Modélisation et Résolution du problème de tournées de véhicules dynamique multi-tour avec overtime

Résumé : Après trois décennies de son introduction, le DVRP est toujours un domaine fertile pour de nouvelles études. L'évolution technologique, qui continue de progresser de jour en jour, a permis une meilleure communication entre les différents acteurs de ce modèle. Ceci a motivé les chercheurs à introduire de nouvelles variantes du DVRP et à utiliser des algorithmes plus complexes pour leur résolution. Parmi ces variantes, on trouve le DVRP multi-tour (MTDVRP) avec overtime (MTDVRPOT).

Traditionnellement, dans un VRP, les véhicules retournent au dépôt avant la fin du temps de travail. Cependant, en réalité, plusieurs contraintes peuvent survenir et empêcher les véhicules d'être à l'heure, au dépôt. Dans le cas dynamique, nous sommes censés répondre aux demandes le jour même de leur arrivée. Néanmoins, pour les entreprises qui disposent d'une flotte limitée, il n'est pas toujours facile de trouver une solution qui assure tout le service tout en respectant le temps normal de travail. Il sera, alors, très pertinent de donner aux véhicules un temps supplémentaire pour terminer leurs services, surtout s'il y a une forte demande.

Cette thèse introduit et résout le MTDVRPOT. Il s'agit d'un problème d'optimisation combinatoire bi-objectif. Pour sa résolution, nous proposons trois démarches. La première est basée sur une méthode exacte itérative, la seconde est un système de colonies de fourmis hybride alors que la troisième est un algorithme mémétique.

Mots clés : Problème de tournées de véhicules dynamique, Multi-tour, Overtime, Optimisation combinatoire, Algorithme mémétique, Algorithme de colonie de fourmis, Métaheuristique

---

Abstract: Even, after three decades of its introduction, DVRP is still a fertile area for further studies. Technological evolution, which continues to progress day by day, has enabled better communication between the various actors in this model. This motivated the researchers to introduce new variants of DVRP and use more complex algorithms for their resolution. Among these variants is the multi-tour DVRP (MTDVRP) with overtime (MTDVRPOT).

Traditionally, in a VRP, vehicles return to the depot before the end of working time. However, in reality, several constraints can arise and prevent vehicles from being on time at the depot. In the dynamic case, we are expected to respond to requests the same day of their arrival. However, for companies with a limited fleet, it is not always easy to figure out a solution that ensures the entire service while respecting the normal working hours. Therefore, it would be very relevant to give vehicles an extra time to complete their services, especially if there is a high demand.

This thesis introduces and solves the MTDVRPOT. This is a bi-objective problem of combinatorial optimization. For its resolution, we propose three approaches. The first is based on an exact iterative method, the second is a hybrid ant colony system while the third is a memetic algorithm.

Keywords: Dynamic vehicle routing problem, Multi-tour, Overtime, Combinatorial optimization, Memetic algorithm, Ant colony system, metaheuristics

Khaoula OUADDI

Modélisation et résolution du problème de tournées de véhicules dynamique multi-tour avec overtime

Thèse N° : 194/ST2I

Année : 2020

Année : 2020 **Thèse N° : 194/ST2I**

جامعة محمد الخامس بالرباط
Université Mohammed V de Rabat

École Nationale Supérieure d'Informatique et d'Analyse des Systèmes
Centre d'Études Doctorales en Sciences des Technologies de l'Information et de l'Ingénieur

THÈSE DE DOCTORAT

MODÉLISATION ET RÉSOLUTION DU PROBLÈME DE TOURÉES DE VEHICULES DYNAMIQUE MULTI-TOUR AVEC OVERTIME

Présentée par

Khaoula OUADDI

Le 31/12/2020

Formation doctorale : Informatique
Structure de recherche : Smart Systems Laboratory (SSL)

JURY

Professeur Rachid ELLAIA PES, EMI, Université Mohammed V de Rabat	**Président**
Professeur Youssef BENADADA PES, ENSIAS, Université Mohammed V de Rabat	**Directeur de thèse**
Professeur Fatima-Zahra MHADA PH, ENSIAS, Université Mohammed V de Rabat	**Co-Encadrant de thèse**
Professeur Souad EL BERNOUSSI PES, Faculté des Sciences, Université Mohammed V de Rabat	**Rapporteur**
Professeur Mohammed DOUIMI PES, ENSAM, Université Moulay Ismail, Meknès	**Rapporteur**
Professeur Adil BELLABDAOUI PH, ENSIAS, Université Mohammed V de Rabat	**Rapporteur**
Professeur Fatima OUZAYED PH, ENSIAS, Université Mohammed V de Rabat, Rabat	**Examinateur**
Professeur Ahmed EL HILALI ALAOUI Professeur-Expert, Université Euromed, Fès	**Examinateur**

Abstract

Being a member of the VRP family, Dynamic VRP (DVRP) has gained great interest among operational research scientists. Even, after three decades of its introduction, DVRP is still a fertile area for further studies. Technological evolution, which continues to progress day by day, has enabled better communication between the various actors in this model, namely: managers, vehicle drivers and customers. This motivated the researchers to introduce new variants of DVRP and use more complex algorithms for their resolution. Among these variants is the multi-tour DVRP (MTDVRP) with overtime (MTDVRPOT).

Traditionally, in a VRP, vehicles return to the depot before the end of working time. However, in reality, several constraints can arise and prevent vehicles from being on time at the depot. In the dynamic case, we are expected to respond to requests the same day of their arrival. However, for companies with a limited fleet, it is not always easy to figure out a solution that ensures the entire service while respecting the normal working hours. Therefore, it would be very relevant to give vehicles an extra time to complete their services, especially if there is a high demand.

This thesis introduces and solves the MTDVRPOT. This is a bi-objective problem of combinatorial optimization. For its resolution, we propose three approaches. The first is based on an exact iterative method, the second is a hybrid ant colony system while the third is a memetic algorithm.

Carried out tests have shown that the memetic algorithm demonstrated superior performance terms of objective function while the ant colony system is better in terms of execution time. Otherwise, the first approach remains limited to small size instances.

Keyword:

Dynamic vehicle routing problem, Multi-tour, Overtime, Combinatorial optimization, Memetic algorithm, Ant colony system, metaheuristics.

Résumé

Étant membre de la famille VRP, le VRP dynamique (DVRP) a suscité un grand intérêt auprès des spécialistes de la recherche opérationnelle. Après trois décennies de son introduction, le DVRP est toujours un domaine fertile pour de nouvelles études. L'évolution technologique, qui continue de progresser de jour en jour, a permis une meilleure communication entre les différents acteurs de ce modèle, à savoir : les gestionnaires, les chauffeurs des véhicules et les clients. Ceci a motivé les chercheurs à introduire de nouvelles variantes du DVRP et à utiliser des algorithmes plus complexes pour leur résolution. Parmi ces variantes, on trouve le DVRP multi-tour (MTDVRP) avec overtime (MTDVRPOT).

Traditionnellement, dans un VRP, les véhicules retournent au dépôt avant la fin du temps de travail. Cependant, en réalité, plusieurs contraintes peuvent survenir et empêcher les véhicules d'être à l'heure, au dépôt. Dans le cas dynamique, nous sommes censés répondre aux demandes le jour même de leur arrivée. Néanmoins, pour les entreprises qui disposent d'une flotte limitée, il n'est pas toujours facile de trouver une solution qui assure tout le service tout en respectant le temps normal de travail. Il sera, alors, très pertinent de donner aux véhicules un temps supplémentaire pour terminer leurs services, surtout s'il y a une forte demande.

Cette thèse introduit et résout le MTDVRPOT. Il s'agit d'un problème d'optimisation combinatoire bi-objectif. Pour sa résolution, nous proposons trois démarches. La première est basée sur une méthode exacte itérative, la seconde est un système de colonies de fourmis hybride alors que la troisième est un algorithme mémétique.

Les tests effectués ont montré la supériorité de l'algorithme mémétique en termes de la fonction objectif alors que l'algorithme de colonie de fourmis est le meilleur en termes du temps d'exécution. En revanche, la première démarche reste limitée aux instances de petite taille.

Mot clé :

Problème de tournées de véhicules dynamique, Multi-tour, Overtime, Optimisation combinatoire, Algorithme mémétique, Algorithme de colonie de fourmis, métaheuristiques.

Remerciement

Je n’aurais jamais pu aboutir à ce stade sans le soutien d’un bon nombre de personnes qui ont été à mes côtés tout au long des années de thèse. Je tiens à exprimer ma plus sincère gratitude à toutes les personnes qui m'ont aidé à mener à bien cette thèse.

Je remercie, en premier lieu, mon directeur de thèse Pr Youssef BENADADA pour m’avoir accueilli au sein de son équipe. Je lui dis merci pour tout le temps qu’il m’a accordé, ses directifs scientifiques et pédagogiques ainsi que ses qualités humaines. J’aimerais également lui exprimer à quel point je suis reconnaissante pour son soutien aux moments les plus difficiles de ce parcours doctoral.

J’adresse mes chaleureux remerciement à ma co-encadrante Pr Fatima-Zahra MHADA pour avoir passé beaucoup de temps sur mon travail, pour ses conseils, son écoute et sa franchise. Je me considère très chanceuse d’avoir travaillé avec elle.

Je remercie tous les membres de la jurée de cette thèse pour le temps qu’ils ont accordé pour lire et évaluer mon travail.

Je souhaite aussi exprimer ma gratitude à tout le personnel de la CNRST pour les moyens qu’ils ont mis en œuvre pour me donner accès à la plateforme MARWAN. Je tiens à témoigner toute ma gratitude à Mme Bouchra RAHIM pour son assistance technique et logistique.

Enfin, je suis très reconnaissant à toute ma famille de m'avoir soutenu dans ce long parcours, et en particulier à mes parents, mon mari et mes petites filles qui sont nées pendant les années de thèse.

Table des matières

Liste des figures

Liste des tableaux

Introduction

De nombreux problèmes d'optimisation sont réellement soumis à des environnements dynamiques. Dans un environnement statique, on considère que l'espace de recherche, la fonction objectif ainsi que les variables sont tous constants par rapport au temps. Certes, cette façon permet de réduire la complication de la résolution, mais elle ne reflète pas la réalité des choses, parce que le dynamisme est le caractère dominant de la vie réelle. En effet, les entrées, les procédures et les sorties d'un système réel sont toutes variables par rapport au temps. En optimisation combinatoire, le système est un problème à optimiser dont l'objectif est de trouver la meilleure solution dans un ensemble discret. Les entrées, les procédures et les sorties sont respectivement, les données, les algorithmes de résolution et la solution optimale. Cette dernière n'est pas toujours atteignable du fait que beaucoup de problèmes d'optimisation combinatoire sont très difficiles à résoudre. En outre, le dynamisme ajoute un autre degré de difficulté. Toutefois, l'évolution accélérée du marché, les exigences croissantes des clients ainsi que la disponibilité technologique ont imposé aux chercheurs de revoir leur attitude. De là, une nouvelle classe de problème a vu le jour. Il s'agit des problèmes d'optimisation combinatoires dynamiques. Cette classe regroupe les problèmes, dont au moins l'une des composantes change dans le temps, à savoir ; la fonction objectif, les variables, les contraintes et la solution optimale. Dans ce genre d'optimisation, on assume, dès le début, que l'information disponible n'est pas complète et que la solution fournie initialement est susceptible de changer si de nouvelles informations seront disponibles. C'est un défi multi-compliqué. Du point de vue algorithmique, les algorithmes de résolution doivent être capables de mettre à jour leurs données, en continu, en donnant des résultats optimaux dans des durées courtes. Du point de vue exécutif, les dispatcheurs doivent disposer des moyens technologiques et organisationnels qui permettent de collecter les nouvelles informations, d'un côté, et de mettre en application les nouvelles solutions, de l'autre.

Aujourd'hui, la classe des problèmes d'optimisation combinatoire dynamique contient un bon nombre de problèmes qui ont été traités dynamiquement. Le problème de bin packing (Ivković & Lloyd, 1993), le problème de couverture minimale des sommets (Ivkovic & Lyyold, 1994), le problème du plus court chemin (Klein & Subramanian, 1998) et le problème du voyageur de commerce (C. Li et al., 2006) sont tous des exemples.

Le problème de voyageur de commerce est un cas particulier du problème de tournées de véhicules (VRP) qui constitue un champ d'études très intéressant, de la recherche opérationnelle, vu son utilité

dans les systèmes logistiques. Le but d'un VRP est de trouver les tournées qu'une flotte de véhicule doit effectuer pour visiter un ensemble de nœuds en minimisant les coûts considérés. Ce problème est directement lié aux systèmes de transport et de distribution où beaucoup d'opérations sont soumises à des restrictions temporelles strictes. En effet, les clients exigent des délais temporels limités pour leurs services. Le niveau de respect de ces délais est un facteur clé dans la compétitivité d'une société de transport. En outre, dans le contexte des urgences, qui forme un secteur très attaché au domaine du transport, le délai de réponse ne reste plus un facteur de différenciation compétitive, mais devient un point de haute criticité, puisque la vie d'un ou plusieurs personnes en dépend. C'est pourquoi, la version dynamique du VRP a reçu un grand intérêt depuis son introduction par (Psaraftis, 1980).

Le VRP dynamique (DVRP) est un VRP où l'information requise pour planifier les tournées varie avec le temps. Cette variation induit un changement de la solution résultante ainsi que les procédures de résolution. Ceci étant, l'implémentation du DVRP nécessite une plateforme technologique capable de manipuler les véhicules en fonction des besoins actualisés. Heureusement, les technologies de l'information et de la communication ont récemment connu un grand progrès. En conséquence, les flottes de véhicules peuvent, désormais, être gérées en temps réel. En effet, les systèmes d'information géographique (SIG) sont capables de fournir l'emplacement actuel des véhicules. Le système de positionnement global (GPS) met à jour les routes possibles selon la position courante. De surcroît, les capteurs de flux de trafic sont en mesure de fournir des données actualisées sur les estimations du temps de parcours des arcs d'un réseau routier, alors que les téléphones cellulaires permettent de recevoir les nouvelles requêtes des clients en temps réel. À partir de ces données, on peut mettre à jour les itinéraires à suivre, dès qu'un nouvel événement se déclenche. Cela permettra non seulement d'améliorer le niveau de service, mais encore d'éviter des dégâts non souhaités surtout dans le contexte des urgences. Tous ces facteurs justifient l'intérêt croissant pour les modèles de transport dynamiques ou en ligne que le DVRP représente.

Malgré toute l'importance donnée au DVRP, il existe encore des variantes moins étudiées. Le DVRP multi-tour (DVRPMT) est l'une de ces variantes. La version statique de ce problème est largement étudiée dans la littérature. Toutefois, les recherches sur le DVRPMT sont très rares. Ce problème est caractérisé par une flotte généralement réduite. Conséquemment, servir tous les clients pendant le temps légal du travail n'est pas toujours possible. De ce fait, de nombreuses études qui traitent le cas statique du VRP multi-tour (VRPMT) permettent aux véhicules d'utiliser un temps supplémentaire appelé « overtime ». L'intérêt de l'overtime se manifeste lorsqu'on n'arrive pas à terminer tout le service durant le temps normal du travail. En d'autres termes, la tolérance de

l'overtime permet aux managers de servir le maximum de clients avant de clôturer la journée de travail surtout dans le cas d'une forte demande. Sachant que dans le cas dynamique, on est sensé servir le maximum de clients le même jour où ils ont effectué leurs requêtes, la tolérance de l'overtime est, certainement, plus bénéfique dans ce cas, surtout s'il y en a des requêtes tardives.

Par ailleurs, l'objectif le plus classique des VRPs est la minimisation de la distance puisqu'elle représente le coût principal de l'activité du transport. Or, plusieurs autres facteurs peuvent contribuer au coût tel que le nombre de véhicules, leurs tailles ou leurs vitesses de roulement. Dans le cas où les véhicules utilisent un overtime, ce dernier présente un coût additionnel pour l'entreprise, notamment en termes de payement des conducteurs, ainsi que les surveillants des dépôts. En effet, ce coût peut être pris en compte de deux manières différentes ; soit en considérant l'overtime total qui reflète l'overtime performé par l'ensemble des véhicules, soit en considérant l'overtime maximal qui représente l'overtime réalisé par le véhicule qui revient le dernier au dépôt. Dans les deux cas, on a besoin de considérer l'overtime comme objectif à minimiser.

Cette thèse est une contribution aux travaux de recherche sur le DVRP, dans laquelle l'objectif final est de résoudre le DVRP multi-tour avec overtime (MTDVRPOT). Dans ce cadre, nous considérons la minimisation de l'overtime maximal performé comme objectif à côté de la minimisation de la distance. Parce que la minimisation de l'overtime maximal permet de créer un équilibre entre les overtimes performés par les différents véhicules de tel sort que le dernier véhicule revient le plus tôt possible. De cette façon, nous allons minimiser à la fois le coût des conducteurs et celui des surveillants du dépôt. Chose que nous ne pouvons pas garantir si nous considérons l'overtime total.

Selon nos recherches, nous sommes les premiers à traiter cette variante du DVRP. Pour ce faire, nous avons d'abord modélisé le MTDVRPOT sous forme de programme mathématique non linéaire en nombre entier. À base de cette modélisation, nous avons développé une heuristique de résolution dont le principe est d'appliquer une méthode exacte de façon itérative. Ensuite, nous avons développé deux autres approches à base de métaheuristiques. La première approche est basée sur le système de colonie de fourmis (ACS) et la seconde est un algorithme mémétique (MA). Afin de comparer les deux métaheuristiques avec des approches de la littérature, nous les avons testés sur des benchmarks conçus pour une variante très classique du DVRP. Par la suite, nous avons proposé de nouveaux jeux de tests inspirés de ces benchmarks et adaptés au MTDVRPOT. Pour pouvoir tester les trois approches, nous avons conçu des problèmes de tailles variées.

Les résultats des travaux de recherche présentées dans cette thèse ont donné lieu à plusieurs articles scientifiques. Certains de ces articles ont été publiés alors que d'autres sont acceptés avec quelques modifications que nous avons effectuées. Dans ce qui suit, nous présentons la liste de nos publications triées par ordre chronologique :

1. Ouaddi, K., Benadada, Y., & Mhada, F. Z. (2016). Multi period dynamic vehicles routing problem: Literature review, modelization and resolution. *Proceedings of the 3rd IEEE International Conference on Logistics Operations Management, GOL 2016*.
2. Ouaddi, K., Benadada, Y., & Mhada, F. Z. (2017). *Ant Colony system approach for dynamic vehicles routing problem multi tour*. *2nd International Conference on Big Data Cloud and Applications, BDCA 2017; Tetuan; Morocco;*, 1–6.
3. Ouaddi, K., Benadada, Y., & Mhada, F.-Z. (2018). Ant Colony System for Dynamic Vehicle Routing Problem with Overtime. *International Journal of Advanced Computer Science and Applications* (Q4, SJR 2019: 0.16), *9*(6), 306–315.
4. Ouaddi, K., Mhada, F.-Z., & Benadada, Y. (2020). Memetic algorithm for multi-tour dynamic vehicle routing problem with overtime (MDVRPOT). *International Journal of Industrial Engineering Computations* (Q1, SJR 2019: 0.98), *March*.
5. Ouaddi, K., Benadada, Y & Mhada, F.-Z. (..). Heuristic based on iterative exact method for multi-tour dynamic vehicle routing problem with overtime. *International Journal of Logistics Systems and Management (*Citescore 2019 : 1.9). Accepté et modifié, en attente de publication.
6. Ouaddi, K., Mhada, F.-Z., & Benadada, Y. (2021). Etude comparative entre trois méthodes de résolution du problème tournées de véhicules dynamique multi-tour avec overtime (MTDVRPOT), *Proceedings of the 5th IEEE International Conference on Logistics Operations Management, GOL 2020.*

Ce rapport qui se compose de 5 chapitres présente le bilan de ce travail :

- Chapitre 1 : Dans ce chapitre, nous présentons l'étude bibliographique élaborée sur le VRP et le DVRP, notamment les notions de base, les différents concepts qu'il faut distinguer ainsi que les taxonomies et les approches de résolutions proposées.

- Chapitre 2 : Dans ce chapitre nous décrivons le problème MTDVRPOT avec sa modélisation mathématique. Ensuite, nous présentons une heuristique basée sur la résolution

itérative du modèle mathématique. Après avoir expliqué les jeux de tests, nous présentons les résultats numériques vers la fin du chapitre.

- Chapitre 3 : Ce chapitre est consacré à la démarche basée sur l'ACS. Dans un premier temps, nous présentons l'algorithme proposé pour la version statique du problème. Ensuite, nous présentons l'algorithme de résolution du problème dynamique. Avant de l'appliquer sur le MTDVRPOT, nous le testons sur des benchmarks classiques du DVRP dans le but de comparer sa performance avec une autre approche de la littérature. A la fin de ce chapitre nous allons présenter les jeux de tests conçus pour le MTDVRPOT ainsi que les résultats numériques.

- Chapitre 4 : Ce chapitre est dédié à l'adaptation de l'algorithme mémétique au MTDVRPOT. À l'égard de l'ACS, l'algorithme est testé sur des benchmarks classiques du DVRP afin de le comparer avec d'autres démarches de la littérature. A la fin du chapitre, les résultats des tests numériques seront présentés.

- Chapitre 5 : Dans ce chapitre nous élaborons une comparaison entre les résultats des trois démarches. Ensuite, les contributions de cette thèse seront discutées.

Chapitre 1

Généralités et revue de littérature

1.1 Introduction

Après plus d'un demi-siècle de la première introduction du problème de tournées de véhicules (VRP) (G. Dantzig et al. 1954), plusieurs sous-classes de ce problème ont été définies. À l'intérieur de chaque sous-classe, on trouve plusieurs variantes qui chevauchent, parfois, avec d'autres sous-classes. Le problème de tournées de véhicules dynamique (DVRP) (Psaraftis, 1980) est l'une des sous-classes les plus récentes du VRP. Le caractère principal du DVRP est le dynamisme de ses entrées et ses sorties : c'est-à-dire que le plan de routage initial est susceptible d'être changé si l'on reçoit des modifications concernant les requêtes ou bien les routes. On peut résumer les modifications possibles dans les trois points suivants :

- Modifier la quantité, le type ou la location d'une requête déjà planifiée.
- Ajouter ou supprimer une requête du plan de routage en cours.
- Changer les routes planifiées par d'autres routes pour échapper à la congestion du réseau.

Le DVRP inclut aujourd'hui un large nombre de variantes et sous-variantes. L'objectif principal de ce chapitre est de présenter le DVRP, clarifier les points qui le distingue des autres versions du VRP, parler du degré du dynamisme comme étant un attribut caractérisant du DVRP, présenter les intérêts spécifiques du DVRP et les travaux qui le traitent ainsi que leurs taxonomies. De surcroit et afin de cerner toute la littérature en relation avec le MTDVRPOT, nous présentons une revue de littérature sur les variantes du VRP, autres que le DVRP, qui chevauchent avec notre problème.

1.2 VRP dynamique (DVRP)

Le problème de tournée de véhicules (VRP) est un problème d'optimisation introduit initialement par Dantzig & Ramser (1959) et dont l'objectif principal est de minimiser le coût de transport. Plusieurs facteurs peuvent impacter ce coût explicitement ou implicitement (ex. : la distance parcourue, le nombre de véhicules utilisés, le respect des délais temporaires...). Sachant que l'espace de recherche est limité, par des contraintes spécifiant la capacité des véhicules, les créneaux horaires et parfois des exigences supplémentaires en relation avec les conducteurs et l'environnement, le but

est de trouver une solution optimale. Toutefois, le VRP est un problème NP-difficile (Lenstra & Kan, 1981). Raison pour laquelle, une solution optimale est souvent inatteignable, avec certitude. Ceci a fait du VRP, à côté de sa grande utilité dans la vie réelle, un champ de recherche toujours fertile depuis cinquante ans. Néanmoins, pour les problèmes de petite taille, un bon nombre d'études ont réussi à trouver l'optimum pour différentes variantes du VRP.

Ayant défini le VRP, nous allons parler du DVRP. Larsen et al. (2008) ont défini le DVRP comme étant la façade dynamique du VRP générique ou conventionnel « The DVRP is the dynamic counterpart of the generic vehicle routing problem (VRP) ». Dans la version conventionnelle du VRP, toutes les informations qui concernent les requêtes à servir sont connues en avance. Les itinéraires ainsi planifiés sont définitifs. En revanche, dans le DVRP, toute l'information nécessaire, ou au moins une partie, n'est disponible qu'en cours de l'exploitation. Autrement dit, les entrées du problème peuvent changer après la construction des itinéraires initiaux. L'objectif dans un DVRP est de considérer ces changements dans le plan actuel en permettant la mise à jour des itinéraires initiaux. Si aucun changement n'a affecté les informations, on se retrouve dans le cas d'un VRP statique (VRPS). Ceci étant, le VRPS peut être considéré comme un cas particulier du DVRP et non pas le contraire. La figure 1 illustre un exemple simpliste du DVRP.

1.2.1 VRP statique (VRPS) et VRP dynamique (DVRP)

Nous venons de définir le DVRP comme étant la façade dynamique du VRP. Dans cette partie,

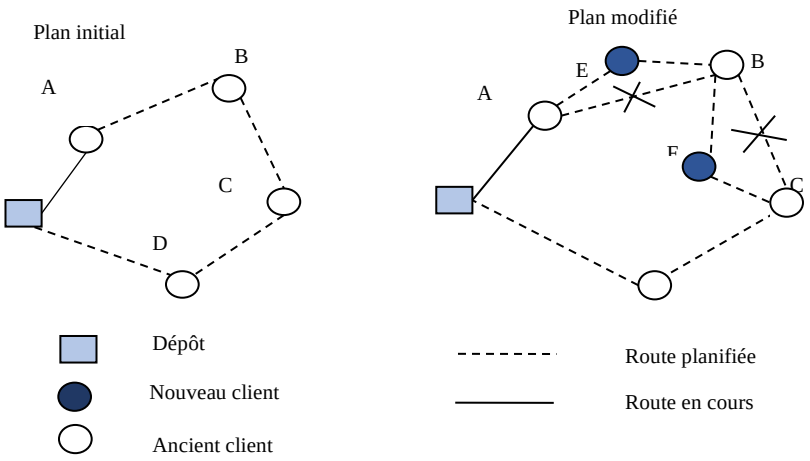

Figure 1: Exemple simpliste du DVRP

nous mettons le point sur les principales différences entre le DVRP et le VRPS. Bianchi (2000) a parlé des différences en termes de temps, des informations futures, de la fonction objectif et de la stratégie :

- **Le temps** : Le temps est la principale dimension, présent dans les problèmes dynamiques et absent dans le cas statique. Dans le cas dynamique, le temps est une caractéristique intégrale de chaque instance. Si on définit le VRP statique par $P=(X, C, F)$, où X est le vecteur variable de décision, C est l'espace vectoriel auquel appartient X (défini par les contraintes du problème) et F est la fonction objectif, le DVRP est défini alors par $P(t)=(X(t), C(t), F(t))$. Dans cette définition, tous les composants sont des variables de temps. Ceci nécessite, au moins, la capabilité de connaitre la position des véhicules à tout moment donné pour pouvoir mettre à jours les variables de décision et les contraintes. Puisque, la fonction objectif peut, aussi, changer au cours du temps, nous devons avoir la possibilité de communiquer en temps réel avec les chauffeurs en vue de les rediriger en cas de besoin en leur donnant les nouveaux itinéraires.

- **Informations futures :** Dans le cas statique, on assume que toutes les informations sont connues, à l'avance. Or, dans le cas dynamique qui domine la vie réelle, seulement, une partie de l'information est connue en avance. Le reste se dévoile ou change dynamiquement avec le temps. Le futur peut être complètement ambigu comme il peut être partiellement connu si on dispose des prévisions stochastiques. Le modèle dynamique est caractérisé par la flexibilité de réaction avec les informations futures. Chose qui n'est pas présente dans le cas statique.

- **La fonction objectif :** la définition de la fonction objectif est un aspect non trivial dans les processus dynamiques. Dans le but de suivre la progression de la fonction objectif dans le temps, le problème dynamique peut être considéré comme une série de problèmes statiques (sous-problèmes). Ainsi, le DVRP hérite des objectifs classiques définis dans le VRP conventionnel. Néanmoins, la nature dynamique du problème conduit à l'émergence de nouveaux objectifs. De façon générale, la fonction objectif est une combinaison de différentes mesures liées à la nature du système. Par exemple, si certaines informations stochastiques sont liées au problème, elles devraient être prises en compte par la fonction objectif. Un autre exemple apparaît dans les services d'urgence où l'intérêt consiste à minimiser le temps de réponse qui correspond au décalage prévu entre le moment où la demande de l'utilisateur survient et son heure de service.

- **Stratégie :** La stratégie est la formule décisionnelle qui implique quelle action devrait être prise à chaque état d'avancement du problème. Cela peut concerner la manière dont le véhicule doit être positionné dans la zone de service. De même que la manière dont les

demandes doivent être traitées. Par exemple, on peut envisager un nouveau sous-problème statique chaque fois qu'une nouvelle requête arrive ou bien attendre un certain temps pour créer un sous-ensemble de nouveaux clients, ou bien considérer une représentation spatiale des requêtes et servir le client non servi le plus proche au véhicule, etc. Ci-après, nous allons décrire quelques stratégies utilisées dans la littérature (Khouadjia, 2011) :

- *Premier arrivé, premier servi (FIFO) :* les demandes sont servies dans l'ordre dans lequel elles sont reçues par le répartiteur.
- *Moyenne de la file d'attente (SQM) :* localisez le véhicule à la position médiane dans la région de service et servez les clients conformément à la stratégie FIFO. Lorsque le service est terminé, le véhicule revient à la médiane.
- *Le plus proche voisin (NN) :* stratégie gourmande, dans laquelle le véhicule répond à la demande non traitée la plus proche.
- *Voyageur de commerce (TS) :* les demandes sont regroupées dans un ensemble de taille donnée. Une fois qu'un ensemble de demandes a été rassemblé, le problème de voyageur de commerce (TSP) est résolu. Les demandes sont satisfaites en suivant la tournée optimale qui commence et se termine au dépôt.
- *Partitionnement (PART) :* la zone de service est divisée en sous-régions. Un véhicule visite les sous-régions dans un ordre qui permet de passer d'une sous-région à une autre adjacente. Les demandes dans chaque sous-région sont traitées dans un ordre FIFO.
- *Waiting (WAIT) :* il s'agit de trouver un temps d'attente optimal pour le véhicule afin de maximiser la probabilité qu'un nouveau client puisse être intégré à l'une des tournées (déjà planifiée), tout en réduisant la longueur moyenne de déviation nécessaire pour servir ce client.

1.2.2 DVRP et VRP stochastique (SVRP)

Le VRP stochastique (SVRP) est une catégorie différente du DVRP. Toutefois, la non-disponibilité des informations complètes au début de l'optimisation, dans les deux modèles, peut créer une certaine ambiguïté. Ritzinger et al. (2016) ont définit le SVRP comme suit: *"The stochastic VRP (SVRP) is basically any VRP where one or more parameters are stochastic, meaning that some future events are random variables with a known probability distribution".* Si l'une ou plusieurs informations du problème ont un caractère incertain, le problème est stochastique (. Habituellement, les informations sur les événements à venir sont disponibles via l'historique des données qui peuvent être converties en modèles d'information. Les problèmes SVRP (Rouky et al., 2018) sont, souvent, modélisés sous la forme d'un programme stochastique (Birge & Louveaux, 1997). De façon générale,

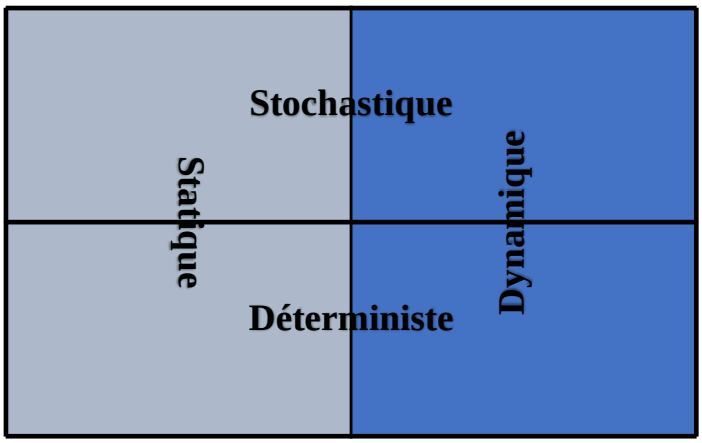

Figure 2: Différentes classes de problèmes (statique, dynamique, stochastique et déterministe)

la modélisation d'un programme stochastique est faite en deux phases. Dans un premier temps, on construit une solution en se basant sur les informations stochastiques préalablement disponibles. Ensuite, au fur et à mesure que les informations réelles se dévoilent, on procède à un recours correctif de la solution initiale. Le coût généré par ce recours devra être pris en compte, éventuellement, dans la conception de la solution initiale. Par exemple, dans le cas d'un VRP avec quantité demandée stochastique, nous allons construire un plan de routage initial à base des prévisions sur les quantités demandées sachant que les quantités réelles ne vont être connues qu'à la livraison. Le décalage entre les quantités prévues et réelles pourra produire un manque chez quelques clients. L'action corrective pourra être, la planification d'une nouvelle tournée pour livrer le manque réalisé. Une excellente revue sur le SVRP est présente dans Gendreau et al. (1996). Le lecteur intéressé trouvera, aussi, une étude, très marquante, dédiée aux applications des métaheuristiques sur l'optimisation combinatoire stochastique dans Bianchi et al. (2009).

Par rapport au DVRP, le point principal qui le distingue du SVRP est que ce dernier, à la base statique, exploite les données stochastiques pour faire un plan durable. Alors que le DVRP est un modèle, à la base déterministe, qui permet la mise à jour du premier plan en réponse aux nouvelles informations. Néanmoins, le DVRP peut être stochastique, si une partie des données utilisées dans l'optimisation est stochastique. Dans le cas contraire (déterministe), on n'utilise que des données certaines. La figure 2 illustre les classes de problème dynamique, statique, stochastique et déterministe et les chevauchements possibles entre elles.

1.2.3 Degré de dynamisme

Avant de modéliser un problème dynamique, nous aurons besoin de savoir à quel point ce problème est dynamique. Autrement dit, nous aurons besoin de quantifier son dynamisme. L'aspect dynamique d'un problème est lié au nombre des événements dynamiques et des coordonnées spatiales et temporaires de ces événements. Lund et al. (1996) et Larsen, (2000) ont défini le degré de

dynamisme d'un problème. On suppose que l'horizon de planification est l'intervalle *[0,T]*. n_s est le nombre des requêtes statiques et n_d est le nombre de requêtes dynamiques. Soit $t_i \in [0, T]$ le temps d'apparition de la requête *i*. Ainsi, les requêtes statiques ont toutes un $t_i = 0$, alors que les requêtes dynamiques ont des $t_i \in]0, T]$. Lund et al. (1996) ont défini le degré de dynamisme comme suit :

$$dod = \frac{n_d}{n_s + n_d}$$

Par la suite, Larsen, (2000) a généralisé cette formule pour prendre en compte le temps d'apparition des requêtes. L'auteur a défini une nouvelle formule qu'il a appelée le degré de dynamisme effectif :

$$edod = \frac{\sum_{i=1}^{n_s + n_d} (\frac{t_i}{T})}{n_s + n_d}$$

où $t_i \in [0,T]$ est le temps d'apparition de la requête i.

Le degré de dynamisme effectif représente alors une moyenne du temps de réception des requêtes par rapport au dernier moment possible de réception. On peut facilement voir que $edod$ est compris entre 0 et 1. Plus les requêtes arrivent en retard, plus le $edod$ s'approche de 1 et plus le problème est dynamique. Larsen, (2000) a étendu la définition du $edod$ pour tenir compte des fenêtres de temps si elles sont considérées. Dans ce cas, on suppose que $[a_i, b_i]$ est l'intervalle de temps où la requête i doit être livrée. Le $edod_{tw}$ (tw : réfère *time window* ou fenêtres de temps) est définit comme suit :

$$edod_{tw} = \frac{\sum_{i=1}^{n_s + n_d} (T - (b_i - t_i))/T}{n_s + n_d}$$

Si les fenêtres de temps ne sont pas imposées, on aura $a_i = t_i$ et $b_i = T$ et puis : $edod_{tw} = edod$.

Ainsi, le degré de dynamisme est une valeur qui permet de catégoriser les problèmes en niveaux de dynamisme. D'après Larsen, (2000), les compagnies d'approvisionnement et de distribution appartiennent au niveau dynamique faible. Les entreprises de service et de réparation des appareils électroménagers appartiennent au niveau moyen. Alors que les services d'urgences et les services de taxis ont des niveaux de dynamisme élevés.

1.2.4 Intérêts du DVRP

L'intérêt du DVRP dans les modes de transport actuels devient indiscutable. Toutefois, il y a des domaines plus concernés que les autres. Khouadjia, (2011) a listé quelques applications de ce modèle dans la vie réelle :

- **Les sociétés d'approvisionnement et de distribution :** dans les systèmes gérés par les vendeurs, les sociétés de distribution estiment le niveau de stock des clients de manière à les reconstituer avant l'épuisement des stocks. Généralement, les demandes sont connues à l'avance. Toutefois, certains clients peuvent épuiser leurs stocks et demandent des livraisons urgentes.

- **Services des courriers :** il s'agit des services des courriers express internationaux qui doivent répondre aux demandes des clients en temps réel. La charge est collectée sur différents sites clients et doit être livrée sur un autre site. Le colis à livrer est ramené sur un terminal distant pour être traité et expédié ultérieurement. Les livraisons constituent un problème de routage statique puisque les destinataires sont connus par le chauffeur. Cependant, la plupart des demandes de collecte sont dynamiques, car ni le chauffeur ni le planificateur ne savent où les collectes vont avoir lieu.

- **Entreprises de services de secours et de réparation :** les services de réparation font généralement appel à une entreprise des services publics (secours en cas de panne, électricité, gaz, eau, égouts, etc.) qui répond aux demandes des clients en matière d'entretien ou de réparation. Les demandes de services arrivent de manière dynamique et, généralement, demandent un service urgent.

- **Systèmes Dial-a-Ride :** les systèmes Dial-a-Ride sont principalement utilisés dans les systèmes de transport adaptés à la demande et destinés à servir les petites communautés ou les passagers ayant des besoins spécifiques (personnes âgées, personnes à mobilité réduite). Les clients de ce type peuvent réserver un voyage un jour à l'avance (clients statiques) ou bien faire une demande à court terme (clients dynamiques).

- **Services d'urgence :** ils comprennent les services de police, de lutte contre les incendies et les ambulances. Par définition, ces problèmes sont purement dynamiques puisque tous les clients sont inconnus et arrivent en temps réel. Dans la plupart des cas, les itinéraires ne sont pas formés, car les demandes sont généralement traitées avant l'apparition d'une nouvelle demande. Le problème est alors d'affecter le meilleur véhicule (par exemple, le plus proche de la nouvelle demande). Les méthodes de résolution reposent sur des analyses de localisation permettant de décider où envoyer les véhicules d'urgence en échappant à l'embouteillage.

- **Services de taxi :** la gestion des taxis est encore un autre exemple de problème d'acheminement dynamique dans la vie réelle. Dans la plupart des systèmes de taxi, le pourcentage de clients dynamiques est très élevé par rapport aux clients statiques. C'est-à-dire que très peu de clients sont connus par le planificateur avant que le taxi ne quitte la centrale au début de la journée de travail.

1.3 Taxonomie et revue de littérature

La littérature du DVRP est une littérature assez riche. De ce fait, plusieurs variantes sont traitées,

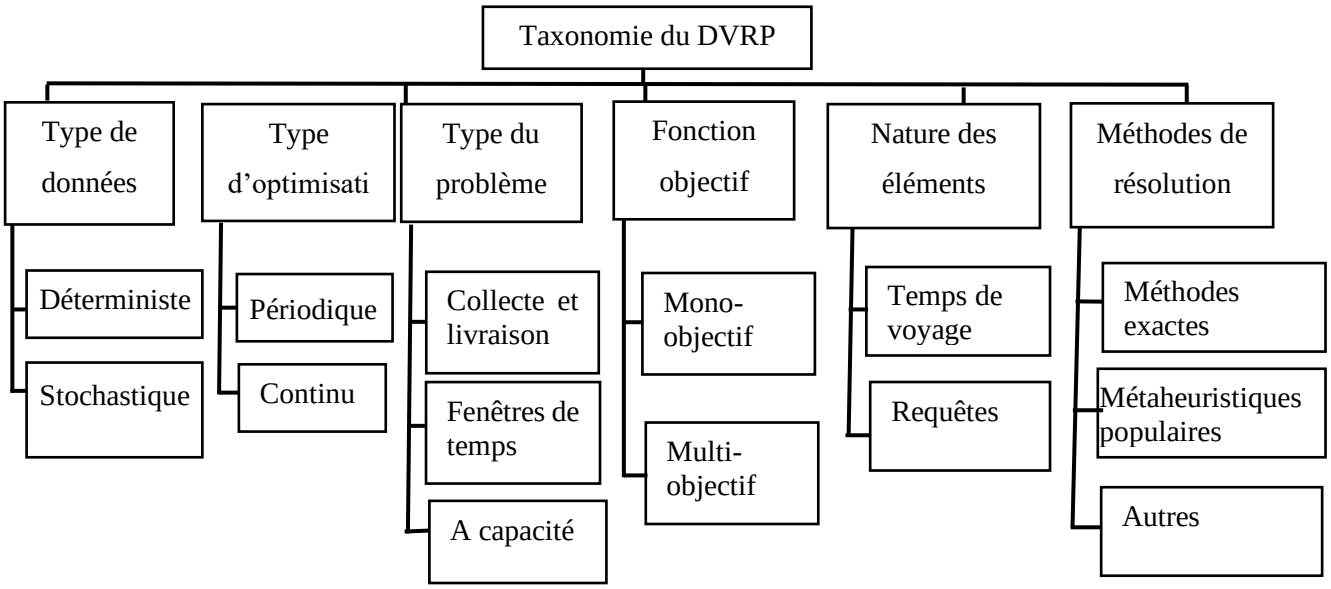

Figure 3 : Taxonomie du DVRP

avec plusieurs approches. En outre, le DVRP est un problème plus général que le VRP conventionnel. Il chevauche avec le VRP stochastique, le VRP déterministe et le VRP avec temps de voyage dépendant. Ceci étant, plusieurs classifications sont possibles pour ce problème. Larsen, (2000) a classifié les problèmes DVRP selon leur degré de dynamisme. Il a parlé de trois niveaux de dynamisme ; le niveau faible, le niveau moyen, et le niveau élevé. Alors que Pillac et al. (2013) ont proposé, quant à eux, une autre classification qui considère la démarche de traitement des requêtes et la nature des données préalablement disponibles. Ils ont, ainsi, catégorisé le DVRP en quatre groupes. Le DVRP déterministe, le DVRP stochastique, le DVRP périodique et le DVRP continu. Récemment Psaraftis, (2016) ont proposé une taxonomie plus précise du DVRP vu qu'elle considère plusieurs façades de ce problème. Ils ont défini onze (11) critères de classifications, à savoir ; le type du problème, le contexte logistique, le mode de transport, la fonction objectif, la taille de la flotte, la contrainte de temps, la contrainte de capacité des véhicules, la possibilité de rejeter des clients, la nature des données dynamiques, la nature stochastique et la méthode de résolution.

Dans cette thèse nous adoptons une classification plus générale, basée sur les critères de (Pillac et al. 2013) et quatre des onze critères utilisés par Psaraftis, (2016). La figure 3 présente les critères de cette taxonomie.

1.3.1 Type de données

Les données utilisées dans l'optimisation d'un DVRP peuvent avoir une nature déterministe, comme elles peuvent comprendre des données stochastiques. Un DVRP est déterministe si toutes ses entrées sont connues avec certitude (ex. : Montemanni et al. (2005) ; Khouadjia et al. (2012)) et aucune de ses entrées n'est stochastique. Réciproquement, le DVRP est stochastique s'il comprend au moins une entrée stochastique. Par exemple, Schyns, (2015) a traité le problème de routage des camions de ravitaillement dans un aéroport, en considérant à la fois les requêtes effectuées et les requêtes probables dans le futur. Quant à Albareda-Sambola et al. (2014), ils ont défini un indice de compatibilité géographique qui permet d'estimer le bénéfice potentiel de reporter le service d'un client lorsqu'un autre, très compatible avec lui, a une grande probabilité d'avoir une demande pour les périodes de temps suivantes. Cette politique permet d'identifier l'ensemble des clients à visiter durant chaque période de temps en se basant sur leurs indices de compatibilité et leurs probabilités de demander le service. Un autre exemple est celui de Yang et al. (2004) qui ont trouvé que la méthode OPTUN qui considère quelques futures requêtes probables, est celle qui a donné les meilleurs résultats parmi cinq autres approches déterministes présentées dans le même article. Une autre implémentation qui exploite les données stochastiques pour guider les véhicules dans les zones où il y a une demande probable (approche proactive) est proposée par Orozco & Barceló, (2012). De leur part Hvattum et al. (2006) ont développé une heuristique dont le principe est d'affecter un ensemble de requêtes prometteuses aux véhicules au début de chaque intervalle dépendamment de leurs fréquences d'apparition dans les scénarios stochastiques possibles. L'algorithme utilise, ensuite, la méthode de Branch-and-Regret pour les fusionner à fin d'avoir une seule solution optimale.

Récemment Ritzinger et al. (2016) ont présenté une revue synthétique sur le DVRP stochastique qui combine entre le caractère dynamique du DVVRP et le caractère stochastique du SPTV.

1.3.2 Type d'optimisation

Pillac et al. (2013) différencie entre deux types d'optimisation, l'optimisation continue et l'optimisation périodique. L'optimisation continue est effectuée tout au long de la journée. La mémoire adaptative (Taillard et al. 2001), dont le rôle est de garder les informations sur les bonnes solutions, est souvent utilisée dans ce genre d'optimisation. Chaque fois que les données disponibles changent, une procédure de décision regroupe les informations de la mémoire pour mettre à jour le

plan de routage actuel. L'avantage est que la capacité de calcul est maximisée, éventuellement au détriment d'une implémentation plus complexe. Il convient de noter que, du fait que les tournées en cours sont sujet de modification à tout moment, les véhicules ne connaissent pas leur prochaine destination avant la fin du traitement de la demande. L'une des premières études exploitant le concept de mémoire adaptative pour effectuer une optimisation continue, est celle présentée par Gendreau et al. (1999).

Quant à l'optimisation périodique, elle commence au début de la journée par une première optimisation qui produit une solution initiale. Ensuite, une procédure d'optimisation résout périodiquement le problème statique correspondant à l'état actuel, soit chaque fois que les données disponibles changent, soit à des intervalles de temps fixes. L'optimisation périodique présente l'avantage de pouvoir s'appuyer sur des algorithmes développés pour le routage statique. Le principal inconvénient est qu'elle doit être effectuée avant la mise à jour du plan de routage. On est obligé, dans ce cas, de minimiser le temps d'exécution des algorithmes d'optimisation. L'approche de Montemanni et al. (2005) est un bon exemple de l'optimisation périodique. Le principe de cette étude est de décomposer la période de planification en sous-périodes. Au début de chaque sous-période, ils relancent l'optimisation d'un nouveau problème. Ce dernier contient les requêtes non encore livrées en plus des requêtes arrivantes durant la période précédente. Plusieurs chercheurs ont utilisé la même démarche dans leurs approches de résolution du DVRP. Khouadjia et al. (2012) ; Mańdziuk & Żychowski, (2016) ; Hanshar & Ombuki-Berman, (2007) et Chen & Xu, (2006) sont tous des exemples de l'optimisation périodique. La décomposition en période de temps n'est pas la seule représentation de l'optimisation périodique définie par Pillac et al. (2013). En effet, les démarches qui lancent une nouvelle optimisation à chaque apparition d'une nouvelle requête forment la seconde façade de l'optimisation périodique. Dans l'une des premières études sur DVRP dynamique, Psaraftis, (1980) a basé son approche de résolution sur ce type d'optimisation continue. L'approche de Schyns, (2015) est un autre exemple de cette optimisation.

1.3.3 Type de problème

Le type de problème réfère aux caractéristiques générales du problème telles que la contrainte de fenêtres de temps, la contrainte de capacité, la nature du service (collecte et livraison par exemple) ou le nombre de dépôts. Ces caractéristiques sont généralement les points de classification ordinaires du VRP conventionnel. Plusieurs types de problèmes sont présents dans la littérature. Dans cette partie, nous présentons quelques exemples pour clarifier la notion.

Le premier exemple est le VRP à capacité (CVRP). Dans ce problème les véhicules ont une capacité de charge limitée et doivent collecter ou livrer des articles à divers clients. Les articles, quant à eux, se caractérisent par une mesure, telle que le poids ou le volume. L'objectif est de livrer ou collecter les articles au moindre coût, sans dépasser la capacité maximale des véhicules. Dans le cas dynamique, on parle du DVRP à capacité (CDVRP) qui a les mêmes caractéristiques du CVRP projetées dans un contexte dynamique (Khouadjia et al. (2012) ; Montemanni et al. (2005) ; Hanshar & Ombuki-Berman, (2007)).

Le problème de collecte et livraison et un autre type, très connu, du VRP. Dans ce problème, on doit livrer des articles à des clients et collecter d'autres. Les sociétés de transport des courriers sont les plus concernées par ce modèle. Dans certains cas, les actions de collecte et de livraison se font chez le même client simultanément. Nous parlons, alors, du VRP avec collecte et livraison (VRPPD) simultanées. À l'égard du CDVRP, le DVRP avec collecte et livraison DVRPPD est l'implémentation dynamique du VRPPD. Benyahia & Potvin, (1998) ; Cheung et al. (2008); Haghani & Jung, (2005) et Hemert & Poutré, (2010) sont tous des exemples des études qui traitent le DVRPPD.

Un autre exemple est le VRP avec fenêtre de temps (VRPTW) introduit par Solomon, (1983). La principale caractéristique de ce problème est la définition des périodes de temps (appelées fenêtres de temps) durant lesquelles les clients doivent être servis. La façade dynamique du VRPDTW est le DVRP avec fenêtre de temps (DVRPTW). Parmi les articles qui étudient cette variante du DVRP, nous citons l'article de Berger et al. (2007), celui de Ghannadpour et al. (2014) et celui de Hong, (2012).

Avant de finir avec les exemples des types du problème, il est important de parler du VRP multi-tour (VRPMT). La particularité de ce type de problème réside dans le fait que les véhicules peuvent effectuer plusieurs tournées pendant la même journée. La version statique du VRPMT est largement étudiée dans la littérature (section 1.4.2). Cependant les recherches sur le DVRP multi-tour (DVRPMT) sont encore peu nombreuses (Ouaddi et al. 2018, 2020). D'ailleurs, c'est la raison qui nous a poussés à étudier ce problème.

1.3.4 Fonction objectif

Dans les premières définitions du VRP, l'objectif était restreint à minimiser la distance parcourue comme étant le facteur principal qui définit le coût de transport (Stewart & Golden, (1984); Gillett & Miller, (1974)). Avec le temps, et en réponse aux exigences du marché, d'autres facteurs ont été inclus dans les objectifs des VRPs étudiés. Dans le cas du DVRP, la littérature présente une diversité

d’objectifs d’optimisation ; tandis qu’il y a des études qui se limitent à optimiser un seul objectif (mono-objectif), d’autres proposent des approches multi-objectifs.

On dit qu’un problème est mono-objectif, lorsqu’il a un seul objectif à optimiser. La solution optimale est, clairement, celle qui optimalise cet objectif. Dans le cas du DVRP, les optimisations mono-objectives considèrent, généralement, la distance totale parcourue comme objectif à minimiser. Khouadjia et al. (2012) ; Montemanni et al. (2005) ; Hanshar & Ombuki-Berman, (2007) et Mańdziuk & Żychowski, (2016) sont des exemples de l’optimisation du DVRP d’un point de vue mono-objectif.

Lorsque le problème d’optimisation a un seul objectif, nous pouvons évaluer facilement les solutions. Par contre, dans le cas multi-objectif, ces derniers peuvent avoir des natures conflictuelles. La difficulté de ce genre de problème réside dans le manque d’une relation d’ordre total qui permet d’évaluer les solutions. En effet, il est possible de trouver une solution meilleure que les autres sur un nombre d’objectifs, mais moins bonne sur le reste. De ce fait, la solution n’est plus une solution optimale unique, mais un ensemble de solutions qui réalise le meilleur compromis entre les objectifs du problème. Ceci nécessite une relation qui peut identifier les meilleurs compromis. La dominance Pareto est la relation la plus utilisée pour atteindre ces compromis. On appelle l’ensemble des meilleurs compromis **l’ensemble du front Pareto**. Les solutions appartenant au front Pareto, appelées aussi points non dominés, réalisent un équilibre, de sorte qu’on ne peut améliorer l’un des objectifs, qu’en dégradant au moins un autre.

Ceci étant, les démarches proposées pour la résolution du DVRP multi-objectif n’adoptent pas toutes la dominance Pareto. En fait, il est possible de classer ces démarches en deux catégories : les approches Pareto et les approches non Pareto. La première catégorie comprend les approches qui réalisent une optimisation parallèle des objectifs en utilisant la dominance Pareto. Par exemple Kaiwartya et al. (2015) utilisent la dominance Pareto pour traiter un DVRP à cinq objectifs. Un autre exemple sur la dominance Pareto pour le DVRP est présent dans l’étude faite par Ghannadpour et al. (2014).

En revanche, les approches non Pareto cherchent à ramener le problème multi-objectif en un ou plusieurs problèmes mono-objectifs. La scalarisation est l’une des techniques utilisées pour effectuer cette manipulation. Il existe plusieurs méthodes de scalarisation. Toutefois, la scalarisation pondérée ou linéaire est la plus présente dans la littérature du DVRP. Son principe est de formuler le problème multi-objectif en un problème d'optimisation mono-objectif et de résoudre le dernier à plusieurs reprises en faisant varier les coefficients de pondération.

Soit $\text{Min}\ (F(x)) = (f_1(x), f_2(x), \dots, f_k(x))\ , x \in D$ le problème d'optimisation multi-objectif. La sacalarisation linéaire consiste à attribuer un poids p_i à chaque objectif f_i et optimiser la somme : $Min\ \sum_{i=1}^{k} p_i * f_i(x)$

Chang et al. (2003) utilisent la scalarisation pondérée pour optimiser un DVRP qui considère la minimisation du temps de voyage et la minimisation du temps d'attente. Quant à Chen et al. (2006), ils ont défini une fonction pondérée de trois objectifs à savoir : le temps de voyage, le temps d'attente avant le service et le temps d'attente avant le départ dans tous les nœuds. De leurs parts Gendreau et al. (2006) considèrent un cas spécifique du DVRP et optimisent le temps de voyage, l'overtime maximal performé, la satisfaction des clients avec une seule fonction objectif pondérée.

De notre part, nous avons présenté des approches non Pareto pour le MTDVRPOT (Ouaddi et al. 2018, 2020). Dans le cas présent, nous avons conçu des relations d'évaluation dédiées.

1.3.5 Nature des éléments dynamiques

Les éléments dynamiques dans un DVRP sont des nouvelles informations qui concernent les requêtes ou le temps de voyage. Ces informations peuvent inclure l'ajout, l'annulation ou la modification d'une requête. L'ajout consiste à insérer une requête, récemment faite, dans le plan de routage du reste de la journée (Montemanni et al. 2005). Tandis que l'annulation consiste à supprimer une requête, non encore servie du plan de routage. Alors que le besoin de modification se manifeste si, par exemple, le client demande de changer la quantité initialement demandée, ou la localisation de son service (Christiansen & Lysgaard, (2007) ; Ozbaygin & Savelsbergh, (2019)). Le deuxième élément dynamique que le DVRP peut considérer est le temps de voyage. Ce cas est plus présent dans les milieux urbains où il est, parfois, difficile de prévoir le temps de parcours des réseaux à cause de la grande congestion surtout pendant les heures de pointe. En fait, les temps de trajet résultent d'un processus dynamique lié à la congestion du trafic. Il est clair que les temps de déplacement dépendent beaucoup du nombre de véhicules occupant la route et de leur vitesse. Ainsi, une route qui réalise la distance minimale n'est pas forcément la route qui assure un temps de voyage optimal. Ceci étant, le but dans un DVRP avec temps de voyage dépendant est de trouver des routes les plus courtes en termes de temps de voyage si les routes qui assurent la distance minimale sont congestionnées. La littérature du DVRP est de plus en plus riche en études qui abordent le temps de voyage dépendant (ex. : Sabar et al. (2019) ; Musolino et al. (2013) ; Yang et al. (2005) et Ichoua et al. 2003)).

1.3.6 Méthodes de résolution

Les approches adoptées dans la résolution des DVRP varient d'une étude à l'autre, selon les contraintes du problème et la nature du DVRP en étude. Les méthodes exactes et les métaheuristiques

sont les bases de la plupart de ces approches. Toutefois, il existe des approches basées sur des heuristiques spécifiées. Dans cette section, nous présentons les démarches de résolution tout en distinguant entre les différentes métaheuristiques, les méthodes exactes et d'autres heuristiques.

1.3.6.1 Approches à base de méthodes exactes

Les méthodes exactes sont des algorithmes qui résolvent un problème d'optimisation à l'optimalité. Une méthode exacte triviale est l'énumération. Son principe est d'énumérer toutes les solutions réalisables afin de choisir la meilleure. Plusieurs méthodes ont été développées pour effectuer des énumérations intelligentes afin de trouver l'optimum en moins de temps. Par exemple, la méthode de séparation et évaluation ou branch-and-bound (Land & Doig, 1960) et la méthode de branch-and-cut (Padberg & Rinaldi, 1991) sont des méthodes exactes classiques qui se basent sur l'organisation intelligente de la recherche pour accélérer sa convergence vers l'optimum. Un autre exemple est la programmation dynamique proposée par Bellman, (1957) dont le principe est de partitionner le problème d'origine et résoudre les sous-parties. Un autre algorithme exact a été proposé par Hwang et al. (1993) basé sur la stratégie « diviser pour conquérir » pour le problème de voyageur de commerce euclidien. Si le problème est modélisé mathématiquement, l'algorithme du simplex développé par Dantzig et al. (1955) ainsi que les méthodes de points intérieurs (Potra & Wright, 2000) sont les plus appropriées pour trouver la solution optimale. Ceci étant, l'efficacité d'une méthode exacte reste, généralement, limitée aux instances de petite taille. Dans le reste de cette sous-section, nous présentons les approches qui utilisent des méthodes exactes pour résoudre le DVRP. Bien qu'elles soient basées sur des méthodes exactes, ces approches ne sont pas toutes des algorithmes exacts dans le sens qu'elles ne garantissent pas l'optimalité.

L'une des premières études sur le DVRP est celle de Psaraftis, (1980). Il a adopté une approche de programmation dynamique pour résoudre le problème dial-a-ride en temps réel. Son objectif était de trouver le meilleur itinéraire à chaque occurrence d'une nouvelle demande. Il a d'abord résolu le problème statique. Ensuite, il a étendu l'algorithme du cas statique pour inclure de nouvelles requêtes.

Christiansen & Lysgaard, (2007) traitent un DVRP à capacité avec des demandes stochastiques. Ils proposent une approche basée sur la résolution d'une formulation partitionnée. Le sous-problème de génération de colonnes associé a été résolu en utilisant un algorithme de programmation dynamique qui utilise une procédure de branchement pour garantir l'intégralité.

En devisant la journée de travail en périodes de temps, Yang et al. (2004) ont proposé deux approches basées sur la ré-optimisation des nouvelles instances du problème statique à l'aide du CPLEX. À base du même principe de décomposition, Chen & Xu, (2006) traitent un DVRP avec

fenêtres de temps et flotte infinie. Durant chaque période de temps, ils génèrent dynamiquement un ensemble de colonnes correspondant aux tournées de la période suivante. À l'étape de décision, CPLEX résout un problème linéaire binaire dont les variables indiquent si ces colonnes (tournées) feront partie du plan suivant ou non. La solution ainsi obtenue est pseudo optimale, car les colonnes générées ne couvrent pas toutes les tournées possibles. Cependant son temps d'exécution est assez court en comparaison avec les méthodes exactes.

Récemment, Ozbaygin and Savelsbergh (2019) ont proposé une approche basée sur la méthode Branch-and-price et la génération de colonnes pour résoudre le DVRP avec des lieux de livraison en itinérance. Dans ce problème, les emplacements des clients peuvent changer. Cela implique une modification des plans de routage initialement conçus. La méthode proposée donne un nouveau plan de routage optimal et adapté à chaque fois qu'un client révèle le changement de son emplacement de service.

Sur un autre plan, plusieurs études, qui traitent le DVRP avec temps de voyage dépendent se sont servies de l'algorithme de Djikstra dans leurs approches. Bartlett et al. (2014) ont traité le problème de routage dynamique des pièces dans l'industrie des grands engins. Ils ont modélisé le plan de transport en tant que graphe dirigé avec des poids sur les arcs. L'article a défini une formule permettant de calculer le temps de passage sur les arcs du réseau et puis appliquer l'algorithme de Djikstra pour trouver le plus court chemin. Yousefi et al. (2014) ont discuté la possibilité de changer l'itinéraire du véhicule durant l'exécution selon l'état de congestion du réseau. Une architecture système, contenant des liaisons inter nœuds et inter véhicules, est proposée pour collecter les informations sur l'état actuel du réseau. Pour rediriger les véhicules vers des routes moins congestionnées, l'article a proposé deux démarches à base de l'algorithme de Djikstra qui permettent de proposer aux véhicules la bonne route. La première, appelée one-step, génère une seule solution alors que la deuxième, appelée step-by-step, propose une route à chaque intersection selon les mises à jour du trafic.

De notre part, nous avons proposé une heuristique basée sur l'application itérative d'une méthode exacte pour résoudre le MTDVRPOT. Les détails de cette approche sont présentés dans le chapitre 2 de ce rapport.

1.3.6.2 Approches basées sur des métaheuristiques

Si les méthodes exactes n'arrivent pas à résoudre un problème, alors on fait appel aux méthodes approchées. Ces dernières se composent de deux catégories : les heuristiques spécifiques et les métaheuristiques. Les heuristiques sont des procédures, dépendantes du problème à résoudre, qui

permettent de naviguer intelligemment dans l'espace de recherche, afin d'obtenir une solution pseudo-optimale, dans un temps raisonnable. Les métaheuristiques sont, aussi, des heuristiques, mais elles sont évoluées et plus générales de telle sorte qu'on peut les appliquer à une grande variété de problèmes d'optimisation (Talbi, 2009 ; Labadie et al. 2016). Beaucoup de métaheuristiques sont inspirées de phénomènes naturels. Par exemple, le système de colonies de fourmis imite le comportement collectif des fourmis pour trouver le plus court chemin entre leur nid et la source de nourriture. Un autre exemple est celui des algorithmes évolutionnistes inspirés de la théorie de l'évolution.

Dans cette section, nous allons présenter les approches de résolution proposées pour le DVRP et basées sur des métaheuristiques.

- **Les algorithmes évolutionnistes (EA)**

Les algorithmes évolutionnistes sont définis par Dasgupta & Michalewics, (1997) comme étant des procédures de recherches, à usage général, basé sur la sélection naturelle et la génétique des populations. L'algorithme génétique, l'algorithme mémétique, la programmation évolutionniste, les stratégies d'évolution et la programmation génétique sont tous des algorithmes évolutionnistes.

Plusieurs études utilisent l'algorithme génétique pour résoudre le DVRP. Ces études focalisent dans leur grande partie sur le DVRP dans le domaine de transport des courriers qui se caractérise par le service de collecte et de livraison. L'un des premiers travaux sur le DVRP à base de l'algorithme génétique (GA) est celui fait par Benyahia & Potvin, (1998). Dans cet article, chaque véhicule est associé à un vecteur de valeurs d'attribut reflétant l'effet d'insertion de nouvelles demandes de service dans sa route actuelle. Grâce à cette description d'attribut, une fonction d'utilité visant à rapprocher le processus de décision d'un répartiteur humain expert se construit à travers la programmation génétique. Cette dernière est caractérisée par le codage de solutions sous forme de structure d'arbre. Les étapes d'évolution des solutions (chromosome) ressemblent aux étapes de l'algorithme génétique. Les résultats des calculs sont présentés sur les demandes de service recueillies à partir d'une société de services de messagerie. La comparaison est faite avec un modèle de réseau neuronal et une politique de répartition simple.

Sur un problème semblable Cheung et al. (2008) ont aussi appliqué leur propre implémentation de l'algorithme génétique. Cette fois, le problème est caractérisé par des fenêtres de temps dures où la collecte et la livraison des courriers doivent se faire. Pour le cas statique, on génère initialement un ensemble de solutions faisables, à l'aide de quelques procédures d'insertion, de fusion des routes et de raffinement, dédiées. Ces solutions feront l'objet, ensuite, d'entrées (chromosomes) pour l'algorithme génétique. Un chromosome est composé de plusieurs gènes qui sont des combinaisons

de points de collecte et de livraison de chaque requête avec le véhicule affecté pour le service. Les changements dynamiques provenant durant la journée comportent les nouvelles requêtes de service ainsi que les changements possibles dans le temps de voyage. Pour répondre à ces variations, l'article propose une heuristique à base des mêmes procédures utilisées dans l'algorithme statique. En résultats, plus on a des fenêtres de temps large, plus l'algorithme de ré-optimisation dynamique est efficace. Dans le même contexte, Haghani & Jung (2005) ont proposé une solution à base du GA pour le problème de transport des courriers dynamiques, sauf que cette fois la livraison des courriers collectés ne se fait que dans la période de planification (journée) suivante. Ainsi les requêtes dynamiques comprennent uniquement les nouvelles demandes de collectes qui seront livrées le lendemain. La performance de l'algorithme génétique est évaluée en comparant ses résultats avec des méthodes exactes et une autre méthode de minoration, proposée dans le même article, sur des tests générés aléatoirement.

Toujours avec l'algorithme génétique, mais cette fois sur une autre variante du DVRP, Hanshar & Ombuki-Berman, (2007) ont adopté la démarche périodique déterministe qui décompose la journée de travail en intervalles de temps. Ils proposent une solution basée sur le GA pour le DVRP à capacité. La contribution principale dans leur travail a été la façon dont ils représentent un chromosome dans les optimisations dynamiques.

La démarche périodique déterministe a été adoptée, aussi, par Mańdziuk & Żychowski, (2016) qui proposent une solution basée sur l'algorithme mémétique (MA) pour le DVRP à capacité. De notre part, nous avons proposé une approche à base de l'algorithme mémétique (Ouaddi et al. 2020) pour résoudre le DVRP multi-tour avec overtime. Les détails de ce travail seront présentés dans le chapitre 4 de ce rapport.

Hemert & Poutré, (2010) ont introduit la notion de régions fructueuses, qui sont des pôles, où il y a plus de clients potentiels probables. Le problème traité cette fois est un problème de collecte de charges depuis des clients et de livraison à un seul dépôt central. Les requêtes de collectes arrivent dynamiquement durant la journée. Un algorithme évolutionniste, qui n'utilise pas l'opérateur de croisement, tourne en alternance avec un simulateur du problème de routage. Les sorties de ce dernier sont les entrées de l'EA ce qui permet de répondre plus efficacement aux demandes de service. En résultats, l'article donne des points de transition où l'utilisation des régions fructueuses peut être utile.

Une étude plus récente faite par Barkaoui et al. (2015) vise à étudier l'impact de l'exploitation de futures requêtes potentielles dans la génération des plans de routage dans un environnement dynamique afin de maximiser la satisfaction des clients. L'évaluation de la stratégie proposée a été réalisée à l'aide d'une hybridation de l'algorithme génétique précédemment conçu par Berger et al.

(2007) pour le DVRP avec fenêtres de temps. La modification apportée prend en compte la satisfaction des clients qui se calcule sur plusieurs visites successives du même client.

- **L'algorithme de Recherche Tabou (TS)**

Depuis sa première introduction par Glover, (1986), l'algorithme de recherche tabou a eu de nombreuses applications dans la recherche opérationnelle de façon générale et sur les VRPs de façon particulière. Le DVRP a été de sa part un champ fertile pour différentes implémentations de cette métaheuristique. L'utilisation d'une mémoire adaptative est une caractéristique qui marque beaucoup de travaux à base du TS. L'algorithme proposé par Gendreau et al. (1999) était la base de plusieurs approches qui combinent la mémoire adaptative avec le TS pour résoudre le DVRP. Il a été utilisé par Ichoua et al. (2000) avec une petite modification qui permet de considérer la destination courante du véhicule comme partie du nouveau problème à optimiser. Tandis que le problème d'origine considère les destinations courantes des véhicules comme dépôt fictif dans le nouveau problème, la modification apportée traite ces destinations comme clients en considérant la position actuelle dans la route comme début pour le nouveau problème. Cette façon de voir permet de trouver des solutions meilleures dans le cas où il y a de nouvelles requêtes plus proches à un véhicule que sa destination courante. Ichoua et al. (2003) ont adapté la même approche au DVRP avec temps de voyage dépendant. Dans cette version du problème, les requêtes sont stables, mais le temps de parcours des liens du réseau dépend de la période de la journée ou le voyage est fait.

De leurs parts, Chang et al. (2003) ont appliqué l'algorithme TS sur le DVRP avec fenêtre de temps, collecte et livraison. Tandis que Attanasio et al. (2004) et Beaudry et al. (2010) ont adopté le TS au problème Dial-a ride dynamique. Quant à Liao & Hu, (2011), ils ont utilisé l'algorithme TS pour tenir compte des variations courantes dans le temps de parcours des liens du réseau.

- **L'algorithme de Recherche au Voisinage (NS)**

La Recherche par Voisinage est une métaheuristique pour la résolution de problèmes d'optimisation combinatoire et globale, dont l'idée de base est le changement systématique de voisinage au sein d'une recherche locale. Un nombre d'études récentes se réfère à cette métaheuristique pour donner des solutions aux DVRP. Parmi ces études, l'article d'Azi et al. (2012) dont l'idée est de fixer les routes, dès le départ, des véhicules depuis le dépôt et la décision à prendre pour les requêtes dynamiques concerne uniquement leurs acceptations ou leur rejet selon leur proximité des routes déjà planifiées. À cette fin, les auteurs ont développé une adaptation de l'algorithme de recherche par voisinage large qui exploite les prévisions sur les demandes pour créer une population de futurs scénarios probables à base desquels les routes seront créées. L'approche d'Albareda-Sambola et al. (2014) expliquée dans la section 1.3.1 est, aussi, basée sur le NS. Une autre

démarche à base de cet algorithme a été proposée par Hong, (2012) pour résoudre le DVRP avec fenêtres de temps dures. Il utilise le mécanisme retirer-réinsérer de cet algorithme pour insérer les nouvelles requêtes dans les routes déjà planifiées. Gendreau et al. (2006) ont proposé un algorithme NS pour résoudre un DVRP avec collecte, livraison et fenêtre de temps. L'utilisation de l'overtime est tolérée et la contrainte de capacité est non considérée. Pour réagir aux nouvelles requêtes, les auteurs ont adopté une démarche qui combine la recherche au voisinage et la mémoire adaptative, dans une vision continue et déterministe. Lin et al. (2014) utilisent une heuristique appelée Impact (Loannou et al. 2001) qui permet de calculer l'impact de l'insertion d'un nouveau client entre deux clients appartenant déjà à la route, en combinaison avec le NS pour résoudre le DVRP dans le domaine de transport des courriers. D'autres adaptations du NS au DVRP ont été proposées par Mostepha R. Khouadjia et al. (2012) et par Messaoud et al. (2013).

- **Le système de colonie de fourmis (ACS)**

À notre connaissance, la première application du système de colonie de fourmis sur le DVRP est due à Montemanni et al. (2005). Des travaux précédents ont appliqué l'ACS au problème de voyageur de commerce dynamique tel que le travail de Guntsch & Middendorf, (2001). Dans leur article, Montemanni et al. (2005) ont décomposé la période de planification en sous-périodes. À la fin de chaque période de temps, ils lancent l'optimisation du problème contenant les requêtes non encore livrées avec les nouvelles requêtes collectées durant cette période. L'article applique, chaque fois, le même algorithme avec une modification du taux de phéromone initiale sur les arcs du réseau. Une approche similaire a été utilisée par Gambardella et al. (2000) dans l'implémentation des outils logiciels DyvOil et Antroute.

Dernièrement, Mavrovouniotis & Yang, (2015) ont utilisé l'ACS pour développer un générateur d'optimum dynamique pour le DVRP appelé le DBGP. De sa part, Schyns, (2015) a traité le DVRP avec fenêtres de temps dures, livraison partielle et flotte hétérogène. L'objectif est d'optimiser le routage des camions de ravitaillement dans un aéroport. Le problème est caractérisé par la non-visibilité de la période de planification. Ainsi l'optimisation se fait à base de la situation actuelle des camions et leurs capacités restantes en considérant à la fois les requêtes valables et celles probables dans le futur. L'optimisation se fait à chaque apparition d'une nouvelle donnée. L'article propose une adaptation du système de la colonie de fourmis, de Montemanni et al. (2005), pour prendre en considération les contraintes du problème en cours. De notre part, nous avons adopté une démarche basée sur l'ACS de Montemanni et al .(2005) pour résoudre le DVRP multi-tour et multi-objectif. Cette contribution (Ouaddi et al. 2018) sera décrite en détail dans le chapitre 3.

- **L’optimisation par essaim de particules (PSO)**

L'optimisation par essaims des particules est une métaheuristique introduite par Kennedy & Elberhart, (1995) inspirée du comportement des oiseaux dans leur déplacement collectif. Cette méthode se base sur la collaboration des individus entre eux. Elle a, d'ailleurs, des similarités avec l’algorithme de colonies de fourmis, qui s'appuient lui aussi sur le concept d'auto-organisation. L’idée vient du fait qu'un groupe d'individus peu intelligents peut posséder une organisation globale complexe dont le but est d’atteindre un objectif collectif.

Mostepha R. Khouadjia et al. (2012) ont proposé une démarche qui combine la décomposition en intervalle de temps avec la mémoire adaptative pour adapter la méthode d’optimisation par essaim de particules au CDVRP. Dans la méthode d’origine, chaque particule est définie par sa position actuelle et sa meilleure position, ce qui nécessite l’existence d’une mémoire pour sauvegarder les états des particules. L’adaptation proposée ici consiste à utiliser les informations réunies précédemment de l’espace de recherche pour permettre à l’algorithme de commencer la recherche depuis la meilleure solution trouvée. L’optimisation par essaim de particules a été utilisée dans une démarche périodique déterministe, aussi, par Kaiwartya et al. (2015), pour résoudre le DVRP multi-objectif.

1.3.6.3 D’autres méthodes

Dans cette section, nous allons présenter des approches intéressantes, mais qu’on n’a pas pu intégrer parmi les méthodes exactes ou les métaheuristiques. Parmi les articles les plus cités dans la littérature du DVRP, l’étude faite par Bent & Hentenryck, (2004), qui propose une approche à plans multiple (MPA) et une autre approche à scénario multiple (MSA). Le MPA est une généralisation de l’algorithme TS (Taillard et al. 1997). Cette méthode génère constamment des plans de routage conforme aux décisions à prendre et supprime ceux incompatibles en précisant un plan dit distingué et cohérent avec les autres plans pour garantir le service des requêtes dynamique. Pour choisir ce plan, les auteurs ont défini une fonction de consensus permettant de choisir le plan le plus similaire aux autres. De sa part, le MSA est une généralisation prédictive du MPA. Tandis que le MPA ne tient compte que des requêtes connues ou déterministes, le MSA considère l’aspect probabiliste du problème en intégrant les futures requêtes probables qui s’obtiennent par échantillonnage de leurs distributions probabilistes. L’algorithme les projette par la suite dans le plan distingué pour générer une solution qui optimise leurs insertions en cas de leurs réalisations. Pillac et al. (2012) ont proposé une implémentation orientée événement du MSA appelée jMSA. Ce framework permet une meilleure portabilité de l’algorithme MSA afin de faciliter son déploiement dans les systèmes d’aide à la décision, consacrée au routage dynamique. Une autre adaptation du MSA au domaine maritime a été proposée par Tirado et al. (2013).

Hu, (2001) a développé un framework d'évaluation du trafic, à base de la simulation, qui permet de tenir compte des informations en temps réel sur l'état du réseau. Tandis que Van et al. (2008) ont adopté une approche à base de la théorie de la file d'attente pour résoudre le problème de tournées de véhicules avec temps de voyage dynamique à cause de la congestion du réseau. Musolino et al. (2013) ont proposé un framework à quatre procédures pour dessiner les routes des véhicules d'urgence : 1) La procédure de l'affectation reverse (RA) et l'affectation dynamique (DA) qui reçoit des informations courantes sur la performance du réseau pour ajuster les paramètres du modèle d'affectation dynamique jusqu'à ce que les sorties du modèle soient le plus proche des informations en ligne. 2) La procédure de l'affectation dynamique qui simule les interactions entre les flux des demandes et les données réseau et fournit une estimation de la variation du temps de voyage des véhicules sur le réseau. 3) La procédure de prédiction du temps de voyage reçoit les variations de la durée de voyage et fournit une fonction calibrée continue de voyage. 4) La procédure du problème de tournées de véhicules dynamique qui reçoit le flux de l'information concernant les véhicules d'urgence et la fonction calibrée continue et fournit la route optimale pour les véhicules d'urgence.

1.4 VRP statique : revue de littérature

De façon générale, les DVRPs peuvent être modélisés en série de problèmes VRP statiques avec des caractéristiques différentes. Par exemple, le DVRP à capacité et un seul dépôt avec flotte homogène, peut être décomposé en série de problèmes VRP statiques à capacité. Cependant la flotte homogène et l'uni-dépôt ne caractérisent que le premier problème. Alors que les autres problèmes auront des flottes hétérogènes et plusieurs points de départ. Si nous appelons, ces points de départ, des dépôts, nous aurons des problèmes multi-dépôts. C'est le cas du MTDVRPOT auquel nous nous intéressons dans cette thèse. De plus le MTDVRPOT est un problème multi-tour et multi objectif. Afin d'avoir une revue de littérature englobante, nous allons présenter, dans cette section, les travaux concernés par le VRP statique à capacité, avec flotte hétérogène, multi-tour, multi-dépôt et multi-objectif. Pour une lecture plus approfondie, Laporte, (2009) ; Cordeau et al. (2007) et Eksioglu et al. (2009) présentent des études plus riches sur la littérature du VRP statique.

1.4.1 VRP à capacité

Golden et al. (1977) ont donné la formulation mathématique de ce problème dans le cas d'un seul dépôt et ils l'ont appelé VRP générique :

$$min \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} \sum_{k=1}^{NV} d_{ij} x_{ij}^{k} \qquad (1)$$

Sous les contraintes suivantes

$$\sum_{\substack{i=1 \\ i\neq j}}^{N}\sum_{k=1}^{NV} x_{ij}^{k} = 1 \qquad j: 2\ldots N \tag{2}$$

$$\sum_{i=}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N} q_i x_{ij}^{k} \leq Q_k \qquad k: 1..NV \tag{3}$$

$$\sum_{\substack{i=1 \\ l\neq i}}^{N} x_{li}^{k} = \sum_{\substack{j=1 \\ l\neq i}}^{N} x_{il}^{k} \quad , \qquad l: 1..N, k: 1..NV \tag{4}$$

$$\sum_{\substack{j=2 \\ j\neq 1}}^{N} x_{1j}^{k} \leq 1 \qquad k = 1,..,NV \tag{5}$$

$$\sum_{i=}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N} t_i^k x_{ij}^k + \sum_{i=}^{N}\sum_{\substack{j=1 \\ j\neq i}}^{N} t_{ij}^k x_{ij}^k \leq T_k \tag{6}$$

$$X \in S \tag{7}$$

$$x_{ij}^{k} \in \{0,1\} \quad k: 1 \ldots NV \ , i: 1 \ldots N, j: 1 \ldots N, i \neq j \tag{8}$$

Avec

V : Ensemble des clients

N : Nombre de clients

NV : Nombre de véhicules

Q_k: Capacité du véhicule k

q_i : Demande du client i

T_k : Temps maximal permis pour une tournée effectuée par le véhicule k

t_i^k: Temps nécessaire au véhicule k pour livrer ou récupérer au client i

t_{ij}^k: Temps de voyage du client i vers le client j par le véhicule k

d_{ij} : Distance minimale entre i et j

$x_{ij}^k = \begin{cases} 1 \ \ si\ l'arc\ (i,j) est\ traversé\ par\ le\ vehicule\ k \\ o \ \ sinon \end{cases}$

$x_{ij} = \sum_1^{NV} x_{ij}^k$

$S = \{(x_{ij}) : \sum_{i\in Q}\sum_{j\notin Q} x_{ij} \geq 1, \ \forall \mathrm{QCV}\}$

L'objectif (1) est de minimiser la distance totale parcourue (1). L'équation (2) garantit que chaque client doit être servi une et une seule fois. L'inégalité (3) s'assure que la demande totale des clients servis par le véhicule k ne dépasse pas la capacité de ce véhicule. La conservation du flux est garantie

par l'équation (4). L'équation (5) pour garantir que chaque véhicule effectue au maximum une tournée. Alors que l'inégalité (6) traduit le fait que la somme du temps de service et le temps voyage d'un véhicule ne doit pas dépasser le temps maximal permis pour ce véhicule. L'équation (7) élimine les sous-tours. Finalement (8) dit que la variable x_{ij}^{k} est binaire.

La version la plus classique du VRP appelée généralement VRP à capacité (CVRP), est un cas spécifique de cette formulation. Il s'agit d'un problème de livraison à partir d'un seul dépôt vers un ensemble de clients à l'aide d'une flotte de véhicules homogène (c'est-à-dire, ils ont la même capacité de charge et roulent avec la même vitesse). Plusieurs travaux se sont intéressés à cette variante (ex : Taillard, (1993) ; Eydi & Javazi, (2012) ; Stewart & Golden, (1984) ; Gillett & Miller, (1974) ; Gendreau et al. (2008)). Ceci étant, la contrainte de capacité est omniprésente dans la littérature du VRP. Néanmoins, Letchford & Eglese, (1998) et Jaillet, (2008) sont des contre-exemples, parce qu'ils ne considèrent pas la capacité parmi les contraintes.

1.4.2 VRP multi-tour (VRPMT)

Cette variante notée dans la littérature par VRPMT (Vehicle Routing Problem with Multiple Trips) a été introduite pour la première fois par Fleischmann, (1990). Contrairement au modèle classique, dans cette variante un véhicule peut réaliser plusieurs tournées durant la même journée de travail. Taillard et al. (1996) ont proposé une méthode à base de l'algorithme de recherche tabou pour résoudre le VRPMT en permettant l'utilisation d'un temps supplémentaire, travaillé en plus des heures normales de travail (overtime), avec payement de pénalité. Le but est de minimiser la distance totale parcourue ainsi que la pénalité sur l'overtime. Pour tester leur méthode, ils ont proposé des benchmarks avec une flotte restreinte et un temps limité. Ensuite, plusieurs, travaux se sont intéressés à ce problème sous sa forme définie par Taillard et al. (1996) et ont utilisé ses benchmarks pour valider leurs approches. Brandão & Mercer, (1997) font partie des premiers à adopter ces benchmarks pour leurs tests. Ils ont traité un cas réel avec une flotte hétérogène, fenêtres de temps, possibilité de réutiliser les véhicules et tolérance de l'overtime. Le problème est multi-objectif et il a été résolu via un algorithme TS à trois phases. Sur les mêmes benchmarks, Petch & Salhi, (2003) ont testé leur heuristique multi-phase. Dans, la première phase, on utilise l'algorithme de gain généralisé de Yellow, (1970), pour résoudre le VRP. Une deuxième phase de recherche locale est appliquée ensuite sur les solutions de la première phase pour générer les solutions du VRPMT. La dernière phase consiste à construire de nouvelles solutions pour compléter le nombre requis de solutions via le partitionnement des tournées. La principale différence dans ce problème par rapport aux précédents est que la fonction objectif inclut la minimisation de l'overtime maximal effectué au lieu de minimiser l'overtime total. C'est-à-dire : $min\left\{\underset{k=1..m}{max}\ OT_k\right\}$ avec OT_k est l'overtime réalisé par le véhicule k. Un

peu plus tard Salhi & Petch, (2007) ont proposé une approche à base de l'algorithme génétique pour le VRPMT, en plus d'une heuristique dédiée pour le problème d'affectation des tournées aux véhicules (bin packing). Olivera & Viera, (2007) ont présenté une nouvelle modélisation mathématique. Il s'agit de la formulation par chemin. Dans cette formulation, on suppose qu'on dispose de toutes les routes possibles. Chaque route correspond à une tournée faisable. L'étude propose une approche à base de l'algorithme de recherche combinée avec la mémoire adaptative. Ils ont aussi utilisé les benchmarks de Taillard et al. (1996) pour les tests. Une autre formulation mathématique a été proposée, ensuite, par Mingozzi et al. (2013). Cette fois les auteurs considèrent la liste des plannings possibles. Ils ont défini un planning par l'ensemble des tournées effectuées par un seul véhicule. Quant à la résolution, les auteurs ont utilisé une méthode exacte à base de la relaxation lagrangienne. Ils l'ont testé sur 52 instances des benchmarks de Taillard et al. (1996) avec un maximum de 120 clients. Ils ont réussi à résoudre 42 instances en atteignant l'optimalité. Prins, (2002) a travaillé sur le VRPMT avec flotte hétérogène. Il a traité un cas réel multi-objectif. Le premier objectif consiste à minimiser la durée totale de distribution et le second consiste à minimiser le nombre de camions utilisés. Pour résoudre ce problème, l'étude propose une nouvelle heuristique dédiée, plus des adaptations de plusieurs heuristiques connues. La nouvelle heuristique a pu donner des résultats satisfaisants. Ayadi & Benadada, (2013) ont également développé un algorithme mémétique pour le VRPMT qui tolère l'overtime. Ils ont proposé deux formulations (par arc et par chemin). Comme ils ont adopté une représentation des chromosomes multiligne. Les tests ont été effectués sur le benchmark de Taillard et al. (1996). Seixas & Mendes, (2013) quant à eux, ils ont proposé un algorithme de génération de colonne, une heuristique constructive et une approche basée sur le TS pour résoudre leur VRPMT avec des fenêtres temporelles. Récemment, Cheikh et al. (2015) ont utilisé une approche à base de l'algorithme de recherche par voisinage pour résoudre le VRPMT. L'algorithme était capable de trouver un nombre minimal de solutions non réalisables en comparaison avec d'autres études de la littérature.

1.4.3 VRP avec flotte hétérogène (HVRP)

Depuis son introduction par Golden et al. (1984), le VRP avec flotte hétérogène a pu être un champ fertile de recherches. Ce problème dont l'abréviation connue est HVRP considère généralement une flotte limitée ou illimitée de véhicule avec capacité limitée et un coût fixe d'utilisation. De façon générale, son objectif est de déterminer la composition de la flotte ainsi que les tournées des véhicules. Il y a deux types majeurs du HVRP : (1) « Fleet size and Mix Vehicle Routing Problem » (FSM) introduit par Golden et al. (1984) et traite une flotte illimitée, et (2) « Heterogeneous Fixed Fleet Vehicle Routing Problem » introduit par Taillard, (1999) dans lequel la flotte est limitée. Récemment, Koç et al. (2016) ont classifié ce problème en cinq variantes principales :

1) FSM avec coût de véhicules fixe ou variable. Ce problème a été introduit par Ferland & Michelon, (1988) et noté FSM (F, V).
2) FSM avec coût de véhicules fixe, introduit par Golden et al. (1984) et noté FSM(F)
3) FSM avec coût de véhicules variable, introduit par Taillard, (1999) et noté FSM(V)
4) HF avec coût de véhicules fixe ou variable, introduit par Li et al. (2007) et noté HF (F, V)
5) HF avec coût de véhicule variable introduit par Taillard, (1999) et noté HF (V)

Les études de Fleischmann, (1990) ; Brandão & Mercer, (1997) ; Prins, (2002) et Seixas & Mendes, (2013) (décrites dans la section 1.4.2) se sont intéressées au VRPH multi-tour.

1.4.4 VRP multi-dépôt

Comme son nom l'indique, la principale caractéristique d'un VRP multi-dépôt (VRPMD) est le fait qu'il y en a plusieurs dépôts au lieu d'un seul. Pour ne pas s'éloigner du problème initial, nous allons nous concentrer, dans cette revue, sur le VRPMD multi-tour, ou, avec une flotte hétérogène. Plusieurs études se sont intéressées au VRPMD avec une flotte hétérogène. Cependant, elles sont très rares dans le cas du VRPMD multi-tour. Salhi & Sari, (1997) ont traité le premier cas. Leur problème est caractérisé par un nombre fixe de dépôts ayant une capacité illimitée, un nombre limité de types de véhicules, mais avec un nombre illimité de véhicules de chaque type. Les véhicules ont une capacité connue et un coût fixe ou variable (FSM (F,V)). L'objectif est de déterminer la composition du parc ainsi que les visites des véhicules à un coût minimal. Le même problème a été abordé par Said et al. (2014) en utilisant un algorithme de recherche à voisinage variable et par Bolaños et al. (2018) en utilisant un algorithme génétique modifié. Xu et al. (2012) ont examiné le cas d'un VRPMD avec une flotte hétérogène et fenêtres de temps. Quant à Dondo & Cerdá, (2007) et Irnich, (2000), ils ont examiné les VRPMD avec une flotte hétérogène, collecte et livraison et des fenêtres temporelles. Cependant, la seule étude où nous avons trouvé un problème VRPMD multi-tour est celle de Levy et al. (2014). Dans cette étude qui considère également une flotte hétérogène, l'objectif est de générer des tournées pour satisfaire les contraintes de livraison de carburant tout en permettant aux véhicules de faire le plein à partir de n'importe quel dépôt en cas de besoin. L'article utilise deux approches basées sur l'algorithme de recherche au voisinage.

1.4.5 VRP multi-objectif (VRPMO)

Parmi les premières études du VRPMO, on trouve celle de Gambardella et al. (1999). Le problème qu'ils ont traité est un VRP avec fenêtre de temps (VRPTW). Le premier objectif est de minimiser le nombre de véhicules utilisés alors que le second objectif consiste à minimiser la distance totale

parcourue. Les auteurs ont utilisé un système de colonie de fourmis avec deux colonies. Chaque colonie minimise un seul objectif tout en coopérant entre elles par échange d'information via la mise à jour des traces de phéromone. En se basant sur une approche similaire Pellegrini et al. (2007) ont résout un VRP riche dont l'objectif est de minimiser à la fois la distance totale parcourue et le temps d'attente. Quant aux Chávez et al. (2016), ils ont adopté l'algorithme de colonie de fourmis Pareto proposé par Doerner, (2004) pour l'optimisation de la sélection des portefeuilles au VRP multi-dépôt avec collecte et livraison. Sur un VRPTW, Ombuki et al. (2002) ont proposé une approche à deux phases. Dans la première phase, ils utilisent l'algorithme génétique pour minimiser le nombre de véhicules utilisés, ensuite ils appliquent un algorithme de recherche tabou local en vue de minimiser la distance totale parcourue. Toujours avec le VRPTW Ombuki et al. (2006) ont proposé un algorithme génétique multi-objectif. La principale différence de cet article par rapport aux précédents est le fait de considérer les deux objectifs simultanément en utilisant le classement Pareto des individus de l'algorithme génétique (Fonseca & Fleming, 1993). L'algorithme génétique et sa fonctionnalité multi-objectif ont été utilisés, aussi, par Ghoseiri & Ghannadpour, (2010) pour résoudre le même problème, décrit précédemment. Les auteurs ont, aussi, proposé une formulation mathématique par objectif pour ce problème. De leurs parts Zhou et al. (2013) ont adopté cette démarche multi-objectif de l'algorithme génétique pour la résolution de VRPTW sauf que les objectifs visés cette fois sont la distance totale parcourue et l'équilibre de la distance parcourue par rapport à la flotte active. Pour des objectifs similaires Melián-Batista et al. (2014) ont utilisé la métaheuristique de recherche dispersée (scatter search metaheuristic), introduit par Glover et al. (2000), pour résoudre le VRPTW d'une société réelle. Sur un autre cas réel avec données floues et stochastiques Gupta et al. (2010) ont aussi appliqué l'algorithme génétique multi-objectif. À côté de la minimisation de la distance et la taille de la flotte utilisée, cette étude considère la maximisation de la satisfaction des clients et la minimisation du temps d'attente comme objectifs.

1.5 Conclusion

Dans ce chapitre, nous avons présenté les notions indispensables pour comprendre le DVRP. Nous avons, également, présenté les classifications proposées pour ce problème. A base de quelques critères adoptés par ces classifications, nous avons présenté une taxonomie à cinq critères. Dans le dernier paragraphe de ce chapitre, nous avons présenté une revue de littérature du VRP statique en considérant, uniquement, les variantes en relation avec le problème étudié dans cette thèse. Les contributions algorithmiques de cette thèse feront l'objet des chapitres suivants.

Chapitre 2

Problème de tournées de véhicules dynamique multi-tour avec overtime : Description, modélisation mathématique et heuristique de résolution

2.1 Introduction

Traditionnellement, la modélisation mathématique est la première étape pour traiter un problème d'optimisation. D'une part, le modèle mathématique permet de cerner toutes les composantes du problème à savoir : l'objectif, les contraintes qui définissent les périmètres de l'espace de recherche et les variables qui définissent les sorties de l'optimisation. D'une autre part, il permet d'avoir une approche de résolution automatique, si nous arrivons à le résoudre.

Pour rester dans les normes et puisqu'on est les premiers à introduire le MTDVRPOT, nous avons proposé un modèle mathématique uni-période. Il s'agit d'un programme bi-objectifs, non linéaire et en nombre entier. Sur la base de ce modèle, nous avons proposé une méthode de résolution qui couvre toute la journée et dont le principe est de résoudre le modèle mathématique d'une façon itérative.

Dans ce chapitre, nous allons, d'abord, présenter une description détaillée du MTDVRPOT. Ensuite nous allons expliquer les différentes composantes de notre modèle. La méthode de résolution sera présentée par la suite, alors que les jeux de tests et les résultats computationnels seront présentés en dernier lieu.

2.2 Description du problème

2.2.1 Gestionnaire des événements

Le gestionnaire des événements est l'interface responsable d'organiser la communication entre les données provenant de l'extérieur et le système interne de la société. Il s'agit d'un ensemble de règles de gestion qu'on applique sur les nouvelles requêtes, avant de les traiter algorithmiquement, ainsi que

les ordres qu'on donne, par la suite, aux conducteurs (figure 4). Dans le cas du DVRP traité dans cette thèse, nous adoptons la démarche périodique de Montemanni et al. (2005). Le principe de cette démarche est de décomposer la journée de travail en intervalles de temps. Le rôle du gestionnaire des événements est de créer un problème statique pour chaque intervalle de temps et exécuter l'algorithme d'optimisation en vue d'obtenir une solution qui couvre le reste de la journée. Le problème du premier intervalle de temps considère les requêtes reçues pendant la journée précédente et non servies le même jour. Nous commençons l'optimisation du problème suivant vers la fin de l'intervalle de temps (période). Ce problème considère les clients arrivants durant la première période et les clients non encore servis (les clients servis ne sont plus considérés, mais ils seront stockés dans la solution complète finale). À ce moment, un camion peut se trouver :

- Soit en train de livrer un client
- Soit sur la route depuis un client source vers un client destination
- Soit en route vers, ou dans le dépôt central.

Dans le premier et le deuxième cas le client courant ou le client destination sont considérés comme emplacement actuel du véhicule. Nous les appelons dépôts fictifs. Dans ce cas, la capacité du véhicule est sa capacité résiduelle. Quant au troisième cas, nous considérons que le véhicule est stationné initialement dans le dépôt central.

À la fin de chaque intervalle de temps, la meilleure solution trouvée pour le problème correspondant est examinée, et les requêtes dont le temps de traitement commence à partir du début de la période suivante sont transmises aux conducteurs correspondants. En d'autres termes, le plan de routage de la période suivante est tracé selon les positions actuelles des véhicules et la solution obtenue à la fin de la dernière optimisation. Ainsi on communique à chaque conducteur les clients qu'il doit servir et dont le temps de service appartient à la période suivante. Ce modèle considère trois paramètres nécessaires :

n_{ts} : Représente le nombre d'intervalles de temps qui composent une journée de travail,

T_{co} : *Cut-off time* qui représente l'heure limite de réception des requêtes dynamiques. Les requêtes qui arrivent après T_{co} sont reportées à la journée suivante, où elles seront considérées comme clients du problème de la première période. Ce paramètre permet d'éviter la surcharge des requêtes à la fin de la journée.

T_{ac} : *Advanced commitment time* qui permet au système de trouver le plan adéquat de la période suivante et communiquer les nouveaux itinéraires aux chauffeurs avant de quitter le dernier emplacement. C'est une durée de temps qui sépare la fin d'une période du début de la période qui la

suive. Le but principal du T_{ac} est d'avoir un temps intermédiaire pour tourner le programme d'optimisation et communiquer avec les conducteurs sans toucher au temps des deux périodes. Dans le reste de ce rapport, nous considérons que le T_{ac} est négligeable (T_{ac}= 0).

2.2.2 Problème de tournées de véhicules dynamique multi-tour avec overtime (MTDVRPOT).

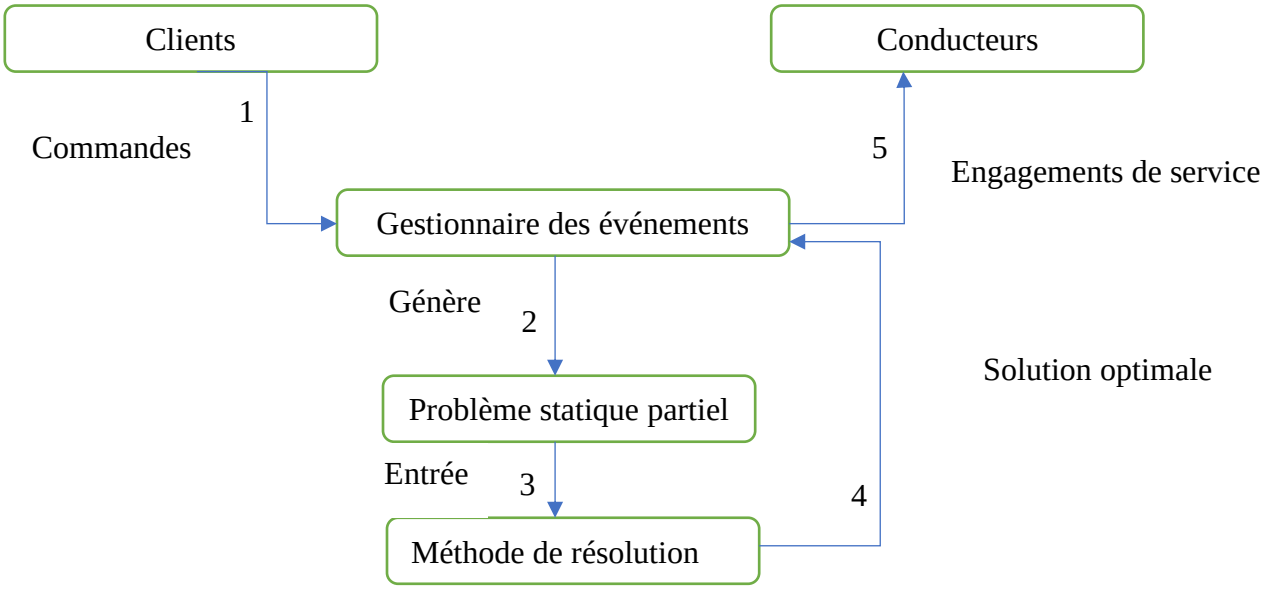

Figure 4: Architecture du gestionnaire des événements

Nous allons considérer le cas d'une société de distribution qui possède une flotte homogène. C'est-à-dire que les véhicules ont la même capacité et roulent avec la même vitesse. Les clients font des requêtes tout au long de la journée, et la société doit servir le nombre maximal de requêtes le même jour. La société dispose d'un nombre limité de véhicules. De ce fait, et si nécessaire, nous permettons aux véhicules d'effectuer plusieurs tournées durant la même journée, pour pouvoir livrer tous les clients en cas d'une grande demande. La journée de travail est divisée en plusieurs périodes de temps égales. Ainsi, les demandes qui arrivent au cours d'une période sont rassemblées pour être insérées dans le plan de routage de la période suivante. Au début de journée, les véhicules partent du dépôt central avec un plan de routage contenant l'ensemble des requêtes arrivantes après T_{co} de la journée d'avant. À la fin de la première période, les positions courantes des véhicules sont marquées. Si le véhicule est en entrain de livrer un client ou en route vers un client, ce dernier est considéré comme un dépôt fictif lors de la période suivante. Une optimisation est effectuée pour déterminer les nouveaux itinéraires qui contiendront les nouveaux clients et les clients restants de la période précédente. La figure 5 illustre un exemple de ce problème. La figure 5. A montre le plan de la première période, tout en spécifiant la position de chaque véhicule à la fin de cette période. Alors que la figure 5.B montre le plan de la deuxième période à son début. Initialement, les deux véhicules

commencent leurs visites à partir du dépôt central. Le premier plan du véhicule 1 est de servir les clients A, B, C et D et de revenir au dépôt central. Quant au véhicule 2, son plan est de servir le client E et de revenir au dépôt central. T^k est le temps nécessaire au véhicule k pour atteindre la destination suivante. Ce temps commence à la fin de la période en cours et se termine lorsque le véhicule k arrive à sa destination. À la fin de la première période, le véhicule 1 est en route vers le client B et le temps nécessaire pour atteindre ce client est T^1. Ainsi, nous considérons B comme un dépôt fictif dans la deuxième période, alors que le véhicule 2 est considéré stationné dans le dépôt central sachant qu'il lui faut un temps T^2 pour l'atteindre. De plus, à la fin de la première période, nous avons quatre nouveaux clients (F, G, H et I). Les itinéraires sont ensuite reprogrammés afin d'avoir un nouveau plan qui contient à la fois de nouveaux clients et des clients non encore servis (B, C et D).

Or, une solution qui assure tout le service et respecte le temps légal n'est pas toujours abordable. Pour éviter ce dilemme, nous autorisons, en cas de besoin, l'utilisation d'un temps supplémentaire qu'on appelle « overtime ». Toutefois, l'overtime ne doit pas dépasser une durée maximale appelée overtime légal maximal.

Reste à noter que les véhicules commencent leurs tournées depuis le dépôt central et y reviennent soit pour s'approvisionner pour une nouvelle tournée soit pour clôturer leur journée de travail. L'objectif est de minimiser la distance totale parcourue ainsi que l'overtime maximal performé.

2.3 Modèle mathématique

Dans cette partie, nous présentons la modélisation mathématique d'un seul sous-problème. Un sous-problème est le problème lié à une seule période de temps. Il contient les nouveaux clients et les clients restants du plan de routage de la période précédente. En plus, les capacités et les positions des véhicules ainsi que les dépôts fictifs dépendent tous de la partie réalisée de la solution de la période

précédente. Ceci étant, le plan résultant de la résolution d'un sous-problème, doit couvrir tout le temps

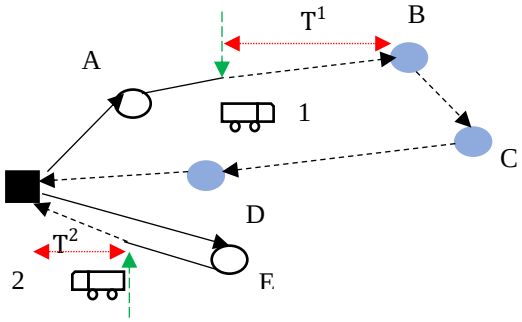

Figure 5.A: Plan de la première période à sa fin

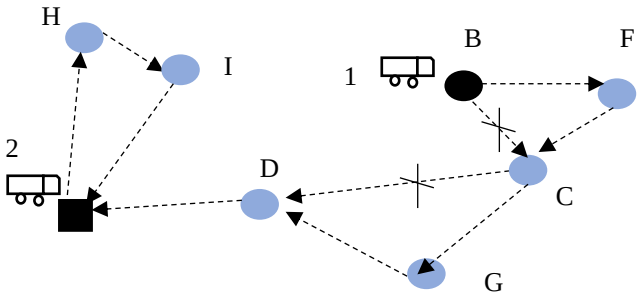

Figure 5.B: Plan de la deuxième période à son début

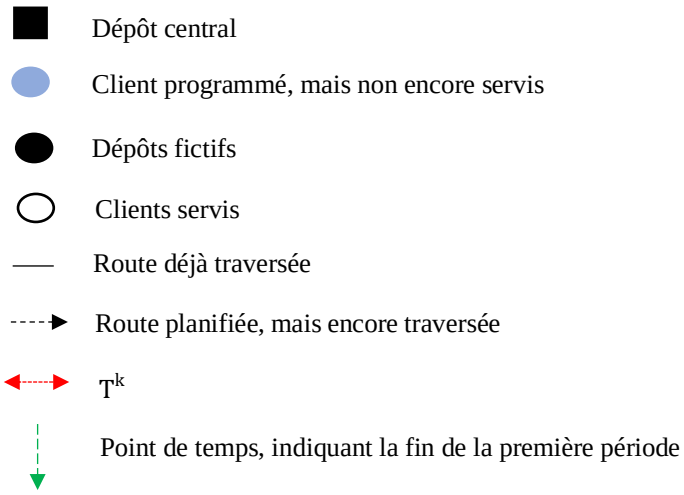

Figure 5: Description du problème

qui reste de la journée de travail. Dans la suite de cette section, nous adoptons les notations suivantes :

0 : Indice du dépôt central.

F : Union des indices du dépôt central et les dépôts fictifs. Pour le problème de la première période de temps D= {0}

I : Liste des clients à servir

K : Nombre de véhicules

n : Nombre maximal de tournées qu'on peut affecter à un véhicule

d_{ij} : Distance entre le nœud i et le nœud j

t_{ij} : Temps de voyage entre le nœud i et le nœud j

Q_k : Capacité restante du véhicule k au début de la période.

Q : Capacité totale ou initiale d'un véhicule.

T^k : Le temps nécessaire pour le véhicule k pour atteindre son dépôt (fictif ou central).

T_p : Temps normal disponible. Il commence au début de la période, en question, et se termine à la fin du temps normal de la journée de travail.

T : Le temps total normal de la journée de travail

αT : Overtime maximal permis.

q_i : Quantité demandée par le client i.

Afin de distinguer les tournées de chaque véhicule, nous définissons S_k l'ensemble des n tournées que le véhicule k ($k \in \{1, .., K\}$) peut effectuer ;

$$S_k=\{m*(k-1)+q,\ q = 1..\ m\}, \qquad (1)$$

Q^r est la capacité disponible pour la tournée r. Une tournée peut commencer depuis le dépôt central ou bien depuis l'un des dépôts fictifs. Si une tournée r commence du dépôt central, $Q^r = Q$. Sinon $Q^r = Q_k$ avec Q_k; est la capacité restante du véhicule correspondant à ce dépôt fictif.

On définit les variables :

$$x_{ij}^r = \begin{cases} 1, \text{ si le noeud j est visité après le noeud i dans la tournée r} \\ 0, \text{ sinon} \end{cases} \qquad (2)$$

et

π_i : Une variable utilisée pour empêcher la création des sous-tours. Elle peut être interprétée par la position du client $i \in I$ dans une tournée. (3)

Les deux objectifs du problème sont :

$$\min \sum_{r \in S_T} \sum_{i \in I \cup F} \sum_{\substack{j \in (I \cup \{O\}) \\ j \neq i}} d_{ij}\, x_{ij}^{r} \tag{4}$$

$$\min(OT^{p}) \tag{5}$$

Avec

$$S^{T} = \bigcup_{k \in \{1,..,K\}} S_k \tag{6}$$

$$OT^{P} = \max_{k \in \{1,..,K\}} (OT_k) \tag{7}$$

et

$$OT_k = \max\left(0, \sum_{r \in S_k} \sum_{i \in I \cup F} \sum_{\substack{j \in (I \cup \{0\}) \\ j \neq i}} t_{ij} x_{ij}^{r} - T_p + T^{k}\right) \tag{8}$$

OT^{P} et OT_k réfèrent à l'overtime maximal, qu'on cherche à minimiser, et à l'overtime du véhicule indexé par k.

On pose pour $d \in (F \setminus \{0\})$ et $r \in S^{T}$

$$f_{dr} = \begin{cases} 1, \text{ Si la tournée r commence depuis le dépôt fictif d} \\ 0, \text{ Sinon} \end{cases} \tag{9}$$

Les contraintes du problème sont modélisées comme suit :

$$\sum_{i \in I \cup F} \sum_{\substack{j \in (I \cup \{0\}) \\ j \neq i}} q_i x_{ij}^{r} \leq Q^{r}, \quad r \in S^{T} \tag{10}$$

$$\sum_{r \in S_k} \sum_{i \in I \cup F} \sum_{\substack{j \in (I \cup \{O\}) \\ j \neq i}} t_{ij} x_{ij}^{r} - T_p + T^{k} \leq \alpha T, \quad k \in \{1,.. ,K\} \tag{11}$$

$$\sum_{r \in S^{T}} \sum_{\substack{j \in (I \cup \{0\}) \\ j \neq i}} x_{ij}^{r} = 1, \; i \in I \cup (F \setminus \{0\}) \tag{12}$$

$$\sum_{\substack{i \in I \cup F \\ i \neq j}} x_{ij}^{r} = \sum_{\substack{i \in (I \cup \{0\}) \\ i \neq j}} x_{ji}^{r}, \quad j \in I, \; r \in S^{T} \tag{13}$$

$$\sum_{r\in S^T}\sum_{i\in(I\cup(F\setminus\{0\}))} x_{i0}^{r} = \sum_{r\in S^T}\sum_{i\in I} x_{0i}^{r} + \sum_{r\in S^T}\sum_{d\in(F\setminus\{0\})} f_{dr} \quad (14)$$

$$\sum_{j\in(I\cup\{0\})} x_{dj}^{r} = f_{dr}\,,\ r\in S^T,\ d\in F\setminus\{0\} \quad (15)$$

$$\sum_{i\in I\cup(F\setminus\{0\}} x_{i0}^{r} \le 1\,,\ r\in S^T \quad (16)$$

$$\sum_{r\in S^T}\sum_{\substack{i\in I\cup F\\ j\neq i}} x_{ij}^{r} = 0,\ j\in(F\setminus\{0\}) \quad (17)$$

$$\pi_j \ge \pi_i + 1 - \mathrm{card}(I\cup\{0\})\left(1 - \sum_{r\in S^T} x_{ij}^{r}\right),\ (i,j)\in I^2 \quad (18)$$

$$x_{ij}^{r}\in\{0,1\},\ (i,j)\in I\cup F,\ \forall r\in S^T\ ,\ \pi_i \ge 0, i\in I \quad (19)$$

La contrainte (10) garantit le respect de la capacité restante de chaque véhicule. L'inégalité (11) montre que l'overtime réalisé par chaque véhicule ne doit pas dépasser l'overtime maximal permis. Le fait que chaque client est visité une seule fois est assuré par la contrainte (12). Quant à l'équation (13), elle représente la conservation de flux au niveau des clients ; chaque véhicule qui visite un client doit le quitter après l'avoir servi. Alors que la conservation de flux au niveau du dépôt central est assurée par la contrainte (14). Quant à la contrainte 15, elle impose aux véhicules stationnés dans les dépôts fictifs, d'effectuer, au moins, une tournée. La contrainte (16) est construite pour éviter que plusieurs tournées partagent le même indice r. La contrainte (17) indique qu'un dépôt fictif ne peut pas être une destination, et la contrainte (18) interdit la création des sous-tours. Finalement, la contrainte (19) spécifie la nature numérique des variables de décision.

2.4 Résolution par une heuristique basée sur une méthode exacte itérative [HMEI]

Comme déjà mentionnée, la journée de travail est divisée en intervalles de temps (périodes). Ainsi, le problème global (c'est-à-dire le problème de toute la journée) est un ensemble de sous-problèmes. Chaque sous-problème représente la question posée au début de chaque nouvelle période et qui consiste à trouver un plan de routage optimal et actualisé pour servir les nouvelles requêtes et les requêtes restantes de la période précédente. Pour résoudre ce problème, nous proposons une heuristique basée sur la résolution itérative des modèles mathématiques (décrits dans la section 2.3) de chaque sous-problème. Ainsi, au début de la journée de travail, nous résolvons le problème

contenant les clients arrivants après l'heure T_f de la veille. Dans ce stade, il n'y a pas de dépôts fictifs puisque tous les véhicules sont initialement stationnés dans le dépôt central. Tous les véhicules ont la même capacité Q, et le temps de voyage normal disponible pour le problème de cette période T_p est égal à la durée de la journée de travail T.

À la fin de chaque intervalle de temps, nous collectons les données nécessaires pour les entrées du modèle mathématique de la période suivante. Certaines informations sont liées à la solution de la période précédente, à savoir :

- Les dépôts fictifs
- Le T^k de chaque véhicule qui dépend de sa position actuelle.
- La capacité restante de chaque véhicule Q_k qui dépend des clients que ce véhicule a servis au cours des périodes précédentes.
- Les clients à servir : ça inclut les clients non encore servis de la période précédente et les nouveaux clients.

Une fois les données nécessaires sont prêtes, nous procédons à la résolution du modèle mathématique. La procédure de la figure 6 présente les étapes de l'heuristique HMEI. Il a été codé avec java, tandis que, nous utilisons la bibliothèque CPLEX pour java, pour résoudre les modèles mathématiques.

Par ailleurs, notre modèle mathématique a deux objectifs. Pour évaluer les solutions, nous considérons que la meilleure solution est celle qui minimise la distance totale parcourue tout en respectant le temps de travail légal. Si nous n'arrivons pas à trouver une telle solution, la meilleure solution est, alors, celle qui minimise l'overtime maximal effectué. Ainsi, dans la première période, au début, nous ne visons que l'objectif de minimiser la distance totale parcourue. Dans cette étape, nous ne considérons pas l'overtime dans le modèle mathématique [le αT est remplacé par 0 dans la contrainte (11)). Si une telle solution est trouvée, elle sera la solution adoptée pour cette période. Sinon, nous lançons une autre optimisation du modèle mathématique en visant le deuxième objectif [minimisation de l'overtime maximal réalisé]. Dans ce cas, la formulation originale des contraintes du modèle mathématique est gardée. Les problèmes des périodes suivantes auront certainement moins de temps pour compléter le service des clients restants en plus de certains nouveaux clients. Si nous considérons que deux problèmes de deux périodes successives commencent en même temps, ces problèmes auraient les mêmes caractéristiques sauf l'ensemble de clients à servir qui est plus grand dans le problème de la deuxième période. Ainsi, si nous n'arrivons pas à trouver une solution

réalisable au problème de la première période, nous ne la trouverons pas pour celui de la deuxième période.

Méthode de résolution

1. Résoudre le modèle mathématique du problème de la première période de temps.
2. Trouver la position de chaque véhicule à la fin de cette période, calculer T^k et C_k et identifier le dépôt [fictif ou central] correspondant à chaque véhicule.
3. Trouver tous les clients non encore servis durant la période en cours et les rassembler avec les nouveaux clients [ceux qui ont effectué des requêtes durant cette période] pour construire l'ensemble des clients à servir dans le problème de la période suivante.
4. Résoudre le modèle mathématique du problème de la période suivante.
5. Répéter les étapes 2, 3 et 4 jusqu'à la dernière période avant T_f
6. Calculer la distance totale parcourue et l'overtime maximal performé

Figure 6 : Heuristique basée sur une méthode exacte itérative [HMEI]

Dans le cas dynamique, le problème de la deuxième période commence après que les véhicules parcourent une partie des itinéraires, en ne considérant que le temps restant et les clients restants. De ce fait, il existe moins de solutions possibles pour le problème de la deuxième période. Par conséquent, si on ne trouve pas de solution optimale qui minimise la distance en respectant le temps normal de travail, dans une période donnée, on ne la trouverait pas dans toutes les périodes suivantes. Dans ce cas, nous optimisons directement l'overtime maximal performé, pour toutes les périodes suivantes. La figure 7 illustre le séquencement de l'optimisation sur 3 périodes.

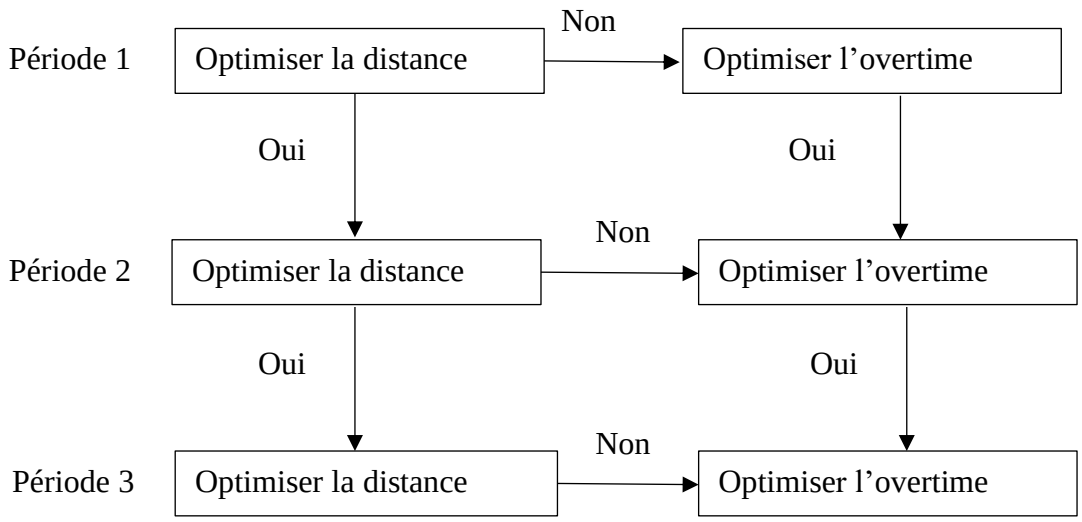

Figure 7: Séquencement de l'optimisation pour un exemple à 3-periode

2.5 Résultats numériques

Pour tester l'approche HMEI, nous adoptons le même principe des benchmark de Taillard et al. (1996). En effett nous nous sommes basés sur le problème C-50 de ces benchmarks pour générer plusieurs instances en modifiant le nombre de véhicules disponibles (m) et l'horizon temporel disponible T. Ce dernier est limité à $T = [1{,}1 * z^* / m]$, avec z^* est la valeur de la meilleure solution trouvée par Rochat & Taillard, (1995) pour le VRP statique. En plus, la journée de travail est divisée en 25 intervalles de temps (Montemanni et al., 2005). Quant à l'heure d'arrivée des requêtes des clients, elle est proportionnelle à l'heure d'arrivée proposée par (Kilby et al. 1998).

Cependant, la résolution d'une instance de 50 clients est loin d'être abordable avec l'HMEI, car elle se base sur une méthode exacte itérative. De ce fait, nous nous sommes limités à résoudre des instances avec moins de clients. Le premier problème, appelé C-25, compte 25 clients et il est généré en éliminant 25 clients des clients de la première période du problème C-50. Le deuxième problème, appelé C-30, comprend 30 clients et il est généré en éliminant 20 clients des clients de la première période du problème C-50. Le troisième problème, appelé C-36, contient 36 clients et il est généré en éliminant 14 clients des clients de la première période du problème C-50. Enfin, le quatrième problème, appelé C-40, comprend 40 et il est généré en éliminant 10 clients des clients de la première période du problème C-50. Afin de tester l'optimisation des deux objectifs du modèle mathématique, la capacité des véhicules est fixée égale à la moitié (Q =80) de la capacité des véhicules du problème d'origine, pour les deux premiers problèmes (C-25 et C-30), car ces deux problèmes ont un nombre de clients de -50% et -40% par rapport au nombre de clients du problème d'origine. Alors que l'on conserve la même capacité du problème d'origine (Q=160) pour les deux derniers problèmes (C-36 et C-40) qui ont un nombre de clients de -28% et -20% du nombre de clients du problème d'origine.

En variant m, le nombre de véhicules disponibles, entre 1 et 5, nous avons généré cinq instances de chaque problème. De plus, l'overtime maximal autorisé pour chaque instance est fixé à un quart de l'horizon temporel normal, et le temps et la distance sont considérés comme équivalents. Les tests ont été effectués sur l'infrastructure de calcul haute performance (HPC) du CNRST. Cette infrastructure est composée de 19 nœuds qui offrent les capacités suivantes :

- 760 cœurs (68 TFlops)
- 108 TB de stockage
- 5,2 TB de mémoire
- 2 cartes GPU

Ces nœuds sont interconnectés par un réseau à très faible latence (OPA) à 100 Gbps ce qui permet d'optimiser les performances pour les calculs parallèles. Cette infrastructure est connectée au réseau MARWAN par une liaison 5 Gbps qui assure la fluidité dans l'utilisation et le transfert des données des universités.

Tableau 1 : Résultats numériques de l'HMEI

	m	**T**	**HMEI**	
			Distance	**Overtime**
C-25	1	577	512,16	0
	2	289	482,51	0
	3	192	506,85	0
	4	144	534,83	14,17
	5	115	527,58	21,76
C-30	1	577	541,42	0
	2	289	554.95	0
	3	192	535.75	0
	4	144	571,09	8,98
	5	115	543,64	4,54
C-36	1	577	500,37	0
	2	289	487,79	0
	3	192	551,64	2,29
	4	144	522,46	0
	5	115	518,49	0
C-40	1	577	539,12	0
	2	289	510,03	0
	3	192	-	-
	4	144	549,60	0
	5	115	572,11	5,10

Le tableau 1 présente les détails des résultats numériques. C, T et m sont respectivement la capacité des véhicules, l'horizon temporel normal et le nombre de véhicules disponibles. La distance est la distance totale parcourue ; elle représente la distance parcourue par tous les véhicules pendant toute la journée. Tandis que l'overtime est l'overtime maximal effectué ; il représente l'overtime effectué par le véhicule qui revient le dernier au dépôt central à la fin de la journée de travail. Nous avons pu avoir une solution pour toutes les instances à l'exception de l'instance C-40-3.

En comparant la solution de l'instance C-25-4 avec l'instance C-30-4, nous remarquons que la solution de l'instance C-30 est meilleure, en termes d'overtime, malgré que le C-30 contient tous les

clients de C -25 plus 5 autres clients appartenant à la première période. Cependant, l'évaluation bi-objective des deux solutions explique cette incohérence. Surtout que la distance parcourue est moins en C-25. La même remarque peut être faite en comparant les solutions des instances C-25-5 et C-30-5. Dans ces tests, nous pouvons dire que chaque instance de C-25 est incluse dans sa réciproque de C-30 (c'est-à-dire : C-25-1 ⊂ C-30-1). De même, les instances de C-36 sont incluses dans celles de C-40. Le premier objectif du problème (distance) est très cohérent avec l'ordre de cette inclusion. Alors que ce n'est pas le cas pour le deuxième objectif (l'overtime). Cela pourrait s'expliquer par le fait qu'un scénario optimal dans une période de temps dépend du scénario de la période précédente. En d'autres termes, en ajoutant certains clients uniquement, dans la première période, on peut avoir des scénarios complétement différents pour la période suivante, car on va avoir d'autres dépôts fictifs et d'autres clients à servir.

Ceci, d'une part, d'autre part, dans un problème comme le MTDVRPOT, à chaque période de temps, souvent, il y a plusieurs scénarios optimaux et non pas un seul. Parce qu'un véhicule peut faire plusieurs tours dans la même journée. Si par exemple, dans la première période de temps, nous constatons que la solution optimale contient trois tournées différentes. Un premier scénario à adopter est d'affecter toutes ces tournées à un seul véhicule. Le deuxième scénario peut consister à attribuer chaque tournée à un véhicule différent, si le nombre de véhicules disponibles le permet. Selon le scénario choisi, les entrées de la période suivante sont décidées. Nous avons essayé de surmonter cette variation en appliquant une heuristique qui standardise l'allocation des tournées aux véhicules après avoir résolu le modèle mathématique de chaque période. Mais la variation demeure. Car, si, par exemple, la solution optimale d'une période de temps contient un itinéraire qui part du dépôt central et qui comprend trois clients, par exemple, cette solution reste optimale si l'ordre des clients de cette tournée est inversé. Mais les dépôts fictifs générés et les clients à servir dans la période suivante ne sont plus les mêmes.

2.6 Conclusion

En conclusion, l'HMEI permet de donner la solution optimale d'une seule période de temps. Cependant, la solution totale n'est pas nécessairement optimale. De plus, nous n'avons pas pu tester l'HMEI que sur des petites instances, ce qui nous a permis de valider notre modèle mathématique. Par contre, les métaheuristiques sont plus appropriées pour résoudre les instances de grande taille des problèmes. Dans le chapitre suivant, nous proposons une démarche à base d'une métaheuristique pour le MTDVRPOT.

Chapitre 3

Algorithme de colonie de fourmis pour la résolution du MTDVRPOT

3.1 Introduction

Le système de colonies de fourmis (ACS) est une métaheuristique proposée par Marco Dorigo et al. (1991) basée sur le comportement biologique des fourmis qui utilisent la phéromone comme moyen de communication. À l'égard de l'exemple biologique, les algorithmes de colonies de fourmis utilisent de simples agents artificiels, appelés fourmis, communiquant entre eux indirectement en utilisant des phéromones artificielles au sein d'une colonie d'optimisation. Les traces de phéromones fonctionnent comme des informations numériques distribuées dans l'ACS sur lesquelles les fourmis se basent pour construire des solutions au problème à résoudre.

L'ACS est une métaheuristique largement utilisée pour résoudre le DVRP (chapitre 1). L'objectif de ce chapitre est de l'adopter au MTDVRPOT. Toutefois, nous aurons besoin de l'adopter d'abord au VRP multi-tour avec overtime (VRPMTOT) statique, pour résoudre le problème de la première période de temps. Ensuite, nous allons étendre sa formulation pour résoudre le problème dynamique. Mais, avant d'effectuer les tests sur le MTDVRPOT, nous allons valider la compétitivité de l'approche sur des benchmarks de la littérature et pour cela, nous allons utiliser les benchmarks de Kilby et al. (1998) dédiés à la version classique du DVRP à capacité. À la fin de ce chapitre, nous présenterons les jeux de tests que nous proposons pour le MTDVRPOT et les résultats de l'ACS sur ces instances.

3.2 Résolution du VRPMTOT statique par le système de colonies de fourmis hybride (ACSH)

Le VRPMTOT statique est caractérisé par une flotte homogène, un seul dépôt, la possibilité que les véhicules effectuent plusieurs tournées pendant la même journée et l'autorisation d'utilisation de l'overtime. Dans cette section, nous décrivons un ACSH adapté à ce type de problème dans le but de comparer ses résultats avec des résultats d'autres algorithmes de la littérature pour le VRPMT statique.

3.2.1 Algorithme de résolution du problème statique

L'ACSH proposé pour résoudre le VRPMTOT statique se déroule sur plusieurs itérations. Mais, avant de commencer les itérations, nous devons initialiser les traces de phéromone (section 3.2.1.1). Ensuite, dans chaque itération chaque fourmi va construire les tournées d'une solution complète selon les règles des transitions (section 3.2.1.2). L'étape suivante consiste à affecter les tournées aux véhicules (section 3.2.1.4) pour procéder à la mise à jour locale des traces de phéromone (section 3.2.1.3). Une fois toutes les fourmis finissent la construction de leurs solutions et avant de passer à une nouvelle itération, nous procédons à la mise à jour globale des traces de phéromone (section 3.2.1.3). À cette étape, l'algorithme doit choisir la meilleure solution de l'itération. Dans le cas statique, nous adoptons le LTR comme critère d'évaluation parce qu'il fait la base de qualification de la performance dans les approches de la littérature. En effet, le LTR (Longest Trip Rate) ou le taux du plus long voyage (Olivera & Viera, 2007) est un indicateur qu'on utilise pour qualifier l'overtime des solutions qui dépasse le temps normal de travail. Il permet de comparer le temps de voyage du véhicule qui revient le dernier au dépôt, à l'horizon de temps que les camions devaient respecter T.

$$LTR = max_{k\in\{1..K\}}(t(k)/T)$$

Avec t(k) le temps mit par le camion k pour réaliser ses tournées. Il est bien évident que la valeur de LTR est supérieure à 1 pour les solutions qui ont un overtime supérieur à 0.

Une fois, la meilleure solution de l'itération (*LocaSolution*) est choisie, elle subit une amélioration par une procédure de recherche locale (3.2.1.5) pour être considérée comme solution globale (*GlobalSolution*) initiale. À la fin des autres itérations, nous comparons le LTR de la meilleure solution de cette itération avec le LTR de la solution globale actuelle pour voir si nous devons laisser la même solution globale ou bien la changer par la meilleure solution de la dernière itération. Le critère d'arrêt décide ensuite le nombre d'itérations après lesquelles l'algorithme doit rendre la meilleure solution trouvée. La figure 8 décrit la procédure de résolution du problème statique par l'ACSH.

Procédure de l'ACSH pour le problème statique
GlobalSolution=null ;
MeuilleurLTRGlobal=+∞ *;*
Pour chaque arc (i,j)
$\tau_{ij} = \tau_0$;
Fin Pour ;
Tant que (*NombreItération*<*NombreItérationTotal*)

LocalSolution=null ;

MeuilleurLTR=+∞ ;

Pour *k : =1* à *m*

Tant que (fourmi *k* n'a pas terminé sa solution)

Choisir le prochain client *j* ;

Fin Tant que

Solution=Affecter le véhicule(*k*) ;

Pour chaque arc *(i,j)* dans Solution

Mettre à jour les traces de phéromone partiellement ;

Fin Pour

Si *(LTR<MeuilleurLTR*)

MeuilleurLTR : =LTR;

LocalSolution : = Solution courrante ;

Fin si

Fin Pour

LocalSolution=RechercheLocale(*LocalSolution*) ; *MeuilleurLTR= LTR(LocalSolution) ;*

Pour chaque arc *(i,j)* dans *LocalSolution*

Mettre à jour les traces de phéromone globalement ;

Fin Pour

Si *(MeuilleurLTR<MeuilleurLTRGlobal*)

MeuilleurLTRGlobal : = MeuilleurLTR ;

GlobalSolution:=LocalSolution ;

Fin si

Fin Tant que

Figure 8 : ACSH pour le VRPMTOT statique

3.2.1.1 Phase d'initialisation

La phase d'initialisation dans l'ACSH comporte les deux points suivants :

- L'initialisation des traces de phéromones par l'affectation d'une valeur positive τ_0 à chaque arc du graphe.
- L'affectation d'un état initial à chaque fourmi. Dans un but de diversification, chaque fourmi est placée sur un nœud sélectionné aléatoirement de la liste des clients à visiter. Sous-entendue, la fourmi a commencé sa tournée au dépôt central puis a visité le premier client où elle est initialement placée.

3.2.1.2 Transition des fourmis

Après la phase d'initialisation, chaque fourmi part du client correspondant et construit sa solution en choisissant les clients un par un en respectant les contraintes de capacité et du temps jusqu'à ce que tous les clients soient visités. Si une fourmi n'arrive pas à ajouter un client dans la tournée actuelle à cause de l'insuffisance de la capacité ou du temps du camion, elle retourne au dépôt central pour recommencer une nouvelle tournée.

Pour passer du nœud i au nœud j, la fourmi doit respecter quelques règles. D'abord, nous posons :

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\tau_{ij}(t).\eta_{ij}^{\beta}}{\sum_{u \in N_i^k(t)} \tau_{iu}(t).\eta_{iu}^{\beta}} & \text{si } j \in N_i^k(t) \\ 0 & \text{sinon} \end{cases}$$

Avec

$\eta_{iu} = \frac{1}{d_{iu}}$: La visibilité de l'arc (i,u), égale à l'inverse de la distance d_{iu} entre les deux nœuds i et u.

$\tau_{iu}(t)$: Le taux de phéromone sur l'arc (i,u) à l'instant t.

β : L'influence relative de la visibilité.

N_i^k: L'ensemble des nœuds que la fourmi k peut visiter après la position i.

Ensuite, nous considérons :

q : Une variable aléatoire uniformément distribuée sur l'intervalle [0,1] .

q_0: Un paramètre de l'intervalle [0,1] qui définit la balance diversification/intensification.

Si $q \leq q_0$, la règle de passage du client i au client j est :

$$j = \text{argmax}_{u \in N_i^k} \left[\tau_{iu}(t).\eta_{iu}^{\beta}\right]$$

Sinon *j* est un client sélectionné aléatoirement avec la probabilité $p_{ij}^{k}(t)$

3.2.1.3 Mise à jour des traces de phéromones

La mise à jour des traces de phéromone est séparée en deux niveaux : une mise à jour locale et une autre globale. La première se fait après la construction d'une solution par une fourmi k sur les arcs qu'elle a visités selon la formule suivante :

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \frac{\rho.\tau_0}{1+OV_k}$$

où

ρ: Facteur d'évaporation fixé à une valeur positive et inférieure à 1. Ce facteur a pour but d'éviter l'accumulation illimitée des traces de phéromones sur les arcs du graphe.

OV_k: Overtime réalisé par la fourmi k après l'affectation des véhicules aux tournées de sa solution.

La mise à jour globale s'effectue à la fin de chaque itération selon la formule suivante :

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho.\Delta\tau_{ij}(t)$$

où les arcs (i,j) appartiennent à la meilleure solution *S* de l'itération.

Avec : $\Delta\tau_{ij} = \frac{1}{L_s}$ et L_s: est le temps total de la solution *S*.

3.2.1.4 Affectation des véhicules

Si le nombre de véhicules dépasse le nombre de tournées, nous affectons un véhicule à chaque tournée. Sinon, nous trions les tournées selon la distance dans un ordre décroissant. L'affectation se fait, ensuite, de façon itérative jusqu'à ce que toutes les tournées soient assignées à l'un des véhicules. La procédure d'affectation des véhicules est détaillée dans la Figure 9.

Procédure de l'affectation des véhicules
Si (*NombreTournees<NombreCamion*) Affecter un camion à chaque tournée ; **Fin si** **Sinon** Trier les tournées de la plus longue à la plus courte ; Affecter un camion à chacune des premières *NombreCamion* tournées ; **Tant que** (il y a des tournées non affectées) Trier les camions selon la longueur totale des tournées affectées dans un ordre croissant ; Affecter la première tournée non affectée au premier camion ; **Fin Tant que** **Fin Sinon**

Figure 9: Procédure d'affectation des véhicules pour l'ACSH du problème statique

3.2.1.5 La Recherche locale

À la fin de chaque itération et avant d'appliquer la mise à jour globale, la meilleure solution trouvée lors de cette itération subit une phase d'optimisation par recherche locale intra et inter tournée. Cette procédure est basée sur les mouvements de recherche locale proposée par Ayadi & Benadada, (2013). Nous appliquons ces mouvements dans l'ordre de la figure 10.

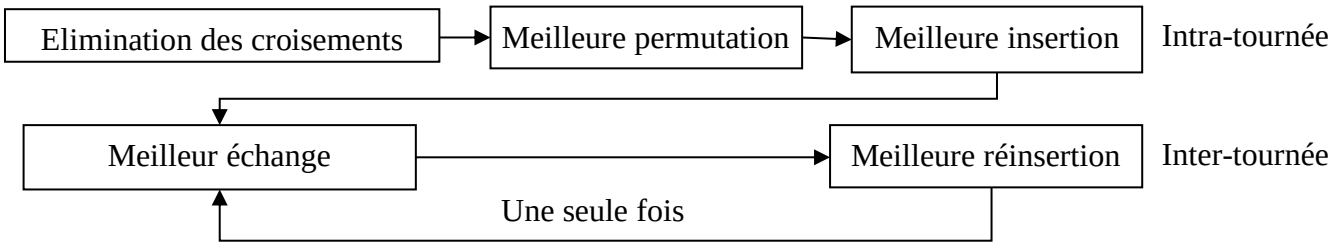

Figure 10: Procédure de recherche locale

Au début, nous appliquons la recherche locale intra-tournée (figure 11) sur chaque tournée isolée afin d'améliorer l'ordre de visite des clients. Cette phase comporte trois mouvements :

- Élimination des croisements s'ils sont trouvés entre les arcs de la tournée.
- Chercher les permutations des clients qui peuvent optimiser la distance de la tournée et effectuer la meilleure permutation trouvée
- Tester des mouvements de réinsertion des clients dans différentes positions et effectuer celui qui minimise le plus la distance.

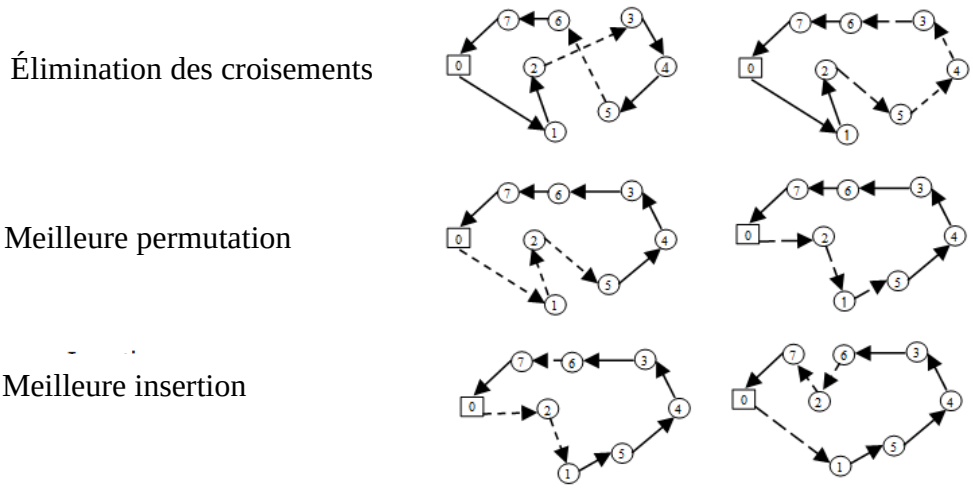

Figure 11 : Mouvements de la recherche locale intra-tournée

Ensuite, nous appliquons la recherche locale inter-tournée. Dans cette phase, les opérateurs d'échange et réinsertion sont appliqués entre deux tournées différentes de la solution (figure 12). Entre chaque paire de tournées de la solution, nous effectuons la meilleure permutation et la meilleure insertion possibles et ceci 2 fois successives. En effet, pour chaque tournée de la solution, nous créons une boucle qui parcoure les tournées restantes pour chercher toutes les échanges possibles entre les deux tournées sélectionnées et réaliser celle qui minimise la distance. Puis, nous cherchons toutes les

réinsertionns possibles entre les tournées sélectionnées pour effectuer celle qui minimise la distance. Nous répétons les deux mouvements une deuxième fois.

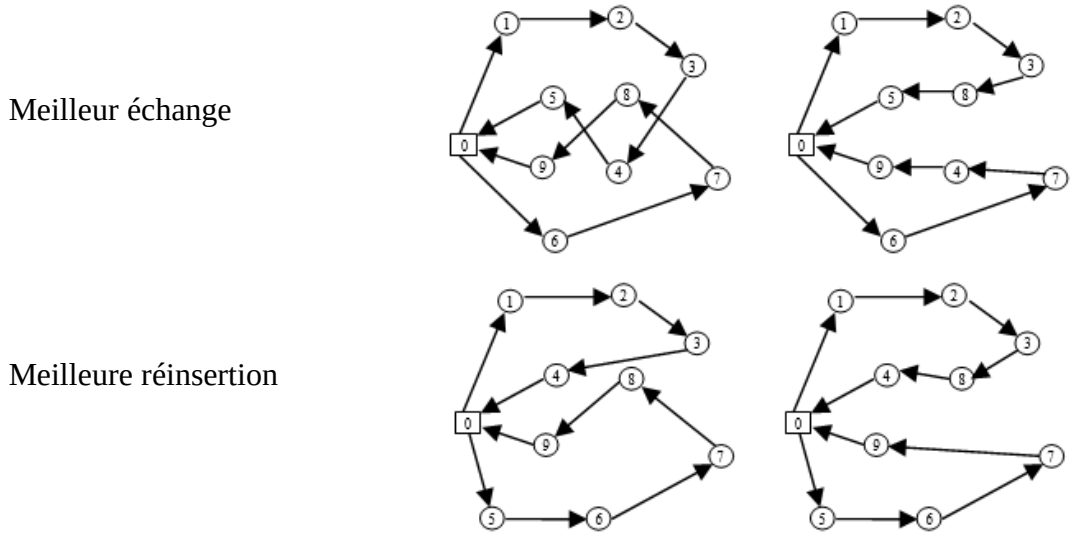

Figure 12 : Mouvement de la recherche locale inter-tournée

3.2.1.6 Évaluation

Dans le cas statique nous adoptons le LTR comme critère d'évaluation puisqu'il est le plus utilisé dans la littérature du VRPMT statique. Ainsi, la meilleure solution choisie est celle qui réalise le LTR le plus bas.

3.2.2 Résultats numériques

3.2.2.1 Paramétrage de l'ACSH

D'après une étude précédemment faite par Gambardella et al. (2000), un bon paramétrage du SCF appliqué au PTV classique est le suivant : $q_0 = 0.9$, $\beta = 1$, $\rho = 0.1$ et $\tau_0 = \frac{1}{n*Cost(PI)}$, ou $Cost(PI)$ est le coût d'une solution trouvée par une heuristique gloutonne. Nous avons utilisé les mêmes paramètres pour l'ACSH, sauf le τ_0 qu'on a fixé à $\tau_0 = 0{,}1$ pour tous les arcs du système. Le choix de cette valeur est basé sur les résultats de plusieurs tests préliminaires.

Le nombre d'itérations est fixé à 200 et le nombre de fourmis est égal au nombre de clients puisqu'on doit placer une fourmi sur chaque client.

3.2.2.2 Benchmarks de Taillard

Les benchmarks sont des problèmes standards de tests dont le but est l'évaluation et la comparaison des algorithmes. La principale raison de tester un algorithme sur de tels problèmes est de comparer

ses résultats avec ceux obtenus par d'autres algorithmes. On prouve, ainsi, la compétitivité ou non de l'algorithme testé. Pour valider l'algorithme statique, nous l'avons testé sur les benchmarks de Taillard et al. (1996) qui sont les jeux de test les plus connus pour le VRPMT. Les résultats obtenus sont référés par ACSH2016 et sont comparés avec les résultats de Ayadi & Benadada, (2013) référés par MA2013.

Taillard et al. (1996) utilisent neuf problèmes de base ; sept problèmes de Christofides et al. (1979) et deux problèmes de Fisher, (1994). Ils ont utilisé les mêmes graphes, les mêmes demandes et les mêmes capacités des camions des problèmes de base, et ont généré 104 instances en proposant plusieurs valeurs de m (le nombre de véhicules disponible) et deux valeurs limites pour le temps de travail $T_1 = [1{,}05\ {}^{z^*}/_{m}]$et $T_2 = [1{,}1\ {}^{z^*}/_{m}]$, avec z^* la valeur de la meilleure solution trouvée par Rochat & Taillard, (1995) pour chaque problème. Comme dans leur benchmark, nous considérons le temps et la distance équivalents. Cependant nous allons tester notre algorithme sur 5 problèmes au lieu de 9, parce que le problème statique n'est pas notre objectif principal. Les résultats présentés dans ce paragraphe correspondent aux jeux de test décrits dans le Tableau 2.

Tableau 2: Jeux de tests de Taillard et al. (1996)

Problème	**z***	**size**	**# instances**	**M**
C1	524,61	50	8	{1..4}
C2	835,26	75	14	{1..7}
C3	826,14	100	12	{1..6}
C5	1291,44	199	6	{1..3}
F134	1162,96	134	6	{1..3}

L'algorithme proposé est codé en Java et exécuté sur une machine de processeur Intel core i5, 2,6 GHz, avec une mémoire de 8 Go de RAM, sous le système d'exploitation Windows 8.

3.2.2.3 Résultats computationnels

Les résultats détaillés sont représentés pour T_1 et T_2 selon la valeur de m (pour plus de détails, voyez le Tableau 20 dans l'Annexe A). Une première observation des résultats permet de conclure qu'on a pu obtenir des solutions meilleures que z* pour plusieurs instances de C1, C2 et F134. Cependant, MA2013 a pu avoir 41 solutions, dont le LTR ≤ 1 alors que l'ACSH2016 n'a pu trouver que 22 solutions. Tout de même, l'ACSH2016 consomme un temps d'exécution faible par rapport à l'approche MA2013 (Tableau 3).

Pour comparer les solutions qui dépassent le temps normal de travail, nous utiliserons la déviation par rapport au LTR qui représente la différence entre la valeur du LTR trouvée par ACSH2016 et celle trouvée par le MA2013:

$$Dev = LTR_{ACSH2016} - LTR_{MA2013}$$

Par contre, le GAP sera utilisé pour comparer les solutions qui respectent le temps normal de travail en comparant leurs fitness f(s) avec la valeur de z*.

$$GAP(s) = 100 * (\frac{f(s)}{z^*} - 1)$$

Le Tableau 4 représente la déviation maximale et minimale pour chaque problème ainsi que le GAP maximal et minimal. La déviation est calculée uniquement pour les instances où les deux approches ACSH16 et MA2013 n'ont pas pu donner une solution qui respecte le temps normal de travail (LTR>1). Tandis que le GAP est calculé pour les instances où les approches ont pu avoir un LTR inférieur ou égal à 1.

Tableau 3: Moyenne du temps d'exécution en secondes

Problème	**ACSH16**	**MA13**
C1	11,5	43
C2	17,28	365
C3	69,33	3751
C5	257 166	5810,58
F134	1246,16	7040,69

Tableau 4: Comparaison du Dev et GAP

Problème	**Dev_{min}**	**Dev_{max}**	**ACSH16**		**MA13**	
			GAP_{min}	**GAP_{max}**	**GAP_{min}**	**GAP_{max}**
C1	0,009	0,023	-2 357	2 011	0,06	4,16
C2	0,085	0,085	-0,327	4 859	0	4,18
C3	0,19	0,19	5 457	8 518	0,26	1,26
C5	-	-	-	-	1,64	6,89
F134	-	-	-3 866	-0,003	0,00	0,34

3.3 Résolution du MTDVRPOT par le système de colonies de fourmis hybride (ACSH)

Pour normaliser les appellations, nous appelons le problème de la première période de la journée *problème statique*. Alors que les problèmes des périodes suivantes sont appelés *problèmes dynamiques.*

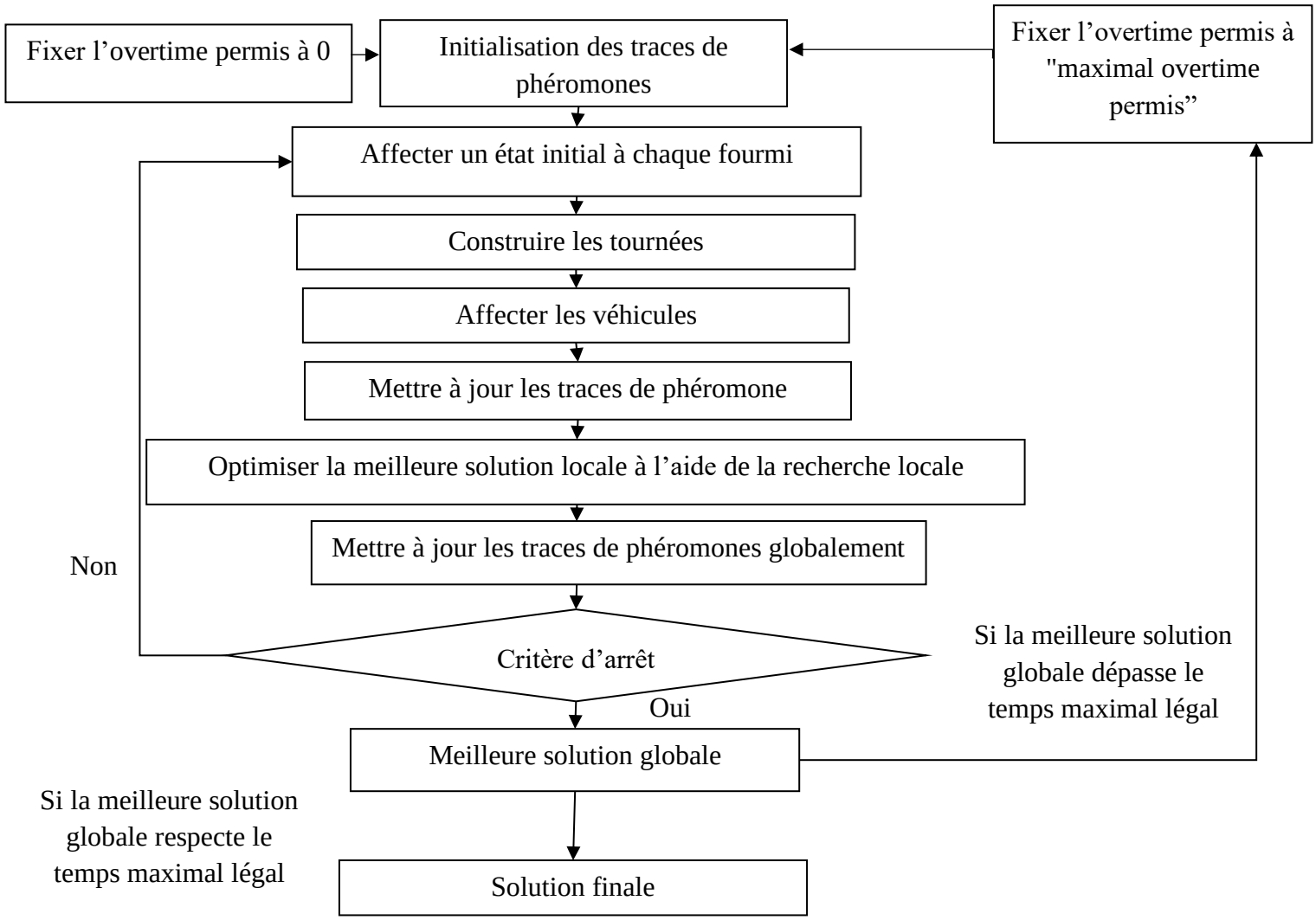

Figure 13 : Organigramme des étapes de l'ACSH pour le MTDVRPOT

3.3.1 ACSH pour le MTDVRPOT

Nous résolvons, d'abord, le problème statique qui contient les demandes de la veille arrivants après l'heure T_{co}. Les étapes de résolution de ce problème sont semblables à celles décrites dans la section 3.2 sauf que dans ce cas, nous essayons, d'abord, de trouver une solution réalisable qui respecte le temps normal en minimisant uniquement la distance. Si une telle solution n'a pas été trouvée, nous passons à la minimisation de l'overtime maximal en respectant l'overtime maximal permis. La figure 13 présente les étapes de cet algorithme.

Pour les problèmes dynamiques, nous suivons les mêmes étapes de la figure 13. En ce qui concerne, l'algorithme de recherche locale (section 3.2.1.5) ainsi que la mise à jour des traces de phéromone (section 3.2.1.3), nous allons utiliser les mêmes procédures utilisées pour le VRPMTOT statique (section 3.2). Les autres phases sont expliquées dans les sections 3.3.1.1, 3.3.1.2 et 3.3.1.3.

3.3.1.1 Phase d'initialisation

Les traces de phéromones sont initialisées sur tous les arcs du réseau selon le mécanisme de conservation des traces de phéromone proposé par Montemanni et al. (2005) :

$$\tau_{ij} = (1 - \gamma_r)\tau_{ij}^{old} + \gamma_r \tau_0$$

Avec

τ_{ij}^{old}: Taux de phéromone sur l'arc (i, j) à la fin de l'optimisation du problème de l'intervalle de temps précèdent.

γ_r: Facteur positif inférieur à 1 qui régularise la conservation du taux de phéromone.

τ_0 : Constante initiale de phéromone pour initialiser la phéromone sur les arcs correspondants aux nouveaux clients.

3.3.1.2 Transition des fourmis

Arès la phase d'initialisation, nous affectons une fourmi à chaque dépôt fictif. La fourmi commence à construire ses tournées en visitant les clients un après l'autre jusqu'à ce que la capacité du véhicule ne supporte plus ou que son temps dépasse la durée restante de l'horizon de planification. Dans ce cas, la fourmi revient au dépôt central et se déplace vers le plus proche dépôt fictif, non encore visité, pour commencer une autre tournée. S'il reste encore des clients non visités après que la fourmi passe par tous les dépôts fictifs, elle revient au dépôt central pour commencer une nouvelle tournée. Elle continue ainsi jusqu'à ce que tous les clients soient visités. La transition d'un client à un autre suit les mêmes règles expliquées dans la section 3.2.1.2.

3.3.1.3 Affectation des véhicules

Dans le cas dynamique, l'affectation des tournées aux véhicules est faite selon la position de chaque véhicule au début de la période traitée. En effet, dans un premier temps, nous affectons les tournées qui commencent à partir des dépôts fictifs aux camions correspondants. Ensuite, les tournées restantes sont triées pour être affectées d'une façon itérative. La procédure complète d'affectation des véhicules pour les problèmes dynamiques est expliquée dans la figure 14

Affecter les tournées commençant depuis un dépôt fictif au camion correspondant

Ordonner les tournées restantes depuis la plus longue à la plus courte ;

Tant que (Il y a des tournées non affectées)

Ordonner les camions selon la distance totale de leurs tournées, dans un ordre ascendant ;

Affecter la première tournée au premier camion ;

Supprimer la première tournée de la liste des tournées non affectées ;

Fin tant que

Figure 14: Affectation des véhicules pour les problèmes dynamiques

3.3.1.4 Évaluation

Le choix de la meilleure solution de l'itération et la meilleure solution globale se fait selon l'objectif visé dans l'optimisation. Au début, on vise la minimisation de la distance. Dans cette phase on ne considère que le temps normal du travail (c'est-à-dire qu'aucun overtime n'est permis). Si on arrive à trouver une telle solution, on choisit toujours celle qui minimise la distance. Sinon on permet l'utilisation de l'overtime, dans la limite de l'overtime maximal permis, et on vise à minimiser le deuxième objectif du problème. Dans ce cas, la meilleure solution est celle qui minimise l'overtime maximal réalisé.

À l'égard de l'HMEI, on essaye d'abord de minimiser la distance, mais une fois, l'algorithme n'arrive pas à trouver une solution qui respecte le temps normal du travail, on ne vise que la minimisation de l'overtime dans toutes les périodes suivantes (figure 7).

3.3.2 Test de l'ACSH sur le CDVRP

Avant de résoudre le MTDVRPOT, nous avons besoin de valider la compétitivité de l'ACSH sur des benchmarks de la littérature. Kilby et al. (1998) ont proposé des benchmarks pour le CDVRP sur lesquels plusieurs chercheurs ont testé leurs approches. Dans cette section, nous décrivons brièvement le CDVRP sous sa forme classique ainsi que les benchmarks de test. Ensuite, nous présenterons les résultats des tests du ACSH sur ces benchmarks en les comparant avec les résultats de Montemanni et al. (2005).

3.3.2.1 Description du CDVRP classique

Le CDVRP classique est décrit comme suit : on doit servir un ensemble de clients à partir d'un seul dépôt. Ces clients arrivent dynamiquement durant la journée de travail. Chaque client i demande une quantité q_i du même produit. On dispose d'une flotte largement suffisante de véhicules homogènes à capacité limitée Q. Un temps de service s_i est associé au client i. Chaque client doit être servi en une seule fois. Le but est de trouver une solution constituée d'un ensemble de tournées visitant tous les clients tout en minimisant la distance totale parcourue. Une solution est réalisable si :

- Chaque client est visité exactement une fois (c'est-à-dire qu'il est inclus dans une tournée unique).
- Chaque tournée commence et se termine au dépôt.
- Le temps consommé durant chaque tournée ne dépasse pas le temps légal de travail
- La somme des quantités commandées par les clients d'une même tournée ne dépasse pas la capacité totale du véhicule.
- Chaque véhicule effectue une seule tournée par jour.

Ce DVRP peut être modélisé en tant que série de VRPs statiques. Chacun d'eux comprend tous les clients connus et non encore servis au moment de l'optimisation.

3.3.2.2 Benchmarks de Kilby

Les benchmarks du CDVRP abordé dans cette partie ont été proposés à l'origine par Kilby et al. (1998). Ils ont dérivé ces jeux de tests d'autres benchmarks conçus initialement pour le VRP statique, à savoir ; 12 problèmes de Taillard, (1993), 7 problèmes de Christofides & Beasley, (1984) et 2 problèmes de Fisher, (1994). La taille de ces problèmes varie entre 50 et 199 clients. Le nom de chaque instance indique le nombre de clients qu'elle contient. Kilby et al. (1998) ont modifié les caractéristiques suivantes :

- Durée de la journée de travail.
- Heure d'apparition de chaque commande : Elle correspond au moment où la commande est devenue connue par le dispatcheur.
- Temps de service de chaque commande : Il représente, la durée nécessaire pour servir le client correspondant.
- Nombre de véhicules : Ils ont défini un nombre de 50 véhicules pour tous les problèmes. Une flotte de taille 50 est largement suffisante pour servir tous les clients durant le temps normal et sans avoir besoin de réutiliser les véhicules.

Ensuite, Montemanni et al. (2005) ont fixé la valeur des deux paramètres T_{ac} et T_{co} à T_{ac}= 0 et T_{co} = 0,5∗ T , avec T : la durée de la journée de travail. Les détails de ces problèmes sont présentés dans le tableau 5.

Tableau 5 : Problèmes de Kilby et al. (1998)

Instance	Nombre de clients	Nombre de véhicules	Capacité des véhicules	Durée de travail	Temps de service
c50	50	50	160	351	15
c75	75	50	140	346	16
c100	100	50	200	399	12
c100b	100	50	200	468	12
c120	120	50	200	794	13
c150	150	50	200	399	10
c199	199	50	200	399	9
f71	71	50	30 000	211	5
f134	134	50	2210	11741	13
tai75a	75	50	1445	769	32
tai75b	75	50	1679	905	26
tai75c	75	50	1122	782	25
tai75d	75	50	1699	789	27
tai100a	100	50	1409	897	30
tai100b	100	50	1842	799	29
tai100c	100	50	2043	905	21
tai100d	100	50	1297	782	23
tai150a	150	50	1544	1062	30
tai150b	150	50	1918	988	27
tai150c	150	50	2021	1081	23
tai150d	150	50	1874	1025	26

3.3.2.3 Résultats numériques

Pour les paramètres de l'ACSH, nous adoptons les mêmes paramètres ($q_0, \beta,\ \rho, \tau_0$ et nombre d'itération) utilisés pour le VRPMTOT statique. Le seul changement concerne le τ_0 pour les problèmes dynamiques où nous posons $\tau_0 = \frac{1}{n_d * Cost(PI)_p}$, avec n_d : *le nombre de clients du problème*

de la période suivante et $Cost\ (PI)_p$ est *la distance de la solution de la période précédente*. Le tableau 6 présente les résultats des tests de l'ACSH sur les problèmes de Kilby et al. (1998) en comparaison avec les résultats de Montemanni et al. (2005). Cinq exécutions ont été réalisées pour chacun des algorithmes.

Meilleure : représente la fonction fitness de la meilleure solution trouvée ; *Moyenne* : représente la moyenne des fitness et *Temps* : représente le temps d'exécution en minute.

Tableau 6 : Résultats numériques de l'ACSH sur les problèmes de Kilby et al. (1998) en comparaison avec Montemmani 2005

	ACSH			**Montemanni 2005**		
	Meilleure	**Moyenne**	**Temps**	**Meilleure**	**Moyenne**	**Temps**
c50	685,81	708,64	0,20	631,3	681,86	4,10
c75	1077,39	1133,46	0,31	1009,36	1042,39	4,10
c100	1052,47	1078,13	1,93	973,26	1066,16	4,10
c100b	948,74	980,12	1,06	944,23	1023,6	4,10
c120	1279,32	1361,18	6,82	1416,45	1525,15	4,10
c150	1459,61	1535,27	4,56	1345,73	1455,5	4,10
c199	1827,81	1888,88	5,64	1771,04	1844,82	4,10
f71	316,6	321,05	4,33	311,18	358,69	4,10
f134	15675,09	16228,39	4,18	15 135,51	16083,56	4,10
tai75a	1821,53	1916,60	0,39	1843,08	1945,2	4,10
tai75b	1555,78	1641,10	0,25	1535,43	1704,06	4,10
tai75c	1556,16	1632,73	0,66	1574,98	1653,58	4,10
tai75d	1514,16	1553,36	0,73	1472,35	1529	4,10
tai100a	2225,45	2392,01	0,96	2375,92	2428,38	4,10
tai100b	2384,79	2446,73	0,73	2283,97	2347,9	4,10
tai100c	1662,35	1720,5	1,18	1562,3	1655,91	4,10
tai100d	2008,47	2094,65	2,96	2008,13	2060,72	4,10
tai150a	3368,62	3465,25	4,63	3644,78	3840,18	4,10
tai150b	3082,21	3254,41	5,65	3166,88	3327,47	4,10
tai150c	2842,62	2968,38	5,7	2811,48	3016,14	4,10
tai150d	3253,3	3321,26	3	3058,87	3203,75	4,10
Total	51 598,28	53 642,2	55,95	50876,23	53794,02	86,24

Le GAP permet de mesurer la différence relative entre deux solutions en comparant leurs fonctions objectives.

$$GAP(s/s') = 100 * \left(\frac{f(s)}{f(s')} - 1\right)$$

Dans cette comparaison $f(s)$ sera remplacée par la fonction objectif de l'ACSH alors que $f(s')$ représentera la fonction objectif de Montemanni 2005.

Tableau 7 : Max. et Min. du GAP de l'ACSH par rapport à Montemanni 2005

	Meilleure	**Moyenne**
Max.	7,94 %	8,03 %
Min.	-10,71 %	-12,04 %
Total	1,39 %	-0,28 %

En termes de fitness, l'ACSH surpasse l'approche de Montemanni 2005 en 6 instances dans la meilleure valeur et en 10 instances dans la moyenne valeur.

On remarque que l'ACSH performe mieux sur les instances de Taillard, (1993) qui sont caractérisées par une distribution mixée uniforme et groupée autour de la zone de service. Dans ces instances, l'ACSH a pu trouver la meilleure solution pour 5 instances et la meilleure valeur moyenne pour 6 instances.

Outre, le tableau 7 reporte la valeur minimale et maximale du GAP, ainsi que le GAP des valeurs totales. Le GAP maximal ne dépasse pas 8,03 %, alors qu'il atteint -12,04 % comme valeur minimale. Pour le total, on peut dire que les deux algorithmes sont comparables puisque le GAP est de 1,39 % pour la meilleure valeur et de -0,28 % pour la valeur moyenne.

3.3.3 Test de l'ACSH sur le MTDVRPOT

3.3.3.1 Jeux de test pour le MTDVRPOT

Pour tester nos algorithmes sur le MTDVRPOT, nous avons besoin des jeux de test adoptés au cas dynamique tout en permettant de vérifier le deuxième objectif du problème (overtime maximal réalisé). Pour répondre à ce besoin, nous proposons des jeux de tests qui combinent le caractère dynamique des benchmarks de Kilby et al. (1998) ainsi que les contraintes sur la flotte et l'horizon du temps présentes dans les benchmarks de Taillard et al. (1996). Dans, ces jeux de tests, nous conservons les mêmes demandes et les mêmes capacités considérées dans les benchmarks de Taillard et al. (1996). Les instances sont générées en proposant plusieurs valeurs de m (nombre de véhicules

disponibles) et des valeurs restreintes de l'horizon temporel T. On pose T=[1,1*z^*/m], avec z^* est la valeur de la meilleure solution trouvée par Rochat & Taillard (1995) pour le VRP statique. L'heure d'arrivée des demandes est proportionnelle à l'heure d'arrivée de Kilby et al. (1998). De plus, l'overtime maximal autorisé pour chaque instance est fixé à un quart de l'horizon temporel normal, et le temps et la distance sont considérés comme équivalents. Pour chaque instance, nous effectuons les tests pour m entre 1 et 5. Pour distinguer ces jeux de test des problèmes de Kilby et al. (1998), la première lettre de chaque problème est mise en majuscule (Tableau 8). Par exemple, l'instance C100-3 correspond à l'instance où m = 3 du problème C100. Une solution est réalisable si l'overtime maximal réalisé ne dépasse pas l'overtime maximal autorisé.

Tableau 8 : Jeux de test pour le MTDVRPOT

Problème	**T**	**Taille**
CMT1	577	50
CMT2	919	75
CMT3	909	100
CMT4	1131	150
CMT5	1421	199
CMT11	114	120
CMT12	902	100
F71	266	71
F134	12 979	134
Thai75a	1780	75
Thai75b	1479	75
Thai75c	1420	75
Thai75d	1502	75
Thai100a	2245	100
Thai100b	2134	100
Thai100c	1547	100
Thai100d	1739	100
Thai150a	3361	150
Thai150b	3000	150
Thai150c	2595	150
Thai150d	2910	150

3.3.3.2 Résultats numériques

Les résultats numériques sont détaillés dans l'Annexe C, tableau 21. Dans ce paragraphe nous présentons une interprétation de ces résultats.

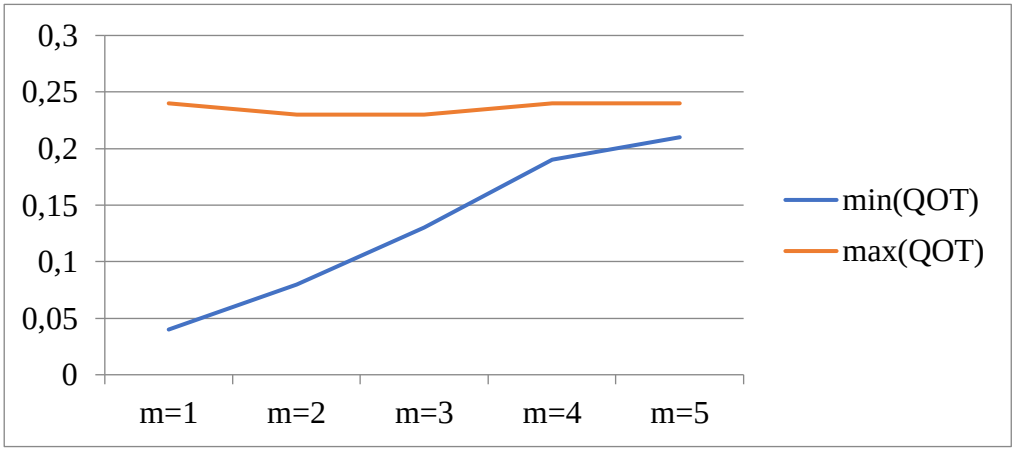

Figure 15 : Max. et min. de la valeur du QOT dépendamment du m dans l'ACSH

Dans un premier temps nous allons examiner une éventuelle corrélation entre l'overtime et la valeur de *m*. Pour ce faire, nous définissons le QOT par :

$$\text{QOT} = \text{Overtime maximal performé/Temps normal de travail}$$

Pour chaque instance, nous avons calculé la valeur moyenne des QOT des trois solutions obtenues. Le graphe de la figure 15 montre la valeur minimale et maximale de la moyenne du QOT en fonction de *m*. Il est clair que plus le *m* augmente plus le QOT minimal augmente. Cependant, la valeur maximale n'est pas très significative par rapport à m. Néanmoins, nous précisons que la valeur maximale du QOT pour $m = 1$ a été obtenue pour l'instance (CMT11-1) qui n'a pas de solution réalisable pour les autres valeurs de *m*. Si nous excluons cette instance de la comparaison, la valeur maximale devient 0,2 (Thai100d). En adoptant cette exclusion, nous pouvons conclure que le max. et le min. du QOT augmentent avec m. Cela se justifie également par le fait qu'on n'arrive pas à trouver des solutions réalisables pour les grandes valeurs de *m*.

Par ailleurs, les cases ombrées (Tableau 21, Annexe C) sont des résultats où la valeur minimale de la distance totale parcourue et la valeur minimale de l'overtime performé ne correspondent pas à la même solution. Selon la procédure d'évaluation adoptée par l'ACSH, si nous visons le deuxième objectif du problème, une solution peut être meilleure si elle réalise un overtime minimal malgré que sa distance ne soit pas minimale. Nous disons que les deux objectifs sont cohérents pour une instance donnée si la meilleure solution (parmi les trois exécutions effectuées) trouvée pour cette instance est une solution qui atteint la distance minimale et l'overtime minimal. Sinon, nous disons que les

objectifs ne sont pas cohérents. Ce cas n'est vrai que si la meilleure solution ne présente pas la distance totale parcourue minimale. Nous définissons le taux de cohérence comme étant le nombre d'instances pour lesquelles les deux objectifs sont cohérents sur le nombre total d'instances résolues :

$$\text{Taux de cohérence} = \frac{\text{Nombre d'instances où les deux objectifs sont cohérents}}{\text{Nombre total d'instances résolues}}$$

Il y a 8 instances sur 62 qui ne réalisent pas cette correspondance ; ce qui représente 13 % des cas. Ainsi dans 87 % des cas, une solution qui minimise l'overtime maximal performé minimise également la distance totale. Ceci étant, le taux de cohérence est de 87 % (figure 16).

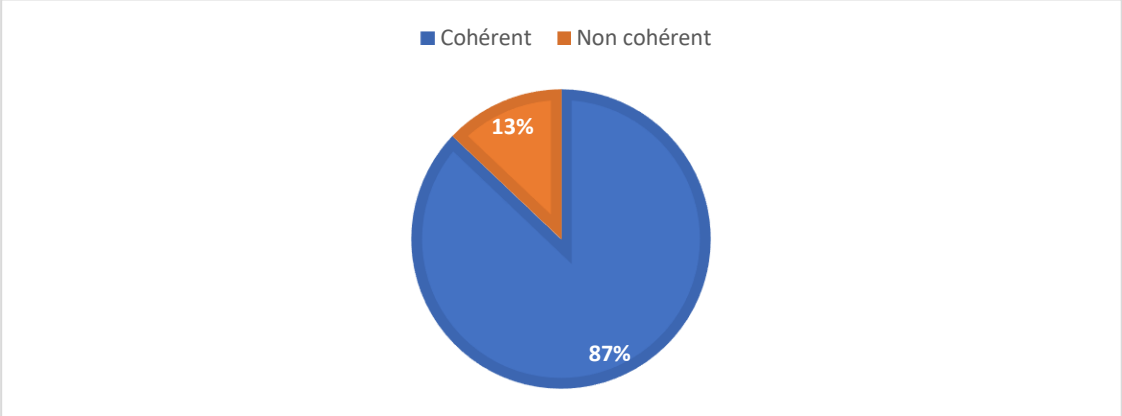

Figure 16 : Cohérence des objectifs du problème pour l'ACSH

3.4 Conclusion

Dans ce chapitre, nous avons présenté une démarche à base de la métaheuristique du système de colonies de fourmis (ACSH). La démarche dédiée au cas statique a pu donner des résultats encourageants surtout, en termes du temps d'exécution. Quant au cas dynamique, l'algorithme proposé a réalisé des résultats assez compétitifs en comparaison avec celles de Montemanni et al. (2005). Ensuite nous avons appliqué l'ACSH sur des jeux de tests conçus pour le MTDVRPOT. Pour mieux évaluer ces résultats, nous avons besoin d'une autre approche appliquée sur les mêmes jeux de tests. Nous proposons ainsi, au chapitre suivant, la résolution du MTDVRPOT par l'algorithme mémétique.

Chapitre 4

Algorithme mémétique pour la résolution du MTDVRPOT

4.1 Introduction

Hart et al., (2005) ont défini les algorithmes mémétiques (MA) comme étant des algorithmes évolutionnistes qui appliquent un processus de recherche local pour affiner les solutions des problèmes difficiles ; "Memetic algorithms are evolutionary algorithms that apply a local search process to refine solutions to hard problems". Le nom "algorithme mémétique" a été utilisé, en premier, par Moscato, (1989) pour désigner les algorithmes qui utilisent des hybridations entre les procédures génétiques telles que le croisement et la mutation avec les procédures de recherches locales.

Selon Krasnogor & Smith, (2005), il est bien connu que les algorithmes évolutionnistes (EA) purs ne sont pas bien adaptés pour affiner la recherche dans les espaces combinatoires complexes, mais s'ils sont combinés avec d'autres approches, ils peuvent grandement améliorer l'efficacité de la recherche. Un algorithme mémétique (MA) est une combinaison d'un EA avec la recherche locale. Cette combinaison a démontré son efficacité par rapport aux EAs traditionnelles dans plusieurs problèmes d'optimisation (Merz & Freisleben, 1999). Le VRP fait partie des problèmes sur lesquels le MA a été appliqué et a pu donner des résultats de haute qualité (Ayadi & Benadada, (2013) ; Mańdziuk & Żychowski, (2016)). En conséquence, les MAs sont de plus en plus acceptés surtout pour les problèmes d'optimisation combinatoire auxquelles appartient le MTDVRPOT.

Dans ce chapitre nous présentons un algorithme mémétique pour le MTDVRPOT. Mais avant de tester cet algorithme sur les jeux de tests du MTDVRPOT, nous l'appliquons sur les benchmarks de Kilby et al. (1998) pour le comparer avec d'autres approches de la littérature.

4.2 MA pour le MTDVRPOT

Dans cette section, nous expliquerons les détails de l'algorithme mémétique (MA) que nous proposons pour la résolution du MTDVRPOT. La figure 17 présente les étapes de cet algorithme. Tout d'abord, nous créons la population initiale à l'aide de l'heuristique d'insertion.

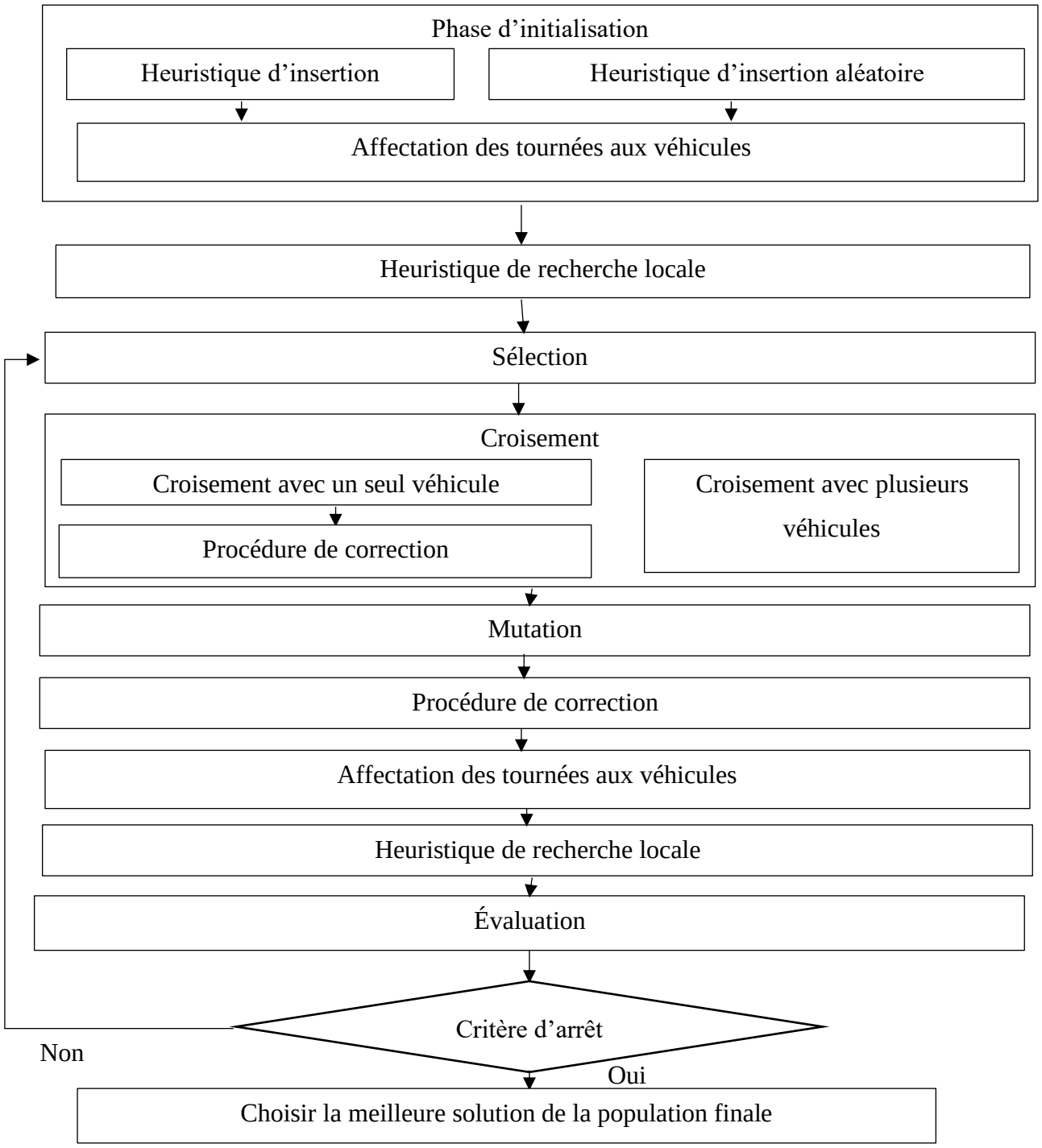

Figure 17 : Étapes du MA

Si le nombre de solutions qu'elle génère est inférieur au nombre requis pour compléter la population initiale, nous appliquons l'heuristique d'insertion aléatoire pour générer de nouveaux individus. Les tournées créées sont attribuées aux véhicules disponibles en fonction de la capacité de chaque véhicule et de la durée restante de la journée de travail. Ensuite, nous appliquons la recherche locale sur les solutions générées afin d'améliorer leurs qualités. L'étape suivante est de sélectionner deux individus de la population. En fonction de la probabilité de croisement (P_c), nous décidons, si ces individus subissent l'étape de croisement ou bien qu'ils passent directement à la mutation. Deux

types de croisement sont utilisés selon le nombre de véhicules disponibles. Si ce nombre est supérieur à un, nous appliquons le croisement à plusieurs véhicules. Sinon, nous utilisons le croisement à un seul véhicule. Si le croisement à un seul véhicule est terminé et que les individus résultants ne respectent plus les contraintes, nous corrigeons les défauts à l'aide de la procédure de correction et nous appliquons la recherche locale avant de passer à la mutation. La mutation est appliquée, ensuite, avec une probabilité de mutation (P_m). Les individus résultants sont corrigés à l'aide de la procédure de correction. Nous affectons les tournées aux véhicules et la recherche locale est appliquée à nouveau pour améliorer la qualité des solutions.

Pour résoudre le problème statique de la première période de temps, nous utilisons l'algorithme de Ayadi & Benadada, (2013). Dans ce qui suit, nous expliquons les procédures utilisées dans la résolution des problèmes dynamiques.

4.2.1 Conception des chromosomes

Les chromosomes de la population devraient refléter efficacement toutes les informations essentielles de la solution. Dans le cas du MTDVRPOT, qui est un problème multi-tour et multi-dépôt, la solution doit représenter les tournées de chaque véhicule, l'ordre de visite des clients de

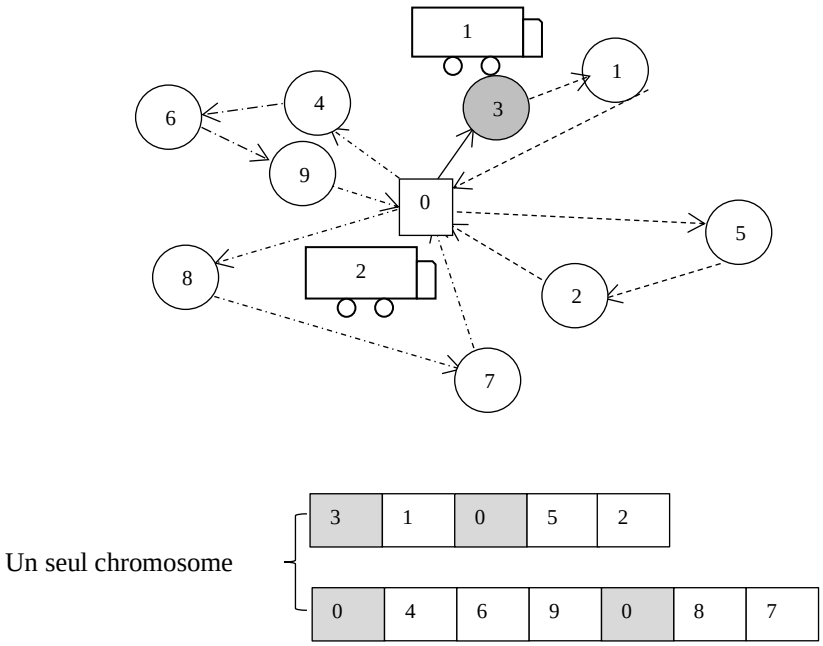

Figure 18 : Codage des chromosomes

chaque tournée ainsi que le dépôt (fictif ou central) du départ. Pour cette raison, nous adoptons un codage multiligne. Chaque ligne représente un voyage complet d'un seul véhicule. Nous désignons par le mot voyage, l'ensemble des tournées effectuées par le même véhicule. Puisqu'on optimise le

problème d'une période intermédiaire, le voyage peut commencer depuis le dépôt central ou bien depuis un dépôt fictif.

La première case représente le dépôt dans lequel se trouve le camion au début de la période de temps. Les autres cases représentent les clients visités en ordre. Un délimiteur (marqué 0) indique le dépôt central pour séparer les tournées effectuées par le même camion. La figure 18 illustre un exemple de codage pour K = 2.

4.2.2 Initialisation de la population

4.2.2.1 Heuristique d'insertion

Le principe de cette heuristique est de chercher toutes les insertions possibles des nouveaux clients dans la partie restante de la solution précédente. Nous cherchons toutes les insertions réalisables du premier client dynamique (nouveau client). Dans chaque solution générée, nous cherchons toutes les insertions réalisables du second client dynamique. En appliquant la même procédure de manière itérative aux clients dynamiques restants, nous construisons un ensemble de solutions réalisables que nous utilisons, par la suite comme population initiale de l'algorithme mémétique (figure 19).

Heuristique d'insertion
$C = \{c_1, c_2, .., c_n\}$ Liste des clients dynamiques
$S = \{t_1, t_2, .., t_m\}$ Tournées du reste de la solution de la période précédente.
Trouver toutes les insertions possibles de c_1 dans S. S_1 est l'ensemble des solutions résultantes
Pour chaque *i* de 2 à *n*
Trouver toutes les insertions possibles de c_i dans chaque solution de S_{i-1}. S_i est l'ensemble des solutions résultantes
Fin Pour
Retourner S_n

Figure 19 : Heuristique d'insertion

4.2.2.2 Heuristique **d'insertion aléatoire**

Cette heuristique n'est utilisée que si le nombre de solutions générées par l'heuristique d'insertion est inférieur au nombre requis pour compléter la population initiale. Depuis chaque dépôt fictif, nous créons une tournée. Les clients sont sélectionnés l'un après l'autre de façon aléatoire. Pour chaque client sélectionné, nous effectuons la première insertion possible dans l'une des tournées précédemment créées. Si aucune insertion n'est possible, une nouvelle tournée, qui commence depuis

le dépôt central et qui visite ce client, est créée. L'algorithme s'arrête lorsque tous les clients sont insérés (Figure 20).

4.2.3 Croisement

Le Croisement est une opération cruciale dans un algorithme génétique. Cet opérateur combine deux chromosomes appelés « parents » pour produire des enfants. Nous utilisons deux méthodes de croisement, selon le nombre de véhicules disponibles. Le premier est appliqué si on ne dispose que d'un seul véhicule (Croisement à un seul véhicule), et le second est utilisé pour plus de véhicules (Croisement à plusieurs véhicules).

Heuristique d'insertion aléatoire

$C = \{c_1, c_2, .., c_n\}$ Ensemble de tous les clients connus et non encore servis.

$D = \{d_1, d_2, .., d_m\}$ Ensemble des dépôts fictifs.

Depuis chaque dépôt fictif de D, une tournée, sans clients, qui revient au dépôt central est créée.

$T = \{t_1, t_2, .., t_m\}$ Ensemble des tournées créées

Pour chaque c_i de C,

S'il y en a t_j de T où c_i peut-être inséré

$t'_j = t_j \cup \{c_i\}$

$T = (T|t_j) \cup t'_j$

$C = C|\{c_i\}$

Fin Si

Sinon

Créer une tournée t_{m+1} qui commence depuis le dépôt central, visite c_i et revient au dépôt central

$T = T \cup t_{m+1}$

$m=m+1$

Fin Sinon

Fin Pour

Retourner *T*

Figure 20 : Heuristique d'insertion aléatoire

4.2.3.1 Croisement à plusieurs véhicules

Pour le croisement à plusieurs véhicules, l'un des parents est considéré comme receveur et le second comme donneur. Le croisement est appliqué pour générer le premier enfant. Ensuite, les rôles sont inversés pour générer le second enfant. Les tournées du véhicule qui consomme le plus de temps

sont supprimées au parent receveur en conservant la première case si elle représente un dépôt fictif. Nous obtenons, ainsi, la première partie de l'enfant. Au niveau du parent donneur, on ne conserve que les clients qui ont été supprimés du receveur en marquant les cases représentant des dépôts fictifs par 0. C'est la deuxième partie de l'enfant. Les deux parties sont combinées, ensuite, pour former le nouvel enfant. L'exemple de la figure 21 illustre un exemple de ce croisement. L'enfant 1 est généré en considérant le parent 1 comme donneur et le parent 2 comme receveur alors que l'enfant 2 est généré en inversant les rôles entre les deux parents.

4.2.3.2 Croisement à un seul véhicule

Dans ce cas, chaque chromosome est représenté par une seule ligne. La première case présente le dépôt fictif dans lequel le véhicule est initialement stationné. Les autres cases présentent les clients visités. Les tournées sont séparées par une case 0. Nous choisissons deux tournées de façon aléatoire. La première tournée appartient au premier parent et la seconde au deuxième parent. Les clients de la première tournée remplacent ceux de la deuxième dans le deuxième parent et les clients de la deuxième tournée remplacent ceux du premier dans le premier parent tout en conservant le même dépôt fictif du parent. Chacun des enfants générés subit la procédure de correction. L'exemple de la figure 22 illustre ce croisement.

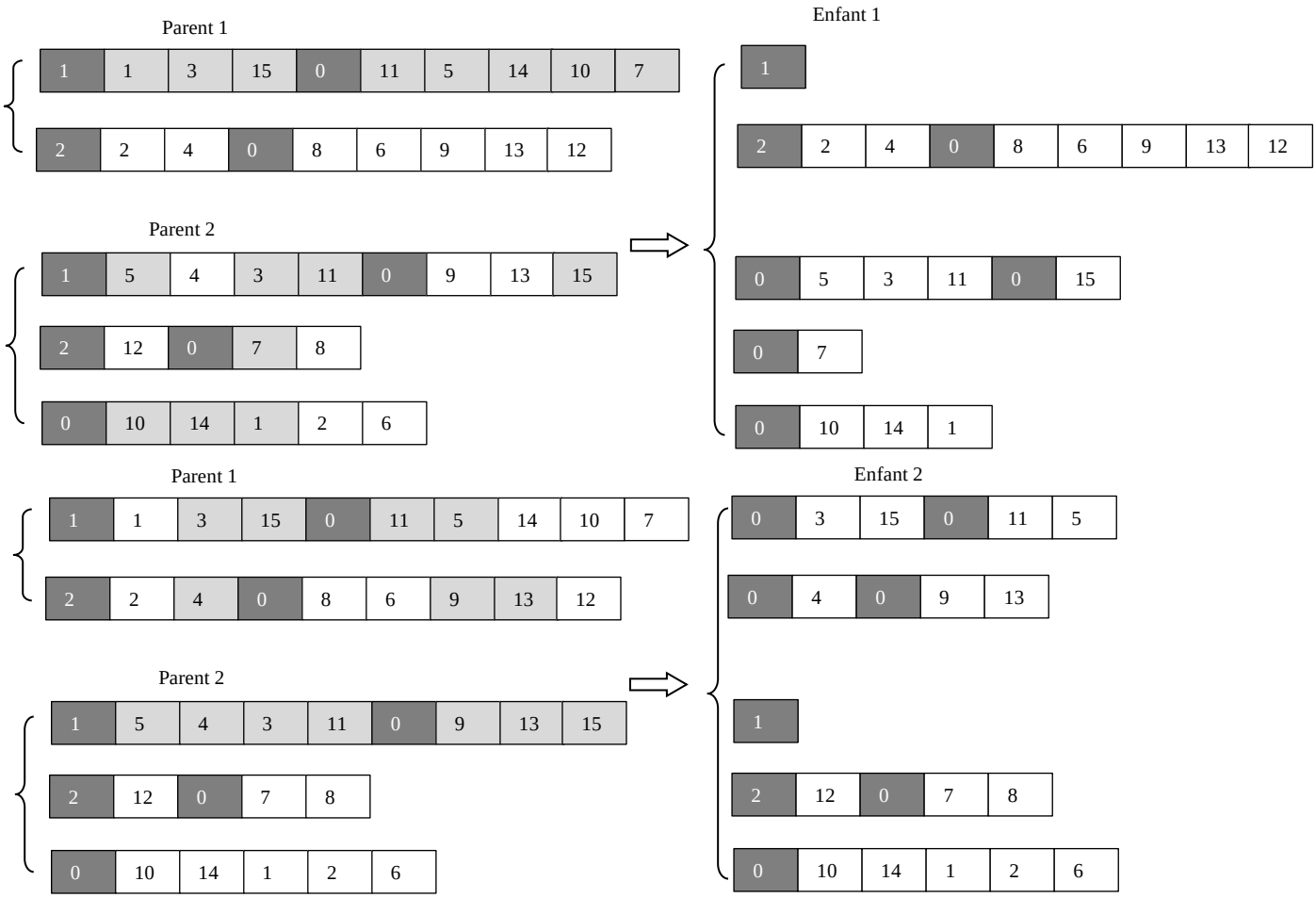

Figure 21 : Croisement à plusieurs véhicules

4.2.4 Mutation

Deux opérations aléatoires sont utilisées dans la phase de mutation pour obtenir une population avec des caractéristiques nouvelles en vue d'étendre la zone de recherche :

- • **Échange aléatoire :** cet opérateur choisit aléatoirement deux clients appartenant à deux tournées différentes et échange leurs positions.

- • **Insertion aléatoire :** cet opérateur choisit deux tournées, aléatoirement. Un client est sélectionné de la première tournée et inséré dans la seconde à une position aléatoire. L'exemple de la figure 23 représente l'opérateur de mutation.

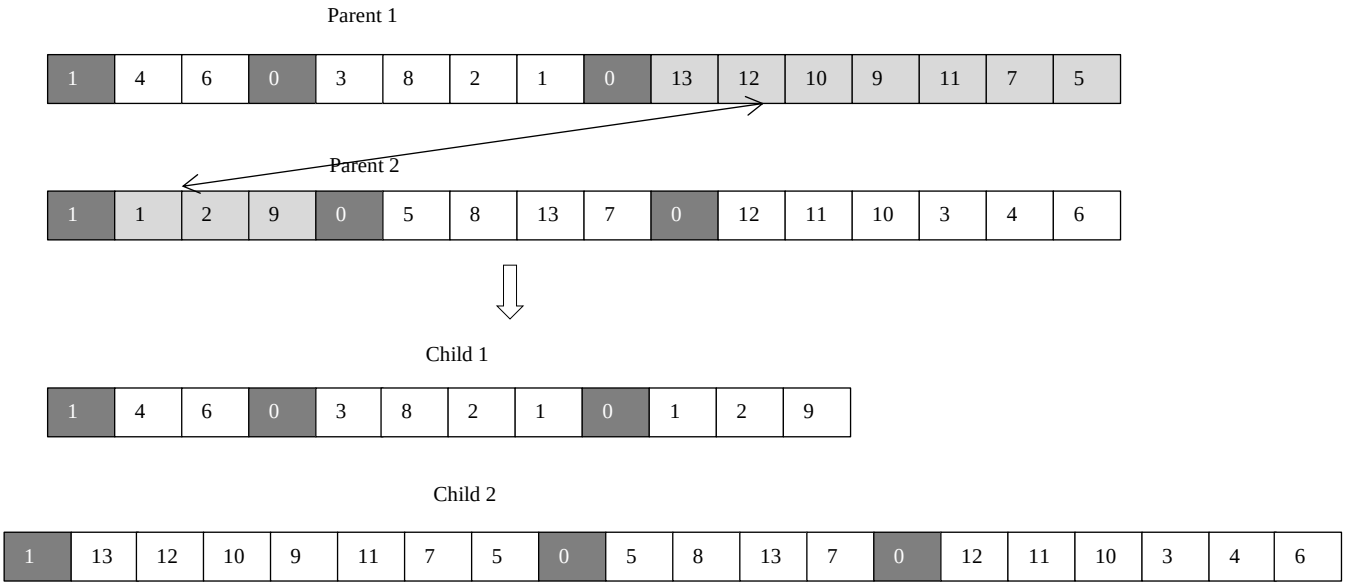

Figure 22 : Croisement à un seul véhicule

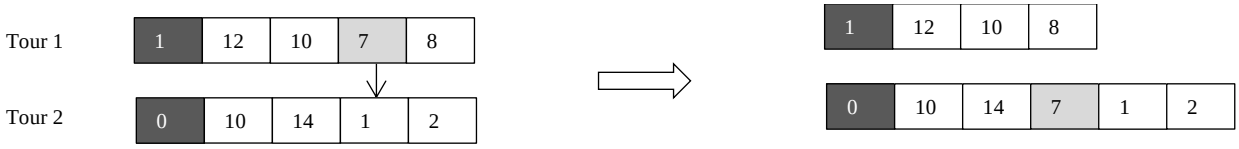

Figure 23. A : Insertion aléatoire

Figure 23.B : Échange aléatoire

Figure 23 : Opérateur de mutation

4.2.5 Procédure de correction

Cette procédure est appliquée aux individus résultants de la phase de croisement d'un seul véhicule et de mutation. Tout d'abord, les clients redondants sont supprimés et l'ordre de visite des clients de chaque tournée est amélioré en vue de minimiser le temps de routage. Si le temps de routage ou la quantité d'une tournée dépassent le temps limite (temps légal plus overtime permis) ou la capacité du véhicule, cette tournée sera divisée en deux ou trois petites tournées. Ensuite, nous essayons d'insérer les clients non visités dans l'une des tournées existantes. Si aucune insertion n'est possible, pour un client, nous lui créons une tournée dédiée qui commence depuis le dépôt central. Enfin, nous utilisons l'algorithme Clarke et Wright pour combiner les tournées courtes. La figure 24 montre la procédure de correction appliquée aux enfants résultants du croisement d'un seul véhicule de la figure 22.

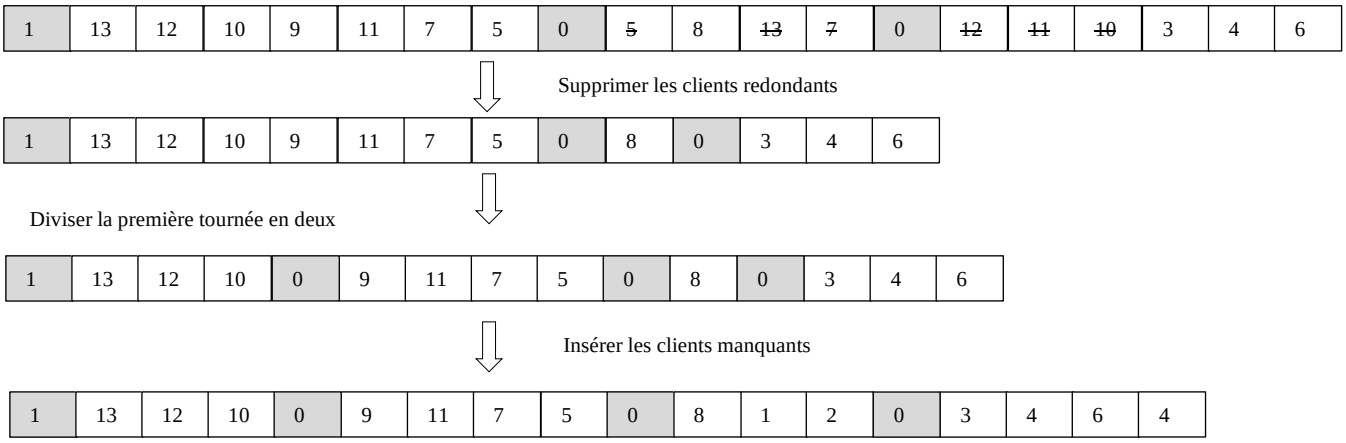

Figure 24 : Procédure de correction

4.2.6 Évaluation

Le MTDVRPOT est un problème multi-objectif. Ainsi, nous avons besoin de définir une relation de dominance afin d'évaluer et comparer les solutions obtenues. Dans notre cas, nous cherchons d'abord à minimiser la distance totale parcourue en respectant le temps légal de planification. Si nous n'arrivons pas à trouver une telle solution, nous passons au deuxième objectif qui consiste à minimiser l'overtime maximal réalisé. Ainsi, nous calculons dans un premier temps, la fitness (distance totale parcourue et overtime maximal réalisé) de chaque solution. Ensuite, les solutions sont ordonnées comme suit : si une ou les deux solutions, à comparer, dépassent le temps légal, celle qui réalise un overtime minimal est la meilleure. Si les deux solutions respectent la contrainte de temps, la meilleure est celle qui a une distance minimale.

4.3 Résultats numériques

4.3.1 Paramétrage du MA

La taille de la population est fixée à 100 pour le premier problème statique et à 20 pour les problèmes dynamiques. Pour la partie statique, la taille de la population (100) permet une meilleure exploration de la zone de recherche, car le temps d'exécution de cette période n'est pas strictement limité. Pour les problèmes dynamiques restants, la génération d'individus réalisables pour la population initiale prend un peu plus de temps. Si la taille de la population initiale est considérablement importante, nous risquons d'augmenter le temps d'exécution. Les tests préliminaires, qui ont été effectués, ont montré que le nombre 20 permet un équilibre entre le temps d'exécution et l'exploration de la zone de recherche. En outre, nous avons fait d'autres tests pour définir les valeurs à adopter pour les probabilités de mutation et de croisement. Les résultats du

tableau 9 (pour l'instance C50) montrent que P_m = 0,4 pour la probabilité de mutation et P_c = 0,9 pour la probabilité de croisement sont les valeurs les plus appropriées.

Tableau 9: Résultats des tests préliminaires sur la probabilité de croisement et de mutation

	P_m	P_c	P_m	P_c	P_m	P_c	P_m	P_c
	0.1	**0.9**	**0.1**	**0.7**	**0.4**	**0.9**	**0.4**	**0.7**
1	648,8		627,64		563,46		618,48	
2	641,89		619,12		616,59		616,22	
3	636,05		609,23		617,27		630,4	
4	616,39		620,23		593,76		624,53	
5	652,57		621,11		624,08		611,67	
Min.	616,39		609,23		563,46		611,67	
Moyenne	639,14		619,46		603,03		620,26	

Selon le cas traité, deux critères d'arrêt sont fixés pour le MA:

- Soit qu'il exécute 100 itérations sans aucune amélioration de la fonction objectif
- Soit qu'il exécute 200 itérations complètes.

Le MA est codé en Java et exécuté sur une machine (Intel Core i5) pour les tests sur le MTDVRPOT et sur une machine (Intel Core i7) pour les tests sur le CDVRP classique.

4.3.2 Test du MA sur le CDVRP classique

À l'égard de l'ACSH, nous testons d'abord le MA sur les benchmarks de Kilby et al. (1998) afin de comparer ses performances avec d'autres approches de la littérature. Le tableau 10 présente les résultats obtenus à l'aide du MA, de l'algorithme génétique de Hanshar & Ombuki-Berman, (2007) et du système de colonies de fourmis de Montemanni et al. (2005).

Tableau 10 : Résultats du MA sur CDVRP classique comparés avec ceux du Hanshar 2007 et de Montemmani 2005

	MA	**Hanshar 2007**	**Montemmani 2005**
c50	563,46	570,89	631,3
c75	977,73	981,57	1009,36
c100	967,02	961,1	973,26
c100b	889,88	881,92	944,23
c120	1258,19	1303,59	1416,45

c150	1282,55	1348,88	1345,73
c199	1623,09	1654,51	1771,04
f71	316,13	301,79	311,18
f134	1381,27	1552,88	1513,55
tai75a	1776,56	1782,91	1843,08
tai75b	1434,1	1464,56	1535,43
tai75c	1473,97	1440,54	1574,98
tai75d	1391,19	1399,83	1472,35
tai100a	2208,31	2232,71	2375,92
tai100b	2211,63	2147,7	2283,97
tai100c	1486,75	1541,28	1562,3
tai100d	1767,57	1834,6	2008,13
tai150a	3327,58	3328,85	3644,78
tai150b	3057,77	2933,4	3166,88
tai150c	2668,34	2612,68	2811,48
tai150d	3061,13	2950,61	3058,87

Comme le montre le tableau 10, le MA fournit de meilleurs résultats que l'algorithme génétique (Hanshar 2007) en 13 instances (62%) et l'algorithme de Montemanni 2005 en 19 instances (90%). De plus, le temps d'exécution maximal de notre MA ne dépasse pas 8,5 min, alors qu'il a touché 12,5 min dans le cas de Hanshar 2007.

On revient vers le GAP pour comparer les solutions trouvées par le MA avec celles trouvées par les méthodes de Hanshar 2007 et de Montemanni 2005. Le tableau 11 présente le GAP maximal et minimal lié à ces deux algorithmes. Le GAP maximal ne dépasse pas 5% dans le meilleur des cas, tandis que le GAP minimal atteint environ -12%. La figure 28 de l'annexe B décrit les tours de la solution qui réalise le GAP minimal par rapport à Hanshar 2007. Il s'agit d'une solution de l'instance f134.

Tableau 11: Min et Max GAP du MA relativement à Hanshar 2007 et Montemmani 2005

GAP	Hanshar 2007	Montemanni 2005
Max.	4,75%	1,59%
Min.	-11,05%	-11,97%

4.3.3 Application du MA sur le MTDVRPOT

Les résultats numériques sont détaillés dans l'Annexe C, tableau 21. Dans ce paragraphe nous présentons une interprétation de ces résultats.

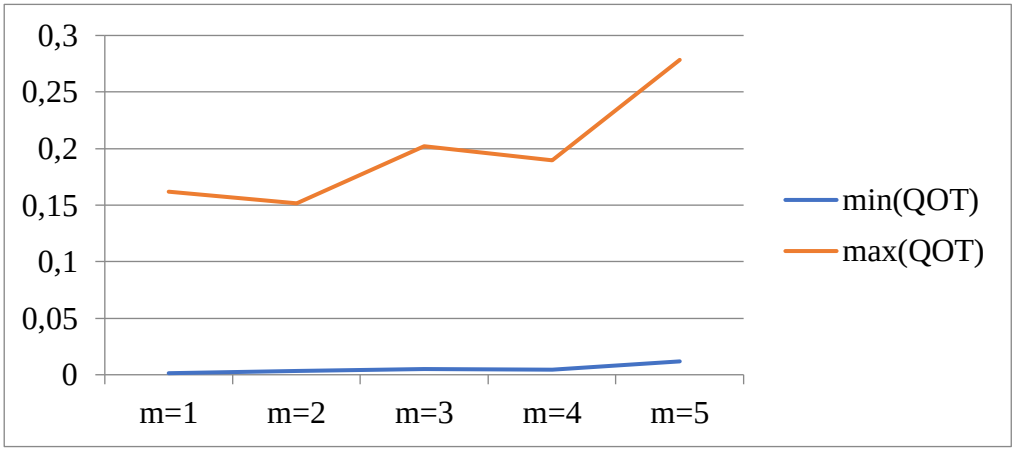

Figure 24: Max et Min de la valeur du QOT dépendamment du m dans le MA

Le graphe de la Figure 25 montre la valeur minimale et maximale de la moyenne du QOT en fonction de m. Dans le cas du MA, plusieurs solutions n'ont pas d'overtime. Les résultats de ces solutions ne sont pas reportés dans le graphe de la figure 25. Malgré les petites déviations de la courbe, il est clair que le max(QOT) augmente avec le m. La même remarque est vraie pour le min(QOT) sauf qu'il y une certaine stabilité pour m égale à 1, 2, 3 et 4.

Concernant la cohérence des objectifs du problème, la figure 26 présente en pourcentage les solutions où les deux objectifs sont cohérents et celles où ils sont incohérents. En conclusion les résultats du MA sur le MTDVRPOT sont cohérents à 92%

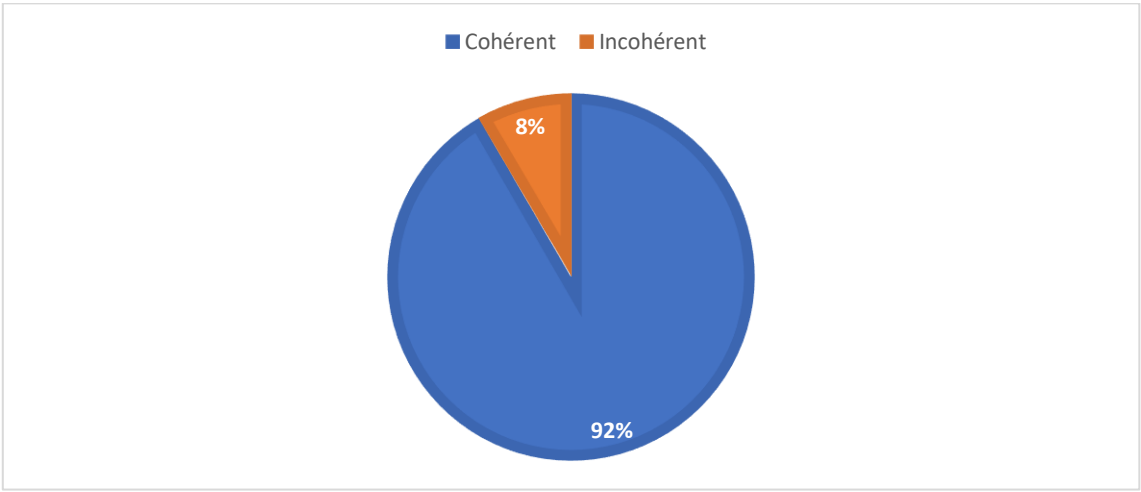

Figure 25 : Cohérence des objectifs pour le MA

4.4 Conclusion

Dans ce chapitre, nous avons proposé un algorithme mémétique pour le MTDVRPOT. Les opérations de croisement, de mutation et de recherche locale sont des adaptations des opérations d'un autre algorithme mémétique, de la littérature, conçu pour le problème statique. Cependant, les heuristiques de génération de la population initiale ainsi que l'affectation des véhicules sont développées pour la première fois pour le MTDVRPOT. Les résultats du MA sur le DVRP à capacité classique sont généralement meilleurs que les autres approches de la littérature. Finalement, les résultats sur le MTDVRPOT ont réalisé un taux de cohérence très élevé entre les deux objectifs du problème. Dans le chapitre suivant, nous allons présenter une comparaison entre les différentes démarches proposées, dans cette thèse, pour le MTDVRPOT.

Chapitre 5

Comparaison des algorithmes et discussion

5.1 Introduction

L'objectif de ce chapitre est de comparer les résultats des trois algorithmes entre eux. Or, les métaheuristiques sont testées sur des instances de grandes tailles puisque l'objectif principal d'une métaheuristique est de pouvoir résoudre des instances de grandes tailles, alors que les méthodes exactes ne supportent, généralement, que des instances de petite taille en garantissant l'optimum. Bien que l'HMEI proposé dans cette thèse est basé sur l'application itérative d'une méthode exacte, le caractère dynamique du MTDVRPOT ne permet pas qu'elle garantisse l'optimum.

Par ailleurs nous avons proposé deux types d'instances pour le MTDVRPOT ; les instances de petite taille (section 3.4) sur lesquelles nous avons testé l'HMEI et les instances de grande taille (section 4.2.3.1) sur lesquelles l'ACSH et le MA ont été appliqués. Dans ce chapitre, et afin de comparer l'ACSH et le MA avec l'HMEI, nous appliquons les métaheuristiques sur les petites instances puisqu'on ne peut pas tester l'HMEI sur les grandes. Outre, nous allons comparer les résultats, des grandes instances, obtenus par l'ACSH avec celle du MA. À la fin, nous allons discuter l'évaluation bi-objectifs proposée dans les contributions de cette thèse.

5.2 Comparaison des résultats sur les petites instances

Dans cette section, nous présentons les résultats de l'ACSH et du MA sur les petites instances et nous les comparons avec ceux du HMEI (tableau 12). Les solutions en gris représentent la meilleure solution trouvée pour chaque instance. Le choix de la meilleure solution est basé sur les mêmes critères d'évaluation utilisés dans les approches de résolution. C'est-à-dire que la meilleure solution est celle qui réalise la distance minimale si l'overtime réalisé est 0. Sinon, la meilleure solution est celle qui minimise l'overtime maximal réalisé.

Nous constatons que le MA a réussi à trouver la meilleure solution pour 12 instances. Quant à l'HMEI, il a pu trouver la meilleure solution pour 8 instances sachant qu'elle ne résout pas l'instance C-40-3. Alors que l'ACSH a pu fournir une seule meilleure solution (C-25-4). En plus, il réalise la distance minimale pour l'instance C-30-4, même si sa solution complète n'est pas la meilleure selon les critères d'évaluation adoptés.

Tableau 12 : Résultats des trois méthodes sur des petites instances

	m	T	C	HMEI		ACSH		MA	
				Distance	Overtime	Distance	Overtime	Distance	Overtime
C-25	1	577	80	512,16	0	525,32	0	512,16	0
	2	289	80	482,51	0	516,57	0	520,73	0
	3	192	80	506,85	0	499,07	0	497,96	0
	4	144	80	534,83	14,17	537,9	1,23	550,18	7,1
	5	115	80	527,58	21,76	539,18	9,76	548,03	7,5
C-30	1	577	80	541,42	0	538,28	0	523,73	0
	2	289	80	554,95	0	577,21	5,57	552,85	0
	3	192	80	535,75	0	571,41	0	534,76	0
	4	144	80	571,09	8,98	556,78	8,83	580,21	8,12
	5	115	80	543,64	4,54	584,98	15,47	572,6	7,72
C-36	1	577	160	500,37	0	558,21	0	527,25	0
	2	289	160	487,79	0	535,32	0	542,9	0
	3	192	160	551,64	2,29	540,02	0	532,09	0
	4	144	160	522,46	0	548,52	0	498,98	0
	5	115	160	518,49	0	599,1	21,6	549,4	6,1
C-40	1	577	160	539,12	0	556,28	0	536,49	0
	2	289	160	510,03	0	562,57	0	563,85	0
	3	192	160	-	-	619,79	20,26	578,01	5,38
	4	144	160	549,6	0	555,71	1,82	556,54	0
	5	115	160	572,11	5,1	622,73	17,27	557,18	4,7

Afin de comparer la performance des trois approches sur les instances de tests, nous avons calculé les valeurs du GAP des solutions obtenues par l'ACSH par rapport aux solutions de l'HMEI pour les deux objectifs du problème. Nous avons fait la même chose pour l'ACSH par rapport au MA et pour le MA par rapport à l'HMEI.

Le tableau 13 présente le Max, le Min et la moyenne de ces GAPs sachant qu'on ne rapporte que les instances pour lesquelles $f(s_1)$ et $f(s_2)$ sont toutes les deux différentes de 0. À partir de ces résultats, il est clair que l'HMEI donne les meilleurs résultats, puisque le GAP maximal par rapport aux deux autres approches est toujours plus grand que la valeur absolue du GAP minimal et la moyenne des GAPs est souvent positive. La seule exception où la moyenne du GAP est négative correspond à l'overtime du MA/HMEI. Toutefois cette valeur n'est pas très significative puisqu'il ne

représente que les solutions de 5 instances, parmi 20, où l'overtime du MA et du HMEI est strictement positif.

Tableau 13 : Max, Min et moyenne des valeurs du GAP

	ACSH/HMEI		ACSH/MA		MA/HMEI	
	Distance	**Overtime**	**Distance**	**Overtime**	**Distance**	**Overtime**
Min.	-2,5	-91,3	-4,0	-82,6	-4,4	-65,5
Max.	15,5	240,7	11,7	276,5	11,2	70
Moyenne	4,4	66,2	2,8	122,1	1,9	-12,5

D'une autre part, le MA donne de meilleurs résultats que l'ACSH avec un GAP maximal de 11,7 % pour la distance et de 276,5 % pour l'overtime. Alors que les valeurs minimales sont -4 % et -82,6 %. Les valeurs moyennes reflètent aussi la supériorité du MA.

Tableau 14 : Moyenne du temps d'exécution en secondes

	HMEI	**ACSH**	**MA**
Temps (s)	43129,9	4	7

Par ailleurs, le temps d'exécution du HMEI est incomparable avec les temps d'exécution des autres approches (tableau 14). Ceci est normal puisque l'HMEI est basé sur la résolution des modèles mathématiques avec des méthodes exactes. Cependant, le temps d'exécution du MA est très bas, tenant compte qu'il a pu réaliser la meilleure solution pour 60 % des problèmes et que ses résultats sont très comparables avec l'HMEI (voir la moyenne du GAP). Ainsi, si nous devons recommander l'une des trois méthodes pour résoudre des problèmes semblables, nous allons choisir le MA.

5.3 Comparaison des résultats des grandes instances

Dans ce paragraphe, nous comparons les résultats obtenus par le MA avec ceux obtenus par l'ACSH sur les instances décrites dans la section 3.3.3.1. Trois exécutions des deux algorithmes sont prises en compte pour chaque instance. Le tableau 15 fournit le nombre de solutions réalisables trouvées par chaque algorithme.

Tableau 15 : Nombre de solutions réalisables obtenues pour chaque problème par le MA et l'ACSH

	MA	ACSH
C50	4	4
C75	5	4
C100	5	5
C100b	5	4
C120	2	1
C150	5	0
C199	5	0
F71	3	2
F134	4	2
Tai75a	5	3
Tai75b	5	4
Tai75c	3	4
Tai75d	5	4
Tai100a	5	5
Tai100b	5	4
Tai100c	5	3
Tai100d	5	2
Tai150a	5	4
Tai150b	5	3
Tai150c	5	2
Tai150d	5	2

Le MA obtient des solutions réalisables pour 96 des 105 instances. Alors que l'ACSH aboutit à la réalisabilité en 62 instances uniquement. Parmi toutes les instances, il n'y en a qu'une seule pour laquelle l'ACSH a pu trouver une solution réalisable alors que le MA ne le peut pas (Thai75c-4). En outre, le MA fournit de meilleurs résultats que l'ACSH pour les deux objectifs du problème dans 94 instances, et l'ACSH fournit de meilleurs résultats que le MA pour seulement deux instances (C100 -2, Thai75c -1). Pour C100-2, la solution apportée par l'ACSH est meilleure en termes de distance ; cependant, l'overtime réalisé par le MA est meilleur. Par ailleurs, l'ACSH fournit une meilleure solution pour les deux objectifs dans l'instance Thai75c-1. Le tableau 16 présente les valeurs de GAP maximales et minimales du MA relativement à l'ACSH pour les deux objectifs du problème. Les résultats détaillés des deux algorithmes sont présentés dans l'Annexe C.

Tableau 16 : GAP relatif du MA par rapport à l'ACSH pour les deux objectifs du problème

GAP	Distance	Overtime
Min	-92,11 %	-100 %
Max	2,5 %	69,49 %

Étant donné que les deux algorithmes ont été exécutés sur la même machine, nous pouvons, nettement, comparer leurs temps d'exécution. Le tableau 17 montre le temps d'exécution moyen des instances de chaque problème pour les deux algorithmes. Aucun des deux algorithmes ne domine, complètement, en termes de temps d'exécution.

Tableau 17 : Temps d'exécution moyen en secondes

	MA	**ACSH**	**GAP**
C50	42,4	40,7	4,1 %
C75	78,2	42,5	84 %
C100	199	211	-5,6 %
C150	607,9	-	-
C199	1253,7	-	-
C120	513,3	1237	-58,5 %
C100b	196,3	193	1,7 %
F71	145,4	205,9	-29,3 %
F134	1266,3	6138,8	-79,3 %
Tai75a	62,4	42	48,5 %
Tai75b	156,6	55	184,7 %
Tai75c	125,4	40,6	208,8 %
Tai75d	201,3	81,4	147,2 %
Tai100a	302,7	112,7	168,5 %
Tai100b	288,9	117,3	146,2 %
Tai100c	252,8	202,5	24,8 %
Tai100d	389,7	206	89,1 %
Tai150a	584,6	328,5	77,9 %
Tai150b	979,1	614,4	59,3 %
Tai150c	1038,6	647	60,5 %
Tai150d	528,4	517,1	2,1 %

Pour plusieurs problèmes, l'ACSH fonctionne beaucoup plus rapidement que le MA. C'est le cas des problèmes : C75, Tai75b, Tai75c, Tai75d, Tai100a, Tai100b, Tai100d, Tai150a, Tai150b et Tai150c, dans lesquels le GAP dépasse -50 % et atteint 208 %. Ces problèmes constituent 47 % du nombre total de problèmes. En contrepartie, il existe des problèmes pour lesquels le MA fonctionne plus rapidement (C150, F134). Dans ces cas, le GAP est inférieur à -50 %. Cependant, leur nombre (9 % du nombre total de problèmes) reste minime par rapport au nombre d'instances dans lesquelles

l'ACSH obtient des résultats plus rapidement que le MA. Pour les problèmes restants, les deux algorithmes ont consommé un temps d'exécution comparable (-50 % <GAP <50 %). Il convient de noter que deux problèmes (C100b, C120) ne sont pas inclus dans cette comparaison puisque l'ACSH n'a trouvé aucune solution pour aucune de leurs instances.

Le dernier point à discuter dans cette comparaison est la cohérence des objectifs du problème dans les résultats des deux algorithmes. Nous avons déjà prouvé que les résultats du MA sont cohérents à 92 % alors que ceux de l'ACSH sont cohérents à 87 %. Cependant, ces pourcentages sont calculés selon le nombre de solutions réalisables trouvées par chaque algorithme. La figure 26 présente le nombre exact des solutions pour lesquelles les deux objectifs sont cohérents ou incohérents pour les deux algorithmes. Les deux algorithmes ont réalisé un nombre égal de solutions incohérentes. Ainsi la différence du taux de cohérence réalisé est, plutôt, due à la différence du nombre de solutions réalisables trouvées par chaque algorithme.

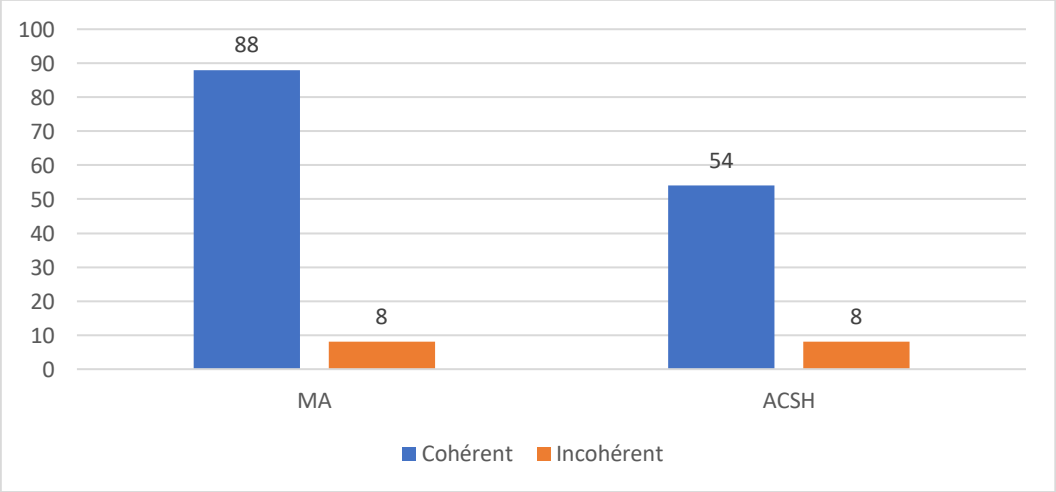

Figure 26 : Comparaison de la cohérence des résultats des deux algorithmes

5.4 Discussion de l'évaluation bi-objectifs du MTDVRPOT

On a proposé trois approches de résolution pour le MTDVRPOT. Bien qu'il s'agisse d'un problème bi-objectifs, aucune des approches proposées n'utilise la dominance Pareto pour évaluer les solutions. En fait, nous avons adopté des relations d'évaluation spécialement conçues pour le problème étudié. En examinant, les trois méthodes de résolution, nous trouvons que l'HMEI et l'ACSH utilisent la même procédure d'évaluation, alors que le MA applique une autre démarche. Les solutions de l'HMEI et l'ACSH sont évaluées d'une façon, purement, mono-objective, mais sur deux niveaux. Dans, le premier niveau, nous ne considérons que l'objectif de minimiser la distance. Si nous n'arrivons pas à trouver une solution qui respecte le temps normal du travail, nous considérons

l'objectif de minimiser l'overtime maximal, sans dépasser la durée maximale permise. En revanche, dans le cas du MA, la distance parcourue et l'overtime maximal sont des attributs qui caractérisent tous les individus de la population. L'évaluation se fait en comparant les valeurs de ces deux attributs pour chaque paire d'individus ; si l'un des individus dépasse le temps normal du travail, le meilleur est celui qui minimise l'overtime maximal, sinon le meilleur est celui qui minimise la distance parcourue. Bien que la méthode d'évaluation adoptée pour le MA est différente, la relation d'ordre définie dans les trois approches est la même. En d'autres termes, les trois approches favorisent le premier objectif (distance) si la solution respecte le temps normal du travail. Une fois, le temps normal n'est plus respecté, la priorité de l'optimisation, passe au deuxième objectif (overtime).

En revenant vers la nature du problème et ses objectifs, on comprend pourquoi, cette relation d'ordre. D'abord, rien ne montre que les objectifs sont conflictuels, mais rien ne prouve qu'ils sont proportionnels. Pour cette raison, nous avons défini le taux de cohérence calculé à base des résultats de trois exécutions de chaque algorithme. Ce taux qualifie la proportionnalité des deux objectifs selon l'algorithme de résolution. En fait, les solutions montrent que les deux objectifs sont généralement proportionnels ou cohérents. Donc, nous pouvons dire que, d'une façon générale, si on optimise l'un des objectifs, on optimise implicitement l'autre. En outre, il y a deux situations possibles pour le MTDVRPOT ; soit que le temps normal est suffisant pour terminer tout service, soit qu'il est insuffisant. Dans le premier cas, la minimisation de l'overtime n'a aucun sens, donc la décision normale et logique est de choisir la solution qui minimise la distance. Par contre, dans le deuxième cas, nous pourrons avoir un conflit entre les objectifs si la solution qui minimise l'overtime n'est pas optimale en termes de distance. Or, la minimisation de l'overtime est prioritaire, dans ce cas, parce que le coût des employés en overtime est généralement très élevé par rapport à leur coût en temps normal. En plus, le taux de cohérence très élevé (87 % au moins) entre les objectifs du problème nous encourage à nous concentrer sur la minimisation de l'overtime puisqu'il est prioritaire dans le deuxième cas. Ainsi, la relation d'ordre qu'on a défini, est très logique d'un point de vue décisionnel.

Outre, nous avons proposé une méthode à base de l'ACSH pour le VRPMTOT statique, en utilisant une autre approche d'évaluation. Cette fois, l'algorithme optimise les deux objectifs au fur et à mesure de son exécution ; le choix de la meilleure solution de chaque itération minimise le LTR tandis que la procédure de recherche locale minimise la distance. Une autre particularité de cette démarche réside dans la nature des benchmarks de tests ainsi que l'indice LTR. En effet, les benchmarks de Taillard et al. (1996) considèrent que le temps de service est nul, en même temps notre ACSH considère que le temps et la distance sont équivalents. Ceci fait que le LTR permet d'évaluer les deux objectifs du problème. D'une part le LTR représente explicitement l'overtime maximal, parce que, plus le LTR

est petit plus le dernier véhicule revient tôt au dépôt. D'autre part, si nous minimisons le LTR, nous minimisons implicitement la distance parce qu'on minimise la distance du camion qui consomme la plus grande distance puisque, dans le cas statique, tous les camions ont le même temps de départ. Cependant dans le cas dynamique, cette déduction n'est pas correcte, parce que les véhicules ne commencent pas tous leurs voyages au début de la journée. En fait, au début de chaque période le nouveau plan de routage décide si on a besoin d'utiliser un véhicule additionnel ou non. Ainsi, dans le cas dynamique le LTR n'est pas significatif pour la distance comme dans le cas statique. C'est la raison pour laquelle nous ne l'avons pas adopté comme critère d'évaluation pour le MTDVRPOT.

5.5 Conclusion

Nous avons proposé différentes démarches d'évaluation bi-objective avec des jeux de tests de tailles variées. Cette diversité nous a permis d'élaborer une comparaison assez englobante, même si, nous sommes les seuls (jusqu'à présent), à traiter ce problème.

En conclusion, le MA est l'algorithme le plus recommandé pour résoudre le MTDVRPOT que ça soit pour les petites ou les grandes instances.

Conclusion générale

L'objectif de cette thèse était de résoudre le problème de tournées de véhicules dynamique multi-tour avec overtime MTDVRPOT. À notre connaissance, nous sommes les premiers à traiter ce problème, tel qu'il est défini dans cette thèse. La complexité du MTDVRPOT vient, non seulement, de son appartenance à la classe VRP, mais aussi de son caractère dynamique et bi-objectifs.

D'abord, nous avons présenté une modélisation mathématique de ce problème. Sur la base de cette modélisation, nous avons développé une approche de résolution qui exécute une méthode exacte de façon itérative (HMEI). Ensuite, nous avons adopté deux métaheuristiques pour implémenter des algorithmes de résolution pour le MTDVRPOT. Le premier est un système de colonies de fourmis hybride (ACSH) et le deuxième est un algorithme mémétique (MA). Or, la résolution du problème dynamique nécessite d'abord la résolution du problème statique. Ce dernier constitue, de sa part, un champ de recherche très vaste. Dans ce travail, nous avons abordé la version dynamique et statique du VRPMTOT en précisant les frontières de chaque problème. Outre, nous avons appliqué nos algorithmes métaheuristiques sur une version classique du CDVRP pour pouvoir comparer leurs performances avec d'autres travaux de la littérature. Les résultats de l'expérimentation sur le CDVRP classique ont prouvé l'efficacité de nos démarches. En revanche, la comparaison des résultats des trois approches sur le MTDVRPOT montre la supériorité de l'algorithme mémétique. Nous pouvons résumer les contributions de cette thèse dans les points suivants :

- Une description détaillée du DVRP et des points qui le distinguent du VRP statique et du VRP stochastique. Nous avons dessiné les périmètres de chaque problème en précisant les chevauchements possibles entre leurs variantes.

- Une revue de littérature qui couvre le DVRP ainsi que les variantes du VRP statique qui ont des liaisons avec le MTDVRPOT.

- Une description du MTDVROT avec modélisation mathématique du sous-problème lié à une période de temps.

- Le développement de la méthode de résolution HMEI. Nous avons, également, proposé des jeux de tests adaptés à son application sur le MTDVRPOT.

- La résolution du VRPMTOT statique à l'aide de l'ACSH. Les résultats démontrent que l'ACSH est compétitif par rapport à une autre approche de la littérature. Le point fort de l'ACSH était le temps d'exécution.

- La résolution du DVRP à capacité classique à l'aide de l'ACSH pour comparer les performances de ce dernier avec une autre approche de la littérature.

- La proposition des jeux de tests pour le MTDVRPOT et leurs résolutions par l'ACSH. Les résultats montrent que les deux objectifs du problème sont cohérents à 87 %.

- La résolution du CDVRP classique à l'aide du MA pour comparer les performances de ce dernier avec des approches de la littérature. Le MA a montré sa supériorité par rapport aux autres algorithmes.

- La résolution du MTDVRPOT avec le MA : Les résultats montrent que les deux objectifs du problème sont cohérents à 92 %.

- Finalement, nous avons comparé les résultats des trois approches sur les instances proposées pour le MTDVRPOT. Cette comparaison montre la surperformance du MA et de l'HMEI en termes de fonction objectif. En revanche, l'ACSH est le meilleur en termes de temps d'exécution.

Ce travail peut être la base de plusieurs futurs travaux que nous proposons comme perspectives :

- Appliquer d'autres métaheuristiques sur le MTDVRPOT et voir laquelle est la plus appropriée.

- Le MA a démontré sa performance dans la résolution du DVRP et du MTDVRPOT. Il sera intéressant de l'appliquer sur d'autres variantes du DVRP, telles que le DVRP avec temps de voyage dépendant et le DVRP avec collecte et livraison.

- Il sera intéressant, aussi, d'ajouter d'autres contraintes et objectifs au MTDVRPOT. Puisqu'on tolère l'utilisation de l'overtime, on peut avoir des clients qui exigent la livraison pendant le temps normal de travail et d'autres qu'on peut retarder. Il sera pertinent de donner un indice de priorité à chaque client. Outre, on peut ajouter comme objectif ; maximiser cet indice dans les premiers clients de chaque tournée, et comme contrainte ; les clients qui ont un indice de priorité supérieur à un seuil prédéfini doivent être servis pendant le temps normal.

- L’ACSH est un algorithme qui a prouvé son importance surtout en termes de temps d’exécution. Toutefois, il a besoin d’amélioration au niveau des performances de la fonction objectif. Peut-être sa combinaison avec d’autres heuristiques donnera de meilleurs résultats.

- L’exploitation des données stochastiques pourrait construire un autre axe de recherche surtout si on pouvait effectuer des simulations de l’arrivée des nouvelles requêtes. A base des résultats de la simulation, on construit des plans de routage préalables. Puisque le problème est dynamique, on peut changer ces plans, au fur et à mesure qu’on reçoit les requêtes réelles.

En guise de conclusion, l’optimisation des problèmes dynamiques est un champ très vaste et plus en plus intéressant. Dans cette thèse, nous avons pu fournir une contribution dans l’optimisation du DVRP qui nous a permis de confronter la complexité de ce genre de problème et de voir son importance théoriquement. Il sera, alors très intéressant d’implémenter ces démarches dans des logiciels informatiques en vue de les mettre en application sur le champ industriel.

Références

Albareda-Sambola, M., Fernández, E., & Laporte, G. (2014). The dynamic multiperiod vehicle routing problem with probabilistic information. *Computers and Operations Research*, *48*, 31–39.

Attanasio, A., Cordeau, J. F., Ghiani, G., & Laporte, G. (2004). Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. *Parallel Computing*, *30*(3), 377–387.

Ayadi, R., & Benadada, Y. (2013). Memetic algorithm for a multi-objective vehicle routing problem with multiple trips. *International Journal of Computer Science and Applications*, *10*(2), 72–91.

Azi, N., Gendreau, M., & Potvin, J. Y. (2012). A dynamic vehicle routing problem with multiple delivery routes. *Annals of Operations Research*, *199*(1), 103–112.

Barkaoui, M., Berger, J., & Boukhtouta, A. (2015). Customer satisfaction in dynamic vehicle routing problem with time windows. *Applied Soft Computing Journal*, *35*, 423–432.

Bartlett, K., Lee, J., Ahmed, S., Nemhauser, G., Sokol, J., & Na, B. (2014). Congestion-aware dynamic routing in automated material handling systems. *Computers and Industrial Engineering*, *70*(1), 176–182.

Beaudry, A., Laporte, G., Melo, T., & Nickel, S. (2010). Dynamic transportation of patients in hospitals. *OR Spectrum*, *32*(1), 77–107.

Bellman, R. (1957). *Dynamic programming*. Press, Princeton University.

Bent, R. W., & Hentenryck, P. Van. (2004). Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, *52*(6), 977–987.

Benyahia, I., & Potvin, J. Y. (1998). Decision support for vehicle dispatching using genetic programming. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.*, *28*(3), 306–314.

Berger, J., Barkaoui, M., & Boukhtouta, A. (2007). A hybrid genetic approach for airborne sensor vehicle routing in real-time reconnaissance missions. *Aerospace Science and Technology*, *11*(4), 317–326.

Bianchi, L. (2000). Notes on dynamic vehicle routing - tha state of the art-. In *Technical Report IDSIA-05-01* (Vol. 66).

Bianchi, L., Dorigo, M., Maria, L., & Gutjahr, W. (2008). A survey on metaheuristics for stochastic combinatorial optimization. *Nat Comput*, *8*, 239–287.

Birge, J. R., & Louveaux, F. (1997). *Introduction to stochastic programming* (Springer (ed.)).

Bolaños, R. I., Escobar, J. W., & Echeverri, M. G. (2018). A metaheuristic algorithm for the multi-depot vehicle routing problem with heterogeneous fleet. *International Journal of Industrial Engineering Computations*, *9*(4), 461–478.

Brandão, J., & Mercer, A. (1997). A tabu search algorithm for the multi-trip vehicle routing and and scheduling problem. *European Journal of Operational Research*, *100*(97), 180–191.

Chang, M. S., Chen, S., & Hsueh, C. F. (2003). Real-time vehicle routing problem with time windows and simultaneous delivery/pickup demands. *Journal of the Eastern Asia Society for Transportation Studies*, *5*(30), 2273–2286.

Chávez, J. J. S., Escobar, J. W., & Echeverri, M. G. (2016). A multi-objective pareto ant colony algorithm for the multi-depot vehicle routing problem with backhauls. *International Journal of Industrial Engineering Computations*, *7*(1), 35–48.

Cheikh, M., Ratli, M., Mkaouar, O., & Jarboui, B. (2015). A variable neighborhood search algorithm for the vehicle routing problem with multiple trips. *Electronic Notes in Discrete Mathematics*, *47*, 277–284.

Chen, H., Hsueh, C.-F., & Chang, M.-S. (2006). The real-time time-dependent vehicle routing problem. *Transportation Research Part E*, *42*, 383–408.

Chen, Z. L., & Xu, H. (2006). Dynamic column generation for dynamic vehicle routing with time windows. *Transportation Science*, *40*(1), 74–88.

Cheung, B. K. S., Choy, K. L., Li, C. L., Shi, W., & Tang, J. (2008). Dynamic routing model and solution methods for fleet management with mobile technologies. *International Journal of Production Economics*, *113*(2), 694–705.

Christiansen, C. H., & Lysgaard, J. (2007). A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research Letters*, *35*(6), 773–781.

Christofides, N., & Beasley, J. E. (1984). The period routing problem. *Networks*, *14*(2), 237–256.

Christofides, N., Mingozzi, A., & Toth, P. (1979). The vehicle routing problem. *Combinatorial Optimization. Wiley, Chichester*, 315–338.

Cordeau, J. F., Laporte, G., Savelsbergh, M. W. P., & Vigo, D. (2007). Chapter 6 Vehicle Routing. *Handbooks in Operations Research and Management Science*, *14*(C), 367–428.

Dantzig, G. B., Ordern, A., & Wolfe, P. (1955). The Generalized Simplex Method. *Pacific Journal of Mathematics*, *5*(2), 183–195.

Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science*, *6*(1), 80–91.

Dantzig, G., Fulkerson, R., & Johnson, S. (1954). Solution of a Large-Scale Traveling-Salesman Problem. *Journal of the Operational Research Society of America*, *2*(4), 393–410.

Dasgupta, D., & Michalewics, Z. (1997). *Evolutionary Algorithms in Engineering Applications* (Springer (ed.)).

Doerner, K. (2004). Pareto Ant Colony Optimization: A Metaheuristic Approach to Multiobjective Portfolio Selection. *Annals of Operations Research*, *131*, 79–99.

Dondo, R., & Cerdá, J. (2007). A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows. *European Journal of Operational Research*, *176*(3), 1478–1507.

Dorigo, M., Colorni, A., & Maniezzo, V. (1991). Distributed Optimization by ant colonies. *Proceedings of the First European Conference on Artificial Life, or D*, 134–142.

Eksioglu, B., Vural, A. V., & Reisman, A. (2009). The vehicle routing problem: A taxonomic review. *Computers and Industrial Engineering*, *57*(4), 1472–1483.

Eydi, A., & Javazi, L. (2012). A novel heuristic method to solve the capacitated arc routing problem. *International Journal of Industrial Engineering Computations*, *3*(5), 767–776.

Ferland, J. A., & Michelon, P. (1988). The vehicle scheduling problem with Multiple Vehicle Types. *Journal of Operational Research Society*, *39*(6), 577–583.

Fisher, M. (1994). Optimal solution for vehicle routing problems K-trees. *Operations Research Society of America*, *42*(4), 626–642.

Fleischmann, B. (1990). *The vehicle routing problem with multiple use of the vehicles*.

Fonseca, C. M., & Fleming, P. J. (1993). Genetic Algorithms for Multiobjective Optimization :Formulation Discussion and Generalization. *5th International Conference on Genetic Algorithms, July*.

Gambardella, L. M., Taillard, É., & Agazzi, G. (1999). MACS-VRPTW : A multiple ant colony system for vehicle routing problems with time windows. *Technical Report IDSIA*, 1–17.

Gambardella, L., Rizzoli, A. E., Oliverio, F., Casagrande, N., Donati, A., Montemanni, R., & Lucibello, E. (2000). Ant Colony Optimization for vehicle routing in advanced. *Search, January*.

Gendreau, M., Guertin, F., Potvin, J.-Y., & Taillard, É. (1999). Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching. *Transportation Science*, *33*(4), 381–390.

Gendreau, M., Guertin, F., Potvin, J. Y., & Séguin, R. (2006). Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries. *Transportation Research Part C: Emerging Technologies*, *14*(3), 157–174.

Gendreau, M., Iori, M., Laporte, G., & Martello, S. (2008). A Tabu Search heuristic for the vehicle routing problem with two-dimensional loading constraints. *Networks*, *51*(1), 4–18.

Gendreau, M., Laporte, G., & Séguin, R. (1996). Stochastic vehicle routing. *European Journal of Operational Research*, *88*(1), 3–12.

Ghannadpour, S. F., Noori, S., Tavakkoli-Moghaddam, R., & Ghoseiri, K. (2014). A multi-objective dynamic vehicle routing problem with fuzzy time windows: Model, solution and application. *Applied Soft Computing Journal*, *14*(PART C), 504–527.

Ghoseiri, K., & Ghannadpour, S. F. (2010). Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Applied Soft Computing Journal*, *10*(4), 1096–1107.

Gillett, B. E., & Miller, L. R. (1974). A Heuristic Algorithm for the Vehicle-Dispatch Problem. *Operations Research*, *22*(2), 340–349.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, *13*(5), 533–549.

Glover, F., Laguna, M., & Martí, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics*, *29*(3), 652–684.

Golden, B., Assad, A., Levy, L., & Gheysens, F. (1984). The fleet size and mix vehicle routing problem. *Computers and Operations Research*, *11*(1), 49–66.

Golden, B. L., Magnanti, T. L., & Nguyen, H. Q. (1977). Implementing vehicle routing algorithms. *Networks*, *7*(2), 113–148.

Guntsch, M., & Middendorf, M. (2001). Pheromone modification strategies for ant algorithms applied to dynamic tsp. *LNCS*, *2037*, 213–222.

Gupta, R., Singh, B., & Pandey, D. (2010). Multi-Objective Fuzzy Vehicle Routing Problem: A Case Study. *Int. J. Contemp. Math. Sciences*, *5*(29), 1439–1454.

Haghani, A., & Jung, S. (2005). A dynamic vehicle routing problem with time-dependent travel times. *Computers and Operations Research*, *32*(11), 2959–2986.

Hanshar, F. T., & Ombuki-Berman, B. M. (2007). Dynamic vehicle routing using genetic algorithms. *Applied Intelligence*, *27*(1), 89–99.

Hart, W. E., Krasnogor, N., & Smith, J. E. (2005). Rcent Advances in Memetic Algorithms. In *Studies in Fuzziness and Soft Computing* (Vol. 166, Issue 1).

Hemert, J. I. va., & Poutré, J. A. La. (2010). *Dynamic Routing Problems with Fruitful Regions: Models and Evolutionary Computation*. 692–701.

Hong, L. (2012). An improved LNS algorithm for real-time vehicle routing problem with time windows. *Computers and Operations Research*, *39*(2), 151–163.

Hu, T. Y. (2001). Evaluation framework for dynamic vehicle routing strategies under real-time information. *Transportation Research Record*, *1774*, 115–122.

Hvattum, L. M., Løkketangen, A., & Laporte, G. (2006). Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic. *Transportation Science*, *40*(4), 421–438.

Hwang, R. Z., Chang, R. C., & Lee, R. C. T. (1993). The Searching over Separators Strategy To Solve Some NP-Hard Problems in Subexponentiai Time. *Algorithmica*, *9*, 398–423.

Ichoua, S., Gendreau, M., & Potvin, J.-Y. (2000). Diversion Issues in Real-Time Vehicle Dispatching. *Transportation Science*, *34*(4), 426–438.

Ichoua, S., Gendreau, M., & Potvin, J. Y. (2003). Vehicle dispatching with time-dependent travel times. *European Journal of Operational Research*, *144*(2), 379–396.

Irnich, S. (2000). Multi-depot pickup and delivery problem with a single hub and heterogeneous vehicles. *European Journal of Operational Research*, *122*(2), 310–328.

Ivković, Z., & Lloyd, E. L. (1993). Fully dynamic algorithms for bin packing: Being (mostly) myopic helps. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *726 LNCS*(2), 224–235. https://doi.org/10.1007/3-540-57273-2_58

Ivkovic, Z., & Lyyold, E. (1994). Fully dynamic maintenance of vertex cover. *Graph-Theoretic Concepts in Computer Science*, 99–111.

Jaillet, P. (2008). A Priori Solution of a Traveling Salesman Problem in Which a Random Subset of the Customers Are Visited. *Operations Research*, *36*(6), 929–936.

Kaiwartya, O., Kumar, S., Lobiyal, D. K., Tiwari, P. K., Abdullah, A. H., & Hassan, A. N. (2015). Multiobjective dynamic vehicle routing problem and time seed based solution using particle swarm optimization. *Journal of Sensors*, *2015*.

Kennedy, J., & Elberhart, R. (1995). Particle swarm optimization. *IEEE International Conf. on Neural Networks*, 6556–6559.

Khouadjia, Mostepha R., Sarasola, B., Alba, E., Jourdan, L., & Talbi, E. G. (2012). A comparative study between dynamic adapted PSO and VNS for the vehicle routing problem with dynamic requests. *Applied Soft Computing Journal*, *12*(4), 1426–1439.

Khouadjia, Mostepha Redouane. (2011). *Solving Dynamic Vehicle Routing Problems: From Single-solution Based Metaheuristics to Parallel Population Based Metaheuristics*.

Kilby, P., Prosser, P., & Shaw, P. (1998). Dynamic VRPs: A Study of Scenarios. *Report APES-06-1998*, *April*, 1–11.

Klein, P. N., & Subramanian, S. (1998). A Fully Dynamic Approximation Scheme for Shortest Paths in Planar Graphs. *Algorithmica*, *22*, 235–249.

Koç, Ç., Bektaş, T., Jabali, O., & Laporte, G. (2016). Thirty years of heterogeneous vehicle routing. *European Journal of Operational Research*, *249*(33), 1–14.

Krasnogor, N., & Smith, J. (2005). A tutorial for competent memetic algorithms: Model, taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation*, *9*(5), 474–488.

Labadie, N., Prodhon, C., & Prins, C. (2016). Metaheuristics for vehicle routing problems. In *Metaheuristics*.

Land, A., & Doig, A. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, *28*(3), 427–520.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, *43*(4), 408–416.

Larsen, A. (2000). The Dynamic Vehicle Routing Problem. In *Kgs. Lyngby, Denmark: Technical University of Denmark (DTU)* (Issues IMM-PHD; No. 2000-73).

Larsen, A., Madsen, O. G., & Solomon, M. M. (2008). Recent Developments in Dynamic Vehicle Routing Systems. In *The Vehicle Routing Problem: Latest Advances and New Challenges.* (pp. 199–218).

Lenstra, J. K., & Kan, A. H. G. (1981). Complexity of Vehicle Routing and Scheduling Problems. *Networks*, *11*, 221–227.

Letchford, A. N., & Eglese, R. W. (1998). The rural postman problem with deadline classes. *European Journal of Operational Research*, *105*(3), 390–400.

Levy, D., Sundar, K., & Rathinam, S. (2014). Heuristics for routing heterogeneous unmanned vehicles with fuel constraints. *Mathematical Problems in Engineering*.

Li, C., Yang, M., & Kang, L. (2006). A New Approach to Solving Dynamic Traveling Salesman Problems. *Simulated Evolution and Learning*, 236–243.

Li, F., Golden, B., & Wasil, E. (2007). A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. *Computers and Operations Research*, *34*(9), 2734–2742.

Liao, T. Y., & Hu, T. Y. (2011). An object-oriented evaluation framework for dynamic vehicle routing problems under real-time information. *Expert Systems with Applications*, *38*(10), 12548–12558.

Lin, C., Choy, K. L., Ho, G. T. S., Lam, H. Y., Pang, G. K. H., & Chin, K. S. (2014). A decision support system for optimizing dynamic courier routing operations. *Expert Systems with Applications*, *41*(15), 6917–6933.

Loannou, G., Kritikos, M., & Prastacos, G. (2001). A greedy look-ahead heuristic for the vehicle routing problem with time windows. *Journal of the Operational Research Society*, *52*(5), 523–537.

Lund;, K., Madsen;, O. B. G., & Rygaard, J. M. (1996). Vehicle routing problems with varying degrees od dynamism. *Rapport Technique, IMM, the Departement of Methematical Modeling, Technical University of Denmark*.

Mańdziuk, J., & Żychowski, A. (2016). A memetic approach to vehicle routing problem with dynamic requests. *Applied Soft Computing Journal*, *48*, 522–534.

Mavrovouniotis, M., & Yang, S. (2015). Ant algorithms with immigrants schemes for the dynamic vehicle routing problem. *Information Sciences*, *294*, 456–477.

Melián-Batista, B., De Santiago, A., Angelbello, F., & Alvarez, A. (2014). A bi-objective vehicle routing problem with time windows: A real case in Tenerife. *Applied Soft Computing Journal*, *17*, 140–152.

Merz, P., & Freisleben, B. (1999). A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem. *Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999*, *3*, 2063–2070.

Messaoud, E., Alaoui, A. E. H., & Boukachour, J. (2013). Hybridation de l'algorithme de colonie de Fourmis avec l'algorithme de recherche à grand Voisinage pour la résolution du VRPTW statique et dynamique. *INFOR*, *51*(1), 41–51.

Mingozzi, A., Roberti, R., & Toth, P. (2013). An exact algorithm for the multitrip vehicle routing problem. *INFORMS Journal on Computing*, *25*(2), 193–207.

Montemanni, R., Gambardella, L. M., Rizzoli, A. E., & Donati, A. V. (2005). Ant Colony System for a Dynamic Vehicle. *Journal of Combinatorial Optimization*, 327–343.

Moscato, P. (1989). On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. *Caltech Concurrent Computation Program, C3P 826*.

Musolino, G., Polimeni, A., Rindone, C., & Vitetta, A. (2013). Travel Time Forecasting and Dynamic Routes Design for Emergency Vehicles. *Procedia - Social and Behavioral Sciences*, *87*, 193–202.

Olivera, A., & Viera, O. (2007). Adaptive memory programming for the vehicle routing problem with multiple trips. *Computers and Operations Research*, *34*(1), 28–47.

Ombuki, B. M., Nakamura, M., & Osamu, M. (2002). A hybrid search based on genetic algorithms and tabu search for vehicle routing. *6th IASTED Intl. Conf. On Artificial Intelligence and Soft Computing (ASC 2002)*, *May*, 176–181.

Ombuki, B., Ross, B. J., & Hanshar, F. (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, *24*(1), 17–30.

Orozco, J.-A., & Barceló, J. (2012). Reactive and Proactive Routing Strategies with Real-Time Traffic Information. *Procedia - Social and Behavioral Sciences*, *39*, 633–648.

Ouaddi, K., Benadada, Y., & Mhada, F.-Z. (2018). Ant Colony System for Dynamic Vehicle Routing Problem with Overtime. *International Journal of Advanced Computer Science and Applications*, *9*(6), 306–315.

Ouaddi, K., Mhada, F.-Z., & Benadada, Y. (2020). Memetic algorithm for multi-tours dynamic vehicle routing problem with overtime (MDVRPOT). *International Journal of Industrial Engineering Computations*, *March*.

Ozbaygin, G., & Savelsbergh, M. (2019). An iterative re-optimization framework for the dynamic vehicle routing problem with roaming delivery locations. *Transportation Research Part B: Methodological*, *128*, 207–235.

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

Pellegrini, P., Favaretto, D., & Moretti, E. (2007). Multiple ant colony optimization for a rich vehicle routing problem: A case study. *Knowledge-Based Intelligent Information and Engineering Systems*, 627–634.

Petch, R. J., & Salhi, S. (2003). A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. *Discrete Applied Mathematics*, *133*(1–3), 69–92.

Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, *225*(1), 1–11.

Pillac, V., Guéret, C., & Medaglia, A. L. (2012). An event-driven optimization framework for dynamic vehicle routing. *Decision Support Systems*, *54*(1), 414–423.

Potra, F. A., & Wright, S. J. (2000). Interior-point methods. *Journal of Computational and Applied Mathematics*, *124*(1–2), 281–302.

Prins, C. (2002). Efficient Heuristics for the Heterogeneous Fleet Multitrip VRP with Application to a Large-Scale Real Case. *Journal of Mathematical Modelling and Algorithms*, *1*(2), 135–150.

Psaraftis, H. N. (1980). A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem. *Transportation Science*, *14*(2), 130–154.

Psaraftis, H. N., Wen, M., & Kontovas, C. A. (2016). Dynamic vehicle routing problems: Three decades and counting. *Networks*, *67*(1), 3–31.

Ritzinger, U., Puchinger, J., & Hartl, R. F. (2016). A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research*, *54*(1), 215–231.

Rochat, Y., & Taillard, É. D. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, *1*(1), 147–167.

Rouky, N., Boukachour, J., Boudebous, D., & Alaoui, A. E. H. (2018). A Robust Metaheuristic for the Rail Shuttle Routing Problem with Uncertainty: A Real Case Study in the Le Havre Port. *Asian Journal of Shipping and Logistics*, *34*(2), 171–187.

Sabar, N. R., Bhaskar, A., Chung, E., Turky, A., & Song, A. (2019). A self-adaptive evolutionary algorithm for dynamic vehicle routing problems with traffic congestion. *Swarm and Evolutionary Computation*, 1018–1027.

Salhi, S., & Sari, M. (1997). A multi-level composite heuristic for the multi-depot vehicle fleet mix problem. *European Journal of Operational Research*, *103*(1), 95–112.

Salhi S, & Petch, R. J. (2007). A GA based heuristic for the vehicle routing problem with multiple trips. *Journal of Mathematical Modelling and Algorithms*, *6*(4), 591–613.

Salhi, Said, Imran, A., & Wassan, N. A. (2014). The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation. *Computers and Operations Research*, *52*, 315–325.

Schyns, M. (2015). An ant colony system for responsive dynamic vehicle routing. *European Journal of Operational Research*, *245*(3), 704–718.

Seixas, M. P., & Mendes, A. B. (2013). Column generation for a multitrip vehicle routing problem with time windows, driver work hours, and heterogeneous fleet. *Mathematical Problems in Engineering*, *2013*.

Solomon. (1983). Vehicle Routing and Scheduling with Time Window Constraints: Models and Algorithms. *Ph.D. Dissertation, Dept. of Decision Sciences, University of Pennsylvania.*, *130*(2), 556.

Stewart, W. R., & Golden, B. L. (1984). A Lagrangean relaxation heuristic for vehicle routing. *European Journal of Operational Research*, *15*(1), 84–88.

Taillard, É. (1993). Parallel iterative search methods for vehicle routing problems. *Networks*, *23*(8), 661–673.

Taillard, É. (1999). A heuristic column generation method for the heterogeneous fleet VRP. *RAIRO (Recherche Opérationnelle/ Operations Research)*, *33*, 1–14.

Taillard, É., Badeau, P., Gendreau, M., Guertin, F., & Potvin, J.-Y. (1997). A tabu search heuristic for vehicle routing problem wth soft time windows. *Transportation Science*, *31 (2)*, 170–186.

Taillard, É., Gambardella, L. M., & Gendreau, M. (2001). *Adaptive memory programming : A unified view of metaheuristics*. *135*.

Taillard, É., Laporte, G., & Gendreau, M. (1996). Vehicle routeing with multiple use of vehicles. *Journal of the Operational Research Society*, *47*(8), 1065–1070.

Talbi, E.-G. (2009). *Metaheuristics : from design to implementation*. John Wiley & Sons.

Tirado, G., Hvattum, L. M., Fagerholt, K., & Cordeau, J. F. (2013). Heuristics for dynamic and stochastic routing in industrial shipping. *Computers and Operations Research*, *40*(1), 253–263.

Van Woensel, T., Kerbache, L., Peremans, H., & Vandaele, N. (2008). Vehicle routing with dynamic travel times: A queueing approach. *European Journal of Operational Research*, *186*(3), 990–1007.

Xu, Y., Wang, L., & Yang, Y. (2012). A New Variable Neighborhood Search Algorithm for the Multi Depot Heterogeneous Vehicle Routing Problem with Time Windows. *Electronic Notes in Discrete Mathematics*, *39*, 289–296.

Yang, J., Jaillet, P., & Mahmassani, H. (2004). Real-time multivehicle truckload pickup and delivery problems. *Transportation Science*, *38*(2), 135–148.

Yang, S., Hamedi, M., & Haghani, A. (2005). Online dispatching and routing model for emergency vehicles with area coverage constraints. *Transportation Research Record*, *1923*, 1–8.

Yellow, P. . (1970). A Computational modification to the savings method of vehicle scheduling. *Operational Research Quarterly*, *12*, 281–283.

Yousefi, S., Abbasi, T., & Anvari, Z. (2014). Transportation routing in urban environments using updated traffic information provided through vehicular communications. *Journal of Transportation Systems Engineering and Information Technology*, *14*(5), 23–36.

Zhou, W., Song, T., He, F., & Liu, X. (2013). Multiobjective vehicle routing problem with route balance based on genetic algorithm. *Discrete Dynamics in Nature and Society*, *2013*.

Annexe A

Tableau 18: État de l'art du DVRP

					Problème	**Fonction objectif**			**Résolution**
	D	**S**	**P**	**C**			**R**	**T**	
(Albareda-Sambola et al., 2014)		*	*		CP-TW	Min (Coût de transport)	*		Variable NS
(Attanasio et al., 2004)	*		*		DaR-CP-TW	Max (Nombre de requêtes servies)	*		TS Parallèle
(Azi et al., 2012)		*	*		TW-MT	Min (Distance totale parcourue) & Max (Nombre de clients servis)	*		Large NS
(Barkaoui et al., 2015)		*		*	CP-TW	Min (Distance totale - Overtime) & Max (Nombre de clients satisfaits)	*		GA-RC
(Bartlett et al., 2014)	*		*		RPIGE	Min (Congestion)		*	AD
(Beaudry et al., 2010)	*		*		DaR-CP-TW	Min (Temps de voyage total - Retard total- Précocité totale)	*		IS-TS
(Bent & Hentenryck, 2004)		*	*		TW-CP	Max (Nombre de clients servis)	*		ASM
(Benyahia & Potvin, 1998)	*		*		PD	Min (Coût)& Max (Clients servis – Satisfaction des clients et des conducteurs)	*		GP
(Berger et al., 2007)	*			*	CP-TW	Min (Les requêtes rejetées - Le retard chez clients - Le temps total)	*		GA-RC
(Chang et al., 2003)	*			*	CP-TW-PD	Min (Temps de voyage-Temps d'attente avant le service)	*		TS
(Z. L. Chen & Xu, 2006)	*		*		CP-TW-FI	Min (Distance totale)	*		GC-CPLEX
(Cheung et al., 2008)	*		*		TW-PD	Min (Temps)	*	*	GA-IS-RL
(Christiansen & Lysgaard, 2007)		*	*		CP	Min (Distance totale)	*		GC-PD-BP
(Gendreau et al., 1999)	*			*	TW	Min (Distance totale-Le retard du service chez les clients)	*		TS
(Gendreau et al., 2006)	*			*	PD-TW-OT	Min (Temps Total-Overtime totale-Retard total chez les clients)	*		TS
(Haghani & Jung, 2005)	*		*		PD-TW	Min (Coût d'usage des véhicules - Cout de routage- Dissatisfaction des clients)	*	*	GA
(Hanshar & Ombuki-Berman, 2007)	*		*		CP	Min (Distance totale)	*		GA
(Hemert & Poutré, 2010)		*		*	CO	Max (Nombre de charges transportées)	*		EA
(Ichoua et al., 2000)	*			*	TW	Min (Distance-Le retard du service chez les clients)	*		TS
(Mostepha R. Khouadjia et al.,	*		*		CP	Min (Distance totale)	*		PSO - variable NS
(Mańdziuk & Żychowski, 2016)	*		*		CP	Min (Distance totale)	*		MA
(Montemanni et al., 2005b)	*		*		CP	Min la distance totale	*		ACS
(Musolino et al., 2013)		*		*	CP	Min (Temps total)	*	*	RA-DA-TTP-OM
(Psaraftis, 1980)	*		*		DaR-CP	Min (Temps - Dissatisfaction des clients)	*		PD
(Schyns, 2015)		*	*		CP-TW-H	Min (Distance totale) & Max (réactivité)			ACS
(Yang et al., 2004)		*	*		PD	Min (Distances à vide -Services retardés-Requêtes rejetées)	*		IS-RL-CPLEX
(Yousefi et al., 2014)	*		*		TVD	Min (temps de voyage)		*	AD

D : Déterministe
S : Stochastique
P : Périodique
C : Continue
R : Requête
T : Temps de voyage

Tableau 19 : État de l'art du VRP statique

	Type du problème	**Fonction objectif**	**Résolution**
(Ayadi & Benadada, 2013)	CP-MT-OT	Min (Temps de voyage-Overtime maximal)	MA
(Bolaños et al., 2018)	CP-MD-H	Min (Cout d'utilisation des véhicules-Distance totale)	GA
(Brandão & Mercer, 1997)	CP-MT-TW-H-OT	Min (Temps de voyage-Overtime total -Temps d'attente-Nombre de véhicules utilisés)	TS
(Chávez et al., 2016)	CP-MD-PD	Min (Distance totale -Temps total - Energie consommée)	ACS Pareto
(Cheikh et al., 2015)	CP-MT-OT	Min (Temps de voyage -Overtime total- Pénalité de surcharge)	NS
(Fleischmann, 1990)	CP-MT-H-TW-OT	Min (Distance)	CW
(Gambardella et al., Agazzi 1999)	CP-TW	Min (Nombre de véhicules utilisés - Distance totale)	ACS
(Levy et al., 2014)	CP-MT-H-MD	Min (Carburant total consommé par tous les véhicules)	VND-Variable NS
(Mingozzi et al., 2013)	CP-MT-OT	Min (Temps de voyage)	Relaxation LP
(Olivera & Viera, 2007)	CP-MT-OT	Min (Temps de voyage -Overtime total- Pénalité de surcharge)	TS-Mémoire adaptative
(Ombuki et al., 2002)	CP-TW	Min (Nombre de véhicules utilisés - Distance totale)	GA-TS
(Ombuki et al., 2006)	CP-TW	Min (Nombre de véhicules utilisés - Distance totale)	GA multi-objectif
(Petch & Salhi, 2003)	CP-MT-OT	Min (Overtime maximal)	CW-RL
(Prins, 2002)	CP-MT-H-OT	Min (Temps de voyage-Nombre de camions utilisés)	Heuristique dédiée
(Said et al., 2014)	CP-MD-H	Min (Cout d'utilisation des véhicules-Distance totale)	Variable NS-CPLEX
(Salhi & Sari, 1997)	CP-MD-H	Min (Cout d'utilisation des véhicules-Distance totale)	Heuristique dédiée
(Seixas & Mendes, 2013)	CP-MT-H-TW-OT	Min (Distance totale -Nombre de véhicules utilisés)	GC-TS
(Taillard et al., 1996)	CP-MT-OT	Min (Distance)	TS-HBP
(Xu et al., 2012)	CP-MD-H-TW	Min (Distance total -Temps d'attente -Pénalité de violation des fenêtres de temps)	Variable NS

Annexe B

Tableau 20: Résultats numériques du ACSH2016 et du MA2013

		m	MA 2013		ACS 2016	
			Coût	LTR	Coût	LTR
C1 z*= 524,6	T1	1	524,92	0,953	514.981	0.936
		2	536,56	0,998	523.768	0.991
		3	561,01	1,023	550.538	1.032
		4	547,1	1,027	536.329	1.050
	T2	1	524,94	0,91	521.038	0.903
		2	530,79	1	512.242	0.941
		3	556,61	0,993	535.160	0.992
		4	546,43	0,985	538.937	1.003
C2 z*= 835,2	T1	1	835,77	0,953	842.325	0.960
		2	839,58	0,956	846.619	0.969
		3	836,18	0,964	872.351	1.043
		4	835,77	0,974	886.157	1.038
		5	839,71	0,996	845.569	1
		6	861,88	0,999	857.787	1.065
		7	885,57	1,034	869.800	1.119
	T2	1	835,26	0,909	832.521	0.905
		2	835,26	0,912	856.500	0.934
		3	835,26	0,916	852.197	1.006
		4	835,77	0,959	875.851	0.963
		5	835,77	0,971	867,698	0,974
		6	842,28	0,997	870,636	1,003

		7	870,19	0,998	909,785	1,069
C3 z*= 826,14	T1	1	830	0,957	876,870	1,011
		2	828,74	0,97	865,569	1,005
		3	829,91	0,97	906,932	1,087
		4	828,74	0,982	870,609	1,027
		5	833,98	0,996	911,458	1,121
		6	867,72	1,008	986,338	1,198
	T2	1	828,26	0,911	896,513	0,986
		2	828,26	0,913	891,445	0,991
		3	829,51	0,931	895,420	0,995
		4	829,54	0,969	871,22	0,999
		5	834,42	0,959	924,410	1,053
		6	836,56	0,994	921,07	1,082
F134 z*= 1162,96	T1	1	1166,96	0,956	1162,91	0,952
		2	1163,53	0,954	1142,409	0,945
		3	1162,97	0,964	1117,997	0,984
	T2	1	1166,71	0,912	1137,630	0,889
		2	1163,53	0,911	1156,821	0,912
		3	1165,98	0,962	1143.229	0.950
C5 z*= 1291,44	T1	1	1313,22	0,968	1417,287	1,045
		2	1312,58	0,969	1479,293	1,092
		3	1313,21	0,969	1465,134	1,129
	T2	1	1316,48	0,926	1469,505	1,034
		2	1380,38	0,972	1493,719	1,054
		3	1378,68	0,97	1467,871	1,091

Annexe C

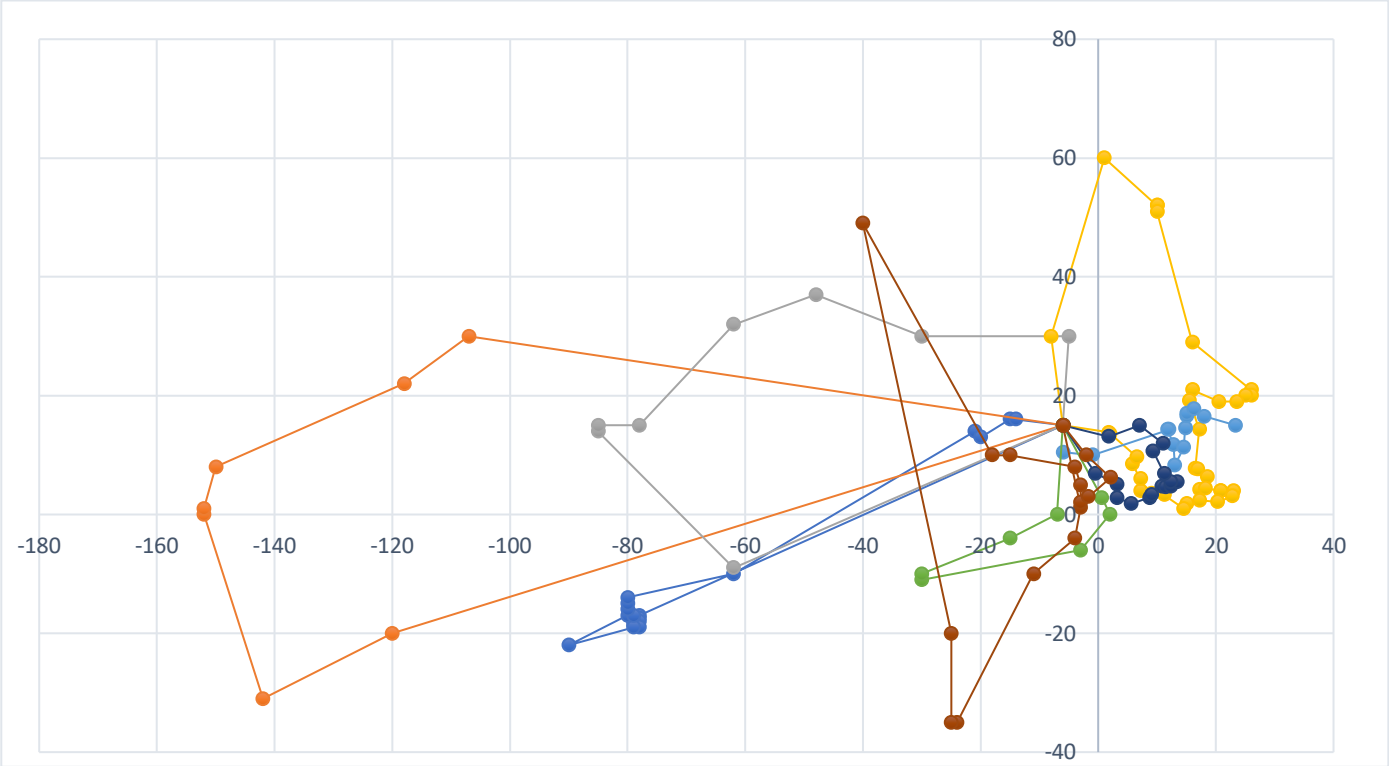

Figure 27: Représentation graphique de la solution où le MA réalise le GAP minimal par rapport à Hanshar2007

Chaque tournée est représentée par une couleur différente sachant que la figure (28) représente le plan de routage d'une journée entière.

Annexe D

Le tableau 21 fournit les résultats numériques détaillés du MA et de l'ACSH sur le MTDVRPOT. **T** est le temps de travail normal tandis que ***m*** représente le nombre de véhicules utilisés. Nous considérons trois exécutions de chaque algorithme pour chaque instance. Les résultats suivis par (*) correspondent aux instances ayant une seule solution réalisable, tandis que ceux suivis par (**) correspondent aux instances ayant deux solutions réalisables. Nous laissons les cases vides pour les instances auxquelles nous n'avons trouver aucune solution réalisable. Les cases ombrées correspondent aux instances où les deux objectifs ne sont pas cohérents. Nous utilisons le tableau 22 pour représenter la meilleure solution trouvée pour ces instances.

Tableau 21 : Résultats du MA et de l'ACSH sur le MTDVRPOT

			MA				**ACSH**			
	m	**T**	**Meilleure**		**Moyenne**		**Meilleure**		**Moyenne**	
			Dist	**Over**	**Dist**	**Over**	**Dist**	**Over**	**Dist**	**Over**
C50	1	577	565,4	0	579,3	8,8	625,3	48,3	629,9	52,9
	2	289	600	11,2	620,5	22,1	658,1	40,7	673,8	53,8
	3	192	613,1	15,2	644,2	21,9	652,5	26,3	672,2	36,2
	4	144	620,5	21,6	642,8	27,3	674*	36*	674*	36*
	5	115								
C75	1	919	911,4	0	920,3	5,4	992,8	73,8	1082,7	163,7
	2	459	931,5	7,3	962,1	22,5	1064	73,6	1079,1	81,5
	3	306	930,8	8,8	948,9	15,2	1069,1	58,8	1092,2	64,3
	4	230	952,1	10,8	983,2	18,3	1046,8	45,1	1060,7	48,8
	5	184	964,4	9,7	985,4	14,9				
C100	1	909	977,4	68,4	986,3	77,3	988,6	79,6	1015,3	106,3
	2	454	991	44	1023,2	58,4	983,5	46,1	1018,8	66,1
	3	303	1008,3	34,3	1057,8	50,5	1015,4	40,7	1064,4	59,9
	4	227	1005,3	26,5	1026,4	40,3	1021,1	39,7	1047	44,7
	5	182	1052	35,5	1067,8	41,5	1114,1*	44,7*	1114,1*	44,7*
C150	1	1131	1304,4	173,4	1313,7	182,7				
	2	565	1249,4	59	1272,1	70,7				
	3	377	1271,9	50,6	1279	51,2				
	4	282	1257,8	32,1	1290,5	41,4				

	5	226	1364,4	52,9	1377	53				
C199	1	1421	1549,3	128,3	1565	144				
	2	710	1589,8	85,1	1613,3	97,2				
	3	473	1598,1	59,1	1609,2	63,1				
	4	355	1633,1	53,7	1641	56,4				
	5	284	1621,3	40,5	1642,4	45,1				
C120	1	1146	1133,1	0	1163,9	22,2	1426,16**	280,16**	1429,115**	283,115**
	2	573	1363*	108,6*	1363*	108,6*				
	3	382								
	4	287								
	5	229								
C100b	1	902	865,9	0	870,1	0	1007,8	105,8	1015	113
	2	451	917,5	7,8	925,1	13	1049,7	75,6	1054	83,3
	3	301	867,6	0	871,3	0,7	982,7	29,5	1038,5	53,4
	4	225	873,3	0	882,7	0	998,6	45,8	1012,4	49,1
	5	180	987,8	27,2	1002,4	29,5				
F71	1	266	277,4	11,4	297,9	31,9	304,8**	38,8**	312,6**	46,6**
	2	133	294,6	15	305,6	20,1	314	24,1	322,2	28,6
	3	89	295	15,9	309	18				
	4	66								
	5	53								
F134	1	1279	1189,5	0	1255,3	12,1	1508,5	229,3	1547,4	268,2
	2	639	1214,2	0	1252,9	2	1500,7*	142,8*	1500,7*	142,8*
	3	426	1267,1	0	1323,3	16,9				
	4	319	1348,8	23,3	1382,7	32,1				
	5	255								
Thai75a	1	1780	1735,1	0	1758	0	1876	96	1982,8	202,8
	2	890	1740,1	0	1774,9	5,1	1990,8	108	2035,6	134,8
	3	593	1704,2	0	1780,2	13,1	2086,2**	108**	2097,3**	119,6**
	4	445	1819	12,1	1850	21,8				
	5	356	1860,2	26,5	1958,2	49,5				

Thai75b	1	1479	1356,7	0	1414	9,6	1478,2	0	1540,2	61,4
	2	740	1358,6	0	1366,6	0	1540,4	32	1579,5	52,2
	3	493	1431,1	0	1433,1	0	1613,2	53,8	1659,3	66,9
	4	370	1581,2	40,4	1622,9	44,7	1658**	64,7**	1679,6**	76**
	5	296	1611,8	71,5	1666,7	82,4				
Thai75c	1	1420	1510	90	1566,1	146,1	1473,1	53,1	1568,3	148,3
	2	710	1524,9	52,6	1531,9	56,1	1651,2	118,7	1686,4	141,2
	3	473	1524,1	56,5	1603,3	74,2	1617,1*	70,3*	1617,1*	70,3*
	4	355					1712,6*	80,5*	1712,6*	80,5*
	5	284								
Thai75d	1	1502	1449,6	0	1502,6	18	1619,2	117,2	1761,2	259,2
	2	751	1453,5	0	1475,4	4,6	1572	35,3	1610,8	65,9
	3	501	1386	0	1416,8	0	1665	64,2	1738,8	90,3
	4	375	1421,3	0	1442	1,7	1761,5**	93**	1806,9**	93,2**
	5	300	1496	13,8	1544,9	32,8				
Thai100a	1	2245	2100	0	2179,9	9,3	2487,7	242,7	2513,6	268,6
	2	1123	2106,5	0	2145,2	0	2621,6	192,8	2683,3	221,1
	3	748	2130,6	0	2247,8	14	2377,9	50,5	2531,4	106,9
	4	561	2169,2	0	2265,9	18,1	2462,6	75,9	2557,7	106,8
	5	449	2322,5	29,3	2349,3	36,3	2626,6**	92**	2653,2**	99,1**
Thai100b	1	2134	2087,9	0	2106	0	2443,4	309,4	2522,8	388,8
	2	1067	2079,9	0	2131,9	8	2584,2	229,4	2633,2	253,2
	3	711	2146,1	7,4	2195,3	22,6	2388,6	95,1	2462,9	118,7
	4	533	2130,6	1,1	2160,4	12,3	2497,6**	96,2**	2528,6**	106,6**
	5	427	2189,1	30,9	2217,9	43,5				
Thai100c	1	1547	1444,7	0	1465,5	0	1567	20	1633,1	86,1
	2	773	1440,8	0	1466	0	1505,8	0	1692,3	80,7
	3	516	1470,8	0	1515,8	2,4	1717,1	60,2	1820,9	95,4
	4	387	1652,7	27,7	1660,2	29,6				
	5	309	1558,8	13,1	1672,4	35,7				
Thai100d	1	1739	1767	28	1783,3	44,3	2017,6	278,6	2102,6	363,6
	2	869	1750,8	6,5	1800,3	31,8	2156,8*	216,7*	2156,8*	216,7*
	3	580	1755,5	5,7	1818,8	26,6				

	4	435	1776,1	14,2	1867,1	35,2				
	5	348	1820	27,9	1882,4	38,4				
Thai150a	1	3361	3464,5	103,5	3534,3	173,3	3772,7**	411,7**	3881,8**	520,8**
	2	1680	3437,3	39,1	3470,2	57,1	3887,2	274,9	4065	360,4
	3	1120	3200,3	0	3402,2	32,7	3819,7*	164,8*	3904,3*	206,4*
	4	840	3411,5	17,9	3460,5	33,4	3934,3	172,3	3956,2	173,6
	5	672	3296,8	0	3357,9	7,9				
Thai150b	1	3000	2943,4	0	3013,1	32	3426,5	426,5	3436,2	436,2
	2	1500	3079,8	40	3135,3	67,8	3281,8	155,4	3405,9	212,1
	3	1000	2973,6	0	3014,1	11,6	3612,9	230,5	3612,9	230,5
	4	750	2947,5	0	3029,8	17,8				
	5	600	3173,8	41,1	3186	47,9				
Thai150c	1	2595	2498,2	0	2599,2	50,5	3006,8	411,8	3086,4	491,4
	2	1297	2481,2	0	2584,2	15,3	3190,4*	312,3*	3190,4*	312,3*
	3	865	2681,5	29,6	2720,6	42,4				
	4	649	2557,1	0	2613,3	10,4				
	5	519	2561,1	0	2747,6	36,3				
Thai150d	1	2910	2875,7	0	2894,4	0	3323	413	3416,5	506,5
	2	1455	2927,3	8,9	2943	16,6	3431,4	271,4	3459,5	289,5
	3	970	2903,7	1,1	2920,8	8,9				
	4	727	2913,7	3,5	2930,5	7,5				
	5	582	2967,2	11,8	3003,4	21,5				

Tableau 22: Meilleure solution des instances ombrées

	m	T	MA		ACSH	
			Distance	Overtime	Distance	Over
C75	3	306	942,9	8,8		
C75	4	230			1069,3	45,1
C100	2	454			989,7	46,1
C150	3	377	1279,4	50,6		
C150	5	226	1389,6	52,9		
C199	4	355	1633,8	53,7		
C100b	2	451			1052,9	75,6
C100b	4	225			1010,6	45,8
C100b	5	180	1022,3	27,2		
F134	1	1279			1549,2	270
Thai75b	4	370	1627	40,4		
Thai75d	4	375			1852,2	93
Thai100a	5	449	2330,4	29,3		
Thai150a	4	840			3946,6	172,3
Thai150b	5	600	3197,4	41,1		
Thai150d	2	1455			3435,1	271,4