8	55.7
A_40_3	4	40	10	30	47.3	55.5
A_40_3	4	40	10	30	47.0	55.9
A_40_5	4	40	10	30	49.5	56.0
A_40_6	4	40	10	30	50.5	56.2
A_40_7	4	40	10	30	47.3	56.0
A_40_8	4	40	10	30	47.4	56.0
A_40_9	4	40	10	30	48.8	55.5
A_40_10	4	40	10	30	47.5	55.5
A_50_1	4	50	12	38	49.8	54.2
A_50_2	4	50	12	38	50.3	54.0
A_50_3	4	50	12	38	49.6	54.3
A_50_4	4	50	12	38	50.0	54.1
A_50_5	4	50	12	38	50.0	54.3
A_50_6	4	50	12	38	51.0	54.9
A_50_7	4	50	12	38	49.3	54.0
A_50_8	4	50	12	38	48.5	55.0
A_50_9	4	50	12	38	48.7	55.2
A_50_10	4	50	12	38	51.7	55.9
A_60_1	4	60	15	45	39.3	55.0
A_60_2	4	60	15	45	41.4	54.4
A_60_3	4	60	15	45	39.7	55.1
A_60_4	4	60	15	45	38.5	55.1
A_60_5	4	60	15	45	39.6	55.3
A_60_6	4	60	15	45	41.3	55.1
A_60_7	4	60	15	45	40.0	55.5
A_60_8	4	60	15	45	39.0	54.5

Suite page suivante

ID de l'instance	$\|D\|$	$\|N\|$	$\|AH\|$	$\|AR\|$	$TSM_{(min)}$	$TDM_{(min)}$
A_60_9	4	60	15	45	37.9	55.6
A_60_10	4	60	15	45	39.9	54.5

Tableau 5.2 Caractéristiques des 30 instances générées à partir des instances principales.

ID de l'instance	$\|D\|$	$\|N\|$	$\|AH\|$	$\|AR\|$	$TSM_{(min)}$	$TDM_{(min)}$
A_174	4	174	46	128	11.9	24.9

Tableau 5.3 Caractéristiques de l'instance basée sur le cas réel de Victoriaville.

5.3 Résultats obtenus avec CPLEX

Dans cette section, nous appliquons le modèle (P_2) présenté par Lamghari et al. [33] aux trois instances principales (A_40, A_50, A_60). Pour chaque instance, le nombre de quarts de travail (S) a été fixé à une valeur correspondant à la borne supérieure obtenue avec l'approche GRASP-MIP3. La durée maximale de chaque quart a été fixée à 600 minutes, conformément aux contraintes opérationnelles. Le temps d'exécution de CPLEX a été limité à 3600 secondes pour chacune des trois instances.

Les résultats obtenus, présentés dans le tableau 5.4, incluent : La valeur de la relaxation linéaire au nœud 0 (***RL***), la borne inférieure (***LB***), la borne supérieure (***UB***), le ***GAP*** relatif entre la borne inférieure et la borne supérieure. De plus, nous indiquons le nombre de nœuds explorés dans l'arbre de branchement lors du processus de séparation et d'évaluation (*Branch-and-Bound*) (***Nb_N_branch***) ainsi que le temps de calcul total en secondes ($Time_{(s)}$).

ID de l'instance	***RL***	***LB***	***UB***	***GAP***(%)	***Nb_N_branch***	$Time_{(s)}$
A_40	123.3	165.5	4925.7	96.6	986850	3600
A_50	213.7	246.5	13793.8	98.2	519484	3600
A_60	264.0	294.2	13850.2	97.8	307210	3600

Tableau 5.4 Résultats obtenus avec CPLEX pour les trois instances

Les résultats obtenus avec CPLEX révèlent plusieurs limitations du modèle présenté dans la section 3.2.1. La borne inférieure demeure très éloignée de la borne supérieure, avec un gap d'environ **96%** pour l'instance A_40 et de **97%** pour A_50 et A_60. Cette disparité reflète la difficulté pour CPLEX de resserrer cet écart, même après une heure d'exécution. La relaxation linéaire au nœud 0 (***RL***) illustre également cette faiblesse : avec une valeur de seulement 123.3 pour A_40, elle reste une approximation peu fiable de la solution optimale, compromettant son utilité pour guider efficacement l'exploration de l'arbre de recherche. La borne inférieure stagne tout au long de l'exécution,

n'atteignant que 165.5 pour A_40, sans amélioration malgré le temps de calcul d'une heure. Le graphique 5.1 ci-dessous montre l'évolution de la borne inférieure pour A_40, mettant en évidence une stagnation rapide et une absence de progrès significatif. Ces résultats mettent en évidence plusieurs faiblesses du modèle : une relaxation linéaire peu performante, un manque d'efficacité dans la recherche au sein de l'arbre de branchement et un temps de calcul élevé sans progrès significatif. Ces observations soulignent la nécessité de recourir à des méthodes alternatives capables de mieux exploiter la structure du problème, de produire des relaxations linéaires plus serrées et d'accélérer la convergence vers des solutions de meilleure qualité.

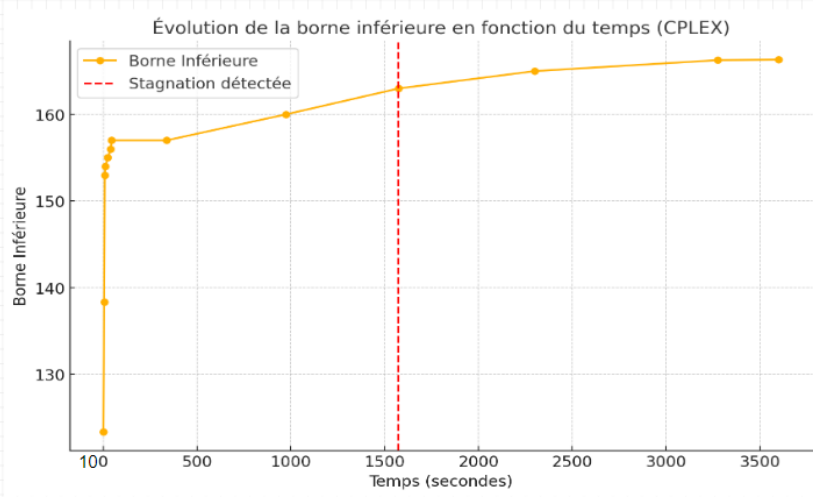


FIGURE 5.1 Évolution de la borne inférieure (LB) pour l'instance A_40 avec CPLEX

5.4 Résultats obtenus avec GRASP-MIP3

Dans cette section, nous présentons les résultats obtenus en combinant la phase de construction GRASP et le modèle MIP3 (P_3). Pour les instances principales (A_40, A_50, A_60), la génération des routes à l'aide de GRASP a nécessité seulement quelques secondes, démontrant l'efficacité de cette méthode pour produire une solution initiale. Les paramètres utilisés pour GRASP-MIP3 sont les suivants :

- *k=5 (la taille de la Liste Restreinte de Candidats (RCL)).*
- *nbR =300 (critère d'arrêt basé sur le nombre maximum de routes générées par tâche)*

Le tableau 5.5 résume les résultats obtenus pour les trois instances principales, incluant le nombre total de routes générées (***Nbr.route***), la valeur $\boldsymbol{LB_{(P_3)}}$ dans le tableau correspond à la borne inférieure obtenue en résolvant le modèle (P_3) restreint aux routes générées par GRASP.

ID de l'instance	*Nbr.route*	*RL*	$LB_{(P_3)}$	*UB*	*GAP*(%)	$Time_{(s)}$
A_40	88063	4749.2	4749.2	4749.2	0	8
A_50	120106	8701.0	9447.5	9447.5	0	50
A_60	147867	8395.4	9530.2	9530.2	0	109

Tableau 5.5 Résultats obtenus avec GRASP-MIP3 pour les instances principales.

Ces résultats montrent que GRASP-MIP3 permet de générer des bornes supérieures de qualité tout en maintenant un temps d'exécution très rapide en quelques secondes. La phase de génération des routes est particulièrement rapide, ce qui en fait une méthode efficace pour produire des solutions initiales exploitables par la suite, comme colonnes initiales dans la méthode DH.

5.5 Résultats obtenus avec DH

Dans cette section, nous présentons les résultats obtenus en appliquant la méthode de génération de colonnes combinée à une heuristique de branchement en profondeur (DH) pour résoudre les instances principales (A_40, A_50, A_60). Le tableau 5.6 résume les résultats de cette approche en indiquant plusieurs métriques clés : le $\boldsymbol{GAP} = \frac{\boldsymbol{UB-LB}}{\boldsymbol{LB}} \times 100$, le nombre total d'itérations effectuées ***(Nbr.itr)***, ainsi que les nombres de colonnes générées dans le problème maître restreint ***(Nbr.col.PMR)***, comprenant à la fois celles générées par GRASP et par les sous-problèmes, et le nombre de colonnes générées spécifiquement par les deux sous-problèmes, nuit et jour ***(Nbr.col.SP)***. De plus, le tableau détaille le temps consacré à la résolution des sous-problèmes ($\boldsymbol{T.SP_{(s)}}$) et du problème maître restreint ($\boldsymbol{T.PMR_{(s)}}$). Enfin, les résultats incluent le nombre total de nœuds de branchement (***Nb.N_branch***) et le temps ($\boldsymbol{Time_{(s)}}$), ainsi que le nombre d'itérations spécifiques à l'heuristique de branchement en profondeur.

ID	*UB*	*LB*	*GAP*(%)	*Nbr.itr*	*Nbr.col.PMR*	*Nbr.col.SP*	$T.PMR_{(s)}$	$T.SP_{(s)}$	*Nb.N_branch*	$Time_{(s)}$
A_40	4749.2	4749.2	0	28	137388	45114	7.88	30.0	1	41,6
A_50	13950.0	7722.9	80.6	19	143497	17408	8.1	520.6	3	531.8
A_60	9644.0	7946.4	21.3	9	173963	23422	9.6	1419.9	1	1434.1

Tableau 5.6 Résultats obtenus avec DH

Les résultats du tableau 5.4 montrent que pour l'instance A_40, la méthode combinée de génération de colonnes et DH a atteint une solution optimale avec un gap 0. Pour l'instance A_50, l'écart élevé (80,62%) s'explique principalement par un mauvais choix des nœuds de branchement, impactant la convergence. De plus, les sous-problèmes ont généré un nombre important de routes, ce qui influence directement la qualité des solutions obtenues.

5.6 Comparaison des résultats

Dans cette section, nous comparons les performances des trois approches : CPLEX, GRASP-MIP3 et DH. L'objectif de cette comparaison est de montrer que notre méthode DH permet de surmonter les faiblesses de CPLEX, tout en offrant des solutions de qualité en un temps raisonnable.
Le tableau 5.7 présente les résultats pour les trois instances principales (A_40, A_50, A_60). Chaque méthode est évaluée selon le temps d'exécution ($\boldsymbol{Time_{(s)}}$), le $\boldsymbol{GAP}$, la valeur de la relaxation linéaire au nœud 0 ($\boldsymbol{RL}$), la borne supérieure ($\boldsymbol{UB}$) et la borne inférieure ($\boldsymbol{LB}$).

ID	CPLEX (P_2)					GRASP-MIP3 (P_3)					DH (GRASP+PM+ SP_s)			
	$\boldsymbol{Time_{(s)}}$	$\boldsymbol{GAP}$(%)	$\boldsymbol{RL}$	$\boldsymbol{UB}$	$\boldsymbol{LB}$	$\boldsymbol{Time_{(s)}}$	$\boldsymbol{GAP}$ (%)	$\boldsymbol{RL}$	$\boldsymbol{UB}$	$\boldsymbol{LB_{(P_3)}}$	$\boldsymbol{Time_{(s)}}$	$\boldsymbol{GAP}$ (%)	$\boldsymbol{UB}$	$\boldsymbol{LB}$
A_40	3600	96.5	123.3	4925.7	165.5	8.0	0	4749.2	4749.2	4749.2	41.6	0.0	4749.2	4749.2
A_50	3600	97.3	213.7	13793.8	246.6	50.0	0	8701.0	9447.5	9447.5	531.8	80.6	13950.0	7722.9
A_60	3600	97.8	264.0	13850.0	294.2	109.0	0	8395.4	9530.2	9530.5	1434.1	21.3	9644.0	7946.4

TABLEAU 5.7 Comparaison des performances des trois approches pour les instances principales.

Le tableau 5.7 illustre les performances des trois approches sur les instances principales. La méthode GRASP-MIP3 se distingue par sa capacité à fournir rapidement des bornes supérieures, surpassant largement celles obtenues avec CPLEX. En effet, GRASP-MIP3 atteint ces résultats en des temps très courts, avec une moyenne d'une minute au lieu de 60 min de CPLEX pour les instances A_40, A_50 et A_60. Quant à notre approche DH, elle démontre non seulement une amélioration significative de la borne inférieure LB, mais réussit également à résoudre certaines instances à l'optimalité. Pour l'instance A_40, DH atteint un gap de 0%, confirmant une solution optimale avec une valeur LB identique à celle de GRASP-MIP3 (4749.2) et une nette supériorité par rapport à CPLEX, où la borne inférieure n'était que de 165.5. Concernant l'instance A_50, la stratégie de branchement actuelle ne s'est pas révélée concluante, suggérant la nécessité d'une amélioration ou d'une combinaison avec une heuristique primale comme GRASP. En effet, l'écart relatif $\frac{\boldsymbol{UB_{(GRASP-MIP3)}}-\boldsymbol{LB_{(DH)}}}{\boldsymbol{LB_{(DH)}}} \times 100$ est de 22.3%, ce qui montre une performance bien meilleure que celle obtenue avec CPLEX. Pour l'instance A_60, DH améliore LB de 294.2 (CPLEX) à 7946.4 et réduit le gap à 21.3% avec un temps de calcul de 1434.1 secondes. Ces résultats soulignent le potentiel de DH dans l'amélioration de la borne inférieure, qui constitue l'objectif principal. Les métaheuristiques comme GRASP permettent d'obtenir de bonnes solutions *(UB)*, mais sans aucune garantie sur la qualité de la solution.

5.7 Résultats sur les 30 instances générées

Dans cette section, nous présentons les résultats obtenus pour les 30 instances générées à partir des trois instances principales (A_40, A_50, A_60). Rappelons que ces instances ont été conçues pour répondre aux besoins opérationnels d'un horizon de planification d'un mois en simulant des

scénarios réalistes avec des instances de grande taille. À l'aide d'une stratégie d'agrégation, nous avons ajusté les temps de service et les temps de déplacement pour ramener ces instances complexes à une taille maîtrisable, tout en conservant leur pertinence opérationnelle.
L'objectif principal de cette analyse est de démontrer l'efficacité de notre approche pour résoudre des problèmes de grande taille. En particulier, nous comparons les performances des trois méthodes (**CPLEX**, **GRASP-MIP3** et **DH**) en termes de temps de calcul, de qualité des solutions (GAP), et des bornes inférieures obtenues. Le tableau 5.8 résume les résultats pour ces instances.

ID	CPLEX (P_2)					GRASP-MIP3 (P_3)					DH (GRASP+PM+ SP_s)			
	$Time_{(s)}$	GAP(%)	RL	UB	LB	$Time_{(s)}$	GAP (%)	RL	UB	$LB_{(P_3)}$	$Time_{(s)}$	GAP (%)	UB	LB
A_40_1	7200	****	495.2	****	601.7	900	14.1	19950.0	23747.8	20394.6	40.3	25.7	23560.2	18729.8
A_40_2	7200	****	488.7	****	599.5	900	15.1	19745.7	23745.0	20157.2	68.0	3.7	18905.0	18220.5
A_40_3	7200	****	473.0	****	587.0	900	17.6	19088.6	23652.0	19468.4	89.8	7.0	18948.2	17701.7
A_40_4	7200	****	470.2	****	582.3	900	17.4	18921.8	23693.8	19567.6	95.0	8.1	18885.2	17459.5
A_40_5	7200	****	471.2	****	585.0	900	14.9	19829.4	23711.8	20171.4	25.6	2.4	18950.5	18491.9
A_40_6	7200	****	486.0	****	597.1	900	17.9	20111.2	23661.5	20557.3	68.5	1.0	18925.2	18733.2
A_40_7	7200	****	497.8	****	602.0	900	17.9	19048.9	23725.0	19470.0	55.6	7.1	18913.8	17658.7
A_40_8	7200	****	494.6	****	600.0	900	17.9	19022.3	23765.8	19501.8	61.8	7.8	18988.2	17604.5
A_40_9	7200	****	476.2	****	585.0	900	15.4	19692.11	23742.0	20078.1	66.0	2.9	18978.2	18435.3
A_40_10	7200	****	491.6	****	598.3	900	17.4	19116.4	23693.8	19567.6	73.5	5.8	18940.0	17889.2
Moy	**7200**	-	484.4	-	**593.8**	**900**	16.5	19452.6	**23713.8**	**19893.4**	**64.3**	**7.1**	**19399.4**	**18092.3**
A_50_1	7200	****	631.2	****	812.7	1500	13.9	23850.0	28341.2	24398.5	129.3	5.7	23657.5	22377.8
A_50_2	7200	****	638.7	****	816.2	1500	12.7	24041.9	28362.5	24754.2	185.8	26.9	28421.2	22390.9
A_50_3	7200	****	628.7	****	808.7	1500	14.3	23764.2	28380.0	24300.9	171.7	6.7	23582.5	22091.4
A_50_4	7200	****	620.0	****	798.7	1500	15.4	23418.7	28400.0	23999.9	146.2	8.0	23680.0	21923.6
A_50_5	7200	****	633.7	****	818.2	1500	13.3	23912.1	28390.0	24602.9	223.6	5.1	23583.8	22438.8
A_50_6	7200	****	644.2	****	819.8	1500	12.3	24229.0	28343.0	24845.5	128.7	24.4	28245.5	22687.7
A_50_7	7200	****	626.7	****	809.2	1500	14.7	23648.7	28376.8	24187.7	234.5	7.2	23644.2	22046.1
A_50_8	7200	****	616.0	****	796.0	1500	15.6	23356.4	28427.2	23978.5	218.1	7.7	23536.0	21833.4
A_50_9	7200	****	617.5	****	794.6	1500	15.9	23440.0	28476.2	23941.5	154.4	7.4	23575.0	21934.1
A_50_10	7200	****	628.7	****	811.0	1500	14.5	23592.8	28397.5	24271.6	139.4	6.3	23671.2	22254.8
Moy	**7200**	-	628.5	-	**808.5**	1500	14.2	23725.4	**28389.4**	**24328.1**	**173.2**	**10.5**	**24559.6**	**22197.8**
A_60_1	7200	****	731.0	****	799.1	1500	12.4	24485.8	28803.5	25218.4	490.9	8.5	23719.8	21846.6
A_60_2	7200	****	773.2	****	835.5	1500	9.1	25369.9	28763.2	26139.5	797.5	4.0	23709.5	22777.0
A_60_3	7200	****	733.0	****	795.0	1500	12.0	24461.4	28724.2	25270.4	762.5	9.1	23771.8	21784.4
A_60_4	7200	****	723.2	****	781.1	1500	14.8	24578.3	28865.8	24578.3	989.5	11.0	23613.2	21269.4
A_60_5	7200	****	741.2	****	814.8	1500	12.0	24546.5	28762.5	25287.4	639.6	7.8	23705.0	21974.5
A_60_6	7200	****	772.2	****	841.3	1500	9.0	25312.1	28691.0	26090.6	365.4	3.6	23687.2	22851.2
A_60_7	7200	****	749.0	****	816.5	1500	11.7	24542.3	28759.0	25383.4	405.4	7.6	23785.2	22099.6
A_60_8	7200	****	741.0	****	806.3	1500	13.3	24334.8	28883.5	25043.2	350.1	9.5	23798.5	21720.9
A_60_9	7200	****	706.7	****	771.1	1500	16.7	23433.9	28873.0	24027.1	637.2	34.0	28221.8	21052.1
A_60_10	7200	****	740.2	****	803.3	1500	16.3	24785.1	28869.2	24159.1	838.7	28.3	28304.0	22056.6
Moy	**7200**	-	741.1	-	**806.4**	**1500**	12.7	24585.0	**28799.5**	**25119.7**	**627.7**	**12.3**	**24631.6**	**21943.2**

TABLEAU 5.8 Résultats obtenus sur les 30 instances générées.

Les résultats obtenus pour les 30 instances générées mettent en évidence les performances contrastées des trois méthodes évaluées. Avec CPLEX, aucune solution n'a pu être trouvée dans un temps de calcul limité à deux heures. Les bornes inférieures fournies sont de mauvaise qualité, et les relaxations linéaires (RL) initiales restent très faibles. Par exemple, pour l'instance A_40_1, la relaxation linéaire est de seulement 495, et la borne inférieure atteinte après deux heures de calcul n'est que de 602. Des résultats similaires ont été observés pour les autres instances, confirmant les limitations de CPLEX pour ces problèmes de grande taille. En revanche, l'approche GRASP-MIP3 a permis de résoudre les 30 instances dans des temps de calcul raisonnables. Le temps de calcul a été fixé à 900 secondes pour les instances A_40_1 à A_40_10, et à 1500 secondes pour les instances générées à partir de A_50 et A_60. Cette approche a démontré son efficacité pour traiter les instances de

grande taille. L'approche DH s'est révélée encore plus performante, résolvant également les instances de grande taille où elle surpasse GRASP-MIP3 en fournissant des résultats de meilleure qualité. Elle a permis en quelques minutes une amélioration significative des valeurs des bornes inférieures (LB) par rapport à celles obtenues avec CPLEX après deux heures de calcul. De plus, DH a réduit le gap de manière efficace, comme pour l'instance A_40_6, où un gap de 1% a été atteint en seulement 68 secondes.

Pour compléter cette analyse, une moyenne a été calculée pour chaque groupe d'instances, en se basant sur les principales métriques : temps de calcul, GAP, bornes inférieures (LB) et supérieures (UB), afin de comparer les performances globales des méthodes sur chaque groupe d'instances, en résumant les résultats sans analyser les 30 cas individuellement. Sur l'ensemble des groupes d'instances, DH surpasse largement CPLEX en obtenant des bornes inférieures renforcées dans des temps de calcul très rapides. Par exemple, pour les instances A_40, DH améliore la borne inférieure moyenne de 593.8 (CPLEX, en 7200 s) à 18092.3 en seulement 64.3 secondes, avec un écart moyen de 7.1%. Pour les instances A_50, la borne inférieure moyenne avec DH est de 22197.8, contre seulement 808.5 avec CPLEX. Pour les instances A_60, DH atteint 21943.2, tandis que CPLEX reste bloqué à 806.4, malgré un temps de calcul vingt fois plus long. Comparé à GRASP-MIP3, DH se distingue également par sa rapidité, tout en obtenant des solutions de très bonne qualité : pour le groupe d'instances A_50, la borne supérieure obtenue avec DH est de 24559.6, contre 28389.4 avec GRASP-MIP3, pour un temps de calcul environ 9 fois inférieur à celui de GRASP-MIP3. Pour les instances A_60, DH atteint une borne supérieure de 24631.6, contre 28799.5 avec GRASP-MIP3, avec un temps également 2.4 fois inférieur.

5.8 Résultats sur l'instance réelle de Victoriaville

Dans cette section, nous présentons les résultats obtenus pour l'instance réelle de **Victoriaville**, qui comprend 174 tâches et constitue une application concrète du problème de balayage printanier. Cette instance repose sur des données provenant du réseau routier de Victoriaville, une municipalité située dans la province de Québec, au Canada. Le réseau modélisé comprend un total de 165 km de voirie, incluant à la fois des segments d'autoroutes et des routes secondaires. Les caractéristiques de cette instance sont présentes dans le tableau 5.3, ainsi la figure 5.2 illustre la répartition de différentes tâches, où les cercles symbolisent les tâches et les carrés noirs indiquent les positions des dépôts [1].

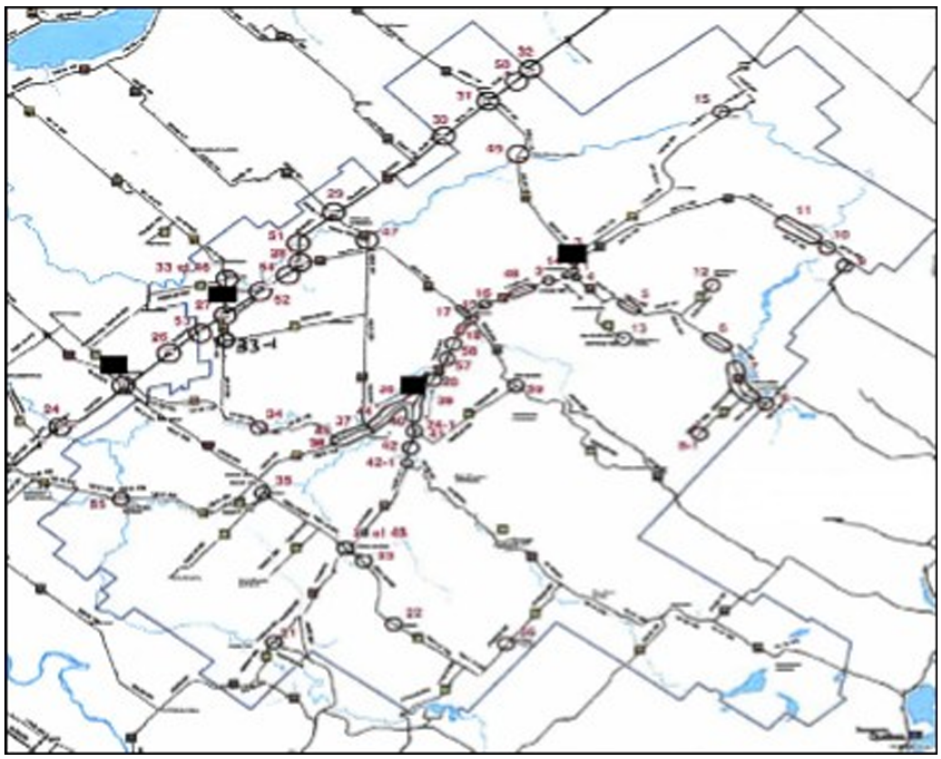


FIGURE 5.2 Illustration de la zone de balayage – Victoriaville, Québec, Canada [1]

Après avoir défini les caractéristiques de l'instance réelle de Victoriaville, nous avons testé les trois approches étudiées – CPLEX, GRASP-MIP3 et DH – afin d'évaluer leur efficacité sur ce cas concret. Ces tests ont permis d'analyser leurs performances en termes de qualité de solution et de temps de calcul. Le tableau 5.9 présente les résultats obtenus pour chacune de ces méthodes.

ID	CPLEX (P_2)					GRASP-MIP3 (P_3)							DH (GRASP+PM+ SP_s)			
	$Time_{(s)}$	$GAP(\%)$	RL	UB	LB	$Time_{(s)}$	GAP (%)	RL	UB	$LB_{(P_3)}$	$Nbr.R$	$T.Nbr.R_{(s)}$	$Time_{(s)}$	GAP (%)	UB	LB
A_174	10800	****	770.5	****	955.0	1500	23.3	25783.7	33821.7	25927.9	220308	500	****	****	****	****

TABLEAU 5.9 Résultats obtenus avec les trois approches sur l'instance A_174.

CPLEX n'a pas permis de résoudre l'instance après trois heures de calcul. La relaxation linéaire obtenue est très mauvaise, et la borne inférieure est restée, sans surprise, stagnante tout au long de l'exécution. En revanche, l'approche GRASP-MIP3 a permis de fournir une borne supérieure au problème, assurant ainsi une première approximation de la solution optimale. Cependant, l'heuristique DH n'a pas pu aboutir à une solution sur l'instance A_174. Après un temps de calcul, l'algorithme n'a généré aucune solution et a finalement interrompu son exécution en raison d'une saturation mémoire. Cette défaillance est due à la taille de l'instance, qui a engendré une explosion combinatoire, notamment au niveau du nombre d'étiquettes générées dans les sous-problèmes, rendant le problème trop complexe pour être traité par DH. Cela souligne les limites de DH face aux

grandes instances et justifie l'intérêt de la stratégie d'agrégation, qui fait l'objet de la section 5.8.1. Elle permet de réduire la taille du problème et de rendre l'approche DH exploitable.

5.8.1 Stratégie d'agrégation

Dans cette section, nous présentons la stratégie d'agrégation mise en place pour réduire la taille de l'instance initiale tout en préservant la structure du problème. Cette agrégation a été réalisée à partir de la solution fournie par l'approche GRASP-MIP3. Certaines de ces routes couvraient plus de 60 tâches, ce qui nécessitait un découpage en plusieurs super-tâches. Afin de simplifier le problème, nous avons introduit une limitation à 10 tâches maximum par super-tâche. Pour les routes contenant plus de 10 tâches, nous avons agrégé chaque ensemble de dix tâches situées sur le même segment en une super-tâche, tout en conservant la structure des itinéraires initiaux. Dans certains cas, certaines super-tâches comptaient moins de 10 tâches lorsque la segmentation ne permettait pas d'atteindre exactement cette limite. À l'inverse, les routes contenant déjà moins de 10 tâches n'ont pas été modifiées : les tâches qu'elles contenaient ont été conservées telles quelles. Après ce regroupement, une nouvelle instance a été générée, réduisant ainsi la taille du problème de 174 tâches à 50 tâches, comprenant 15 super-tâches et 35 tâches non agrégées. Une mise à jour des caractéristiques des tâches a ensuite été réalisée, notamment en ce qui concerne leur type (AR ou AH). Pour chaque super-tâche, le type a été déterminé en fonction des tâches qui la composent :

— Si toutes les tâches appartiennent au type AR, la super-tâche est classée comme AR.
— Si toutes les tâches appartiennent au type AH, la super-tâche est classée comme AH.
— Si une super-tâche contient à la fois des tâches AR et AH, elle est classée comme AH.

Pour les tâches non agrégées, leur type reste inchangé par rapport à l'instance initiale.

L'agrégation des tâches a également nécessité une mise à jour des temps de service et des temps de déplacement dans la nouvelle instance :

— **Temps de service**

- Pour chaque super-tâche, le temps de service est la somme des temps de service des tâches qu'elle contient.
- Pour les tâches non agrégées, leur temps de service reste inchangé.

— **Temps de déplacement**

- **Temps interne aux super-tâches :** Somme des temps de déplacement entre les tâches agrégées, selon la matrice initiale.
- **Temps de déplacement entre deux super-tâches :** Somme des temps de déplacement internes de la première super-tâche, additionnée au temps de déplacement entre son dernier nœud et le premier nœud de la super-tâche suivante.

- **Temps de déplacement entre une super-tâche et une tâche non agrégée :** Somme des temps de déplacement internes de la super-tâche et du temps de déplacement entre son dernier nœud et la tâche non agrégée.
- **Temps de déplacement entre une tâche non agrégée et une super-tâche :** Temps de déplacement entre la tâche et le premier nœud de la super-tâche.
- **Temps de déplacement entre un dépôt et une super-tâche :** Temps de déplacement entre le dépôt et le premier nœud de la super-tâche.
- **Temps de déplacement entre une super-tâche et un dépôt :** Somme des temps de déplacement internes de la super-tâche et du temps de déplacement entre son dernier nœud et le dépôt.

Ces ajustements ont permis de générer une nouvelle instance plus compacte, réduisant la taille de la matrice des temps de déplacement de 178×178 à 54×54, tout en maintenant une structure du problème initial. Les caractéristiques de cette nouvelle instance, décrites dans la section 5.2, sont présentées en détail dans le tableau 5.10.

ID de l'instance	$\|D\|$	$\|N\|$	$\|AH\|$	$\|AR\|$	$TSM_{(min)}$	$TDM_{(min)}$
AG_50	4	50	12	38	41.4	42.9

Tableau 5.10 Caractéristiques de la nouvelle instance issue de l'agrégation de A_174

Nous observons que les valeurs de *TSM* et *TDM* pour cette instance agrégée (41.4 min et 42.9 min) restent comparables à celles des données synthétiques utilisées dans cette étude, présentées dans 5.2.

5.8.2 Résultats obtenus après l'agrégation

L'instance originale A_174 de Victoriaville est de grande taille, ce qui ne permettait pas d'appliquer directement l'approche DH, en raison du temps de calcul très élevé sans trouver une solution. Cependant, grâce à la stratégie d'agrégation présentée dans la section 5.8.1, nous avons pu réduire la taille du problème et exécuter DH sur la nouvelle instance AG_50.

Le tableau 5.11 présente les résultats obtenus avec les trois approches considérées : CPLEX, GRASP-MIP3 et DH, appliquées à cette nouvelle instance.

ID	**CPLEX** (P_2)					**GRASP-MIP3** (P_3)							**DH (GRASP+**PM**+** SP_s**)**			
	$Time_{(s)}$	$GAP(\%)$	RL	UB	LB	$Time_{(s)}$	GAP (%)	RL	UB	$LB_{(P_3)}$	$Nbr.R$	$T.Nbr.R_{(s)}$	$Time_{(s)}$	GAP (%)	UB	LB
AG_50	10800	****	1695.5	****	1777.9	1500	7.5	26776.7	29148	26947.1	372550	58	406	13.51	28931.8	25487.4

Tableau 5.11 Résultats obtenus avec les trois approches sur l'instance AG_50.

Après 3 heures de calcul, CPLEX n'a pas pu fournir une solution et s'est limitée à une relaxation continue très faible de 1695.5. De plus, malgré cette longue durée de calcul, la borne inférieure obtenue est restée stagnante à 1777.98, ce qui empêche CPLEX de converger rapidement. De plus, on observe une nette amélioration du nombre de routes générées. En effet, sur l'instance A_174, GRASP avait généré 220308 routes en 500 secondes, tandis qu'avec la stratégie d'agrégation, le nombre de routes générées est passé à 372550 en seulement 58 secondes, mettant en évidence l'efficacité du processus d'agrégation pour accélérer la génération des solutions. L'approche DH se distingue en surpassant à la fois CPLEX et GRASP-MIP3 sur plusieurs aspects. Elle fournit une relaxation continue plus resserrée et plus représentative de la qualité de la solution optimale, témoignant d'une meilleure qualité de solution avec un gap de 13%. De plus, la borne supérieure obtenue est non seulement meilleure que celle de GRASP-MIP3 sur l'instance agrégée, mais également meilleure que celle trouvée avec GRASP-MIP3 sur l'instance A_174 en passant de 33821.75 à 28932.8. Un des résultats les plus marquants est que grâce à l'agrégation, DH a pu être appliqué sur cette instance réduite, permettant ainsi d'obtenir une solution exploitable en un temps extrêmement raisonnable (406 secondes), avec une meilleure relaxation continue trouvée. Ces résultats démontrent que la stratégie d'agrégation a non seulement permis de rendre l'instance réelle de Victoriaville traitable par DH, mais qu'elle a également permis une meilleure gestion du compromis entre qualité de solution et temps de calcul.

CHAPITRE 6 CONCLUSION

Le travail présenté dans ce document portait sur l'étude et l'amélioration des méthodes dédiées à la résolution du problème de balayage printanier des rues. En s'appuyant sur des approches exactes et heuristiques, l'objectif était de répondre aux défis posés par les instances de grande taille, notamment en améliorant la qualité de la relaxation linéaire et en réduisant les temps de calcul. Dans ce chapitre, nous présenterons une synthèse des travaux réalisés pour améliorer les performances des méthodes utilisées. Ensuite, nous discuterons les limites de la solution proposée et, finalement, nous proposerons des pistes d'amélioration et des perspectives pour les travaux futurs.

6.1 Synthèse des travaux

Dans un premier temps, une nouvelle formulation mathématique du problème a été développée, permettant de mieux modéliser les contraintes opérationnelles et d'améliorer la qualité de la relaxation linéaire. Cette formulation a été accompagnée d'une phase de construction initiale basée sur l'heuristique GRASP, générant un ensemble de routes réalisables servant de base pour les calculs ultérieurs. Ensuite, la décomposition de Dantzig-Wolfe a été appliquée pour reformuler le problème en un problème maître restreint (PMR) et des sous-problèmes de génération de colonnes. Enfin, une heuristique de branchement en profondeur (DH) a été intégrée pour convertir les solutions fractionnaires générées avec la méthode de génération de colonnes en solutions entières. Bien que cette méthode ait été testée principalement sur les 30 instances générées, celles-ci, grâce à des stratégies d'agrégation, permettent de simuler des instances de grande taille en réduisant leur complexité. DH a montré son efficacité en fournissant des solutions de bonne qualité dans des délais raisonnables. Par exemple, elle a permis de résoudre certaines instances de 40 tâches à l'optimalité, avec un gap réduit à 0%, ainsi que d'obtenir un gap de seulement 1% pour certaines instances générées intégrant un grand nombre de tâches. De plus, l'approche a considérablement amélioré les résultats par rapport aux méthodes classiques. Pour l'instance A_60, le gap initial de 97% a été réduit à 21%, mettant en évidence la capacité de cette méthode à resserrer l'écart entre la borne inférieure et la borne supérieure. La stratégie d'accélération adoptée repose principalement sur deux points : l'élimination des tâches déjà couvertes par les colonnes fixées dans le nœud courant, et la suppression des arcs associés aux tâches assignées. Ces ajustements ont permis de réduire la taille des sous-problèmes, accélérant ainsi la convergence vers des solutions entières. Ces travaux ont permis de surmonter les limitations des modèles mathématiques existants pour résoudre ce problème en améliorant la relaxation linéaire et en fournissant des solutions là où CPLEX n'a pas pu converger après trois heures de calcul. L'intégration de DH a permis d'améliorer la qualité des solutions obtenues et de favoriser une convergence plus rapide vers de bonnes bornes, même sur des instances de grande taille issues d'un ensemble de 30 cas générés, ainsi que sur une instance réelle de la ville de Victoriaville,

Québec, Canada, à travers une stratégie d'agrégation spécifiquement développée dans le cadre de cette étude.

6.2 Limitations de la solution proposée et améliorations futures

La solution proposée présente plusieurs limitations qui méritent d'être soulignées. Tout d'abord, bien que les résultats obtenus avec la méthode DH soient globalement satisfaisants, certaines limitations subsistent. Pour certaines instances, l'approche n'a pas permis de produire des solutions de qualité optimale, et le gap reste élevé. De plus, dans le cas des instances simulant des problèmes de grande taille, des phénomènes de cyclage ont été observés, en grande partie en raison de la structure du graphe utilisé, qui est un graphe complet. Cela souligne la nécessité de développer des graphes mieux adaptés ou d'explorer des stratégies d'agrégation supplémentaires pour surmonter cette difficulté. Par ailleurs, les sous-problèmes, bien qu'efficacement résolus, n'ont pas généré un nombre suffisant de colonnes pour améliorer significativement la qualité des solutions dans certains cas.
Pour améliorer les performances de la solution proposée, plusieurs axes de recherche peuvent être envisagés. Une amélioration de la stratégie de branchement pourrait être explorée, en affinant les critères de sélection des variables pour réduire davantage le nombre de nœuds à explorer et accélérer la convergence. Des efforts pourraient être consacrés à la gestion directe des instances de grande taille, sans recourir systématiquement à des stratégies d'agrégation, en exploitant des techniques avancées issues de la génération de colonnes. Parmi celles-ci, la stabilisation de la relaxation linéaire, la réduction dynamique du graphe et l'accélération de la résolution du sous-problème permettent de traiter efficacement de grandes instances sans augmenter excessivement le temps de calcul. Ces stratégies pourraient être explorées afin d'améliorer la scalabilité du modèle et d'assurer une meilleure applicabilité aux contraintes réelles du balayage printanier, notamment en termes de temps de calcul, de faisabilité des tournées et d'optimisation des ressources.

RÉFÉRENCES

[1] A. Lamghari, O. Foutlane et J. Audy, "Decomposition strategies for solving the spring sweeping routing problem," *Les Cahiers du GERAD ISSN*, vol. 711, p. 2440, 2025.

[2] S. Du, M. Akin, D. Bergner, G. Xu et X. Shi, "Material application methodologies for winter road maintenance : a renewed perspective," *Canadian Journal of Civil Engineering*, vol. 49, n°. 1, p. 1–10, 2022.

[3] Á. Corberán et G. Laporte, "Arc routing," *Society for Industrial and Applied Mathematics, Philadelphia, PA*, 2014.

[4] S. Wøhlk, *Contributions to arc routing*. Syddansk Universitetsforlag, 2006.

[5] T. Vidal, "Node, edge, arc routing and turn penalties : Multiple problems—one neighborhood extension," *Operations Research*, vol. 65, n°. 4, p. 992–1010, 2017.

[6] A. Corberán, R. Eglese, G. Hasle, I. Plana et J. M. Sanchis, "Arc routing problems : A review of the past, present, and future," *Networks*, vol. 77, n°. 1, p. 88–115, 2021.

[7] L. Foulds, H. Longo et J. Martins, "A compact transformation of arc routing problems into node routing problems," *Annals of Operations Research*, vol. 226, n°. 1, p. 177–200, 2015.

[8] J. R. Montoya-Torres, J. L. Franco, S. N. Isaza, H. F. Jiménez et N. Herazo-Padilla, "A literature review on the vehicle routing problem with multiple depots," *Computers & Industrial Engineering*, vol. 79, p. 115–129, 2015.

[9] R. B. Lopes, F. Plastria, C. Ferreira et B. S. Santos, "Location-arc routing problem : Heuristic approaches and test instances," *Computers & Operations Research*, vol. 43, p. 309–317, 2014.

[10] F. L. Usberti, P. M. França et A. L. M. França, "Grasp with evolutionary path-relinking for the capacitated arc routing problem," *Computers & Operations Research*, vol. 40, n°. 12, p. 3206–3217, 2013.

[11] C. Martinez, I. Loiseau, M. G. Resende et S. Rodriguez, "Brkga algorithm for the capacitated arc routing problem," *Electronic Notes in Theoretical Computer Science*, vol. 281, p. 69–83, 2011.

[12] Y.-C. Liang, V. Minanda et A. Gunawan, "Waste collection routing problem : A mini-review of recent heuristic approaches and applications," *Waste Management & Research*, vol. 40, n°. 5, p. 519–537, 2022.

[13] J. Zhou, M. Zhang et S. Wu, "Multi-objective vehicle routing problem for waste classification and collection with sustainable concerns : the case of shanghai city," *Sustainability*, vol. 14, n°. 18, p. 11498, 2022.

[14] M. Umam, M. Rizki, M. Hamzah et S. Sutoyo, "Hybrid nearest neighbourhood search–symbiotic organisms search for solving garbage vehicle routing problem," dans *AIP Conference Proceedings*, vol. 2680, n°. 1. AIP Publishing, 2023.

[15] N. Perrier, A. Langevin et C.-A. Amaya, "Vehicle routing for urban snow plowing operations," *Transportation Science*, vol. 42, n°. 1, p. 44–56, 2008.

[16] M. A. Salazar-Aguilar, A. Langevin et G. Laporte, "Synchronized arc routing for snow plowing operations," *Computers & Operations Research*, vol. 39, n°. 7, p. 1432–1440, 2012.

[17] G. Liu, Y. Ge, T. Z. Qiu et H. R. Soleymani, "Optimization of snow plowing cost and time in an urban environment : A case study for the city of edmonton," *Canadian Journal of Civil Engineering*, vol. 41, n°. 7, p. 667–675, 2014.

[18] O. Quirion-Blais, A. Langevin et M. Trépanier, "A case study of combined winter road snow plowing and de-icer spreading," *Canadian Journal of Civil Engineering*, vol. 44, n°. 12, p. 1005–1013, 2017.

[19] N. Perrier, A. Langevin et J. F. Campbell, "A survey of models and algorithms for winter road maintenance. part iv : Vehicle routing and fleet sizing for plowing and snow disposal," *Journal of Operational Research*, 2007.

[20] ——, "A survey of models and algorithms for winter road maintenance. part iii : Vehicle routing and depot location for spreading," *Journal of Operational Research*, 2007.

[21] L. D. Bodin et S. J. Kursh, "A computer-assisted system for the routing and scheduling of street sweepers," *Operations Research*, vol. 26, n°. 4, p. 525–537, 1978.

[22] R. W. Eglese et H. Murdock, "Routeing road sweepers in a rural area," *Journal of the Operational Research Society*, vol. 42, p. 281–288, 1991.

[23] C. A. Blazquez, A. Beghelli et V. P. Meneses, "A novel methodology for determining low-cost fine particulate matter street sweeping routes," *Journal of the Air & Waste Management Association*, vol. 62, n°. 2, p. 242–251, 2012.

[24] C. Cerrone, B. Dussault, B. Golden et E. Wasil, "Multi-period street scheduling and sweeping," *International Journal of Metaheuristics*, vol. 3, n°. 1, p. 21–58, 2014.

[25] B. Golden, J. Nossack, E. Pesch et R. Zhang, "Routing problems with time dependencies or how different are trash collection or newspaper delivery from street sweeping or winter gritting ?" *Procedia Engineering*, vol. 182, p. 235–240, 2017.

[26] C. Yurtseven et M. A. Gökçe, "A novel arc-routing problem of electric powered street sweepers with time windows and intermediate stops," *IFAC-PapersOnLine*, vol. 52, n°. 13, p. 2308–2313, 2019.

[27] T. Parsons, J. Seo et D. Livesey, "Municipal street-sweeping area generation with route optimization," *International Transactions in Operational Research*, vol. 32, n°. 3, p. 1375–1399, 2025.

[28] A. Kraiem, J.-F. Audy et A. Lamghari, "Mixed integer linear programming model for a multi-depot arc routing problem with different arc types and flexible assignment of end depot," *Transportation Research Procedia*, vol. 82, p. 1109–1119, 2025.

[29] ——, "Adaptive large neighbourhood search for the multi-depot arc routing problem with flexible assignment of end depot and different arc types," *Journal of the Operational Research Society*, p. 1–19, 2024.

[30] D. R. Morrison, S. H. Jacobson, J. J. Sauppe et E. C. Sewell, "Branch-and-bound algorithms : A survey of recent advances in searching, branching, and pruning," *Discrete Optimization*, vol. 19, p. 79–102, 2016.

[31] G. B. Dantzig et P. Wolfe, "Decomposition principle for linear programs," *Operations research*, vol. 8, n°. 1, p. 101–111, 1960.

[32] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. Savelsbergh et P. H. Vance, "Branch-and-price : Column generation for solving huge integer programs," *Operations research*, vol. 46, n°. 3, p. 316–329, 1998.

[33] A. Lamghari, O. Foutlane et J.-F. Audy, "Decomposition strategies for solving the spring sweeping routing problem," Département de management, Université du Québec à Trois-Rivières, Trois-Rivières, QC, Rapport, April 2024, unpublished report.

[34] M. G. Resende et C. C. Ribeiro, "Greedy randomized adaptive search procedures : Advances, hybridizations, and applications," *Handbook of metaheuristics*, p. 283–319, 2010.

[35] L. Lozano, D. Duque et A. L. Medaglia, "An exact algorithm for the elementary shortest path problem with resource constraints," *Transportation Science*, vol. 50, n°. 1, p. 348–357, 2016.