| | |
|---|---|
| **Titre:** Title: | Génération de colonnes et sélection d'arcs de recharge pour un problème d'horaires d'autobus électriques |
| **Auteur:** Author: | Yoann Sabatier Montanaro |
| **Date:** | 2025 |
| **Type:** | Mémoire ou thèse / Dissertation or Thesis |
| **Référence:** Citation: | Sabatier Montanaro, Y. (2025). Génération de colonnes et sélection d'arcs de recharge pour un problème d'horaires d'autobus électriques [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/64432/ |



| | |
|---|---|
| **URL de PolyPublie:** PolyPublie URL: | https://publications.polymtl.ca/64432/ |
| **Directeurs de recherche:** Advisors: | Quentin Cappart |
| **Programme:** Program: | Génie informatique |

**POLYTECHNIQUE MONTRÉAL**
affiliée à l'Université de Montréal

# Génération de colonnes et sélection d'arcs de recharge pour un problème d'horaires d'autobus électriques

**YOANN SABATIER MONTANARO**
Département de génie informatique et génie logiciel

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

Mars 2025

**POLYTECHNIQUE MONTRÉAL**
affiliée à l'Université de Montréal

Ce mémoire intitulé :

**Génération de colonnes et sélection d'arcs de recharge pour un problème d'horaires d'autobus électriques**

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

**Daniel ALOISE**, président
**Quentin CAPPART**, membre et directeur de recherche
**Guy DESAULNIERS**, membre et codirecteur de recherche
**Michel GENDREAU**, membre

# REMERCIEMENTS

Je souhaite tout d'abord remercier très sincèrement mes directeurs de recherche, Quentin Cappart et Guy Desaulniers, pour leur aide énorme tout au long de ce projet. Leur expertise, leurs conseils avisés, leur réactivité exemplaire et leur bienveillance ont été des atouts inestimables. Ce travail n'aurait pas été possible sans leur accompagnement et leur soutien constant.

Un immense merci à ma fiancée, Charline Courbon, pour avoir été à mes côtés tout au long de cette aventure. Ton soutien, ta patience et ta présence m'ont été d'une aide précieuse pour surmonter les moments difficiles et avancer sereinement.

Je souhaite également remercier mes amis Mathéo, Alexandre, Bilal, Dora, Hugo et François, avec qui j'ai partagé deux années magnifiques. Ces moments de camaraderie et de partage resteront gravés dans ma mémoire.

Je tiens à exprimer ma gratitude à mes collègues du laboratoire, Max, Gaël et Hugo, pour les échanges enrichissants et les moments conviviaux partagés.

Je remercie sincèrement les membres du jury, Daniel Aloise et Michel Gendreau, pour avoir accepté d'évaluer mon travail de recherche. Merci pour le temps et l'attention que vous consacrerez à l'analyse de ce mémoire.

Enfin, je remercie l'équipe de GIRO inc. pour leurs précieuses idées et leur collaboration tout au long du projet.

Merci à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce mémoire.

# RÉSUMÉ

Dans le cadre de la transition énergétique et de la réduction des émissions de gaz à effet de serre, l'adoption des autobus électriques se généralise, apparaissant comme une solution clé pour rendre les transports en commun durables. Cependant, cette transition soulève de nouveaux défis en termes d'optimisation, notamment en raison des contraintes liées à la recharge des batteries. L'optimisation associée consiste à assigner un ensemble de trajets planifiés à une flotte de véhicules, problème connu sous le nom de *problème d'horaires de véhicule (vehicle scheduling problem)*. Lorsqu'il s'agit de gérer plusieurs dépôts, le problème se complique et devient le *problème d'horaires de véhicules multi-dépôts (multi depot vehicle scheduling problem)*. Dans le cas des autobus électriques, le problème se transforme en *problème d'horaires de véhicules électriques multi-dépôts (multi depot electric vehicle scheduling problem)*, où il devient nécessaire de prendre en compte non seulement les itinéraires, mais également les temps et lieux de recharge.

Une méthode largement reconnue pour résoudre ces problèmes complexes est la *génération de colonnes*, qui divise le problème en deux étapes : un problème maître, qui génère une solution à partir d'un sous-ensemble d'horaires possibles, et des sous-problèmes, qui ajoutent des colonnes (horaires de bus) pour améliorer cette solution. Ces sous-problèmes sont souvent modélisés sous forme de problèmes de plus courts chemins sur un graphe, où les nœuds représentent notamment les trajets à effectuer. Dans les approches précédentes, les graphes utilisaient un réseau espace-temps avec une sélection dynamique des opportunités de recharge lors de la résolution du sous-problème. Cependant, cette approche ne permet pas d'imposer des contraintes ad-hoc sur les transitions entre trajets, ce qui est essentiel pour les sociétés de transport en commun et, donc, pour les entreprises spécialisées en logiciels d'optimisation pour le transport public, telles que GIRO Inc.

Dans ce travail, nous proposons une nouvelle formulation de graphe qui offre la possibilité d'imposer des contraintes supplémentaires sur les arcs reliant les trajets, répondant ainsi aux besoins des opérateurs. De plus, afin de réduire la taille du graphe et d'accélérer les calculs, nous avons intégré des stratégies avancées de sélection d'arcs. Ces stratégies incluent un précalcul des opportunités de recharge les plus pertinentes et une adaptation de l'algorithme *Greedy Randomized Adaptive Search Procedure* proposé par Jacquet et al. [1], spécifiquement modifié pour fonctionner dans le cadre de notre graphe reformulé.

Nos résultats expérimentaux, obtenus sur des instances réalistes dérivées de données de lignes de bus montréalaises, montrent que notre approche atteint des performances comparables à

celles des méthodes de l'état de l'art, tout en offrant une flexibilité supplémentaire pour respecter des contraintes opérationnelles spécifiques. Par ailleurs, nous proposons une méthodologie permettant aux utilisateurs d'ajuster facilement l'équilibre entre la réduction du temps de calcul et le surcoût en fonction de leurs besoins, offrant ainsi une solution adaptable et robuste dans des contextes variés.

# ABSTRACT

In the context of the energy transition and the reduction of greenhouse gas emissions, the adoption of electric buses is gaining momentum, emerging as a key solution to make public transportation sustainable. However, this transition introduces new optimization challenges, particularly due to constraints related to battery recharging. The optimization problem involves assigning a set of timetabled trips to a fleet of vehicles, a problem known as the *Vehicle Scheduling Problem*. When multiple depots are involved, the problem becomes more complex and is referred to as the *Multi Depot Vehicle Scheduling Problem*. In the case of electric buses, the problem evolves into the *Multi Depot Electric Vehicle Scheduling Problem*, requiring not only the planning of routes but also the scheduling of recharging times and locations.

A widely recognized method for addressing such complex problems is *Column Generation*, which divides the problem into two stages: a master problem that generates a solution based on a subset of bus schedules and subproblems that add columns (bus schedules) to improve this solution. These subproblems are often modeled as shortest path problems on a graph, where nodes represent trips and potential recharging opportunities. In previous approaches, these graphs relied on a time-space network with dynamic selection of recharging opportunities during the subproblem resolution. However, this approach does not allow for the enforcement of ad-hoc constraints on transitions between trips, which is essential for public transit companies and, therefore, for software optimization firms specializing in public transportation, such as GIRO Inc.

In this work, we propose a novel graph formulation that pre-selects recharging opportunities before solving the subproblems. This reformulation enables the enforcement of additional constraints on arcs connecting trips, meeting the specific needs of operators. Additionally, to reduce the graph size and accelerate computation, we have implemented advanced arc selection strategies. These include a pre-computation of the most relevant recharging opportunities and an adaptation of the *Greedy Randomized Adaptive Search Procedure* proposed by Jacquet et al. [1], specifically tailored to operate within our reformulated graph.

Our experimental results, conducted on realistic instances derived from Montreal's bus network data, demonstrate that our approach achieves performance comparable to state-of-the-art methods while providing additional flexibility to enforce operational constraints. Moreover, we propose a methodology that allows users to easily balance computation time reduction and solution cost according to their needs, offering a robust and adaptable solution for

diverse operational contexts.

# TABLE DES MATIÈRES

# LISTE DES TABLEAUX

# LISTE DES FIGURES

# LISTE DES SIGLES ET ABRÉVIATIONS

| | |
|---|---|
| VSP | Problème d'horaires de véhicules *(Vehicle Scheduling Problem)* |
| MDVSP | Problème d'horaires de véhicules multi-dépôts *(Multi-Depot Vehicle Scheduling Problem)* |
| EVSP | Problème d'horaires de véhicules électriques *(Electric Vehicle Scheduling Problem)* |
| MDEVSP | Problème d'horaires de véhicules électriques multi-dépôts *(Multi-Depot Electric Vehicle Scheduling Problem)* |
| GC | Génération de colonnes |
| GRASP | Procédure de recherche gloutonne et adaptative aléatoire *(Greedy Randomized Adaptive Search Procedure)* |
| MILP | Programmation linéaire en nombres mixtes *(Mixed Integer Linear Programming)* |
| SP | Sous-problème |
| BB | Méthode de séparation et évaluation (*Branch-and-Bound*) |
| BP | Méthode de génération de colonne en nombres entiers *(Branch-and-Price)* |
| GNN | Réseau de neurones pour graphes *(Graph Neural Network)* |
| CPAIOR | Conférence internationale sur l'intégration de la programmation par contraintes, l'IA et la recherche opérationnelle *(Integration of Constraint Programming, Artificial Intelligence, and Operations Research)* |

# CHAPITRE 1 INTRODUCTION

Le changement climatique est l'un des enjeux majeurs du XXI$^{e}$ siècle, avec des conséquences profondes sur l'environnement, les écosystèmes et les sociétés humaines. Pour atténuer ces effets, une transformation radicale du secteur des transports est nécessaire, notamment du transport routier, responsable de 18 % des émissions de gaz à effet de serre au Canada en 2021 [3]. Les gouvernements et les sociétés de transport sont donc à la recherche de solutions plus durables pour limiter leur impact environnemental. L'électrification des transports en commun apparaît comme une solution évidente pour améliorer la qualité de l'air en milieu urbain et réduire l'empreinte carbone. Par exemple, la ville de Montréal s'est engagée à développer une flotte d'autobus 100 % électrique d'ici 2040 [4].

## 1.1 Importance de l'électrification des transports publics

L'intégration des autobus électriques dans les réseaux de transport public présente de nombreux avantages environnementaux et sociaux. En réduisant la dépendance aux combustibles fossiles, les véhicules électriques contribuent à la diminution des émissions de gaz à effet de serre et à l'amélioration de la qualité de l'air en milieu urbain. Cette transition s'inscrit dans les objectifs climatiques internationaux et soutient le développement de villes plus durables et résilientes [5].

Cependant, cette transition s'accompagne de défis techniques et opérationnels significatifs. Contrairement aux autobus traditionnels, les autobus électriques sont confrontés à des contraintes liées à l'autonomie limitée des batteries, aux temps de recharge plus longs et à la disponibilité des infrastructures de recharge [6]. Ces contraintes nécessitent une planification minutieuse pour assurer un service fiable et efficace.

## 1.2 Définition du problème et enjeux opérationnels

La planification des réseaux de transports en commun est un processus complexe qui peut être divisé en trois niveaux : la planification stratégique, la planification tactique et la planification opérationnelle [7].

Tout d'abord, la planification stratégique porte sur les décisions à long terme. Celles-ci sont extrêmement importantes car elles façonnent l'infrastructure tout entière. Cela inclut la conception du réseau, c'est-à-dire déterminer les lignes de transport, les trajets des bus ou des métros, et les connexions entre les différents points d'intérêt (zones résidentielles,

centres commerciaux, lieux historiques). Dans le cas des bus électriques, la planification stratégique englobe également le positionnement des stations de recharge ainsi que l'acquisition des bornes de recharge nécessaires pour garantir une bonne couverture.

Ensuite vient la planification tactique, qui concerne les décisions à moyen terme (saisonnières ou annuelles). Ses opérations ajustent les services pour répondre aux besoins des passagers. En particulier, elle s'occupe de fixer les fréquences des services sur chaque ligne en fonction de la demande prévue et établit les horaires pour les véhicules. Le but est de coordonner les opérations pour assurer des correspondances minimales et ainsi minimiser les temps d'attente pour les passagers.

Enfin, il y a la planification opérationnelle, à court terme et axée sur la gestion quotidienne des opérations pour offrir le service prévu à un coût minimal. Il s'agit notamment d'affecter des horaires à des véhicules, des véhicules à des conducteurs et de gérer les dépôts de bus.

Ces trois niveaux de planification sont interdépendants et contribuent ensemble à l'efficacité globale des services de transport en commun.

Dans ce travail, nous nous intéressons à la planification opérationnelle et plus particulièrement au processus d'affectation des horaires à des bus dont l'un des problèmes centraux est le *problème d'horaires de véhicules (Vehicle Scheduling Problem ou VSP)*. Il consiste à affecter un ensemble de trajets à une flotte de véhicules tout en respectant diverses contraintes [8]. Cependant, dans les réseaux de transport plus complexes, les véhicules peuvent être répartis sur plusieurs dépôts géographiquement distincts, ce qui donne lieu au *problème d'horaires de véhicules multi-dépôts (Multi-Depot Vehicle Scheduling Problem ou MDVSP)*. Cette version étendue introduit des défis supplémentaires, car il faut non seulement affecter les trajets aux véhicules, mais aussi déterminer à partir de quel dépôt chaque véhicule doit opérer. Le MDVSP est reconnu pour être un problème NP-difficile, en raison de la complexité combinatoire accrue lorsqu'il s'agit de gérer simultanément plusieurs dépôts et une grande flotte de véhicules [9].

L'introduction des autobus électriques dans les réseaux de transport ajoute encore une couche de complexité au MDVSP. Ces véhicules, bien qu'ils présentent des avantages environnementaux significatifs, sont contraints par une autonomie limitée et nécessitent des temps de recharge parfois longs. Le problème devient ainsi le *problème d'horaires de véhicules électriques multi-dépôts (Multi-Depot Electric Vehicle Scheduling Problem ou MDEVSP)*. Le problème d'horaire de bus électriques est NP-difficile, même dans le cas avec un seul dépôt [10].

## 1.3 Défis liés aux autobus électriques

L'utilisation d'autobus électriques introduit de nouvelles contraintes dans la planification opérationnelle. En particulier, il est essentiel de déterminer *où*, *quand* et *combien de temps* recharger les véhicules. La nature non linéaire du processus de recharge des batteries et les contraintes de capacités dans les stations de recharge ajoutent une couche supplémentaire de complexité [11].

L'optimisation dans ce contexte est cruciale pour plusieurs raisons. Premièrement, elle permet de réduire les coûts opérationnels tout en maximisant l'efficacité du réseau. Deuxièmement, elle contribue à l'amélioration de la fiabilité du service en évitant les interruptions dues à une batterie déchargée. Enfin, une planification optimisée permet de réduire les dépenses liées à la maintenance, au personnel et à l'énergie consommée.

## 1.4 Génération de colonnes et état de l'art

Ces dernières décennies, la *génération de colonnes* (GC) est devenue une méthode populaire pour résoudre les problèmes de planification des véhicules à grande échelle [12], y compris le MDEVSP.

### 1.4.1 La méthode de génération de colonnes

La GC est une méthode itérative utilisée pour résoudre des problèmes d'optimisation linéaire de grande taille en décomposant le problème en un *problème maître restreint* (PMR) et des *sous-problèmes* (SPs) [13]. Elle commence par une formulation initiale du PMR avec un ensemble restreint de colonnes, qui est résolu pour obtenir une solution primale et des valeurs duales. Ces valeurs duales sont ensuite utilisées dans les SPs pour identifier de nouvelles colonnes à coûts réduits négatifs, ici modélisés comme des problèmes de plus court chemin dans le cadre du MDEVSP [14]. Les colonnes prometteuses sont ajoutées au PMR, et le processus est répété jusqu'à ce qu'aucune nouvelle colonne améliorante ne soit trouvée. Enfin, la GC est imbriquée dans une méthode de type *branch-and-bound* (BB) résultant en une méthode dite de *branch-and-price* (BP) pour obtenir une solution entière optimale.

### 1.4.2 Limites de l'état de l'art

Malgré ses avantages, la GC rencontre des défis lorsqu'elle est appliquée au MDEVSP. La complexité du problème peut entraîner des espaces de solution très vastes, rendant la méthode coûteuse en termes de calcul, particulièrement pour les instances réelles comprenant de

nombreux trajets et dépôts. Un goulot d'étranglement majeur réside dans la taille du graphe utilisé pour le SP de la GC. Les réseaux de grande échelle entraînent des temps de calcul plus longs, ce qui peut être inacceptable pour les compagnies de transport qui ont besoin de décisions rapides.

Dans les travaux de Gerbeaux et al. [2], l'apprentissage automatique est utilisé pour réduire la taille du graphe, optimisant le processus de planification en prédisant quels arcs sont les plus susceptibles de contribuer à une solution optimale. Bien que cette approche offre une solution efficace, l'intégration de l'apprentissage automatique introduit des défis importants pour des partenaires industriels comme GIRO. Ces défis incluent la gestion et la collecte de grands ensembles de données, la conformité aux réglementations sur la confidentialité des données (telles que le *règlement général de protection des données* en Europe) et l'anonymisation des informations sensibles, ce qui peut devenir complexe et coûteux.

En réponse à cela, Jacquet et al. [1] ont proposé une approche basée sur deux métaheuristiques, la recherche en faisceau [15] et la méthode *Greedy Randomized Search Procedure* (GRASP) [16]. Leur méthode consiste à construire des solutions initiales, pas nécessairement faisables, en explorant de manière gloutonne les arcs les plus prometteurs. Après plusieurs itérations, les arcs sélectionnés dans ces solutions sont rassemblés dans un ensemble global, permettant de réduire la taille du graphe en ne conservant que ceux présents dans cet ensemble, c'est-à-dire, ceux ayant un fort potentiel d'appartenir à la solution optimale. Bien qu'utilisant un paradigme différent, cette méthode atteint des résultats similaires à ceux de Gerbaux et al. [2] en termes de réduction des temps de calcul et d'augmentation du coût.

Cependant, les deux approches précédemment mentionnées reposent sur une formulation de graphe qui détermine les chemins de recharge de manière dynamique durant le processus d'optimisation. Bien que cette approche offre une certaine flexibilité, elle présente des limitations importantes lorsqu'il s'agit de répondre aux exigences opérationnelles strictes des compagnies de transport. En effet, ces dernières souhaitent un contrôle accru sur la séquence des trajets, en ajoutant notamment des contraintes ad-hoc interdisant que deux trajets spécifiques se suivent directement, ce qui est difficilement réalisable avec la formulation actuelle du graphe.

Cette limitation découle principalement de la structure même du graphe utilisé. Dans cette configuration, les nœuds de recharge sont communs à l'ensemble des trajets. Par conséquent, il existe toujours un chemin dans le graphe reliant deux trajets via un nœud de recharge. Cette caractéristique rend impossible l'interdiction complète de la succession de deux trajets en supprimant simplement certains arcs du graphe. En effet, même si des arcs directs entre deux trajets sont supprimés, le graphe conserve des chemins alternatifs passant par les nœuds

de recharge, contournant ainsi les restrictions souhaitées.

## 1.5 Notre contribution : reformulation et réduction du graphe

Ce travail présente deux contributions principales pour résoudre efficacement le MDEVSP. La première est une reformulation novatrice du graphe qui intègre les contraintes supplémentaires imposées par les opérateurs de transport, tout en permettant une plus grande flexibilité dans l'affectation des trajets. Cette reformulation offre une meilleure adaptabilité aux besoins opérationnels concrets. Elle permet désormais d'interdire facilement la succession de deux trajets, même en présence d'opérations de recharge entre eux.

La deuxième contribution porte sur la réduction de la taille du graphe, un aspect clé pour accélérer les calculs tout en maintenant la qualité des solutions. Trois stratégies principales ont été développées à cet effet. La première stratégie consiste en une présélection des opportunités de recharge, permettant de ne conserver que celles jugées essentielles pour une bonne planification. La deuxième stratégie repose sur une réinterprétation des travaux de Jacquet et al. [1], où l'algorithme GRASP est revisité et adapté pour s'intégrer à la nouvelle structure du graphe. Enfin, la troisième stratégie propose une sélection d'arcs *trajet-à-trajet* basée sur une analyse de temps de trajet à vide, visant à identifier les arcs les plus prometteurs pour inclusion dans le graphe.

Les résultats expérimentaux démontrent l'efficacité de ces approches. Pour les petites instances, une réduction de temps de calcul d'en moyenne 63 % a été observée avec un impact négligeable sur la qualité des solutions (surcoût de moins de 1.4 %). Pour les grandes instances, la réduction atteint 45 %, avec une dégradation de seulement 3.7 % des coûts. Ces avancées offrent une solution pratique et efficace aux défis rencontrés dans la planification des réseaux de véhicules électriques, répondant ainsi aux besoins croissants des opérateurs face à l'adoption accrue des autobus électriques dans les villes.

Par ailleurs, les travaux réalisés dans le cadre de ce mémoire ont été soumis à la conférence *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research* (CPAIOR).

## 1.6 Organisation du document

Le reste de ce document est organisé comme suit. Dans le chapitre 2, nous présentons une revue de littérature détaillée sur les méthodes de résolution du MDVSP et de l'*electric vehicle scheduling problem* (EVSP). Dans le chapitre 3, nous décrivons le problème en détail

et proposons un modèle mathématique. Nous présentons ensuite, dans le chapitre 4, la méthodologie de résolution de Gerbaux et al. [2] ainsi que ses limites. Dans le chapitre 5, nous détaillons nos contributions, incluant la nouvelle formulation du graphe et les stratégies de sélection d'arcs. Nous discutons des résultats expérimentaux qui démontrent l'efficacité de notre méthode dans le chapitre 6. Enfin, dans le chapitre 7, nous concluons ce travail et suggérons des pistes pour des recherches futures.

# CHAPITRE 2 REVUE DE LITTÉRATURE

Dans ce chapitre, nous présentons une revue de littérature détaillée des différentes approches pour résoudre le MDEVSP. Nous commençons par les méthodes de résolution du problème classique du MDVSP, puis nous explorons l'évolution des méthodes et des formulations pour l'EVSP. Enfin, nous discutons des travaux récents et des problèmes connexes au MDEVSP.

## 2.1 Le problème de planification MDVSP

Le MDVSP est un problème central en optimisation combinatoire, particulièrement dans les transports publics. L'objectif est de minimiser les coûts d'exploitation en assignant des véhicules, répartis sur plusieurs dépôts, à des trajets prédéfinis. Ce problème est rendu complexe par des contraintes telles que les capacités des dépôts, les temps d'attente, les trajets à vide, et les coûts fixes et variables. Classifié comme NP-difficile [17], il a suscité le développement de nombreuses approches, allant des méthodes exactes aux heuristiques. Dans cette revue, nous ne reprenons pas exhaustivement tous les travaux, mais nous nous concentrons sur ceux ayant eu un impact significatif ou ayant proposé une méthodologie novatrice.

### 2.1.1 Approches exactes

Les premières approches exactes utilisaient des réseaux de connexions, où chaque paire de trajets successifs était représentée comme une connexion possible. Carpaneto et al. [18] ont proposé une méthode de BB combinant des bornes additives basées sur des relaxations. Cette approche permettait de résoudre des instances comportant jusqu'à 70 trajets et trois dépôts en un temps raisonnable. Cependant, la complexité croissante avec l'augmentation du nombre de trajets limitait son application à grande échelle.

La GC a marqué une avancée significative dans la résolution du MDVSP. Ribeiro et Soumis [19] ont reformulé le problème comme un problème de partition d'ensembles, où chaque colonne représentait un itinéraire faisable. En utilisant une relaxation continue et une technique de GC, ils ont pu résoudre des instances ayant jusqu'à 300 trajets et six dépôts en moins d'une heure.

En 2006, Kliewer et al. [20] ont introduit un modèle basé sur les réseaux espace-temps pour résoudre le MDVSP. Contrairement aux modèles par connexion, cette approche réduit significativement la taille des modèles en ne représentant que les événements de départ et d'arrivée comme nœuds distincts. Ils ont montré que cette méthode permettait de résoudre

des instances réalistes comportant jusqu'à 7000 trajets et cinq dépôts en un peu plus de trois heures, en utilisant des solveurs standards comme CPLEX. Ces avancées ont permis l'intégration pratique de ces modèles dans des logiciels de planification pour les systèmes de transport public.

### 2.1.2 Approches heuristiques

Les heuristiques se sont révélées essentielles pour traiter des instances à très grande échelle. Pepin et al. [21] ont comparé cinq heuristiques, incluant une recherche à voisinage large et une recherche tabou, combinées à une évaluation par GC. Ils ont montré que ces méthodes pouvaient produire des solutions avec un écart à l'optimalité de moins de 1% pour des instances de 1500 trajets et huit dépôts en moins de trois heures, offrant ainsi un compromis efficace entre la qualité de la solution et le temps de calcul.

Ainsi, l'évolution des approches pour le MDVSP montre la recherche d'un équilibre entre précision et extensibilité. Les méthodes exactes, bien qu'efficaces pour des instances de taille moyenne, sont limitées par leur complexité computationnelle. Les modèles avancés, comme ceux basés sur les réseaux espace-temps, et les heuristiques modernes, telles que la recherche à voisinage étendu, ont élargi la portée des solutions à des problèmes à grande échelle, offrant des outils pratiques pour les systèmes de transport modernes.

## 2.2 Évolution des méthodes pour le EVSP

L'EVSP est une extension du VSP. Le VSP vise à minimiser le coût total de la flotte en assignant des trajets aux véhicules de manière optimale. Il s'agit d'un problème de flot à coût minimum qui peut être résolu en temps polynomial. L'EVSP, quant à lui, introduit des contraintes supplémentaires liées aux spécificités des véhicules électriques. Ces contraintes incluent principalement la capacité limitée des batteries, le temps de recharge et la disponibilité des stations de recharge. Ces nouvelles contraintes augmentent la complexité du problème et nécessitent des approches novatrices pour garantir des solutions réalistes et efficaces. Perumal et al. [6] proposent une revue de littérature récente et complète sur ce sujet.

### 2.2.1 Historique de la modélisation de la recharge

Dans sa forme la plus élémentaire, l'EVSP repose sur l'hypothèse simplificatrice d'un temps de recharge constant, indépendant du niveau initial de la batterie. Cette approche, qui suppose souvent un remplacement direct des batteries, a été utilisée pour minimiser le nombre de batteries nécessaires, comme l'ont démontré Chao et Xiaohong [22]. Cependant, cette hy-

pothèse simplificatrice présente des limites significatives dans des contextes réels. Supposer un temps de recharge constant ne reflète pas la réalité et la possibilité de recharges partielles permet une flexibilité accrue et une meilleure optimisation des opérations.

Pour répondre à ces limites, Wen et al. [23] ont proposé un modèle de *programmation linéaire mixte* (MILP) combiné à une heuristique de recherche adaptative à voisinage large. Ce modèle intègre une fonction de recharge linéaire, où le temps de recharge est proportionnel à l'énergie ajoutée.

Par ailleurs, Zhang et al. [24] ont démontré l'efficacité d'une politique de recharge partielle dans un cadre multi-dépôts avec des flottes de véhicules hétérogènes. Leur approche a permis de réduire la taille des flottes de 7 % en moyenne, tout en respectant les contraintes opérationnelles liées à la sécurité énergétique des véhicules (20 % de capacité minimale de la batterie).

Même si ces approches linéaires ont longtemps représenté une avancée par rapport aux hypothèses de recharge constante, Montoya et al. [11] ont montré que ces modèles restaient insuffisants pour refléter le comportement réel des batteries, notamment lorsque le niveau de charge s'approche de sa capacité maximale. Cette lacune réside dans l'hypothèse simplificatrice selon laquelle la charge de la batterie est linéairement proportionnelle au temps passé à la station. Or, le processus réel de recharge est non linéaire, caractérisé par une phase initiale où le courant reste constant, suivie d'une phase exponentielle de diminution du courant pour éviter d'endommager la batterie.

Toutefois, l'utilisation de fonctions différentielles pour modéliser exactement la non-linéarité des batteries aurait introduit une complexité excessive, rendant les résolutions impraticables pour des instances à grande échelle. Ainsi, Montoya et al. [11] ont introduit une modélisation par approximation linéaire par morceaux, qui segmente la courbe de recharge en plusieurs segments linéaires définis par des points d'arrêt. Cette approche permet une meilleure prise en compte des deux phases critiques du processus de recharge, tout en restant intégrable dans les modèles d'optimisation comme l'EVSP.

Les résultats numériques de Montoya et al. révèlent que l'approximation par morceaux réduit significativement les erreurs associées aux modèles linéaires. Ils montrent ainsi qu'une approximation linéaire peut sous-estimer les temps de recharge nécessaires, produisant jusqu'à 14 solutions infaisables sur 20 instances et augmentant les coûts de 2,70 % en moyenne [11].

### 2.2.2 Résolution de l'EVSP

Les méthodologies exactes, telles que MILP, offrent des solutions optimales pour des instances de petite taille. Par exemple, Sassi et Oulamara [10] ont démontré que le *problème de planification et de recharge des véhicules électriques* est NP-difficile et ont développé une formulation MILP permettant de résoudre des instances de petite et moyenne taille à l'aide de CPLEX. Cependant, ces approches deviennent impraticables pour des instances de grande taille en raison de leur complexité computationnelle.

Pour des instances plus importantes, les heuristiques et métaheuristiques sont devenues incontournables. La méthode de Wen et al. [23] discutée précédemment s'est avérée efficace pour résoudre des instances complexes ayant jusqu'à 500 trajets en moins de 20 minutes.

Par ailleurs, des études récentes ont exploré l'impact des choix de modélisation sur les performances environnementales et économiques. Pasha et al. [25] ont intégré des données en temps réel sur les coûts énergétiques et les émissions de CO2 mettant en lumière la nécessité d'une approche combinant planification des trajets et gestion intelligente de la recharge, pour maximiser les bénéfices environnementaux et économiques.

En complément, des travaux comme ceux de Savari et al. [26] ont mis l'accent sur la nécessité d'infrastructures de recharge efficaces et sur les interactions entre les véhicules électriques et le réseau électrique. Ces études ont révélé l'importance de modèles prédictifs et d'approches en temps réel pour garantir une recharge optimale et minimiser les impacts sur le réseau électrique.

### 2.2.3 Hybridation de la génération de colonnes pour le MDEVSP

La GC est une méthode largement utilisée pour résoudre le MDEVSP en raison de son efficacité dans la gestion des grandes instances combinatoires. De plus, son intégration avec d'autres approches permet d'améliorer les temps de calcul, offrant ainsi des perspectives prometteuses pour les problèmes à grande échelle.

Le travail de Wang et al. [27] propose une approche hybride où la GC a été combinée avec un algorithme génétique. Spécifiquement, la GC génère un ensemble initial de colonnes (représentant des horaires potentiels) à partir desquelles l'algorithme génétique sélectionne les colonnes optimales. Cette méthode utilise un codage binaire pour représenter les solutions, et des stratégies élitistes pour accélérer la convergence. Les résultats montrent que cette méthode est environ 40 fois plus rapide que les approches exactes comme le BP, tout en maintenant une solution de haute qualité pour des instances jusqu'à 500 trajets.

Dans une approche récente, Gerbaux et al. [2] ont utilisé l'apprentissage supervisé, en parti-

culier des *réseaux de neurones pour graphes* (GNN), pour améliorer l'efficacité de la GC dans le MDEVSP. L'objectif était de réduire la taille du graphe utilisé dans le modèle en éliminant les arcs peu pertinents. Cette méthode combine un algorithme glouton avec le GNN pour identifier les arcs critiques. Les expériences ont montré une réduction de 65 à 87 % du temps de calcul pour les petites instances, avec une perte de qualité limitée à 3.7 %. Cependant, la généralisation du modèle sur des instances différentes reste un défi.

Enfin Jacquet et al. [1] ont exploré une autre approche en utilisant les métaheuristiques GRASP et la recherche en faisceau pour optimiser la GC. Ces méthodes, similaires au travail de Gerbaux et al. permettent de sélectionner des arcs ayant un fort potentiel d'appartenance à la solution optimale, réduisant ainsi la taille du graphe tout en conservant une perte d'optimalité inférieure à 5 %. Cette stratégie s'est avérée particulièrement efficace sur des instances contenant jusqu'à 2500 trajets, avec un gain de temps supérieur à 50 % par rapport aux méthodes traditionnelles. Contrairement à Gerbaux et al., cette approche n'emploie pas d'apprentissage automatique, ce qui simplifie son implémentation tout en offrant une robustesse similaire.

Cependant, une limitation commune à ces deux approches réside dans la structure du graphe utilisée, qui ne permet pas d'ajouter des contraintes ad-hoc spécifiques aux besoins des opérateurs. Cette rigidité limite leur capacité à intégrer des règles opérationnelles complexes ou des scénarios personnalisés. Dans ce travail, nous proposons une méthode qui surmonte cette limitation en offrant une flexibilité accrue pour inclure de telles contraintes, tout en maintenant une efficacité computationnelle élevée.

### 2.3 Synthèse

En résumé, le MDEVSP est un problème complexe qui a été abordé par diverses méthodes, allant des approches exactes aux heuristiques et métaheuristiques. L'introduction des véhicules électriques a ajouté de nouvelles contraintes qui rendent la résolution du problème encore plus difficile. Les méthodes actuelles présentent des limitations en termes de temps de calcul ou de flexibilité opérationnelle. Notre travail vise à proposer une approche pratique et efficace pour résoudre le MDEVSP en tenant compte des contraintes spécifiques liées aux autobus électriques.

# CHAPITRE 3 ÉNONCÉ DU PROBLÈME ET MODÈLE MATHÉMATIQUE

Cette section présente une description détaillée du MDEVSP. Nous commençons par décrire les composants clés du problème, y compris les fonctions de consommation d'énergie et de recharge, puis nous fournissons une formulation mathématique.

## 3.1 Définition du problème

Le MDEVSP consiste à planifier les horaires d'une flotte de bus électriques pour couvrir un ensemble $T = \{t_1, t_2, \ldots, t_N\}$ de $N$ trajets, où chaque trajet $t \in T$ a une heure de départ $q_t^D$, une heure d'arrivée $q_t^E$, un lieu de départ $l_t^D$, et un lieu d'arrivée $l_t^E$. Chaque trajet doit être couvert une fois par un bus.

Les bus sont logés dans un ensemble $D$ de dépôts, avec chaque dépôt $d \in D$ ayant $v_d$ bus. Chaque bus a une capacité de batterie de $\overline{\sigma}$ et commence la journée entièrement chargé, pour retourner à son dépôt d'origine en fin de journée. Les bus peuvent se recharger dans des stations de recharge appartenant à un ensemble $H$, où chaque station $h \in H$ dispose de $u_h$ chargeurs disponibles. La recharge suit une fonction définie comme $\sigma^F = f(\sigma^I, \Lambda)$, où $\sigma^I$ est l'état initial de charge (*state-of-charge* ou SoC), $\Lambda$ est le temps de recharge, et $\sigma^F$ est l'état final de charge. La fonction $f$ est concave et linéaire par morceaux [11], représentant différents taux de recharge évoluant au cours du temps, comme présenté sur la figure 3.1.

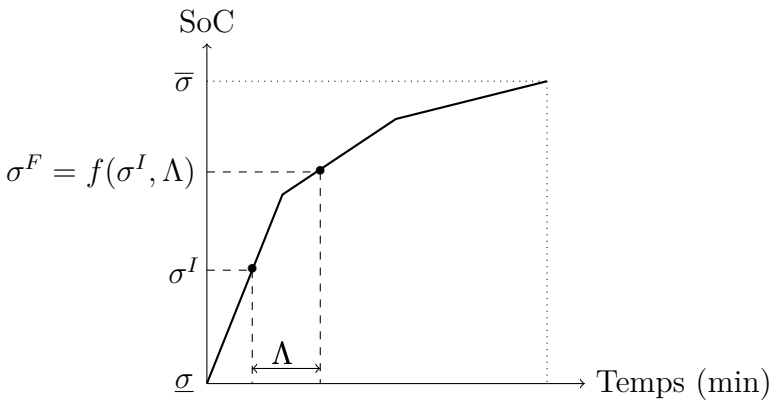



FIGURE 3.1 Fonction de recharge linéaire par morceaux tirée de Gerbaux et al. [2]

Pour gérer les sessions de recharge, le temps est discrétisé en un ensemble $P = \{p_1, p_2, \ldots, p_M\}$ de $M$ intervalles consécutifs de $\delta$ minutes. Chaque session de recharge doit être effectuée sur un nombre de périodes consécutives dans $P$. Une session de recharge ne doit jamais dépasser

la capacité maximale de la batterie, mais le bus occupera le chargeur pendant toutes les périodes de recharge réservées.

L'objectif est d'assigner les trajets aux bus afin de minimiser les coûts opérationnels tout en respectant certaines contraintes. Chaque bus suit un itinéraire débutant et se terminant au même dépôt, avec des niveaux de batterie restant dans l'intervalle $[\underline{\sigma}, \overline{\sigma}]$. Les bus peuvent retourner à n'importe quel dépôt pour une pause, à condition d'y rester au moins $\gamma$ minutes. Pour éviter un temps d'inactivité excessif en dehors d'un dépôt, le temps entre deux trajets consécutifs $t_1$ et $t_2$ est contraint par $q^D_{t_2} - q^E_{t_1} \leq \beta$.

Chaque trajet doit être couvert exactement une fois, et le nombre de bus utilisés dans chaque dépôt ne peut pas dépasser sa flotte disponible. À tout moment, le nombre de bus en recharge ne peut pas dépasser le nombre de chargeurs disponibles dans les stations de recharge.

Les coûts opérationnels incluent des coûts fixes et variables. Un coût fixe $c^F$ est associé à chaque bus utilisé dans la solution. Les coûts variables incluent un coût d'attente $c^W$ par unité de temps pour les bus inactifs hors dépôt, un coût à vide $c^T$ par unité de temps pour les déplacements sans passagers, une pénalité $c^R$ pour le retour au dépôt, et des coûts de recharge composés d'un coût fixe $c^H_F$ pour initier une recharge et d'un coût variable $c^H_V$ par unité de temps passée dans une station de recharge.

### 3.2 Formulation mathématique

Le MDEVSP peut être formulé comme un MILP. Soit $S_d$ l'ensemble des horaires faisables partant du dépôt $d \in D$. Pour chaque horaire $s \in S_d$, nous définissons les notations suivantes :

- $c_s$ : coût total associé à l'horaire $s$.
- $e^t_s$ : paramètre binaire indiquant si l'horaire $s$ couvre le trajet $t$.
- $g^{p,h}_s$ : paramètre binaire indiquant si l'horaire $s$ inclut une recharge à la station $h$ pendant la période $p$.
- $x_s$ : variable binaire égale à 1 si l'horaire $s$ est utilisé dans la solution.

La formulation est la suivante :

$$\min \quad \sum_{d \in D} \sum_{s \in S_d} c_s x_s \tag{3.1}$$

$$\text{sous contraintes :} \quad \sum_{d \in D} \sum_{s \in S_d} e_s^t x_s = 1 \qquad \forall t \in T \tag{3.2}$$

$$\sum_{s \in S_d} x_s \leq v_d \qquad \forall d \in D \tag{3.3}$$

$$\sum_{d \in D} \sum_{s \in S_d} g_s^{p,h} x_s \leq u_h \qquad \forall p \in P, \forall h \in H \tag{3.4}$$

$$x_s \in \{0,1\} \qquad \forall d \in D, \forall s \in S_d. \tag{3.5}$$

La fonction objectif (3.1) minimise les coûts opérationnels totaux. Les contraintes (3.2) garantissent que chaque trajet est couvert exactement une fois. Les contraintes (3.3) limitent le nombre de bus utilisés dans chaque dépôt à la taille de sa flotte. Les contraintes (3.4) restreignent le nombre de recharges simultanées au nombre de chargeurs disponibles dans chaque station. Enfin, les contraintes (3.5) imposent des restrictions binaires sur les variables représentant les horaires des bus. La figure 3.2 présente une représentation visuelle d'une solution au MDEVSP. Pour chaque bus, elle met en évidence les périodes dédiées aux trajets, à la recharge, à l'attente à la recharge, ainsi qu'à l'attente aux dépôts.

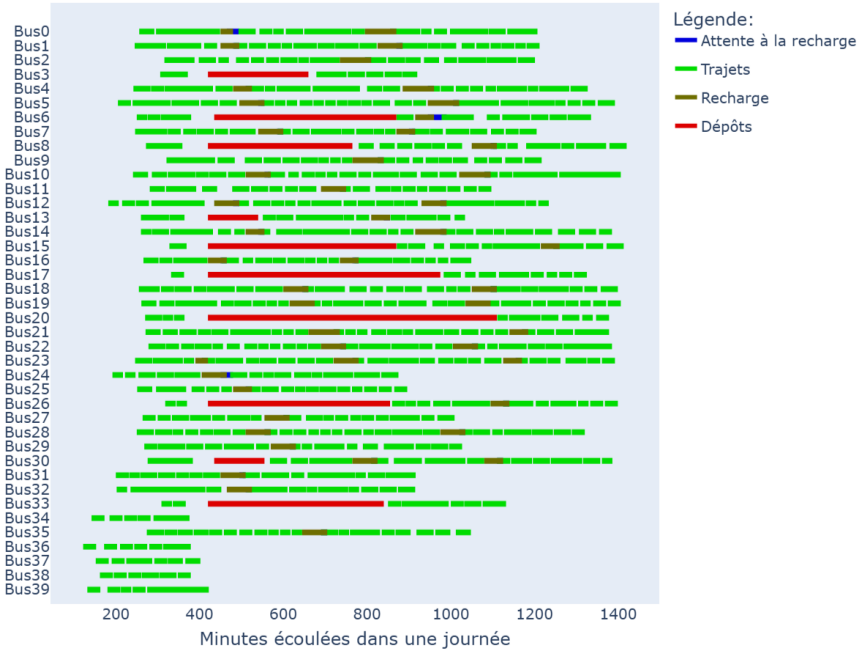



FIGURE 3.2 Représentation visuelle d'une solution du MDEVSP

# CHAPITRE 4 RÉSOLUTION DU PROBLÈME PAR GÉNÉRATION DE COLONNES ET APPRENTISSAGE AUTOMATIQUE

Cette section présente l'heuristique de GC proposée par Gerbaux et al. [2] pour résoudre le MDEVSP. Nous y discutons de la méthodologie proposée et des limites associées.

## 4.1 Génération de colonnes pour le MDEVSP

La GC est une approche itérative conçue pour résoudre efficacement des problèmes d'optimisation linéaire de grande taille. L'idée centrale est de résoudre la relaxation linéaire des contraintes (3.1)–(3.4), appelée le *problème maître* (PM). À chaque itération, des SPs sont résolus pour identifier de nouvelles colonnes (horaires) ayant des coûts réduits négatifs, susceptibles d'améliorer la solution actuelle. Le PMR est une version simplifiée du PM, où seules certaines colonnes représentant des horaires faisables sont incluses. Dans le cadre du MDEVSP, chaque dépôt est associé à un SP formulé comme un problème de plus court chemin avec contraintes de ressources sur un réseau espace-temps [14]. Les grandes étapes de la résolution par GC sont décrites ci-dessous.

1. **Formulation initiale du PMR.** Le PMR est initialisé sans recourir à un ensemble prédéfini d'horaires ou de colonnes initiales. À la place, la GC est démarrée en introduisant des variables artificielles dans le PMR, suivie d'une phase visant à éliminer progressivement ces variables de la solution, permettant ainsi de construire une base réalisable initiale pour le problème.
2. **Résolution du PMR et récupération des variables duales.** À l'itération $\ell$, on résout le PMR restreint aux ensembles d'horaires $S_d^\ell \subseteq S_d, \forall d \in D$. On obtient alors une solution primale et un vecteur de variables duales $(\pi^\ell, \eta^\ell, \nu^\ell)$ associées respectivement aux contraintes de couverture des trajets (3.2), de capacité des dépôts (3.3) et de capacité des stations de recharge. (3.4).
3. **Résolution des SPs et génération de nouvelles colonnes.** Les SPs utilisent les valeurs duales $\pi^\ell$, $\eta^\ell$ et $\nu^\ell$ pour identifier de nouvelles colonnes, c'est-à-dire de nouveaux horaires, ayant un coût réduit négatif. Plus précisément, le coût réduit d'un horaire $s$ est donné par :

$$\overline{c}_s^\ell \;=\; c_s \;-\; \sum_{t \in T} \pi_t^\ell \, e_s^t \;-\; \eta_d^\ell \;-\; \sum_{p \in P} \sum_{h \in H} \nu_{p,h}^\ell \, g_s^{p,h}, \tag{4.1}$$

Si un horaire avec $\bar{c}_s^\ell < 0$ est trouvé, il est ajouté à l'ensemble $S_d^\ell$ et le PMR est mis à jour.

4. **Répétition jusqu'à convergence.** Une fois que les nouvelles colonnes sont ajoutées au PMR, ce dernier est résolu à nouveau, fournissant de nouvelles valeurs duales. Le processus est itératif et se poursuit tant qu'il existe au moins une colonne de coût réduit négatif. En l'absence de telles colonnes, la solution courante du PMR est optimale pour la relaxation linéaire du problème maître.
5. **Passage à une solution entière.** Lorsqu'aucune colonne de coût réduit négatif ne peut être générée, le PMR atteint son optimum linéaire. Toutefois, une solution entière étant nécessaire, des techniques de branchement, détaillées dans une section ultérieure, sont utilisées pour imposer les contraintes d'intégrité. Il est à noter que la GC est appliquée à chaque fois qu'une relaxation linéaire doit être résolue.

### 4.2 Résolution d'un SP

Le SP pour un dépôt $d \in D$ peut être défini sur un réseau acyclique $\mathcal{G}_d = (\mathcal{V}_d, \mathcal{A}_d)$. Le réseau utilisé par Gerbaux et al. [2] est illustré dans la figure 4.1, où $h$ et $d'$ représentent respectivement une station de recharge et un dépôt arbitraire. L'ensemble de nœuds $\mathcal{V}_d$ comprend les nœuds de trajet ($t$), les nœuds de dépôt ($o^d$ et $k^d$), les nœuds de recharge ($r_p^h$, pour les recharges à la station $h$ pendant la période $p$), et les nœuds d'attente ($w_p^h$ pour les arrêts à la station $h$ et $s_p^{d'}$ pour les arrêts au dépôt $d'$ pendant la période $p$). Il y a six types d'arcs dans l'ensemble $\mathcal{A}_d$, définis comme suit :

1. **Arcs de sortie et d'entrée** (en noir). Pour chaque trajet $t \in T$, un *arc de sortie* $(o_d, t)$ est créé, où $o_d$ représente le début de la journée au dépôt $d$. De même, un *arc d'entrée* $(t, k_d)$ relie $t$ à la fin de la journée au dépôt $d$. Pour chaque station $h$, un autre *arc d'entrée* $(w_{p_M}^h, k_d)$ permet à l'horaire de se terminer par une recharge avant de retourner au dépôt.
2. **Arcs trajet-à-trajet** (en rouge). Pour deux trajets distincts $t_i, t_j \in T$, un *arc trajet-à-trajet* $(t_i, t_j)$ est créé si $t_j$ peut suivre $t_i$, vérifiant la contrainte $\rho_{l_{t_i}^E, l_{t_j}^D} \leq q_{t_j}^D - q_{t_i}^E \leq \beta$, où $\rho_{i,j}$ est le temps de trajet entre deux points $i$ et $j$ dans le réseau.
3. **Arcs de retour au dépôt** (en vert). Pour chaque trajet $t \in T$ et dépôt $d \in D$, les *arcs depuis le dépôt* $(s_{p_L}^d, t)$ et les *arcs vers le dépôt* $(t, s_{p_E}^d)$ sont définis, garantissant que les contraintes temporelles pour revenir au dépôt ou en partir sont respectées. Ici, $p_L$ représente la dernière période à laquelle le bus peut quitter le dépôt pour atteindre le trajet $t$, et $p_E$ correspond à la première période à laquelle le bus peut arriver au dépôt après avoir terminé le trajet $t$.

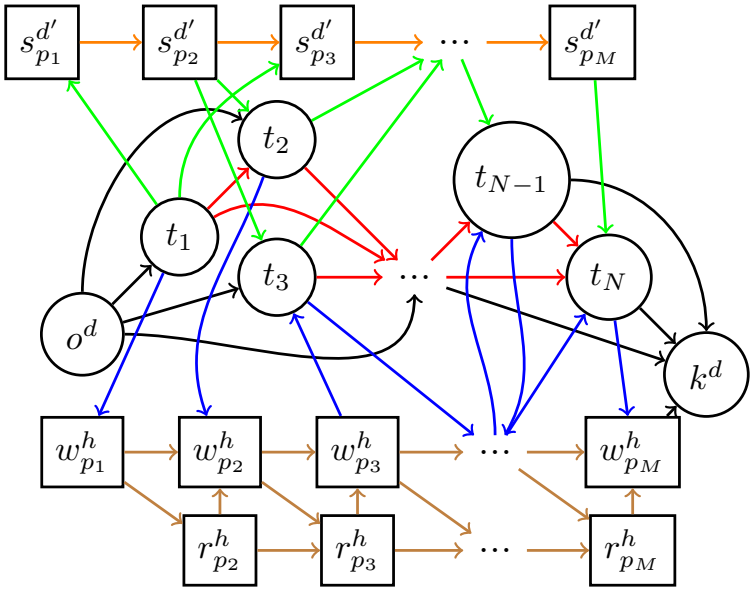



FIGURE 4.1 Réseau $\mathcal{G}_d$ utilisé par Gerbaux et al. [2].

4. **Arcs de connexion avec recharge** (en bleu). Pour chaque station de recharge $h \in H$ et chaque trajet $t \in T$, les *arcs de connexion avec recharge* connectent les trajets aux opérations de recharge. Il existe des *arcs depuis la station* $(w^h_{p_L}, t)$ et des *arcs vers la station* $(t, w^h_{p_E})$, où $p_L$ et $p_E$ sont définis comme ci-dessus, mais en considérant la station $h$ au lieu du dépôt $d$.
5. **Arcs d'attente au dépôt** (en orange). Chaque dépôt $d \in D$ dispose d'*arcs d'attente* $(s^d_{p_i}, s^d_{p_{i+1}})$ entre deux périodes consécutives $p_i$ et $p_{i+1}$.
6. **Arcs de recharge** (en marron). Chaque station de recharge $h \in H$ possède des *arcs d'attente* $(w^h_{p_i}, w^h_{p_{i+1}})$ et des *arcs de recharge* $(r^h_{p_i}, r^h_{p_{i+1}})$, ainsi que des *arcs de début et de fin de recharge* $(w^h_{p_i}, r^h_{p_{i+1}})$ et $(r^h_{p_i}, w^h_{p_i})$ pour les périodes $p_i$ et $p_{i+1}$.

Directement à partir de la définition du coût réduit d'un horaire (4.1), le coût $\overline{c}^\ell_{i,j}$ de chaque arc $(i,j) \in \mathcal{A}_d$ est calculé comme suit :

$$\overline{c}^\ell_{i,j} = \begin{cases} c_{i,j} - (\eta^\ell_d + \pi^\ell_j) & \text{si } i = o_d, j \text{ est un trajet} \\ c_{i,j} - \pi^\ell_j & \text{si } i, j \text{ sont des trajets} \\ c_{i,j} - \nu^\ell_{p,h} & \text{si } j \text{ correspond à une recharge} \\ c_{i,j} & \text{autrement,} \end{cases} \tag{4.2}$$

où

$$c_{i,j} = \begin{cases} \rho_{i,l_j^D} \cdot c^T & \text{si } i = o_d \text{ et } j \text{ est un trajet,} \\ \rho_{i,l_j^D} \cdot c^T + c^R & \text{si } i = s_p^d \text{ et } j \text{ est un trajet,} \\ \rho_{l_i^E,j} \cdot c^T & \text{si } i \text{ est un trajet et } (j = k_d \text{ ou } j = s_p^d), \\ \rho_{l_i^E,l_j^D} \cdot c^T + (q_j^E - q_i^D - \rho_{l_i^E,l_j^D}) \cdot c^W & \text{si } i \text{ et } j \text{ sont des trajets,} \\ \rho_{l_i^E,j} \cdot c^T + (p_E - q_i^E - \rho_{l_i^E,j}) \cdot c^W + c_F^H & \text{si } i \text{ est un trajet et } j \text{ est un nœud d'attente } w, \\ \rho_{i,l_j^D} \cdot c^T + (q_j^D - p_L - \rho_{i,l_j^D}) \cdot c^W & \text{si } i \text{ est un nœud d'attente } w \text{ et } j \text{ est un trajet,} \\ \delta \cdot c_V^H & \text{si } i \text{ et } j \text{ sont des nœuds } w \text{ ou } r, \\ \delta \cdot c_V^H & \text{si } i \text{ est un nœud } w \text{ et } j \text{ est un nœud } r, \\ 0 & \text{autrement.} \end{cases} \tag{4.3}$$

Chaque horaire faisable dans $S_d$ correspond à un chemin allant de $o^d$ à $k^d$ dans $\mathcal{G}_d$. Cependant, il existe des chemins qui ne représentent pas un horaire faisable car l'autonomie de la batterie n'est pas prise en compte dans la structure du réseau.

Gerbaux et al. [2] intègrent ces contraintes comme des ressources et résolvent le SP à l'aide d'un algorithme d'étiquetage. Nous renvoyons à cet article pour une description complète des ressources, des étiquettes et de l'algorithme.

### 4.3 Dérivation de solutions entières avec BP

Pour obtenir des solutions entières au MDEVSP, un algorithme de type BP est employé, combinant la GC avec un processus de BB. L'approche repose principalement sur des branchements heuristiques, qui génèrent un seul nœud enfant à chaque étape, et, dans de rares cas, sur des branchements dichotomiques, qui créent deux nœuds enfants.

Les branchements heuristiques, largement privilégiés dans ce travail, permettent de fixer certaines variables clés à leurs valeurs entières de manière approximative, mais efficace :

- *Fixation de colonnes :* Les colonnes correspondant à des variables ayant des valeurs supérieures à un seuil (typiquement 0.7) dans la solution du PMR sont fixées à 1.
- *Fixation d'inter-tâches :* Les connexions entre trajets (inter-tâches) avec des valeurs dépassant un seuil sont fixées à 1, facilitant la planification entre trajets consécutifs.

Ces branchements heuristiques suffisent généralement pour obtenir rapidement des solutions faisables et de bonne qualité. Nous utilisons donc une heuristique de plongée [28] qui consiste à explorer en profondeur l'arbre de recherche en fixant progressivement des colonnes sélec-

tionnées de manière heuristique à leurs valeurs entières.

Ils sont intégrés soit au niveau du PM, comme pour la fixation de colonnes, soit au niveau des SPs, par exemple en modifiant le graphe des plus courts chemins pour éliminer les arcs non pertinents.

### 4.4 Réduction de la taille du réseau par apprentissage automatique

Gerbaux et al. [2] ont développé une méthode combinant heuristique gloutonne et GNN afin de réduire la taille du réseau espace-temps dans le MDEVSP. D'abord, des instances sont résolues par GC complète pour collecter des données sur les arcs utilisés dans les solutions optimales. Ces données servent ensuite à entraîner un GNN en apprentissage supervisé, permettant de prédire, à partir des caractéristiques des arcs, leur probabilité d'appartenir à une solution optimale. Une fois le modèle entraîné, il est appliqué à de nouvelles instances pour évaluer la pertinence des arcs. Les arcs ayant une probabilité faible sont éliminés, réduisant ainsi la taille du réseau. Enfin, le problème réduit, ne conservant qu'un sous-ensemble d'arcs pertinents, est résolu par GC, ce qui accélère le processus tout en préservant une qualité de solution élevée.

### 4.5 Limitation du réseau $\mathcal{G}_d$

Bien que la méthode de Gerbaux et al. [2] soit efficace, le réseau espace-temps utilisé (figure 4.1) limite la capacité d'imposer efficacement certaines contraintes, comme l'interdiction qu'un trajet spécifique suive un autre en raison des réglementations pour les conducteurs. Par exemple, il est impossible d'empêcher le trajet $t_1$ d'être suivi par le trajet $t_3$ dans le réseau $\mathcal{G}_d$ sans ajouter une ressource supplémentaire à la formulation du SP, car il existe un chemin de $t_1$ à $t_3$ via les nœuds d'attente et de recharge, par exemple : $(t_1, w^h_{p_1}, r^h_{p_2}, r^h_{p_3}, w^h_{p_3}, t_3)$.

Notez que supprimer l'arc $(w^h_{p_3}, t_3)$ du réseau n'est pas non plus souhaitable, car cela empêcherait tout chemin vers $t_3$ en provenance d'un nœud de recharge. Cette flexibilité pour éviter certaines séquences de trajets est cruciale pour respecter les exigences opérationnelles dans des applications réelles.

# CHAPITRE 5 NOUVELLE MÉTHODOLOGIE DE RÉSOLUTION

Dans ce chapitre, nous proposons deux contributions majeures pour dépasser les limites évoquées à la section 4.5. La première porte sur une *reconceptualisation du réseau du sous-problème*, et la seconde vise à *réduire la taille* de ce nouveau réseau afin d'en assurer une résolution efficace.

## 5.1 Reconceptualisation du réseau du SP (Première contribution)

Nous proposons un nouveau réseau acyclique $\mathcal{G}_d^{\mathsf{new}} = (\mathcal{V}_d^{\mathsf{new}}, \mathcal{A}_d^{\mathsf{new}})$ pour modéliser le SP associé au dépôt $d \in D$ (voir figure 5.1). L'ensemble de nœuds $\mathcal{V}_d^{\mathsf{new}}$ inclut toujours les nœuds de trajets ($t$), les nœuds de dépôt ($o^d$ et $k^d$) et les nœuds d'attente au dépôt ($s_p^d$), qui restent inchangés. Cependant, les nœuds de recharge et les nœuds d'attente à une station sont remplacés par deux nouveaux types de nœuds : un nœud *entrée-en-station* $a_p^{t,h}$ et un nœud *sortie-de-station* $b_p^{t,h}$. Ces nœuds représentent respectivement l'entrée et la sortie d'une station de recharge $h$ à la période $p$, immédiatement après l'achèvement du trajet $t$. Ces nœuds sont spécifiques à chaque trajet.

En reprenant notre exemple précédent, nous pouvons maintenant interdire le trajet $t_3$ de suivre $t_1$ en supprimant l'arc $(b_p^{t_1,h}, t_3)$, sans interdire la séquence $t_2$ à $t_3$, car $t_2$ dispose désormais de ses propres nœuds *entrée-en-station* et *sortie-de-station*. Notez qu'une modélisation similaire pourrait s'appliquer pour un retour au dépôt, mais les compagnies de transport autorisent souvent toutes les séquences de trajets lorsque les trajets sont séparés par un retour au dépôt, car elles offrent la possibilité de changer de conducteur.

La deuxième modification que nous apportons au réseau concerne la manière dont les arcs de recharge (en marron) sont définis. Tout d'abord, nous limitons ces arcs aux transitions $(a_p^{t,h}, b_{p'}^{t,h})$ telles que $p' > p$. Soit $\Delta_{p,p'} = p' - p$ le nombre de périodes entre un nœud $a_p^{t,h}$ et un nœud $b_{p'}^{t,h}$. Comme une recharge sur des périodes consécutives doit avoir lieu lorsqu'un bus visite une station de recharge, il existe $\frac{\Delta_{p,p'}(\Delta_{p,p'}+1)}{2}$ options de recharge possibles, une pour chaque paire de périodes de début de recharge et de durée.

Chaque option est représentée par un arc spécifique entre une paire de nœuds $(a_p^{t,h}, b_{p'}^{t,h})$. Comme montré dans la figure 5.1, plusieurs arcs, représentants des options de recharges différentes, peuvent relier une même paire de nœuds $(a_p^{t,h}, b_{p'}^{t,h})$. Tous les autres arcs du réseau initial $\mathcal{G}_d$ restent inchangés.

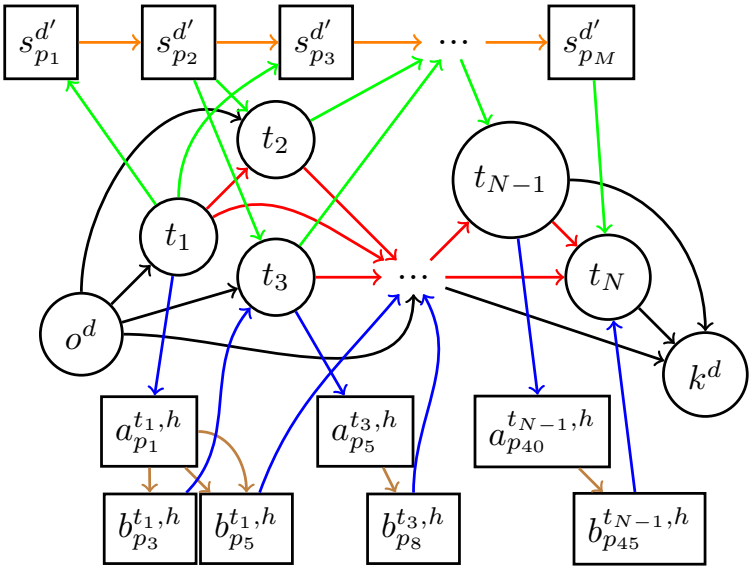



FIGURE 5.1 Nouveau réseau $\mathcal{G}_d^{\mathsf{new}}$ proposé.

## 5.2 Réduire la taille du réseau (Deuxième contribution)

Malheureusement, inclure toutes les options de recharge possibles pour chaque trajet aboutit à un réseau extrêmement volumineux, difficilement exploitable. Par exemple, dans nos ensembles d'instances les plus simples (en moyenne 687 trajets et 1 station de recharge), le nouveau réseau (figure 5.1) contient environ 59 millions d'arcs, alors que le réseau initial (figure 4.1) n'en comprend que 20 000. Cette hausse drastique de la taille nécessite une importante réduction de l'échelle du réseau. Nous proposons trois stratégies pour y remédier :

1. ne sélectionner qu'un sous-ensemble d'options de recharge pertinentes,
2. employer une heuristique (constructive et randomisée) pour identifier les arcs les plus prometteurs à conserver,
3. filtrer les arcs *trajet-à-trajet* en fonction de leur temps de déplacement à vide.

### 5.2.1 Stratégie 1 : Sélection d'options de recharge pertinentes

Intuitivement, de nombreuses options de recharge (ou d'attente) peuvent être éliminées sans risque. Par exemple, il est peu probable qu'un bus recharge un grand nombre de périodes (par ex., plus de 2 heures) en milieu de journée. Nous pouvons donc élaguer les arcs entre deux nœuds $a_p^{t,h}$ et $b_{p'}^{t,h}$ qui correspondent à de tels scénarios peu réalistes. Pour cela, nous introduisons cinq règles d'élagage :

1. Après la fin d'un trajet, les bus ne peuvent se recharger qu'entre $\epsilon_1$ et $\epsilon_2$ périodes,

et ne peuvent attendre qu'entre $\epsilon_3$ et $\epsilon_4$ périodes. Les valeurs $\epsilon_1$, $\epsilon_2$, $\epsilon_3$ et $\epsilon_4$ sont des hyperparamètres définis soit par les exploitants, sur la base de leur expertise, soit par des statistiques agrégées sur l'utilisation du réseau.

2. L'attente n'est autorisée qu'après la recharge et non avant. L'idée est qu'un bus qui doit se recharger devrait prioriser cette action, puis attendre avant d'effectuer le trajet suivant.
3. Nous interdisons l'accès à une station de recharge $h$ depuis un trajet $t$ si la destination de $t$ est trop éloignée de cette station. Formellement, soit $L$ l'ensemble de tous les terminaux de trajet, $\alpha_1 \in [0, 1]$ un hyperparamètre qui contrôle la tolérance à accepter un arc, et $\rho_{i,j}$ le temps de trajet entre les localisations $i$ et $j$. Un arc *connexion-avec-recharge* $(t, a_p^{t,h})$ est conservé uniquement si

$$\rho_{l_t^E,h} \leq \min_{\ell \in L} \rho_{\ell,h} + \alpha_1 \Big( \max_{\ell \in L} \rho_{\ell,h} - \min_{\ell \in L} \rho_{\ell,h} \Big). \tag{5.1}$$

4. De manière similaire, nous interdisons d'accéder à un trajet $t$ depuis une station $h$ si son point de départ est trop éloigné de cette station. Par conséquent, un arc $(b_p^{t,h}, t)$ est conservé seulement si

$$\rho_{h,l_t^D} \leq \min_{\ell \in L} \rho_{h,\ell} + \alpha_1 \Big( \max_{\ell \in L} \rho_{h,\ell} - \min_{\ell \in L} \rho_{h,\ell} \Big). \tag{5.2}$$

5. Soit $\tau_t$ le nombre d'options de recharge restantes après le trajet $t$. Nous ne conservons que $\theta \cdot \tau_t$ de ces options, où $\theta \in [0, 1]$. La sélection s'effectue via une distribution de probabilité pondérée. Formellement, soient $O$ l'ensemble des options de recharges et $W_o$ le nombre de périodes d'attente de l'option $o \in O$. La pondération de chaque option est $\kappa_o = \frac{1}{W_o+1}, \forall o \in O$. Ce schéma de pondération favorise les temps d'attente plus courts. Les arcs menant aux autres options de recharge sont alors supprimés du réseau.

### 5.2.2 Stratégie 2 : Sélection de séquences de trajets prometteuses

Bien que le mécanisme précédent permette de réduire considérablement le nombre d'arcs, il se concentre uniquement sur ceux impliquant une station de recharge. Nous proposons également une méthode pour élaguer d'autres arcs du réseau, en particulier ceux reliant les trajets (en rouge dans la figure 5.1), qui sont très nombreux. Pour ce faire, nous nous inspirons de l'approche GRASP proposée par Jacquet et al. [1], mais nous l'adaptons pour travailler directement sur notre nouveau graphe.

Concrètement, ils proposent une heuristique constructive et randomisée exécutée $K$ fois pour

générer plusieurs horaires. À chaque itération, l'heuristique collecte l'union des arcs identifiés comme prometteurs et les combine avec ceux issus des itérations précédentes. Cette sélection d'arcs est effectuée *a priori*, et lors de la GC, les SPs sont résolus sur les réseaux réduits ainsi constitués, à l'aide d'un algorithme d'étiquetage.

L'algorithme 1 récapitule ce processus. Nous travaillons sur le réseau complet $\left(\mathcal{V}^{\mathsf{new}}, \mathcal{A}^{\mathsf{new}}\right) = \left(\bigcup_{d\in D} \mathcal{V}_d^{\mathsf{new}}, \bigcup_{d\in D} \mathcal{A}_d^{\mathsf{new}}\right)$ (ligne 6). L'ensemble $\mathcal{A}^{\mathsf{TT}}$ stocke les arcs *trajet-à-trajet* utilisés dans les solutions générées durant ce processus. La première boucle principale (lignes 8–33) exécute $K$ itérations de l'heuristique constructive randomisée. À chaque itération, nous cherchons à couvrir tous les trajets de l'ensemble dynamique $T'$ (ligne 9). Tant qu'il reste des trajets non couverts (lignes 10–33), nous initions un nouvel horaire pour un bus partant d'un dépôt $d$ (lignes 11–14). Ce dépôt est choisi aléatoirement parmi ceux non encore utilisés. Une fois tous les dépôts utilisés, la sélection repart aléatoirement de l'ensemble de tous les dépôts. Le premier trajet à couvrir est choisi au hasard parmi les $\mu$ trajets non couverts ayant les plus petits temps de début. Ce trajet est marqué comme couvert (ligne 14).

Les étapes suivantes (lignes 15–33) consistent essentiellement à ajouter des activités à l'horaire jusqu'à atteindre la destination finale $k^d$. Quatre cas se présentent :

1. *Trajets faisables (lignes 17–21).* Depuis le nœud courant $n$, on privilégie les trajets non couverts, sinon on peut sélectionner un trajet déjà couvert. La faisabilité d'un trajet $t$ s'évalue via (i) l'existence d'un arc $(n, t) \in \mathcal{A}^{\mathsf{new}}$, et (ii) un SoC suffisant pour réaliser le trajet $t$ et atteindre le dépôt $d'$ le plus proche à l'issue de ce trajet. Si plusieurs trajets sont possibles, le prochain est choisi aléatoirement parmi un sous-ensemble des meilleurs selon le coût $c_{n,t}$. La sélection aléatoire obéit à :

$$\Phi(n, \mathcal{F}, \alpha_2) \sim \mathsf{Uniform}\left\{ t \in \mathcal{F} \;\middle|\; c_{n,t} \leq c^{\mathsf{min}} + \alpha_2 \left(c^{\mathsf{max}} - c^{\mathsf{min}}\right) \right\}, \tag{5.3}$$

   où $c^{\mathsf{min}}$ est le meilleur coût, $c^{\mathsf{max}}$ le pire coût et $\alpha_2 \in [0, 1]$ un hyperparamètre de tolérance.
2. *SoC faible (lignes 22–29).* Quand le SoC est inférieur à un seuil $\lambda$, le bus est dirigé vers la station de recharge la plus proche (nœud $a_p^{n,h}$). L'option de recharge $((a_p^{n,h}, b_{p'}^{n,h}))$ est choisie aléatoirement après le filtrage effectué en *Stratégie 1*, selon le même principe que l'équation (5.3) avec $\alpha_2 = 1$. Après la recharge, le prochain trajet est sélectionné de façon similaire.
3. *Aucun trajet faisable (lignes 30–31).* Si aucun trajet n'est disponible, le bus patiente au dépôt le plus proche.
4. *Fin de journée (lignes 32–33).* Le bus retourne à son dépôt d'origine $k^d$. Si des trajets

**Algorithm 1:** Heuristique constructive aléatoire pour la sélection d'arcs.

▷ **Pré :** $T$ est l'ensemble de tous les trajets à couvrir, triés par l'heure de départ la plus tôt.
▷ **Pré :** $\alpha_2$ est le pourcentage de trajets à considérer lors de l'extension d'un chemin partiel.
▷ **Pré :** $K$ est le nombre d'itérations de l'heuristique constructive aléatoire.
▷ **Post :** $\mathcal{A}^{\mathsf{grasp}}$, un sous-ensemble d'arcs contenant des horaires couvrant tous les trajets.

$(\mathcal{V}^{\mathsf{new}}, \mathcal{A}^{\mathsf{new}}) := (\bigcup_{d\in D} \mathcal{V}_d^{\mathsf{new}}, \bigcup_{d\in D} \mathcal{A}_d^{\mathsf{new}})$
$\mathcal{A}^{\mathsf{TT}} := \emptyset$ ▷ *Ensemble qui contiendra tous les arcs entre trajets utilisés*
**for** $i$ **de** 1 à $K$ **do**
  $T' := T$ ▷ *Ensemble dynamique contenant tous les trajets à couvrir*
  **while** $T' \neq \emptyset$ **do**
    $d := \mathsf{getNextDepot}()$
    $n := o^d$ ▷ *La journée commence pour un bus au dépôt d*
    $n := \mathsf{getRandomTrip}(n, T', \mu)$ ▷ *Sélection parmi les $\mu$ premiers trajets*
    $T' := T' \setminus \{n\}$
    **while** $n \neq k^d$ **do**
      $\mathcal{F} := \{t \in T \mid (n,t) \in \mathcal{A}^{\mathsf{new}} \ \wedge \ \mathsf{isFeasible}(n,t)\}$
      **if** $\mathcal{F} \neq \emptyset$ **then**
        $t := \Phi(n, \mathcal{F}, \alpha_2)$ ▷ *Sélection aléatoire, eq.* (5.3)
        $\mathcal{A}^{\mathsf{TT}} := \mathcal{A}^{\mathsf{TT}} \cup \{(n,t)\}$ ▷ *L'arc entre trajets est ajouté à l'ensemble S*
        $n := t$ ▷ *Mettre à jour le nœud courant*
        $T' := T' \setminus \{n\}$
      **else if** $\mathsf{getSoC}(n) \leq \lambda$ **then**
        $h := \mathsf{getClosestRechargingStation}(n)$
        $a := \mathsf{getStationNode}(n, h)$ ▷ *Nœud $a_p^{n,h}$ dans le SP*
        $\mathcal{B} := \{b \in \mathcal{V}^{\mathsf{new}} \mid (a,b) \in \mathcal{A}^{\mathsf{new}}\}$
        $b := \Phi(a, \mathcal{B}, 1)$ ▷ *Nœud $b_{p'}^{n,h}$ dans le SP*
        $\mathcal{H} := \{t \in T \mid (b,t) \in \mathcal{A}^{\mathsf{new}} \ \wedge \ \mathsf{isFeasible}(b,t)\}$
        $n := \Phi(b, \mathcal{H}, \alpha_2)$ ▷ *Sélection aléatoire, eq.* (5.3)
        $T' := T' \setminus \{n\}$
      **else if** $\exists\ (n,s) \in \mathcal{A}^{\mathsf{new}} \mid \mathsf{isDepotWaitingNode}(s)$ **then**
        $n := \mathsf{getClosestDepot}(n)$
      **else**
        $n := k^d$
$\mathcal{A}^{\mathsf{grasp}} := \mathcal{A}^{\mathsf{TT}} \cup \left(\mathcal{A}^{\mathsf{new}} \setminus (T \times T)\right)$
**retourner** $\mathcal{A}^{\mathsf{grasp}}$

restent à couvrir, l'itération reprend au sein de la boucle externe (lignes 15–33).

Une fois les $K$ itérations terminées, l'ensemble final d'arcs $\mathcal{A}^{\mathsf{grasp}}$ est obtenu en remplaçant tous les arcs *trajet-à-trajet* par ceux de l'ensemble $\mathcal{A}^{\mathsf{TT}}$ (ligne 34). Le temps d'exécution de cet algorithme est négligeable comparé à celui de la GC exécutée par la suite. Nous gardons les valeurs empiriques proposées par Jacquet et al. [1], à savoir $\mu = 10$, $\lambda = 50\,\%$ et $\alpha_2 = 0.1$.

### 5.2.3 Stratégie 3 : Filtrage des arcs *trajet-à-trajet*

En parallèle de la sélection de l'algorithme 1, nous proposons une autre méthode de sélection des arcs *trajet-à-trajet* (en rouge dans la figure 5.1). Soit $\mathcal{A}^{\mathsf{new}} = \bigcup_{d \in D} \mathcal{A}_d^{\mathsf{new}}$ l'ensemble de tous les arcs, et $(t, t') \in \mathcal{A}^{\mathsf{new}}$ un arc *trajet-à-trajet*. La règle de filtrage conserve uniquement les meilleurs arcs en fonction de $\rho_{l_t^E, l_{t'}^D}$, c'est-à-dire le temps de trajet à vide requis pour démarrer le trajet $t'$ après avoir terminé le trajet $t$. On suit la même idée que l'équation (5.3), avec $\alpha_3 \in [0, 1]$ pour contrôler la tolérance. Le résultat est :

$$\mathcal{A}^{\mathsf{filtered}} = \mathcal{A}^{\mathsf{new}} \setminus \bigcup_{t \in T} \left\{ (t, t') \in \mathcal{A}_t^{\mathsf{TT}} \;\middle|\; \rho_{l_t^E, l_{t'}^D} > \rho_t^{\mathsf{min}} + \alpha_3 \left( \rho_t^{\mathsf{max}} - \rho_t^{\mathsf{min}} \right) \right\}, \tag{5.4}$$

où $\mathcal{A}_t^{\mathsf{TT}} = \left\{ (t, t') \in \mathcal{A}^{\mathsf{new}} \mid t' \in T \right\}$ est l'ensemble des arcs *trajet-à-trajet* issus du trajet $t$, et $\rho_t^{\mathsf{min}}$ (resp. $\rho_t^{\mathsf{max}}$) représente le temps de déplacement à vide minimal (resp. maximal) pour les arcs partant de $t$.

Lorsque $\alpha_3 = 0$, seul l'arc avec le coût le plus bas est sélectionné, correspondant aux trajets suivants les plus proches. À mesure que $\alpha_3$ augmente, davantage d'arcs sont inclus, permettant un ensemble plus large de transitions réalisables, ce qui équilibre l'efficacité en termes de coût et la flexibilité de la solution. Une fois cet ensemble d'arcs déterminé, il peut être combiné avec l'ensemble d'arcs fourni par la sélection basée sur GRASP de deux manières. Premièrement, une intersection des deux ensembles peut être effectuée, ce qui réduit la sélection aux arcs qui sont à la fois efficaces en termes de coût et jugés prometteurs par GRASP. Cette approche renforce le processus de sélection des arcs, conduisant à un graphe plus petit et très ciblé, ce qui peut réduire considérablement le temps de calcul. Alternativement, une union des deux ensembles peut être utilisée pour s'assurer que des arcs potentiellement importants ne sont pas exclus du processus d'optimisation.

# CHAPITRE 6 EXPÉRIENCES ET RÉSULTATS NUMÉRIQUES

L'objectif des expériences menées dans ce mémoire est de comparer la performance de notre reformulation de graphe avec deux approches à l'état de l'art : la méthode basée sur la métaheuristique GRASP [1] et celle utilisant l'apprentissage automatique [2]. Ces comparaisons permettent d'évaluer l'efficacité de notre méthode en termes de qualité des solutions obtenues, de temps de calcul, et de robustesse à travers diverses configurations du problème. Afin d'assurer une comparaison fiable, nous avons réutilisé les mêmes instances que celles employées dans les travaux de Gerbaux et al. [2], dérivées des horaires de transport en commun de Montréal [29]. Ces instances sont générées à partir de données réelles, enrichies par l'ajout de variations aléatoires dans les horaires des trajets, afin de produire des scénarios diversifiés, cohérents et représentatifs des conditions opérationnelles réelles. .

## 6.1 Description des instances

Les instances utilisées dans ces expériences sont organisées en sept sous-ensembles, étiquetés A à G. Ces sous-ensembles couvrent une large gamme de tailles et de configurations, représentant ainsi divers scénarios rencontrés dans la planification des réseaux de transport public. Les caractéristiques principales des instances, résumées dans le tableau 6.1, sont décrites comme suit :

**Sous-ensembles A à D** Les sous-ensembles A, B, C et D sont construits à partir d'un réseau initial comprenant quatre lignes de bus (figure 6.1a). Une *ligne de bus* correspond à un itinéraire fixe reliant deux stations, suivant un ensemble prédéfini d'arrêts intermédiaires. Un *trajet*, quant à lui, désigne l'exécution concrète de cette ligne à un horaire donné, c'est-à-dire un départ précis associé à une ligne déterminée. Ce réseau est caractérisé par des dépôts et des stations de recharge répartis de manière décentralisée, ce qui reflète une infrastructure où la gestion des trajets entre dépôts et stations de recharge peut être plus complexe. Les tailles des sous-ensembles varient en fonction du nombre moyen de trajets à planifier, allant de 568 à 893 trajets par instance. Ces configurations sont représentatives des réseaux de transport public de taille moyenne, tels que ceux trouvés dans des zones suburbaines ou des villes de taille intermédiaire.

**Sous-ensembles E à G** Les sous-ensembles E, F et G, en revanche, sont dérivés d'un réseau plus complexe, comportant huit lignes de bus (figure 6.1b). Ce réseau est structuré autour

d'installations centralisées, notamment des dépôts et des stations de recharge regroupés. Cette configuration simplifie certaines opérations logistiques, mais augmente la complexité des décisions liées à l'ordonnancement des trajets et à l'utilisation optimale des ressources disponibles.

Les sous-ensembles de ce groupe présentent une taille moyenne de 669 à 3115 trajets par instance. Ces dimensions reflètent les défis typiques rencontrés dans les grandes agglomérations urbaines, où la densité des lignes et le volume élevé de trajets imposent des contraintes opérationnelles significatives.

## 6.2 Méthodologie de comparaison

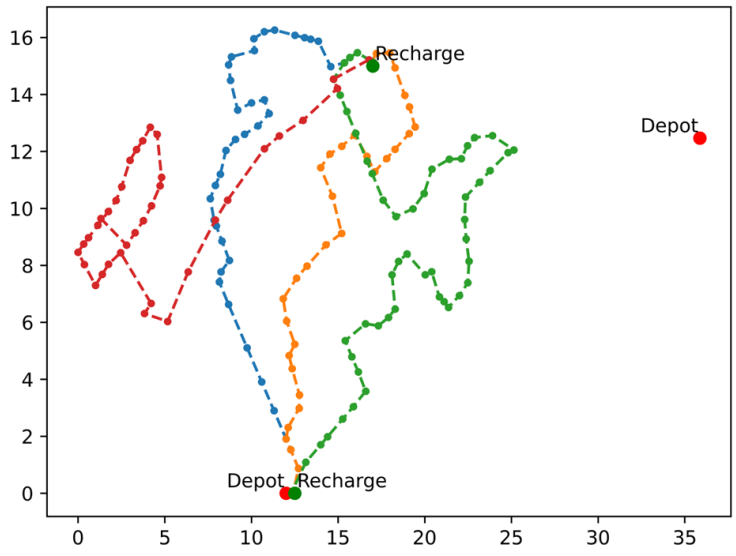
Les expériences reposent sur une configuration de paramètres qui reflète des conditions réalistes pour la planification des autobus électriques. Ces paramètres incluent la vitesse moyenne des véhicules, les coûts fixes d'utilisation et de recharge, ainsi que les capacités minimale et maximale des batteries. Les valeurs utilisées sont résumées dans le tableau 6.2. Elles ont été choisies pour refléter une situation réaliste basée sur des données représentatives du domaine.

Notre méthode est calibrée sur 70 % des instances et validée sur 15 % des instances. Les résultats finaux présentés ci-après sont évalués sur les 15 % restants des instances, conformément au protocole d'évaluation de Gerbaux et al. [2]. Les expériences ont été réalisées sur une machine équipée d'un processeur Intel Core i7-12700 de 12e génération (20 cœurs). L'heuristique de GC a été implémentée à l'aide de la bibliothèque Gencol [30].

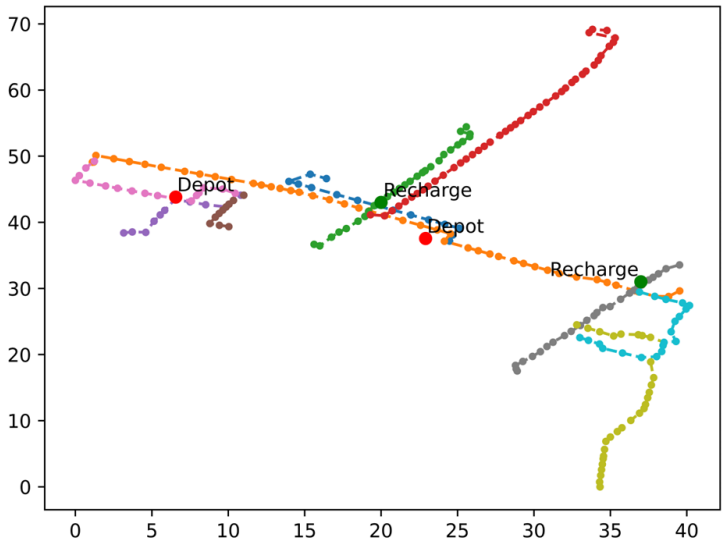
## 6.3 Heuristiques de génération de colonnes comparées

Dans nos expériences, nous comparons les heuristiques suivantes.

1. cgBasic *(baseline)*. L'heuristique de GC de base, qui repose sur les réseaux $\mathcal{G}_d$ (figure 4.1) sans aucune sélection d'arcs pour réduire leur taille. Cette méthode fournit généralement les meilleurs résultats en termes de qualité de solution, mais est prohibitive en termes de temps de calcul.
2. cgHBD *(Gerbaux et al. [2])*. L'heuristique de GC basée sur l'apprentissage automatique de Gerbaux et al. [2].
3. cgStrategy1 *(ablation)*. L'heuristique de GC de base améliorée avec la sélection d'options de recharge pertinentes (*Stratégie 1*) et les nouveaux réseaux $\mathcal{G}_d^{\mathsf{new}}$ (figure 5.1).
4. cgStrategy2 *(ablation)*. L'heuristique de GC améliorée uniquement avec la sélection d'arcs fournie par la *Stratégie 2* avec $K = 5$ itérations. Les réseaux $\mathcal{G}_d$ sont utilisés.




(a) Réseau 1




(b) Réseau 2

FIGURE 6.1 Réseaux de bus des différentes instances

TABLEAU 6.1 Caractéristiques principales des instances.

| Sous-ensemble | Réseau | $\|D\|$ | $\|H\|$ | No. Trajets Moy (Min-Max) | $v_d$ | $u_h$ | No. Instances (calibration/validation/test) |
|---|---|---|---|---|---|---|---|
| A | 1 | 1 | 1 | 687 (568-893) | 60 | 4 | 500 (350/75/75) |
| B | 1 | 1 | 2 | 687 (568-893) | 60 | 2 | 500 (350/75/75) |
| C | 1 | 2 | 2 | 687 (568-893) | 30 | 2 | 500 (350/75/75) |
| D | 1 | 1 | 1 | 687 (568-893) | 60 | 4 | 500 (350/75/75) |
| E | 2 | 2 | 2 | 787 (669-880) | 50 | 2 | 500 (350/75/75) |
| F | 2 | 2 | 2 | 1336 (1152-1474) | 70 | 3 | 200 (140/30/30) |
| G | 2 | 2 | 2 | 2600 (2374-3115) | 200 | 5 | 50 (35/7/8) |

TABLEAU 6.2 Valeur de certains hyperparamètres du MDEVSP

| **Nom du paramètre** | **Valeur** |
|---|---|
| Temps maximum entre deux trajets consécutifs ($\beta$) | 45 min |
| Temps minimum lors d'un retour au dépôt ($\gamma$) | 30 min |
| Période de discrétisation du temps ($\delta$) | 15 min |
| Capacité minimale de la batterie ($\underline{\sigma}$) | 10 kWh |
| Capacité maximale de la batterie ($\overline{\sigma}$) | 100 kWh |
| Coût fixe d'utilisation d'un véhicule ($c^F$) | 1000 |
| Coût d'attente par minute ($c^W$) | 2 |
| Coût d'un voyage à vide par minute ($c^T$) | 4 |
| Coût fixe de retour au dépôt ($c^R$) | 30 |
| Coût fixe de trajet vers la recharge ($c_F^H$) | 30 |
| Coût fixe d'attente à la recharge ($c_V^H$) | 30 |

5. $\mathsf{cgFull}(K,\ \alpha_3,\ \omega)$ *(notre approche complète)*. La matheuristique complète combinant toutes nos contributions (*Stratégies 1, 2, et 3*) utilisant les nouveaux réseaux $\mathcal{G}_d^{\mathsf{new}}$. Il y a trois hyperparamètres qui varient selon nos expériences : le nombre d'itérations $K$ dans l'algorithme 1, la tolérance $\alpha_3$ dans l'équation (5.4), et la décision de prendre soit l'union ($\mathcal{A}^{\mathsf{grasp}} \cup \mathsf{A}^{\mathsf{filtered}}$) soit l'intersection ($\mathcal{A}^{\mathsf{grasp}} \cap \mathsf{A}^{\mathsf{filtered}}$) avec les sous-ensembles d'arcs obtenus par les stratégies 2 et 3 ($\omega \in \{\cup, \cap\}$). Intuitivement, ces options offrent différents compromis entre le temps de calcul et la qualité de la solution.

### 6.3.1 Calibration : choix des hyperparamètres de la stratégie 1

Pour optimiser la performance de la stratégie 1, nous avons calibré les six hyperparamètres suivants : $\epsilon_1$, $\epsilon_2$, $\epsilon_3$, $\epsilon_4$, $\alpha_1$, et $\theta$. La calibration a été effectuée en utilisant les mêmes instances que celles introduites par Gerbaux et al. [2], lesquelles avaient été utilisées pour leur méthode basée sur l'apprentissage automatique. Nous avons conservé leur séparation des instances

en trois groupes distincts : calibration, validation et test. Bien que les instances de validation n'aient pas été utilisées dans notre étude, nous avons conservé cette répartition afin de garantir une meilleure comparabilité des résultats avec les travaux précédents.

Dans un premier temps, nous avons analysé en détail les instances de calibration, en extrayant des statistiques relatives aux opportunités de recharge, telles que la durée des recharges, la distance par rapport aux trajets précédents et suivants, ainsi que d'autres métriques pertinentes. Ensuite, nous avons testé nos solutions sur les instances de test, conformément à cette partition prédéfinie.

En conservant cette même division des ensembles, nous garantissons que notre approche est totalement comparable à celle de Gerbaux et al. [2], permettant ainsi une évaluation directe et équitable des performances entre les différentes méthodes.

- *Paramètres $\epsilon_1$ et $\epsilon_2$ :* Les valeurs de $\epsilon_1$ et $\epsilon_2$ ont été respectivement fixées à 2 et 4. Ce choix repose sur l'analyse de la durée effective des recharges présentes dans les solutions de `cgBasic` pour les instances de calibration. La figure 6.2 illustre la distribution des durées de recharge, confirmant que ces valeurs capturent adéquatement les comportements observés.
- *Paramètre $\epsilon_3$ :* Ce paramètre a été logiquement fixé à 0, car permettre l'absence d'attente après la recharge reflète le comportement le plus fréquemment observé et s'avère essentiel pour respecter les dynamiques typiques des solutions.
- *Paramètre $\epsilon_4$ :* Initialement fixé à 0, ce paramètre entraînait de mauvaises performances sur les instances de grandes tailles (E à G), en raison de la complexité accrue de ces cas. En conséquence, nous avons ajusté $\epsilon_4$ pour autoriser une attente allant jusqu'à 2 périodes, ce qui a considérablement amélioré les performances sur ces instances complexes.
- *Paramètre $\alpha_1$ :* $\alpha_1$ a été fixé à 0.3, comme illustré dans la figure 6.3. Cette valeur permet d'englober plus de 95% des recharges présentes dans les solutions obtenues, garantissant ainsi une couverture efficace tout en évitant une inclusion excessive d'options non nécessaires.
- *Paramètre $\theta$ :* Ce paramètre a été fixé de manière empirique à 0.3, Un compromis équilibré visant à sélectionner un nombre de recharges évitant tout excès ou insuffisance.

En résumé, les six hyperparamètres ont été ajustés manuellement pour obtenir les configurations suivantes :

- Instances A à D : $(\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4, \alpha_1, \theta) = (2, 4, 0, 0, 0.3, 0.3)$.
- Instances E à G : $(\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4, \alpha_1, \theta) = (2, 4, 0, 2, 0.3, 0.3)$.

Cette calibration garantit une adaptation fine des hyperparamètres aux spécificités des différentes catégories d'instances.

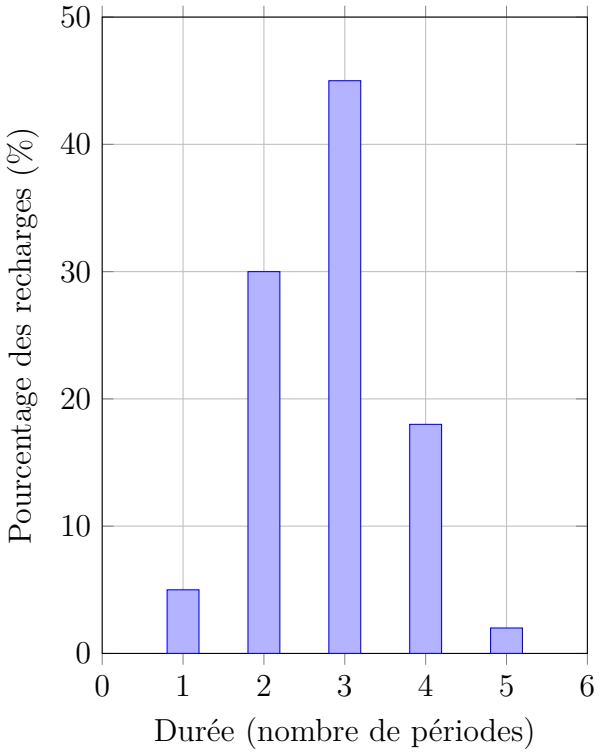
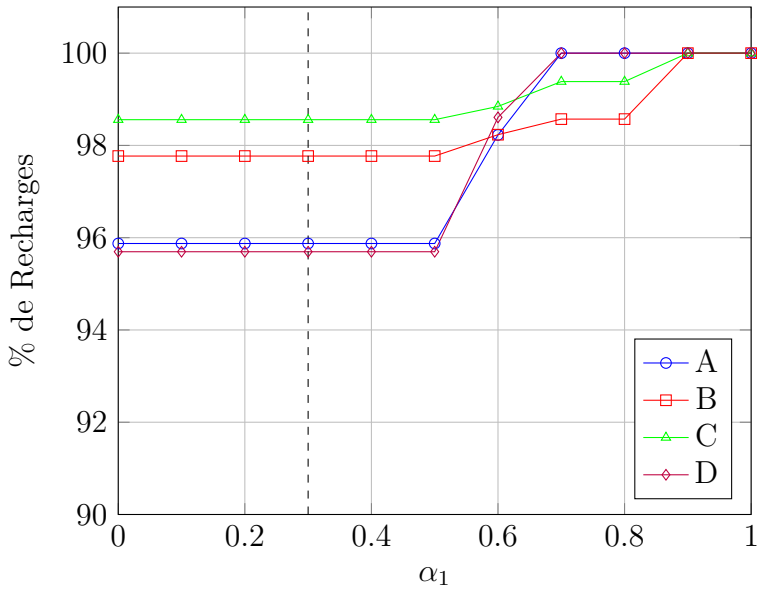



FIGURE 6.2 Distribution des durées de recharge des solutions de `cgBasic` selon leur durée.

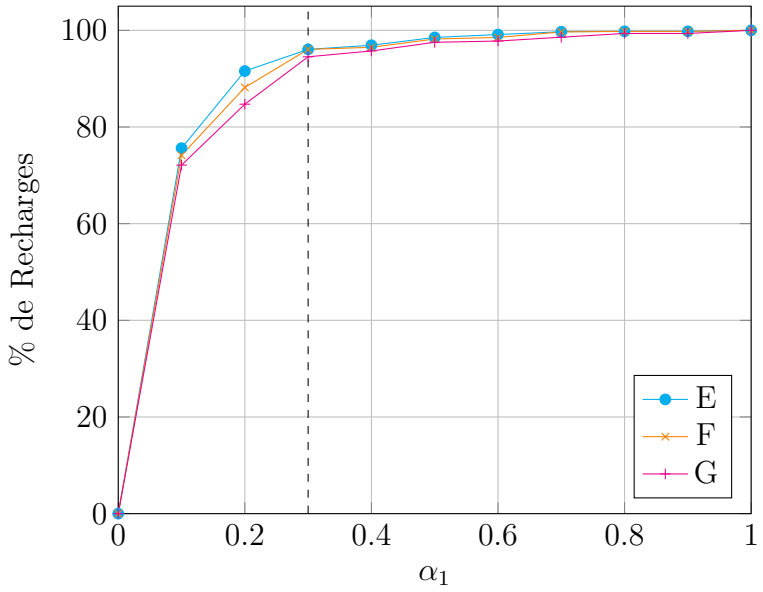
### 6.3.2 Résultats : compromis entre qualité et temps de calcul

L'objectif principal de notre matheuristique (cgFull) est d'atteindre un bon équilibre entre la réduction du temps de calcul et la préservation de la qualité des solutions. Cet équilibre est mesuré par la comparaison avec l'heuristique de GC de base (cgBasic), qui sert de référence. Pour évaluer cet objectif, nous avons exploré les effets de différents hyperparamètres sur les performances de cgFull. Les hyperparamètres considérés sont :

- Le paramètre $K \in \{5, 10\}$, qui contrôle le nombre d'itérations de l'algorithme GRASP effectuées.
- Le paramètre $\alpha_3 \in \{0.1, 0.2, 0.3, 0.5, 1\}$, qui influence la tolérance que nous avons lors de la stratégie 3.
- L'opérateur $\omega \in \{\cup, \cap\}$, qui détermine la manière dont les arcs à élaguer sont regroupés (union ou intersection des ensembles générés).




(a) Sous-figure pour les instances A, B, C et D




(b) Sous-figure pour les instances E, F et G

FIGURE 6.3 Pourcentage de recharges des solutions `cgBasic` où le trajet précédent la recharge satisfait les équations (5.1) et (5.2), en fonction de $\alpha_1$.

**Front de Pareto et interprétation des résultats** Les compromis obtenus entre la qualité des solutions et le temps de calcul sont visualisés à l'aide d'un front de Pareto (figure 6.4). Cette figure illustre, pour chaque configuration de cgFull, la réduction moyenne du temps de calcul en pourcentage (axe vertical) et la différence de coût moyen par rapport à cgBasic en pourcentage (axe horizontal). Chaque point individuel (carré ou cercle) représente les performances moyennes d'une configuration sur toutes les instances de test du sous-ensemble A. Le front de Pareto, représenté par une courbe orange, délimite les configurations offrant le meilleur compromis. Plus précisément :

— Les configurations situées sur le front de Pareto sont les plus performantes, car aucune autre configuration ne les domine simultanément sur les deux axes (temps et coût).

— Les solutions dominées, en revanche, sont représentées par des cercles isolés en dessous ou à droite du front. Elles indiquent des configurations moins efficaces, soit en raison de temps de calcul excessifs, soit en raison d'une dégradation trop importante de la qualité des solutions.

Certaines configurations spécifiques, telles que celles avec $\alpha_3 = 1$ et $\omega = \cup$, ne sont pas incluses dans cette analyse, car elles correspondent à la stratégie alternative cgStrategy1. De plus, les configurations ayant une différence de coût supérieure à 3% ont été exclues, car elles introduisent une dégradation excessive de la solution, ce qui va à l'encontre de notre objectif.

**Configurations sélectionnées** Sur la base de ces résultats, trois configurations spécifiques de cgFull ont été identifiées comme offrant des compromis pertinents entre qualité et temps de calcul. Ces configurations sont décrites ci-dessous, chacune correspondant à un scénario d'application particulier :

1. cgFull(5, 1, $\cap$) : Cette configuration se distingue par une réduction maximale du temps de calcul, atteignant environ 70%. Elle est particulièrement adaptée aux scénarios où la rapidité de décision est cruciale, par exemple dans des contextes où des ajustements fréquents des plans sont nécessaires ou lorsque le temps de calcul est une contrainte stricte.
2. cgFull(5, 0.3, $\cup$) : Cette configuration priorise la qualité des solutions, avec une différence de coût inférieure à 1.4% par rapport à cgBasic. Bien que la réduction du temps de calcul soit légèrement inférieure (environ 55%), elle est idéale pour des applications où la précision et la minimisation des coûts sont essentielles, comme la gestion budgétaire stricte ou les opérations à forte sensibilité au coût.
3. cgFull(10, 0.5, $\cup$) : Cette configuration représente un compromis équilibré entre les deux précédentes. Avec une réduction du temps de calcul d'environ 65% et une différence de coût moyenne de 1.6%, elle convient aux cas où il est nécessaire de combiner

une réduction significative du temps de calcul avec une solution de qualité raisonnable. Cette configuration est particulièrement utile dans des contextes opérationnels mixtes où les deux critères doivent être pris en compte.

**Analyse comparative** La comparaison des trois configurations met en évidence la flexibilité de cgFull pour répondre à différents besoins. En ajustant les hyperparamètres $K$, $\alpha_3$ et $\omega$, il est possible d'adapter l'algorithme à des exigences variées, qu'il s'agisse de réduire le temps de calcul ou de maintenir une qualité optimale des solutions. Ce niveau de personnalisation confère à cgFull un avantage significatif par rapport aux approches alternatives, qui ne permettent pas toujours de telles adaptations.

En conclusion, cette analyse montre que cgFull offre une méthode robuste et adaptable pour résoudre le MDEVSP, avec des performances compétitives tant en termes de temps que de qualité des solutions. Ces résultats soulignent l'importance de l'exploration des hyperparamètres pour maximiser les avantages de l'algorithme dans des contextes réels.

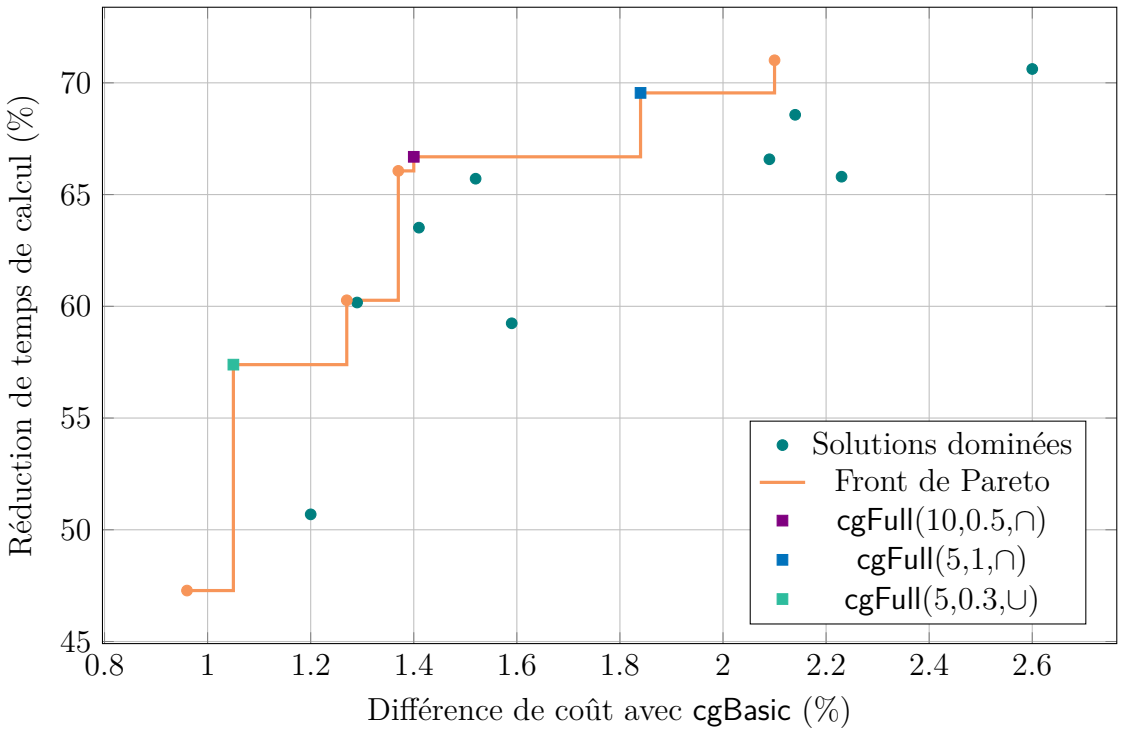



FIGURE 6.4 Front de Pareto de cgFull.

### 6.3.3 Résultats : Performance et robustesse de cgFull

Les résultats comparant cgFull avec les autres heuristiques de GC sont présentés au tableau 6.3. Ce tableau regroupe, pour chaque sous-ensemble d'instances (A à G), plusieurs indicateurs permettant d'analyser à la fois la performance en temps de calcul et la qualité de la solution obtenue.

Les colonnes ont la signification suivante :

- **Temps (sec.)** : le temps de calcul moyen (en secondes) requis pour résoudre le sous-ensemble d'instances considéré.
- **Réd Temps (%)** : le pourcentage de réduction moyen du temps de calcul par rapport à la méthode de base cgBasic. Les valeurs *min*, *moy* et *max* correspondent respectivement à la réduction minimale, moyenne et maximale observée dans le sous-ensemble. Une valeur négative indique une augmentation du temps, tandis qu'une valeur positive indique un gain de temps.
- **Diff Coût (%)** : la différence de coût moyenne (en pourcentage) par rapport à cgBasic. Les valeurs *min*, *moy* et *max* indiquent la détérioration (valeurs positives) ou l'amélioration (valeurs négatives) du coût.
- **Ratio** : le rapport entre la *réduction moyenne du temps* et la *différence moyenne de coût*. Des valeurs plus élevées de ce ratio sont préférables car elles indiquent un meilleur compromis entre gain de temps et préservation de la qualité de la solution.
- **No véhicules** : Le nombre moyen de véhicules requis dans la solution.
- **No Arcs** : Le nombre moyen d'arcs présents dans le graphe du SP.

Pour chaque sous-ensemble d'instances, nous mettons en gras la *meilleure* valeur obtenue en termes de *réduction moyenne de temps*, de *différence moyenne de coût* et de *ratio*.

En analysant en détail les résultats, nous pouvons remarquer plusieurs choses. Tout d'abord, en se concentrant sur la réduction moyenne de temps (*moy*), on remarque que cgHBD obtient souvent la plus forte baisse de temps de calcul pour les sous-ensembles d'instances de taille modérée (A, B, C et D). Par exemple, pour A et B, cgHBD réduit en moyenne le temps de plus de 70%. Cependant, cette approche montre plus de difficultés à généraliser pour de plus grandes instances (notamment F et G), où la réduction de temps moyenne diminue et la dégradation du coût tend à s'accentuer.

Les approches cgFull, quant à elles, atteignent souvent des réductions de temps significatives (souvent autour de 50–70%) tout en proposant des solutions de qualité satisfaisante. Par exemple, pour le sous-ensemble C, cgFull(5,0.3,∪) réalise une réduction moyenne de 55.7% et un ratio très élevé (79.6), ce qui illustre un excellent compromis entre temps de calcul et qualité. Sur les grandes instances (F et G), ces variantes cgFull se montrent plus régulières,

conservant des gains de temps notables.

D'autre part, d'un point de vue de la qualité de la solution, l'heuristique cgStrategy1 produit régulièrement les solutions les plus proches de cgBasic (voire meilleures), avec des écarts moyens de coût très faibles. Toutefois, ce résultat s'accompagne d'une réduction de temps souvent modeste (moins de 40% sur A–E), et quasiment nulle sur les plus grands ensembles.

La méthode cgHBD poursuit un autre extrême : elle sacrifie davantage la qualité pour obtenir des gains de temps supérieurs. La *différence de coût* moyenne associée à cgHBD est donc plus élevée, notamment pour les plus grands sous-ensembles.

Par ailleurs, cgFull est très variable selon ses différentes variantes, mais obtient un compromis intéressant, sous 4.5% de *différence de coût* pour une diminution de temps pouvant atteindre ou dépasser 50% sur les grandes instances. Ce constat est particulièrement visible pour C et G, où la version cgFull(5,0.3,∪) affiche les meilleurs ratios.

Enfin, comme l'illustrent les résultats, cgStrategy2 offre souvent un bon compromis entre temps de calcul et préservation de la qualité des solutions, mais étant basée sur les réseaux temps-espace $\mathcal{G}_d$, elle ne tire pas parti de la flexibilité accrue des réseaux proposés $\mathcal{G}_d^{\mathsf{new}}$. Cette observation est cohérente avec les travaux de Jacquet et al. [1], qui montrent que ces méthodes peuvent conserver une robustesse appréciable sur différentes gammes d'instances.

Pour résumer, si cgHBD est très performante en termes de gain de temps sur des instances de taille moyenne, elle peine à maintenir une qualité stable quand la taille du problème s'accroît fortement. De son côté, cgStrategy1 privilégie avant tout la qualité, mais ne parvient pas à réduire considérablement les temps de calcul, notamment pour les grandes instances. Enfin, les variantes de cgFull (*ex.* cgFull(5,0.3,∪)) démontrent une bonne capacité d'adaptation et un compromis efficace, tout en tirant partie de la flexibilité offerte par les réseaux $\mathcal{G}_d^{\mathsf{new}}$. Ainsi, selon les ressources de calcul disponibles, la taille des instances et les exigences de qualité imposées, chacune de ces heuristiques présente ses avantages propres, sans qu'aucune ne domine clairement l'ensemble des scénarios rencontrés.

TABLEAU 6.3 Résultats moyens pour les sous-ensemble d'instances A à G.

| Instances | Algorithme | Temps (sec.) | Réd Temps (%) | | | Diff Coût (%) | | | Ratio | No. | No. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | min | moy | max | min | moy | max | | vehicules | arcs |
| A | cgBasic | 347 | - | - | - | - | - | - | - | 41.9 | 20,695 |
| | cgHBD | - | 60.1 | **70.8** | 79.3 | -1.7 | 1.4 | 6.9 | 50.6 | 42.2 | 3,850 |
| | cgStrategy1 | 231 | 18.9 | 33.4 | 99.5 | -1.3 | **1.1** | 23.4 | 30.4 | 41.8 | 20,801 |
| | cgStrategy2 | 113 | 54.4 | 66.3 | 75.6 | 0.2 | 1.7 | 5.2 | 39.0 | 42.0 | 4,672 |
| | cgFull(5,0.3,∪) | 140 | 49.0 | 58.7 | 66.8 | -1.3 | **1.1** | 4.8 | **53.4** | 42.0 | 8,712 |
| | cgFull(10,0.5,∩) | 116 | 56.6 | 65.7 | 72.5 | -1.1 | 1.4 | 4.5 | 46.9 | 42.0 | 7,032 |
| | cgFull(5,1,∩) | 106 | 57.6 | 68.5 | 76.7 | -0.7 | 1.8 | 5.3 | 36.1 | 42.1 | 6,770 |
| B | cgBasic | 430 | - | - | - | - | - | - | - | 42.0 | 22,224 |
| | cgHBD | - | 57.9 | **72.0** | 87.1 | -1.6 | 2.1 | 6.3 | 34.3 | 42.4 | 4,137 |
| | cgStrategy1 | 266 | 25.2 | 38.2 | 51.4 | -2.1 | **1.1** | 3.3 | 34.7 | 42.0 | 22,647 |
| | cgStrategy2 | 141 | 58.8 | 66.2 | 74.1 | -1.4 | 1.4 | 4.4 | **47.3** | 42.2 | 5,139 |
| | cgFull(5,0.3,∪) | 192 | 43.1 | 54.4 | 64.9 | -2.1 | 1.2 | 3.5 | 45.3 | 42.3 | 10,722 |
| | cgFull(10,0.5,∩) | 150 | 53.2 | 63.8 | 74.0 | -2.0 | 1.7 | 5.1 | 37.5 | 42.4 | 9,124 |
| | cgFull(5,1,∩) | 135 | 54.7 | 67.2 | 75.0 | -1.3 | 1.8 | 4.3 | 37.3 | 42.4 | 8,860 |
| C | cgBasic | 455 | - | - | - | - | - | - | - | 41.7 | 25,609 |
| | cgHBD | - | 69.6 | **74.4** | 80.0 | -1.8 | 2.1 | 6.2 | 35.4 | 42.1 | 5,643 |
| | cgStrategy1 | 284 | 29.4 | 38.4 | 50.0 | -2.3 | **0.5** | 3.6 | 76.8 | 41.7 | 26,097 |
| | cgStrategy2 | 139 | 60.6 | 68.5 | 76.4 | -1.5 | 1.6 | 5.8 | 42.8 | 41.8 | 6,822 |
| | cgFull(5,0.3,∪) | 203 | 3.9 | 55.7 | 64.8 | -1.9 | 0.7 | 2.9 | **79.6** | 41.7 | 13,734 |
| | cgFull(10,0.5,∩) | 156 | 56.3 | 65.2 | 72.8 | -1.1 | 1.1 | 4.6 | 59.3 | 42.0 | 12,138 |
| | cgFull(5,1,∩) | 146 | 60.0 | 67.3 | 74.1 | -1.3 | 1.1 | 3.6 | 61.2 | 41.9 | 11,886 |
| D | cgBasic | 372 | - | - | - | - | - | - | - | 41.9 | 19,726 |
| | cgHBD | - | 73.0 | **87.5** | 96.3 | -0.8 | 1.6 | 4.7 | **54.7** | 42.3 | 3,866 |
| | cgStrategy1 | 250 | 17.4 | 32.5 | 43.1 | -3.3 | **0.7** | 3.3 | 46.4 | 41.9 | 21,280 |
| | cgStrategy2 | 117 | 52.9 | 67.0 | 77.2 | -2.1 | 1.4 | 8.2 | 47.9 | 42.1 | 4,715 |
| | cgFull(5,0.3,∪) | 166 | -15.3 | 53.3 | 63.8 | -3.2 | 1.2 | 3.8 | 44.4 | 42.2 | 8,847 |
| | cgFull(10,0.5,∩) | 121 | 57.0 | 66.2 | 73.7 | -1.4 | 1.5 | 5.1 | 44.1 | 42.2 | 7,138 |
| | cgFull(5,1,∩) | 109 | 61.4 | 69.3 | 77.6 | -2.9 | 1.6 | 4.7 | 43.3 | 42.1 | 6,833 |
| E | cgBasic | 491 | - | - | - | - | - | - | - | 29.1 | 23,904 |
| | cgHBD | - | 51.7 | **58.6** | 65.9 | -1.6 | 3.7 | 8.3 | **15.8** | 30.3 | 6,371 |
| | cgStrategy1 | 427 | 2.0 | 13.0 | 21.3 | -1.4 | **2.1** | 5.2 | 6.2 | 29.3 | 26,234 |
| | cgStrategy2 | 205 | 50.1 | 58.2 | 66.6 | 0.8 | 4.8 | 9.0 | 12.1 | 30.7 | 7,204 |
| | cgFull(5,0.3,∪) | 323 | 24.5 | 34.2 | 45.9 | -0.2 | 2.9 | 6.6 | 11.8 | 30.4 | 16,116 |
| | cgFull(10,0.5,∩) | 284 | 29.7 | 41.9 | 49.0 | 0.1 | 3.7 | 7.9 | 11.3 | 30.7 | 14,996 |
| | cgFull(5,1,∩) | 274 | 33.5 | 43.9 | 53.0 | 0.4 | 4.5 | 7.1 | 9.8 | 30.9 | 14,784 |
| F | cgBasic | 2147 | - | - | - | - | - | - | - | 45.3 | 55,709 |
| | cgHBD | - | 60.0 | 65.7 | 70.9 | 0.8 | 3.2 | 7.1 | **20.5** | 46.8 | 10,021 |
| | cgStrategy1 | 2158 | -10.4 | -0.6 | 6.8 | 0.6 | **2.3** | 4.0 | -0.3 | 45.6 | 65,341 |
| | cgStrategy2 | 730 | 61.3 | **65.8** | 69.4 | 2.7 | 5.2 | 8.0 | - | 47.6 | 11,230 |
| | cgFull(5,0.3,∪) | 1470 | 23.7 | 31.4 | 39.1 | 0.6 | 3.2 | 5.8 | 9.8 | 47.5 | 34,913 |
| | cgFull(10,0.5,∩) | 1132 | 42.7 | 47.1 | 50.9 | 2.1 | 4.0 | 6.1 | 11.8 | 47.8 | 30,678 |
| | cgFull(5,1,∩) | 1056 | 44.6 | 50.7 | 56.2 | 1.4 | 4.6 | 7.1 | 11.0 | 48.0 | 30,049 |
| G | cgBasic | 17001 | - | - | - | - | - | - | - | 84.5 | 167,520 |
| | cgHBD | - | 31.5 | 52.5 | 62.1 | 3.3 | 4.2 | 6.9 | 12.5 | 87.1 | 17,948 |
| | cgStrategy1 | 17708 | -14.2 | -4.7 | 8.4 | 1.3 | **1.8** | 2.4 | -2.6 | 84.1 | 199,395 |
| | cgStrategy2 | 5217 | 58.6 | **68.8** | 75.0 | 3.8 | 5.8 | 6.9 | 11.9 | 88.3 | 20,763 |
| | cgFull(5,0.3,∪) | 10343 | 27.6 | 38.7 | 44.8 | 1.7 | 2.5 | 3.9 | **15.5** | 86.4 | 92,381 |
| | cgFull(10,0.5,∩) | 7204 | 48.9 | 57.1 | 65.1 | 3.1 | 3.8 | 4.9 | 15.0 | 87.4 | 75,556 |
| | cgFull(5,1,∩) | 6746 | 51.1 | 60.0 | 66.5 | 3.9 | 4.5 | 5.6 | 13.3 | 87.6 | 73,592 |

# CHAPITRE 7 CONCLUSION

Ce mémoire s'inscrit dans le cadre des recherches visant à répondre aux défis de planification des bus électriques dans un contexte multi-dépôts. Cette problématique, centrale dans l'électrification des transports publics, soulève de nombreux défis à la fois opérationnels et computationnels. À travers une méthodologie combinant reformulation du graphe et réduction de la complexité, ce travail apporte des solutions innovantes qui améliorent à la fois la praticité et l'efficacité des approches actuelles. Cette conclusion s'articule autour de trois parties : la synthèse des contributions, l'analyse des limites de la méthode proposée, et les perspectives de recherche future.

## 7.1 Synthèse des contributions

La principale contribution de ce mémoire réside dans l'élaboration d'un cadre méthodologique permettant de résoudre le MDEVSP tout en respectant les contraintes opérationnelles liées aux autobus électriques. Ce cadre se distingue par trois avancées majeures.

D'abord, une nouvelle formulation de graphe a été développée. Contrairement aux approches classiques, cette reformulation intègre directement les contraintes imposées par les opérateurs, telles que les interdictions ou obligations de séquences entre trajets. En structurant le problème à travers un graphe où les opportunités de recharge sont pré-calculées, cette méthode permet de respecter les contraintes spécifiques tout en garantissant une meilleure flexibilité dans les affectations de trajets.

Ensuite, pour pallier la croissance exponentielle de la taille du graphe due à cette reformulation, des stratégies avancées de réduction ont été proposées. Ces stratégies incluent des règles expertes pour éliminer les arcs peu pertinents, une métaheuristique constructive aléatoire inspirée des travaux de Jacquet et al. [1], et une optimisation des options de recharge par l'élagage des scénarios improbables. Ces innovations permettent de réduire significativement la charge computationnelle, tout en maintenant une qualité de solution comparable à celle des approches de l'état de l'art.

Enfin, les résultats expérimentaux obtenus démontrent la robustesse et la praticité de la méthode dans des scénarios réels. Les tests réalisés sur des instances réalistes, dérivées des réseaux de transport de Montréal, montrent des réductions substantielles des temps de calcul. Pour les petites instances, le temps de calcul est réduit de 63 % en moyenne, avec une augmentation des coûts inférieure à 1,4 %. Pour les grandes instances, la réduction atteint

45 %, avec un surcoût moyen de seulement 3,7 %. Ces résultats confirment la capacité de la méthode à répondre aux exigences opérationnelles tout en offrant un compromis optimal entre performance et coût.

## 7.2 Analyse des limites

Bien que ce travail apporte des avancées significatives, certaines limitations subsistent et méritent une attention particulière dans le cadre de travaux futurs.

Tout d'abord, la méthodologie repose sur des hypothèses simplificatrices concernant le comportement des batteries et les temps de recharge. Par exemple, le modèle linéaire par morceaux utilisé pour représenter le processus de recharge ne tient pas compte de la non-linéarité observée dans les batteries réelles, notamment lorsque la capacité se rapproche de sa limite maximale. Cette simplification, bien qu'efficace pour réduire la complexité computationnelle, peut entraîner des écarts entre les solutions obtenues et les besoins opérationnels réels.

De plus, les stratégies de réduction de graphe s'appuient sur des hyperparamètres dont l'ajustement dépend fortement de la configuration spécifique de chaque instance. Cette dépendance aux données limite la généralisation de la méthode à des contextes variés sans ajustements préalables, ce qui pourrait réduire son attractivité pour les compagnies de transport. Par ailleurs, le caractère heuristique de ces stratégies peut introduire une certaine variabilité dans la qualité des solutions obtenues. Néanmoins, leur coût computationnel très faible, comparé aux approches basées sur l'apprentissage automatique, en fait une alternative intéressante dans un cadre opérationnel, notamment pour des applications nécessitant des temps de calcul réduits ou une intégration rapide.

Enfin, la méthode se concentre sur des flottes homogènes, où tous les véhicules possèdent des caractéristiques identiques en termes d'autonomie et de capacité. Or, dans de nombreux réseaux, les flottes sont composées de véhicules hétérogènes, ce qui introduit une couche supplémentaire de complexité qui n'a pas été abordée dans ce travail.

## 7.3 Perspectives de recherche future

Les travaux futurs pourraient explorer plusieurs axes d'amélioration pour renforcer l'efficacité et la généralisation des approches proposées.

Une première piste consisterait à intégrer une modélisation plus fine du comportement des batteries, en incluant des modèles non linéaires ou des approximations par morceaux plus détaillées. Cela permettrait de mieux refléter la réalité des processus de recharge, en particu-

lier pour les scénarios impliquant des recharges partielles ou des fluctuations de la demande énergétique.

Par ailleurs, l'adaptation de cette méthode à des flottes hétérogènes constitue une direction prometteuse. Une telle extension nécessiterait de repenser la formulation des graphes et les stratégies de réduction pour tenir compte des différences entre véhicules, notamment en termes de capacité de batterie, de consommation énergétique et de vitesse de recharge.

Enfin, une extension naturelle de cette recherche serait l'application de cette méthodologie à des contextes de planification plus complexes, tels que les flottes dynamiques, où les trajets ne sont pas fixés à l'avance, ou les réseaux intermodaux, où les autobus partagent l'infrastructure avec d'autres modes de transport. Ces extensions pourraient élargir le champ d'application de la méthode et répondre à des besoins encore plus variés.

## 7.4 Conclusion générale

En conclusion, ce mémoire apporte une contribution au domaine de l'optimisation des réseaux de transport public électrique. En combinant reformulation du graphe, stratégies de réduction avancées et validation expérimentale rigoureuse, ce travail propose une approche innovante et adaptable pour résoudre le MDEVSP. Les résultats obtenus, tant sur le plan théorique qu'opérationnel, soulignent la pertinence de cette méthode dans le cadre de la transition énergétique et de la mobilité durable. À mesure que les réseaux de transport évoluent pour répondre aux défis environnementaux du XXI$^{e}$ siècle, les approches proposées ici constituent un pas en avant vers des solutions plus efficaces et durables.

# RÉFÉRENCES


[1] T. J. Jacquet, "Sélection d'arcs et génération de colonnes pour le problème d'horaires d'autobus électriques," Mémoire de maîtrise, Polytechnique Montréal, avril 2024. [En ligne]. Disponible : https://publications.polymtl.ca/58011/

[2] J. Gerbaux, G. Desaulniers et Q. Cappart, "A machine-learning-based column generation heuristic for electric bus scheduling," *Computers & Operations Research*, vol. 173, p. 106848, 2025. [En ligne]. Disponible : https://www.sciencedirect.com/science/article/pii/S0305054824003204

[3] Transports Canada, "Émissions de gaz à effet de serre - rapport annuel 2023 de transports canada," 2023, accessed : 2024-09-13. [En ligne]. Disponible : https://tc.canada.ca/fr/services-generaux/transparence/gestion-rapports-ministeriels/rapports-annuels-transports-canada/transports-canada-2023/emissions-gaz-effet-serre

[4] Société de transport de Montréal, "Bus Électriques - Électrification du réseau de surface," 2024, accessed : 2024-09-13. [En ligne]. Disponible : https://www.stm.info/fr/a-propos/grands-projets/grands-projets-bus/electrification-du-reseau-de-surface/bus-electriques

[5] United States Environmental Protection Agency, "Global greenhouse gas emissions overview," 2023, accessed : 2024-11-01. [En ligne]. Disponible : https://www.epa.gov/ghgemissions/global-greenhouse-gas-overview

[6] S. S. Perumal, R. M. Lusby et J. Larsen, "Electric bus planning & scheduling : A review of related problems and methodologies," *European Journal of Operational Research*, vol. 301, n°. 2, p. 395–413, 2022. [En ligne]. Disponible : https://ideas.repec.org/a/eee/ejores/v301y2022i2p395-413.html

[7] G. Desaulniers et M. D. Hickman, "Public transit," dans *Transportation*, ser. Handbooks in Operations Research and Management Science, C. Barnhart et G. Laporte, édit. Elsevier, 2007, vol. 14, p. 69–127. [En ligne]. Disponible : https://www.sciencedirect.com/science/article/pii/S0927050706140025

[8] S. Bunte et N. Kliewer, "An overview on vehicle scheduling models," *Public Transport*, vol. 1, n°. 4, p. 299–317, 2009.

[9] B. A. Foster et D. M. Ryan, "An integer programming approach to the vehicle scheduling problem," *Journal of the Operational Research Society*, vol. 27, n°. 2, p. 367–384, 1976. [En ligne]. Disponible : https://doi.org/10.1057/jors.1976.63

[10] O. Sassi et A. Oulamara, "Electric vehicle scheduling and optimal charging problem : complexity, exact and heuristic approaches," *International Journal of Production Re-*

*search*, vol. 55, n°. 2, p. 519–535, 2017.

[11] A. Montoya, C. Guéret, J. E. Mendoza et J. G. Villegas, "The electric vehicle routing problem with nonlinear charging function," *Transportation Research Part B : Methodological*, vol. 103, p. 87–110, 2017. [En ligne]. Disponible : https://www.sciencedirect.com/science/article/pii/S0191261516304556

[12] L. Costa, C. Contardo et G. Desaulniers, "Exact branch-price-and-cut algorithms for vehicle routing," *Transportation Science*, vol. 53, p. 946–985, 2018. [En ligne]. Disponible : https://api.semanticscholar.org/CorpusID:69715266

[13] J. Desrosiers, M. Lübbecke, G. Desaulniers et J. B. Gauthier, "Branch-and-price," Groupe d'études et de recherche en analyse des décisions, GERAD, Montréal QC H3T 2A7, Canada, Les Cahiers du GERAD G-2024-36, oct. 2024. [En ligne]. Disponible : https://www.gerad.ca/fr/papers/G-2024-36

[14] S. Irnich et G. Desaulniers, "Shortest path problems with resource constraints," dans *Column generation*, G. Desaulniers, J. Desrosiers et M. Solomon, édit. Springer, 2005, p. 33–65.

[15] B. P. Lowerre et B. R. Reddy, "Harpy, a connected speech recognition system," *The Journal of the Acoustical Society of America*, vol. 59, n°. S1, p. S97–S97, 1976.

[16] T. A. Feo et M. G. Resende, "A probabilistic heuristic for a computationally difficult set covering problem," *Operations Research Letters*, vol. 8, n°. 2, p. 67–71, 1989.

[17] M. Dell'Amico, M. Fischetti et P. Toth, "Heuristic algorithms for the multiple depot vehicle scheduling problem," *Management Science*, vol. 39, n°. 1, p. 115–125, 1993.

[18] G. Carpaneto, M. Dell'Amico, M. Fischetti et P. Toth, "A branch and bound algorithm for the multiple depot vehicle scheduling problem," *Networks*, vol. 19, n°. 5, p. 531–548, 1989.

[19] C. C. Ribeiro et F. Soumis, "A column generation approach to the multiple-depot vehicle scheduling problem," *Operations Research*, vol. 42, n°. 1, p. 41–52, 1994.

[20] N. Kliewer, T. Mellouli et L. Suhl, "A time–space network based exact optimization model for multi-depot bus scheduling," *European Journal of Operational Research*, vol. 175, n°. 3, p. 1616–1627, 2006. [En ligne]. Disponible : https://www.sciencedirect.com/science/article/pii/S0377221705002274

[21] A.-S. Pepin, G. Desaulniers, A. Hertz et D. Huisman, "Comparison of heuristic approaches for the multiple depot vehicle scheduling problem," *Journal of Scheduling*, vol. 12, n°. 1, p. 17–30, janv. 2009.

[22] J.-Q. Li, "Transit bus scheduling with limited energy," *Transportation Science*, vol. 48, n°. 4, p. 521–539, 2014. [En ligne]. Disponible : https://doi.org/10.1287/trsc.2013.0468

[23] M. Wen, E. Linde, S. Ropke, P. Mirchandani et A. Larsen, “An adaptive large neighborhood search heuristic for the electric vehicle scheduling problem,” *Computers & Operations Research*, vol. 76, p. 73–83, 2016.

[24] A. Zhang, T. Li, Y. Zheng, X. Li, M. G. Abdullah et C. Dong, “Mixed electric bus fleet scheduling problem with partial mixed-route and partial recharging,” *International Journal of Sustainable Transportation*, vol. 16, nº. 1, p. 73–83, 2022.

[25] J. Pasha, B. Li, Z. Elmi, A. M. Fathollahi-Fard, Y. yip Lau, A. Roshani, T. Kawasaki et M. A. Dulebenets, “Electric vehicle scheduling : State of the art, critical challenges, and future research opportunities,” *Journal of Industrial Information Integration*, vol. 38, p. 100561, 2024. [En ligne]. Disponible : https://www.sciencedirect.com/science/article/pii/S2452414X24000050

[26] G. F. Savari, M. J. Sathik, L. A. Raman, A. El-Shahat, H. M. Hasanien, D. Almakhles, S. H. A. Aleem et A. I. Omar, “Assessment of charging technologies, infrastructure and charging station recommendation schemes of electric vehicles : A review,” *Ain Shams Engineering Journal*, vol. 14, nº. 4, p. 101938, 2023.

[27] C. Wang, C. Guo et X. Zuo, “Solving multi-depot electric vehicle scheduling problem by column generation and genetic algorithm,” *Applied Soft Computing*, vol. 112, p. 107774, 2021.

[28] R. Sadykov, F. Vanderbeck, A. Pessoa, I. Tahiri et E. Uchoa, “Primal heuristics for branch and price : The assets of diving methods,” *INFORMS Journal on Computing*, vol. 31, nº. 2, p. 251–267, 2019.

[29] J. Brasseur, “Accélération d’une méthode d’agrégation dynamique de contraintes par apprentissage automatique pour le problème de construction d’horaires de conducteurs d’autobus,” Mémoire de maîtrise, Polytechnique Montréal, 2022.

[30] J. Desrosiers, *GENCOL : Une équipe et un logiciel d’optimisation.* Studia Informatica Universalis, janv. 2010, vol. 8, nº. 2, p. 61–96.