Thèse
de doctorat
de l'UTT

Syrine Roufaida AIT HADDADENE

Modèles et méthodes pour la gestion logistique optimisée dans le domaine des services et de la santé

Spécialité :
Optimisation et Sûreté des Systèmes

2016TROY0027

Année 2016

utt
université de technologie
Troyes

THESE

pour l'obtention du grade de

DOCTEUR de l'UNIVERSITE DE TECHNOLOGIE DE TROYES

Spécialité : OPTIMISATION ET SURETE DES SYSTEMES

présentée et soutenue par

Syrine Roufaida AIT HADDADENE

le 30 septembre 2016

Modèles et méthodes pour la gestion logistique optimisée dans le domaine des services et de la santé

JURY

M. V. T'KINDT	PROFESSEUR DES UNIVERSITES	Président
M. T. GARAIX	MAITRE DE CONFERENCES	Examinateur
M. A. GUINET	PROFESSEUR DES UNIVERSITES	Rapporteur
Mme N. LABADIE	MAITRE DE CONFERENCES - HDR	Directrice de thèse
M. A. MOUKRIM	PROFESSEUR DES UNIVERSITES	Rapporteur
Mme C. PRODHON	MAITRE DE CONFERENCES - HDR	Directrice de thèse

Remerciements

Je tiens à remercier au premier lieu les personnes qui m'ont accompagnées tout au long de cette thèse : mes directrices Nacima Labadie et Caroline Prodhon. Merci à elles pour la confiance qu'elles m'ont accordée, le temps qu'elles ont pu me consacrer et tous les conseils qu'elles m'ont donnés. Je tiens à vous exprimer toute ma gratitude et ma reconnaissance.

Je voudrais ensuite remercier chaleureusement les membres du jury. Merci à Alain Guinet et à Aziz Moukrim pour l'honneur qu'ils m'ont fait d'être les rapporteurs de cette thèse et pour avoir pris le temps de lire ce mémoire dans les moindres détails et juger ce travail. Un grand merci à Thierry Garaix et à Vincent T'kindt d'avoir accepté d'examiner ce travail.

Une thèse, c'est aussi une vie de bureau. C'est pourquoi je tiens à remercier tout particulièrement Elyn Solano, Matthieu Le Berre et Mourad Benttaleb. Nos échanges pas toujours scientifiques, nos déjeuners, ... merci pour la bonne humeur qui a rythmé ces trois ans et qui les ont rendues agréables. Je remercie également les personnes formidables que j'ai rencontrées à l'UTT et qui ont partagé mon quotidien : Jorge Victoria, Julien Autuori, Oussama Masmoudi, Birome Holo Ba, Sona Kandé, Nacef Tazi, Marie-Hélène Mai, Gustavo Bula Duc Nguyen et Mazen Hittawe. Grâce à eux, des moments sympathiques ont illuminé mes trois années. Merci également à mes amis hors UTT : Zaynab Habibi et Noel Youmbi qui ont toujours cru en moi et m'ont apporté tout leur soutien pour finaliser cette thèse.

J'adresse bien de vifs remerciements à Faicel Hnaien et Murat Afsar avec qui j'ai partagé les joies de l'enseignement. Ce fut une expérience très enrichissante à laquelle je crois bien avoir pris goût.

Je dédie ce manuscrit à mes chers parents. Je ne les remercierai jamais assez pour les sacrifices consentis à notre égard, mes deux frères et moi, et sans qui ma vie ne serait pas aussi riche et joyeuse. Je vous remercie infiniment pour votre soutien sans cesse, votre patience et surtout la fierté et la confiance que vous m'apportez. Merci à Oussama, mon cher grand frère, mon bras droit et mon ami d'enfance pour sa présence et ses accompagnements. Merci également à Rany, mon cher petit frère, même s'il ne comprend pas toujours tout ce que je fais de mes journées, m'a toujours apporté joie et bonne humeur.

Enfin, je ne peux terminer sans exprimer toute ma reconnaissance à mon fiancé Idir Gaci.

Je remercie le destin de l'avoir mis sur mon chemin afin qu'il m'apporte la joie et le bonheur.

Merci pour ta compréhension, ton soutien, ta confiance et surtout ta patience.

Encore un grand merci à tous pour m'avoir conduit à ce jour mémorable.

Syrine Roufaida AIT HADDADENE.

Résumé

Cette thèse aborde le problème de tournées de véhicules (VRP) intégrant des contraintes temporelles : fenêtres de temps (TW), synchronisation (S) et précédence (P), appliqué au secteur de soins à domicile, donnant le VRPTW-SP. Il s'agit d'établir un plan de visite journalier des soignants, aux domiciles des patients ayant besoin d'un ou plusieurs services.

Tout d'abord, nous avons abordé ce problème sous angle mono-objectif. Ensuite, le cas bi-objectif est considéré. Pour la version mono-objectif, un Programme Linéaire à Variables Mixtes Entières (PLME), deux heuristiques constructives, deux procédures de recherches locales et trois métaheuristiques à base de voisinages sont proposés : une procédure de recherche constructive adaptative randomisée (GRASP), une recherche locale itérée (ILS) et une approche hybride (GRASP $\times$ ILS).

Concernant le cas bi-objectif, différentes versions de métaheuristiques évolutionnaires multi-objectifs sont proposées, intégrant différentes recherches locales : l'algorithme génétique avec tri par non-dominance version 2 (NSGAII), une version généralisée de ce dernier avec démarrages multiples (MS-NSGAII) et une recherche locale itérée avec tri par non-dominance (NSILS). Ces algorithmes ont été testés et validés sur des instances adaptées de la littérature.

Enfin, nous avons étendu le VRPTW-SP sur un horizon de planification, donnant le VRPTW-SP multi-période. Pour résoudre cette extension, un (PLME) ainsi qu'une matheuristique sont proposés.

Mots-clés : Recherche opérationnelle, Optimisation combinatoire, Transport – Planification, Synchronisation, Soins à domicile, Métaheuristiques, Décision multicritère, Modèles mathématiques.

Abstract

This work addresses the vehicle routing problem (VRP) including timing constraints : time windows (TW), synchronization (S) and precedence (P), applied in Home Health Care sector ; giving the VRPTW-SP. This problem consists in establishing a daily caregivers planning to patients' homes asking for one or several services.

We have started by considering the problem as a single objective case. Then, a bi-objective version of the problem is introduced. For solving the single-objective problem, a Mixed Integer Linear Program (MILP), two constructive heuristics, local search procedures and three local search based metaheuristics are proposed : a Greedy Randomized Adaptive Search procedure (GRASP), an Iterated Local Search (ILS) and a hybrid approach (GRASP $\times$ ILS).

Regarding the bi-objective VRPTW-SP, different versions of multi-objective evolutionary algorithm, including various local research strategies are proposed : the Non-dominated Sorting Genetic Algorithm version 2 (NSGAII), a generalized version of this latter with multiple restarts (MS-NSGAII) and an Iterated Local Search combined with the Non-dominated Sorting concept (NSILS). All these algorithms have been tested and validated on appropriate instances adapted from the literature.

Finally, we extended the VRPTW-SP on a multi-period planning horizon and then proposed a MILP and a matheuristic approach.

Keywords : Operations research, Combinatorial optimization, Transportation – Planning, Synchronization, Home care services, Metaheuristics, Multiple criteria decision making, Mathematical models.

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Contexte

L'utilisation efficace des ressources tout en garantissant des services de qualité stimulent la réflexion logistique dans le secteur de la santé. Les établissements de santé sont amenés à réduire les coûts de consommation des produits, limiter les pertes et fournir un meilleur suivi des services offerts. En effet, l'augmentation des frais de santé entraîne des déficits inquiétants de la sécurité sociale. Ceci impose une réduction des dépenses. Or, le principal poste budgétaire des hôpitaux, défini par le salaire des personnels, étant pratiquement incompressible, les dépenses doivent être diminuées sur les autres postes. Dans ce contexte, les coûts des produits pharmaceutiques ainsi que les déplacements des personnels représentent une partie importante de la masse financière sur laquelle il semble possible d'agir rapidement.

Les établissements d'Hospitalisation à Domicile n'échappent pas à cette tendance. Ces dernières sont des structures de soins alternatives à l'hospitalisation qui permettent d'assurer au domicile du patient des soins médicaux et paramédicaux. Ils sont apparus dans le but d'initier des réductions des frais immobiliers et mobiliers (frais fixes) tout en maintenant un niveau de qualité de service satisfaisant. Cependant, en France, le développement de ces structures s'accélère et des études spécifiques sont nécessaires pour gérer de façon optimisée les dépenses relatives à leur fonctionnement.

L'obligation de contrôle des dépenses de santé semble être une raison nécessaire et suffisante pour entamer des démarches de restructuration de la logistique. Ces problématiques affirment la nécessité d'apporter une contribution significative à la performance du système de santé par la mise à disposition des bons soins ou produits, au bon moment et à moindre coût en préservant la qualité de service.

Cette thèse est consacrée à l'organisation des structures d'hospitalisation et de soins à domicile. Les travaux de recherche dans ce secteur ont comme but de mettre en place une coordination fine et une planification optimisée des ressources humaines et matérielles pour assurer une prise en charge et un suivi de soin de qualité, tout en maîtrisant les coûts. Ce type d'activité permet soit de maintenir à domicile des personnes qui n'étant pas entièrement

dépendantes, nécessitent néanmoins une surveillance minimale, ou de faciliter le retour à une vie normale pour des personnes ayant subi des pathologies lourdes en les autorisant à quitter l'établissement de santé et en proposant de prodiguer les soins à domicile. Les soins apportés sont généralement réalisés à des horaires spécifiques et parfois nécessitent l'intervention simultanée de plusieurs aides soignants qualifiés. L'objectif principal dans ce cas est de planifier, synchroniser et coordonner les activités afin d'apporter une organisation optimisée permettant l'accès aux soins, sans recours à l'hospitalisation classique.

En associant les patients (malades) à des clients et les soignants (personnels de soins) à des véhicules, le problème introduit dans cette thèse peut être défini comme une variante du problème de tournées de véhicules où les soins demandés doivent être réalisés à des horaires spécifiques par des soignants qualifiés. Cette variante sera appelée *VRPTW-SP Vehicle Routing Problem With Time Window, Synchronization and Precedence constraints* où certains clients demandent plusieurs services simultanément ou dans un ordre prédéfini. Les critères d'optimisation concernent la minimisation des coûts de déplacement des personnels de soins tout en maximisant la satisfaction des patients.

Contributions

Cette thèse est organisée comme ceci, nous commençons dans le chapitre 1 par présenter le système d'hospitalisation à domicile en France. En effet, cette étape est nécessaire afin de comprendre son fonctionnement et définir ses besoins.

Avant d'aborder les problèmes étudiés dans le cadre de cette thèse, il est nécessaire de présenter les contributions proches à notre sujet afin de bien situer notre travail. Pour cette raison, un état de l'art sur les problèmes de logistique de transport en général, et particulièrement les problèmes de logistique liés à la santé est présenté dans le chapitre 3. Un rappel des outils de recherche opérationnelle et d'aide à la décision est donné dans le chapitre 2.

Une définition plus précise du VRPTW-SP étudié durant cette thèse est exposée dans le chapitre 4 dans sa version mono-objectif pour lequel une modélisation mathématique est proposée et est testée en utilisant le solveur linéaire *Cplex*. Les heuristiques constructives et les recherches locales que nous avons développées spécifiquement pour le problème étudié sont également présenté dans ce même chapitre.

Ensuite, trois méthodes approchées de type métaheuristique (GRASP, ILS et l'hybridation de ces deux dernières appelée GRASP $\times$ ILS) sont proposées dans le chapitre 5. Celles-ci sont testées et comparées entre elles et aux solutions obtenues par le solveur linéaire.

Le chapitre 6 est consacré à la résolution multi-objectif du problème introduit dans cette thèse, en proposant différentes méthodes approchées de type métaheuristique à base de population.

Les tests sont réalisés sur une version adaptée des instances de la littérature, introduites par Bredström & Rönnqvist (2008) pour un problème similaire à celui considéré ici, mais n'incluant pas initialement certaines caractéristiques spécifiques au problème auquel nous nous intéressons.

Le chapitre 7, propose une introduction détaillée à l'une de nos perspectives. Cette dernière consiste à étendre le VRPTW-SP sur un horizon de planification, la variante déduite est alors appelée *P-VRPTW-SP* pour *Periodic Vehicle Routing Problem With Time Window, Synchronization and Precedence constraints*. Dans ce chapitre, un bref état de l'art sur les problèmes de tournées de véhicules périodiques dans le secteur de la santé est d'abord exposé, la description du problème est définie à l'aide d'une formulation mathématique pour finir par la proposition d'une méthode de résolution de type matheuristique.

Pour terminer, une conclusion sur les travaux de cette thèse ainsi que les perspectives de recherche dans ce domaine sont discutées.

Notons que les modèles et méthodes proposés dans cette thèse sont conçues pour résoudre les différents problèmes rencontrés dans les structures d'hospitalisation à domicile, mais sont également applicables aux problèmes de logistique au sens large.

Chapitre 1

Logistique d'hospitalisation à domicile

1.1 Introduction

Depuis une cinquantaine d'années, le secteur de la santé est en pleine mutation et la France n'y échappe pas. Dans une optique de réduction des coûts et d'amélioration du confort, le nombre de lits dans les établissements hospitaliers ne cesse de baisser entraînant de nouvelles formes de demandes. Les patients n'exigent pas en général une lourde prise en charge mobilisant un plateau technique de haut niveau et quittent plus précocement l'hôpital. C'est le cas par exemple des patients souffrant de pathologies graves, aiguës ou chroniques, pour qui il peut être préférable que la durée de séjour en hôpital soit aussi réduite que possible, d'où l'intérêt de l'apparition des structures de prise en charge aux domiciles des patients et/ou sur différents lieux de vie tels que les écoles, les entreprises, etc.

En France, les structures de santé à domicile sont apparues suite à différents facteurs, essentiellement :

— L'espérance de vie qui augmente de plus en plus, avec une durée de vie moyenne dépassant les 80 ans, engendrant ainsi une croissance du nombre de personnes souffrant de maladies chroniques, donnant lieu à des incapacités fonctionnelles et à des handicaps.

— Le besoin de confort personnel des patients qui demandent une prise en charge les soustrayant le moins possible de leur environnement habituel.

— La réduction des moyens et le manque de lits disponibles des structures de soins classiques.

L'hospitalisation à domicile (HAD) garantit aux patients à la fois la sécurité des soins, et le confort psychologique et physique. Elle permet d'assurer au domicile du malade, pour une durée limitée et parfois révisable en fonction de l'évolution de son état de santé, des soins médicaux et paramédicaux continus et surtout coordonnés pour une meilleure qualité

de service. Mais les soins apportés par ce type de structures requièrent une planification fine des activités des soignants afin de répondre aux besoins et exigences des patients tout en limitant les coûts.

Cette thèse traite les problèmes de tournées de véhicules rencontrés dans le secteur de la santé, particulièrement dans les établissements d'hospitalisation à domicile. Avant de les aborder plus spécifiquement, ce chapitre rappelle dans la section 1.2 l'historique de ces établissements de santé. L'organisation interne est présentée dans la section 1.3 et le déroulement de la prise en charge des patients au sein de ces structures est détaillé dans la section 1.4. Enfin, le chapitre se conclut dans la section 1.5.

1.2 Historique de l'hospitalisation à domicile

1.2.1 Création du concept d'HAD

L'hospitalisation à domicile a quelques décennies d'existence. Elle constitue un mode de prise en charge original permettant aux malades de bénéficier chez eux de soins médicaux et paramédicaux que seuls des établissements hospitaliers peuvent leur prodiguer.

En 1951, à cause du nombre important de malades, une première réflexion autour de la possibilité de soigner les patients à domicile en particulier ceux atteints du cancer, fut apparue en France à l'hôpital Tenon (Paris). À cette époque, l'infirmière avait une double responsabilité, assurer les soins tout en ayant une approche sociale du patient et de son entourage.

Ces structures se sont développées en France à partir de 1957 en s'inspirant des expériences américaines "Home Care" du Professeur E. M Bluestone, chef de service à l'hôpital Montefiore de New York qui, à cause d'une surpopulation importante de patients décida de suivre certains malades, dont l'état de santé ne nécessitait plus une présence continue, directement chez eux. L'intérêt est de privilégier le maintien à domicile du malade, tout en assurant la même qualité de soin qu'en hospitalisation traditionnelle (Hôpital).

L'objectif de la création de cette première structure de soins à domicile, consistait d'une part à désencombrer les hôpitaux vu l'évolution des pratiques médicales qui requéraient de moins en moins une présence indispensable au sein de l'hôpital, et d'autre part satisfaire les malades souhaitant être soignés dans leur environnement familial.

En 1958, une deuxième structure a été créée à Puteaux, connue sous le nom de « santé service » et destinée à des patients atteints de cancer.

Les années qui suivent, voient se multiplier le nombre de structures par la création des services HAD de pédiatrie, kinésithérapie, obstétrique, ergothérapie, diététique et nutrition

parentérale.

Depuis, la capacité d'accueil s'est vu monter en flèche pour passer de 3908 places autorisées (dont 3832 places installées) en 1999 à 4739 places autorisées (dont 4203 places installées) en 2002 pour atteindre les 7266 places autorisées (mais le nombre de places réellement ouvertes semble largement inférieur) en juin 2006, réparties entre 185 structures.

En mai 2009, une enquête par questionnaire a été réalisée par le groupe de travail coordonné par le CCLIN Sud-Ouest *Centre de Coordination et de lutte Contre Les Infections Nosocomiales* et proposée par la DGS *Direction Générale De La Santé* aux HAD de France, a permis d'identifier 12766 places autorisées pour les 148 établissements qui ont répondu à l'enquête parmi 244 établissements existants.

1.2.2 Objectif des structures HAD

L'HAD vise principalement à coordonner les interventions de plusieurs professionnels afin d'assurer au patient les meilleures conditions d'hospitalisation au sein de son environnement familial, de garantir une prise en charge globale et ainsi de réduire ou d'éviter le séjour à l'hôpital. Enfin, cette forme d'hospitalisation tente d'assurer la maîtrise des coûts des prestations dans un contexte de limitation de la progression des dépenses de santé, car si un patient n'était pas pris en charge par un établissement d'HAD, il serait forcément hospitalisé dans un hôpital traditionnel.

Enfin, les structures d'HAD ont pour mission de répondre aux demandes des patients, sans exclusion de durée de séjour ou de gravité de l'état de santé.

1.3 Organisation des établissements HAD

Une structure d'hospitalisation à domicile est un mini réseau comportant un nombre d'acteurs aux compétences variées intervenant dans la prise en charge du patient. Cette dernière est propre aux patients et diffèrent selon le type de besoin. Les principaux acteurs identifiés sont les suivants :

Le responsable de l'établissement : représente l'interface administrative assurant la gestion des différents acteurs.

Les secrétaires : assurent l'accueil téléphonique et physique au siège de l'établissement d'HAD et effectuent toutes les tâches administratives relatives aux patients.

Le médecin coordonnateur : responsable de la prise en charge globale du patient, c'est le référent principal de la structure, qui se charge de donner l'avis médical pour toute admission ou sortie en garantissant la qualité de la prise en charge des malades.

Les infirmier (ère)s : conduisent la mission de coordination des soins des malades. Lors de l'admission d'un patient. L'infirmier (ère) rencontre le malade ainsi que sa famille dans le but de recueillir leurs attentes et ainsi anticiper l'organisation à mettre en place.

Les aides-soignants : réalisent des soins d'hygiène, de confort et de bien-être et assurent une observation du patient permettant d'alerter les infirmier (ère)s.

Intervenants paramédicaux : tels que le kinésithérapeute, le psychomotricien, l'orthophoniste,... qui concourent au projet thérapeutique du patient et se chargent d'établir parfois des comptes rendus de suivi des patients.

L'assistant(e) de service social : sa présence est essentielle au cours de la prise en charge pour le malade et son entourage dans le but d'évaluer la capacité (en terme d'espace) de l'entourage à maintenir matériellement l'hospitalisation à domicile. Il met en œuvre des moyens matériels, humains et financiers pour pallier certaines difficultés qui pourraient compromettre la prise en charge en HAD.

Le psychologue : assure l'accompagnement psychologique du patient mais aussi de son entourage. Il soutient l'équipe de l'établissement d'HAD devant la complexité des enjeux psychologiques liés à certaines situations.

Le logisticien : (ou dans certains cas le pharmacien) gère les aspects logistiques dans l'HAD. Il s'occupe des livraisons du matériel et des médicaments. Il est censé avoir aussi une idée sur le niveau de stock de l'HAD pour passer les commandes en cas de manque.

Des détails sur les rôles des différents intervenants en HAD peuvent être trouvés dans Guinet (2014).

1.4 Organisation de la prise en charge d'un patient

La prise en charge d'un patient au sein des HAD se déroule en quatre étapes essentielles :

1. Les évaluations qui précèdent l'admission :
   — Médicales : réalisées par le médecin coordonnateur de l'établissement d'HAD , pour établir le projet thérapeutique du malade.

— Paramédicales : réalisées par l'équipe paramédicale attachée à l'établissement d'HAD. Il s'agit par exemple, de prendre contact avec les anciens soignants du malade dans le but d'élaborer un projet de soin conjointement avec eux.
— Sociales : réalisées par l'assistant de service social de l'établissement d'HAD afin que le besoin matériel ou humain soit définit de manière à garantir la meilleure qualité de soin.

2. La réunion initiale : il s'agit de regrouper certains professionnels, qui interviendront au domicile du malade au cours de la prise en charge. Cette réunion permet de :
   — Veiller à la bonne installation du malade.
   — Présenter au patient et à son entourage l'équipe de l'HAD en leur expliquant le fonctionnement.
   — Remettre au patient le livret d'accueil.
   — Affiner le projet thérapeutique et préciser les traitements.
3. Le suivi de la prise en charge au cours du séjour : les différents intervenants de l'HAD assurent les soins chaque jour voire plusieurs fois par jour, dans certains cas. Le suivi de la prise en charge s'effectue par les visites des infirmiers au moins chaque semaine au domicile du malade.
4. La fin de la prise en charge : quel que soit le mode de sortie, le médecin coordinateur rédige systématiquement un compte rendu d'hospitalisation qui sera transmis au médecin prescripteur (ou traitant).

D'après les organisations d'HAD, les soins sont généralement organisés pour qu'il y ait au minium un passage par jour au domicile du malade. Ces passages sont aussi nombreux que nécessaire, et des visites de suivi peuvent avoir lieu périodiquement par les infirmiers. Ainsi, plusieurs types d'intervention nécessitent la coordination de différents acteurs (médecins, infirmiers, etc). Plus de détails sur ce mode de prise en charge ainsi que son importance sont soulignés par Guinet (2014).

Durant ce projet de thèse, nous avons eu l'occasion de discuter avec des structures d'HAD, et nous avons remarqué que la planification est généralement faite à la main et au jour le jour selon les types de services. Nous avons également identifier que le nombre de malades nécessitant la coordination de plusieurs acteurs, représentaient environ 10% du nombre total des patients affectés à cette HAD. La planification des déplacements du personnels soignants est une tâche difficile et surtout coûteuse en temps. Elle doit respecter de nombreuses contraintes (disponibilité, compétences, ...), tout en garantissant un niveau de service maximal en limitant les coûts. Ainsi, il est évident qu'un besoin en outil d'aide à la décision est ressenti en HAD pour planifier le travail du personnel chargé des visites. C'est d'ailleurs, dans ce cadre que porte le travail présenté dans cette thèse.

1.5 Conclusion

Au fil des années, les établissements d'hospitalisation et de soins à domicile ont pris de plus en plus d'importance dans le contexte de la santé. L'un des aspects les plus essentiels est la qualité des soins, qui doit être équivalente à ceux donnés dans un hôpital traditionnel. Ce mode de prise en charge a permis d'améliorer également la qualité de service des hôpitaux traditionnels en réduisant les problèmes de taux d'occupation des lits.

Les informations rassemblées et synthétisées dans ce chapitre nous permettent de comprendre le système étudié, et nous serviront davantage dans la définition des problèmes abordés par la suite.

Dans le secteur de la recherche, les travaux portant sur la prise en charge des patients à domicile se développent de plus en plus intégrant des domaines variés (statistique, économie/gestion, recherche opérationnelle, ...). La littérature portant sur les problématiques relatives à la recherche opérationnelle appliquée aux structures d'hospitalisation et de soin à domicile sera exposée en détail dans le chapitre 3.

En attendant, le chapitre suivant rappelle les outils de base de recherche opérationnelle et d'aide à la décision généralement utilisés pour une logistique de transport optimisée.

Chapitre 2

Logistique du transport et outils de recherche opérationnelle

2.1 Introduction

La recherche opérationnelle (RO) a pour but de résoudre les problèmes de décisions ou d'optimisation à l'aide d'outils mathématiques et informatiques. On situe habituellement sa naissance à la deuxième guerre mondiale, lorsque l'état-major britannique fit appel à des équipes de mathématiciens et de physiciens pour l'aider à analyser divers aspects des opérations militaires, même si, les problèmes relevant de la RO sont plus anciens.

Après la guerre, cette approche systématique et scientifique des problèmes de décision a été vite étendue à d'autres domaines tels que l'industrie, la finance ou encore le transport où elle a connu de nombreux succès. Concernant l'optimisation du transport, comme par exemple l'organisation et l'optimisation des tournées d'un ou plusieurs types de véhicules, les méthodes de résolution envisagées sont généralement soit des méthodes exactes dans le but de trouver une solution optimale au problème posé, soit des méthodes approchées, dont le but est de trouver une solution de bonne qualité, sans garantie d'optimalité mais en des temps de calcul plus au moins raisonnables. Des méthodes hybrides combinant les approches exactes et approchées sont également apparues ces dernières décennies.

Dans cette thèse, nous étudions des problèmes d'optimisation combinatoires qui font partie de la catégorie des problèmes de transport, et plus précisément des tournées de véhicules. Ces problèmes sont très répandus en logistique et il en existe une multitude de variantes, plus souvent désignées par des acronymes pour les distinguer les uns des autres.

La principale motivation de l'étude d'une telle catégorie de problèmes revient à son large champs d'applications pratiques et ces implications économiques et environnementales. D'après Toth & Vigo (2014), les frais de transport représentent d'environ 10% à 20% des prix

finaux des marchandises. Les techniques d'optimisation permettent souvent de développer des systèmes de transport flexibles et efficaces répondant aux préoccupations actuelles relatives à l'économie et à la qualité de vie et d'économiser de 5% à 20% sur ces coûts de transport. Les techniques d'optimisation combinatoire et de recherche opérationnelle s'avèrent donc essentielles.

Avant d'aborder ces problèmes et ainsi les résoudre, il est important de comprendre les types de décisions que l'on peut considérer. La section suivante propose une classification des niveaux de décisions. Ensuite, la section 2.3 rappelle les différentes caractéristiques des problèmes de logistique. La section 2.4 est consacrée à la description de deux problèmes très répandus dans le domaine de transport : le problème de voyageur de commerce et le problème de tournées de véhicules. Les variantes les plus courantes sont également décrites. Les méthodes de résolution exactes et approchées permettant de résoudre de tels problèmes sont présentées dans la section 2.5. Enfin, une synthèse des modèles et méthodes est proposée donnant l'orientation du chapitre suivant.

2.2 Niveaux de décision

Une décision est définie comme étant l'action de donner une valeur à une variable inconnue et dont la connaissance permet aux décideurs de faire un choix. Par conséquent, concevoir une chaîne logistique revient à prendre un ensemble de décisions qui peuvent être regroupées en trois classes selon le niveau qu'elles impliquent :

— Stratégique : ce sont des décisions à long terme (d'une durée d'un an ou plus). Il s'agit par exemple de décider du recrutement des soignants.
— Tactique : ce sont des décisions à moyen terme (durent de quelques mois à un an). Il s'agit par exemple de décider de la fréquence de visite des patients dans le cas des tournées périodiques.
— Opérationnel : ce sont des décisions à court terme (qui durent de quelques jours à un mois), dont le but est par exemple d'élaborer les tournées à réaliser au jour le jour.

En raison de la complexité du problème d'optimisation des décisions, ces trois types de décisions sont généralement traités de manière hiérarchique basée sur la portée temporelle.

Les problèmes de logistique étudiés dans cette thèse sont de niveau tactique ou opérationnel.

2.3 Caractéristiques des problèmes de logistique

Dans cette section, nous analysons les différentes caractéristiques des problèmes de logistique, en accentuant sur la logistique du transport, car elle fait l'objet principal de l'étude menée dans cette thèse. Ainsi, nous présentons les aspects que l'on doit prendre en considération afin de donner une modélisation au problème étudié.

2.3.1 Nature des données

En général, les données considérées comportent une part d'incertitude, car il est difficile par exemple de prévoir les demandes, malgré cela il est nécessaire de prendre des décisions stratégiques, tactiques ou même opérationnelles. C'est pourquoi, la plupart du temps, les études sont faites avec des données moyennes ou considérées connues avec certitudes (cas déterministe). C'est le cas de cette étude. Parfois, l'incertitude est prise en compte par des lois de probabilités associées à certaines données (cas stochastique).

2.3.2 Nature du problème

Le problème d'optimisation peut avoir une nature statique, dynamique ou encore périodique (selon l'horizon considéré et relativement à la nature des données, qui peuvent être constantes ou encore peuvent varier dans le temps). L'approche la plus répandue est l'approche statique, considérant des données constantes sur la période de l'étude. Cependant, il est parfois nécessaire d'étudier les cas dynamique et périodique.

2.3.3 Les contraintes

Dans un problème de logistique du transport, on peut avoir une multitude de contraintes qui entrent en jeu telles que les contraintes de capacité des véhicules/dépôts, contraintes temporelles, ... Lorsqu'il s'agit simplement de visiter un ensemble de clients au moindre coût avec un seul véhicule sans considérer sa capacité, le problème est appelé problème du voyageur de commerce ou (TSP) *Traveling Salesman Problem*. C'est le problème de base en logistique de transport où tous les clients sont visités par une seule tournée.

Ce dernier a été généralisé par la suite au problème de tournées de véhicules (VRP pour *Vehicle Routing Problem*) qui se différencie du TSP par le nombre de véhicules et/ou la contrainte de capacité qui exige que la somme totale des demandes des clients n'excède pas la capacité du véhicule, sachant que la demande de chaque client est supposée inférieure à celle ci. Les véhicules peuvent être considérés tous de même capacité (flotte homogène) ou encore de capacités différentes (flotte hétérogène).

D'autres types de contraintes peuvent aussi s'ajouter, telles que les contraintes d'ordonnancement des tâches, des fenêtres de temps, ... Cette variété de contraintes a fait naître de nombreuses variantes de VRP, nous présentons plus en détail certaines d'entre elles dans la section 2.4.

2.3.4 Les objectifs

Le dernier aspect à aborder est relatif aux objectifs visés. Lorsqu'il s'agit d'optimiser un ou plusieurs termes qui ne sont pas contradictoires, ils sont généralement combinés pour formuler une unique fonction-objectif (par exemple minimiser la somme de plusieurs coûts). En revanche, quand plusieurs objectifs opposés sont à optimiser, par exemple la minimisation des distances parcourues ainsi que l'équilibrage de la charge de travail, agréger les objectifs en une fonction unique n'est généralement pas pertinent et mieux vaut alors les séparer. On parle alors des approches multi-objectifs, qui consistent à séparer les fonctions (critères) à optimiser simultanément. Dans ce cas, il est impossible de définir la valeur optimale en toute généralité. Il existe plutôt un ensemble de valeurs optimales, formant une frontière dite de Pareto.

2.3.5 Représentation des problèmes de logistique du transport

En général, les problèmes de tournées de véhicules sont modélisés sur un graphe, souvent noté par $G = (V, E)$. Ce dernier est défini par :

- V, l'ensemble d'objets généralement représentés par des nœuds/sommets et qui peuvent correspondre à une ou plusieurs informations (indice du client, du dépôt, ...).
- E, l'ensemble des relations existantes entre ces objets, représentées par des arcs/arêtes et qui peuvent être pondérés (distance séparant deux nœuds, durée de déplacement, ...).

En logistique de transport, les relations sont souvent orientées (arcs), et la relation entre deux éléments d'un graphe n'est pas forcément symétrique. On parle alors d'un *graphe orienté* et *asymétrique*.

2.4 Modèles mathématiques

Nous allons présenter ici les modèles mathématiques de quelques problèmes de transport proches de notre sujet d'étude et dit "*Less-Than-Truck-Load*" où les visites des clients ne sont pas forcément assurées par des véhicules dédiés mais regroupés en tournées.

2.4.1 Le problème du voyageur de commerce

Le problème du voyageur de commerce est certainement le problème le plus célèbre en optimisation combinatoire. Illustré par le mathématicien *William Rowan Hamilton* en 1859, il consiste en un voyageur de commerce ayant comme mission de visiter plusieurs villes en passant exactement une fois par chacune d'entre elles. Le but est donc de trouver le tour minimisant la distance totale parcourue par ce voyageur. Ce problème revient à trouver un cycle hamiltonien de coût minimal dans un graphe complet. Ce problème a été prouvé NP-difficile (voir le livre de Garey & Johnson (2002)). Ci-dessous, est données la première modélisation mathématique du TSP qui a été proposée par Dantzig & Ramser (1959b) pour modéliser un problème de livraison d'essence : Soit $G = (V, E)$ un graphe complet orienté où $V = \{1, 2, ..., n\}$ est l'ensemble des sommets représentant les clients à visiter et E est l'ensemble des arcs représentant les routes entre ces clients où à chaque arc $(i, j) \in E$ est associé un coût positif C_{ij}. Ce problème utilise des variables binaires x_{ij} telles que :

$$x_{ij} = \begin{cases} 1 & \text{si l'arc (i, j) est utilisé.} \\ 0 & \text{sinon.} \end{cases}$$

Le programme linéaire en nombres entiers est le suivant :

$$\min \sum_{(i,j)\in E} C_{ij}.x_{ij} \tag{2.1}$$

Sous les contraintes :

$$\sum_{i\in V} x_{ij} = 1 \qquad \forall j \in V \tag{2.2}$$

$$\sum_{i\in V} x_{ij} = 1 \qquad \forall i \in V \tag{2.3}$$

$$\sum_{i,j\in S} x_{ij} \leq |S| - 1 \qquad \forall S \subset V : 2 \leq |S| \leq n - 1 \tag{2.4}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in E : i \neq j \tag{2.5}$$

Sous cette formulation, la fonction objectif (2.1) exprime la minimisation de la somme des coûts engendrés par le déplacement sur les arcs. Les contraintes (2.2 et 2.3) sont des contraintes de conservation de flots permettant d'assurer que le voyageur entre et sort exactement une fois de chaque sommet. La contrainte (2.4) est utilisée pour interdire la formation de sous-tours dans la solution. En effet, sans cette contrainte, une solution peut prendre la forme de bulles qui ne sont reliées ni entre elles, ni au dépôt. En pratique, cette dernière est difficile à traiter du fait de sa cardinalité. Il y a en effet $2^{(n-1)} - 1$ sous-ensembles S à vérifier pour un problème de taille $|V| = n$, et par conséquent autant de contraintes.

En général, les contraintes de sous-tours sont relâchées en début de résolution, ensuite des algorithmes de détection des sous-tours permettront d'ajouter les coupes nécessaires et de relancer la résolution. Cette phase d'ajout de coupes/résolution est répétée autant de fois que nécessaire, jusqu'à ce que la solution soit réalisable.

2.4.2 Le problème de tournées de véhicules

Le problème de tournées de véhicules constitue une extension du TSP à plusieurs voyageurs dans laquelle K véhicules de capacité W doivent à partir d'un dépôt commun pour visiter/servir un ensemble de n clients. Initialement proposé par Dantzig & Ramser (1959a), le problème suppose la visite des clients à partir d'un dépôt au moyen d'une flotte de véhicules, avec un coût minimal. Le VRP est un problème NP-difficile, il généralise le problème du voyageur de commerce (TSP), qui consiste à visiter un ensemble de clients avec un seul véhicule. Plusieurs formulations mathématiques ont été proposées pour modéliser ce problème. Un état de l'art récent et détaillé de ces modélisations est dressé dans le livre de Toth & Vigo (2014). Nous présentons ici une formulation à trois indices connue par sa simplicité et proposée par Fisher & Jaikumar (1981). Ce modèle est défini sur un graphe orienté $G = (V, E)$ où $V = \{0, 1, 2, ..., n\}$ l'ensemble des nœuds et E l'ensemble des arcs. Le nœud 0 représente le dépôt et chaque $i \in V - \{0\}$ est un client à qui il faudrait livrer une quantité q_i de marchandises . Chacun des arcs $(i, j) \in E$ possède un poids C_{ij} représentant le coût de déplacement des véhicules du client i vers j. Cette modélisation nécessite n^2K variables de décision binaires x_{ijk} et nK variables y_{ik}, telles que :

$$x_{ijk} = \begin{cases} 1 & \text{si l'arc (i, j) est parcouru par le véhicule k.} \\ 0 & \text{sinon.} \end{cases}$$

et

$$y_{ik} = \begin{cases} 1 & \text{si le client i est visité par le véhicule k.} \\ 0 & \text{sinon.} \end{cases}$$

Le programme linéaire est le suivant :

$$\min \sum_{(i,j)\in E} \sum_{k\in K} C_{ij} x_{ijk} \tag{2.6}$$

Sous les contraintes :

$$\sum_{i\in V} q_i . y_{ik} \leq W \qquad \forall k \in K \tag{2.7}$$

$$\sum_{k\in K} y_{0k} = K \tag{2.8}$$

$$\sum_{k\in K} y_{ik} = 1 \qquad \forall i \in V \setminus \{0\} \tag{2.9}$$

$$\sum_{j\in V} x_{ijk} = y_{ik} \qquad \forall i \in V, \forall k \in K \tag{2.10}$$

$$\sum_{i\in V} x_{ijk} = y_{jk} \qquad \forall j \in V, \forall k \in K \tag{2.11}$$

$$\sum_{i,j\in S} x_{ij} \leq |S| - 1 \qquad \forall S \subset V \setminus \{0\} : 2 \leq |S| \leq n-1, \forall k \in K \tag{2.12}$$

$$x_{ijk} \in \{0,1\} \qquad \forall (i,j) \in E : i \neq j, \forall k \in K \tag{2.13}$$

$$y_{ik} \in \{0,1\} \qquad \forall i \in V, \forall k \in K \tag{2.14}$$

Sous cette formulation, la fonction objectif (2.6) exprime la minimisation des coûts de déplacement des véhicules sur les arcs. La capacité des véhicules est respectée à l'aide des contraintes (2.7). Le nombre de véhicules utilisés ne peut pas dépasser le nombre de véhicules disponibles dans le dépôt, ceci est géré par la contrainte (2.8). Les contraintes (2.9) quant à elles assurent que chaque client soit visité exactement une fois. Les contraintes (2.10 et 2.11) garantissent que chaque véhicule visitant un client reparte. Enfin, de même que pour le TSP, les contraintes (2.12) permettent l'élimination des sous-tours. Le reste des contraintes fixent la natures des variables.

2.4.3 Le problème de tournées de véhicules avec fenêtres de temps

Lorsqu'un véhicule doit visiter/servir un client, comme c'est le cas des soignants devant se déplacer aux domiciles des malades pour leur apporter des soins particuliers, souvent ces services doivent se réaliser dans des plages horaires spécifiques. Le respect de ces plages horaires ajouté au VRP défini le problème de tournées de véhicules avec fenêtres de temps (VRPTW pour *Vehicle routing problem with time windows*). Les premières applications traitées dans la littérature concernaient la distribution de courriers (Pullen & Webb (1967)), la planification de transport (Knight & Hofer (1968)), ... Depuis, ce problème a suscité une grande attention de la part des chercheurs et des industriels car

il apporte une composante fréquemment rencontrée dans différentes problématiques telles que la distribution des marchandises, le transport des écoliers, la logistique maritime, la planification des horaires de vols, ... Pour résoudre ce problème, Solomon (1987) a développé une heuristique efficace intégrant la notion du décalage des heures de passage induit par l'ajout d'un client à une tournée d'un véhicule donné. Il a construit un jeu d'instances qui fait encore référence aujourd'hui pour l'évaluation des algorithmes de résolution proposés pour le VRPTW. Par la suite, plusieurs références se sont imposées par leurs importantes contributions apportées à la résolution du VRPTW, par exemple Kindervater & Savelsbergh (1997) ont introduit des techniques permettant de vérifier en $O(1)$ la possibilité d'insérer ou pas un client dans la tournée d'un véhicule, tout en respectant les fenêtres horaires. Gehring & Homberger (1999) ont proposé deux métaheuristiques à base de population ainsi que de nouveaux jeux d'instances pour le VRPTW. Depuis, les méthodes heuristiques et métaheuristiques ont largement été utilisées pour résoudre ce problème, telles que celles proposées par Bräysy & Gendreau (2005), Labadi *et al.* (2008), Nagata (2007) et Bräysy *et al.* (2010).

De manière générale, le VRPTW désigne l'ensemble des problèmes de tournées de véhicules pour lesquels les services réalisés par les véhicules sont soumis à des fenêtres de temps. Dans ce contexte, les fenêtres de temps peuvent avoir deux types : souples ou dures. Les fenêtres souples autorisent que le service ait lieu en dehors de la plage horaire exigée par le client en question mais en contre partie une pénalité sera alors infligée. En revanche, les fenêtres dures n'autorisent en aucun cas le début de service en dehors du créneau spécifié. Une modélisation de ce problème peut être obtenue simplement en ajoutant au modèle à trois indices présenté précédemment pour le VRP, les contraintes suivantes :

$$t_{ik} + s_i + T_{ij} - M.(1 - x_{ijk}) \leq t_{jk} \qquad \forall i, j \in V \setminus \{0\}, \forall k \in K \tag{2.15}$$

$$a_i \leq t_{ik} \leq b_i \qquad \forall i \in V \setminus \{0\}, \forall k \in K \tag{2.16}$$

La variable t_{ik} représente la date de début de service chez le client i par le véhicule k, s_i est la durée de service chez le client i, T_{ij} est le temps de trajet entre les clients i et j, a_i et b_i sont respectivement l'heure de début au plus tôt et au plus tard du service chez le client i. La contrainte (2.15) assure la cohérence des instants de visite. En effet, elle vérifie que pour deux clients (i, j) visités consécutivement par le même véhicule, le début de service chez le client j ne peut commencer qu'à partir de fin de service du client i augmentée du temps de déplacement nécessaire pour se rendre vers le client j. L'utilisation des variables temporelles dans cette contrainte permet également l'élimination automatique des sous-tours ce qui conduit à la suppression des contraintes (2.12) dans le modèle dédié au VRP. Enfin,

la contrainte (2.16) assure que les demandes des clients soient servies dans leurs fenêtres de temps. M est une constante de valeur très grande.

2.4.4 Le problème de tournées de véhicules périodiques

Le problème de tournées de véhicules périodiques (PVRP pour *Periodic Vehicle Routing Problem*) est une extension du VRP sur un horizon de planification prédéfini. Dans ce problème chaque client doit être visité un certain nombre de fois dans un horizon H constitué de P périodes. A chaque client est associé une fréquence de jours de visites qui peut être soumise à des contraintes d'espacement, par exemple un client peut demander trois visites par semaine (i.e. $|H| = 7$), mais avec au moins deux jours de décalage entre chaque deux visites.

Ce problème a été utilisé pour modéliser différentes applications réelles, en particulier la collecte des déchets ménagers. Cette dernière reste l'application la plus répandue pour ce problème. La première description a été faite par Beltrami & Bodin (1974) où le but était de minimiser la distance totale parcourue tout en respectant les passages requis pour chaque client. Plus tard, Lacomme *et al.* (2005) et Chu *et al.* (2006) ont modélisé la collecte des déchets comme un problème de tournées sur arcs périodique. Plus récemment, Michallet *et al.* (2014) ont utilisé le PVRP pour modéliser le problème de tournées des convoyeurs de fonds dans lequel des fenêtres de temps et des contraintes de sécurité sont à respecter.

Les méthodes heuristiques et métaheuristiques ont été utilisées pour résoudre ce problème. A notre connaissance, la meilleure métaheuristique actuellement est l'algorithme génétique proposé par Vidal *et al.* (2013). Des méthodes exactes ont également été proposées, citons par exemple l'approche par relaxation lagrangienne de Baldacci *et al.* (2008) qui permet de résoudre optimalement des instances jusqu'à 100 clients et 6 périodes.

2.4.5 Le problème de tournées de véhicules avec ramassage et livraison

De manière générale, le problème de ramassage et de livraison (PDP pour *Pick-up and Delivery Problem*) est un problème dans lequel des clients peuvent demander qu'une certaine quantité de marchandises soit livrée ou collectée sur leurs sites. Plus souvent, un couple origine/destination est considéré pour chaque produit. Velasco *et al.* (2012) ont proposé un algorithme génétique bi-objectif pour résoudre le PDP appliqué à la distribution des huiles.

Le PDP devient le *Dial-a-Ride Problem* (DARP) lorsqu'il s'agit de transporter des passagers au lieu de marchandises. Cette variante inclut une qualité de service avec la minimisation des temps d'attente/retour par exemple ; généralement utilisée pour modéliser

le problème de transport des malades ou des personnes âgées depuis leurs domiciles et vers l'hôpital. Un état de l'art détaillé sur ces problèmes peut être trouvé dans l'article de Parragh *et al.* (2008) et Hartl (2008).

Dans ce qui a précédé, nous avons présenté une description rapide de quelques modèles de base qui existent pour modéliser le VRP et ses extensions relatives aux variantes étudiées dans cette thèse. Dans ce qui suit, une vue d'ensemble sur les méthodes de résolution des problèmes de tournées de véhicules est proposée. A ce niveau, notre but n'est pas de détailler le fonctionnement de ces différentes méthodes, mais plutôt d'avoir un aperçu global sur leurs principes de base afin d'identifier les méthodes que nous pourrons adapter aux problèmes nous intéressants.

2.5 Méthodes de résolution

Comme tout problème classique d'optimisation combinatoire, le problème de tournées de véhicules a été résolu par des méthodes exactes et des méthodes approchées. Dans ce qui suit, une classification générale de ces approches est donnée.

2.5.1 Méthodes exactes

La plupart des problèmes de tournées de véhicules sont formulés sous forme d'un *Programme Linéaire en Nombres Entiers* (PLNE) dont le but est de minimiser une fonction linéaire à variables entières en respectant un ensemble de contraintes qui sont également linéaires. Résoudre un PLNE nécessite le recours à des méthodes dédiées, on peut citer par exemple les algorithmes de séparation et évaluation, les méthodes de coupes, les méthodes de génération de colonnes, la programmation dynamique, la relaxation lagrangienne, ... Ces méthodes sont aussi dites "méthodes complètes".

Ces approches peuvent avoir l'avantage d'être moins coûteuses en terme de temps de calcul et d'espace mémoire lorsqu'une borne supérieure (pour le problème de minimisation comme le VRP) de **bonne qualité** lui sont préalablement fournie. Généralement, celle ci est donnée à l'aide de métaheuristiques efficaces.

2.5.1.1 Algorithmes de séparation et évaluation

Un algorithme de séparation et évaluation également appelé *Branch and Bound Algorihm*, repose sur une méthode arborescente de recherche de la solution optimale, il s'agit d'énumérer l'ensemble des solutions possibles de façon intelligente pour retrouver la meilleure.

La méthode se base d'abord sur la séparation (Branch) de l'espace des solutions en sous-ensembles de plus en plus petits. L'exploration de ces solutions utilise ensuite une évaluation optimiste (Bound) pour borner les sous-ensembles, ce qui permet de ne considérer que ceux susceptibles de contenir une solution potentiellement meilleure que la solution courante.

L'algorithme peut avoir des performances expérimentales très différentes, en fonction de la structure de données utilisée pour la liste des nœuds. La stratégie utilisée pour explorer l'arbre des solutions et le choix du prochain nœud à séparer sont également cruciaux. Plusieurs options peuvent être envisagées, à savoir :

— **Profondeur d'abord** : Cette stratégie favorise les nœuds les plus éloignés de la racine (ceux de profondeur la plus élevée). L'avantage de ce type de stratégie est qu'elle mène rapidement à une solution réalisable (borne supérieure en minimisation) tout en économisant la mémoire.

— **Largeur d'abord** : Cette stratégie avantage plutôt les nœuds les plus proches de la racine. Elle est généralement moins efficace que les autres stratégies.

— **Le meilleure d'abord** : Cette stratégie consiste à explorer à chaque itération, le nœud possédant la meilleure évaluation. L'avantage de cette stratégie est souvent d'améliorer rapidement la solution provisoire et par conséquent, d'accélérer la recherche puisqu'il sera plus facile d'ignorer des nœuds qui possèdent une mauvaise évaluation par rapport à la meilleure solution.

Des détails sur cette approche peuvent être trouvés dans la plupart des livres traitant de problèmes d'optimisation combinatoire, comme Balas & Toth (1983) ou encore Lacomme *et al.* (2003).

2.5.1.2 Algorithmes de coupes

Les méthodes de coupes sont basées sur la relaxation de certaines contraintes du problème étudié dans le but de simplifier le problème initial. Il s'agit donc d'enlever une partie des contraintes du problème initial et de résoudre le problème relaxé. La solution obtenue fournit une borne qui n'est pas toujours de bonne qualité. L'idée principale est d'ajouter ensuite un sous ensemble de contraintes (**coupes**) qui n'excluent aucune solution réalisable, et qui sont violées dans la solution optimale du problème relaxé. L'algorithme répète ce processus jusqu'à ce qu'une solution réalisable au problème initiale soit obtenue ou une solution égale à celle trouvée par une méthode approchée soit obtenue ou encore qu'aucune contrainte violée ne soit identifiée.

Cette approche est dite polyédrale car une solution d'un programme linéaire correspond à un sommet du polyèdre délimité par les contraintes et qui constitue l'enveloppe convexe

des solutions possibles. Lorsque des contraintes sont relaxées, le polyèdre résultant est plus large que le précédent et sa solution optimale peut se situer hors du polyèdre original. L'objectif de l'ajout de coupes est donc de tracer un polyèdre relaxé en excluant une partie de l'espace convexe non réalisable, de manière à ce que la solution optimale du problème relaxé corresponde à une solution réalisable, et donc optimale, pour le problème original. La performance d'une méthode de coupes dépend principalement de l'efficacité de sa procédure de génération de coupes.

Un algorithme de branchement et de coupes appelé *Branch and Cut Algorithm* peut être réalisé en combinant ces deux derniers algorithmes. Il consiste à utiliser la méthode des coupes pour calculer une bonne borne (inférieure pour un problème de minimisation, supérieure sinon) au cours de la phase d'évaluation de la méthode de séparation et évaluation précédemment présentée, afin d'améliorer la performance.

Notons qu'il existe des applications informatiques génériques (solveurs linéaires tels que Cplex, Lindo, Xpress, Gurobi, ...) qui permettent de résoudre les problèmes modélisés par des programmes mathématiques en variables binaires ou entières. En effet, ces solveurs utilisent les méthodes exactes citées précédemment, et peuvent fournir une solution optimale en un temps réduit par rapport à une énumération complète. Cependant, pour les instances de grande taille des problèmes NP-difficiles, les durées d'exécution restent encore trop importantes pour pouvoir être utilisables dans des applications réelles. Malgré les progrès réalisés (notamment en matière d'applications informatiques), le temps nécessaire pour trouver une solution d'un problème NP-difficile risque d'augmenter exponentiellement en fonction de la taille du problème, c'est pourquoi il faut généralement se tourner vers les méthodes approchées.

Des détails sur les méthodes exactes ainsi que des applications aux problèmes de tournées de véhicules peuvent être trouvés dans les livres de Wolsey (1998), Lacomme *et al.* (2003) et Wolsey & Nemhauser (2014). Plusieurs applications aux problèmes de tournées existent dans la littérature, nous pouvons citer par exemple les article de Lysgaard *et al.* (2004) et Cordeau (2006).

2.5.2 Méthodes approchées

Les méthodes approchées, aussi dites "*incomplètes*" ont comme but de générer des solutions réalisables de bonne qualité pour des problèmes NP-difficiles, mais sans garantie d'optimalité, et en temps de calcul souvent inférieur à celui des méthodes exactes. Elles sont généralement de complexité polynomiale, ce qui nous permet de traiter des instances de grande taille.

L'efficacité d'un tel algorithme est jugée selon deux approches :

— En comparant le temps de calcul ainsi que la valeur de la solution obtenue avec la meilleure solution connue ou une borne inférieure.

— En calculant le ratio dans le pire des cas entre la valeur de la solution obtenue et l'optimum.

Les sections suivantes récapitulent les méthodes approchées les plus courantes, utilisées généralement pour résoudre des problèmes de tournées de véhicules.

2.5.2.1 Heuristiques

Les heuristiques sont des méthodes souvent basées sur *le bon sens*, permettant d'obtenir des solutions réalisables. La solution construite résulte en général d'une succession de décisions élémentaires, généralement sélectionnée de manière *gloutonne*.

Dans la pratique, les heuristiques sont conçues pour être spécifiques à des problèmes particuliers. Pour les problèmes de tournées, plusieurs types d'heuristiques peuvent être rencontrés dans la littérature.

Heuristiques de construction :

— **L'heuristique du plus proche voisin** (PPV) : Génère une solution réalisable par construction, en ajoutant à chaque itération un élément à une solution partielle. C'est un processus d'insertion qui peut s'exécuter de manière séquentielle ou parallèle. Dans sa version séquentielle, une tournée est initialisée à la fois, au départ du dépôt, visitant à chaque fois le client *disponible* (non encore inséré) le plus proche. Lorsque l'insertion devient impossible sur cette tournée, le véhicule retourne au dépôt et une nouvelle tournée est ouverte. Dans la version parallèle, à chaque itération une tournée est choisie puis le plus proche voisin du dernier client déjà ajouté à cette tournée ou le dépôt dans le cas de la première insertion, est ajouté, si l'insertion est toujours possible, sinon cette tournée est fermée et une autre tournée est choisie.

Typiquement, cette heuristique trouve son intérêt dans des instances où les clients sont regroupés en cluster (très proche les uns des autres). En revanche, elle sera généralement moins bonne sur des instances où les clients sont éloignés les uns des autres (éparpillés).

— **L'heuristique de la meilleure insertion** (Solomon (1987)) : Inspirée du principe du PPV, cette heuristique procède à une insertion successive des clients dans une solution, à la position optimisant un critère choisi. Pour le VRPTW, une route est initiée avec un nœud choisi grâce à un critère d'initialisation. Dans l'article de Solomon, deux critères sont proposés : soit le nœud le plus éloigné du dépôt, soit celui ayant une date

de fin de sa fenêtre de disponibilité au plus tôt. Ensuite, au cours des itérations, ces critères sont utilisés conjointement (optimiser la somme de ces deux critères). Enfin, les insertions se répètent jusqu'à ce que l'on ne puisse plus insérer de clients dans la tournée en question, auquel cas une nouvelle tournée est démarrée. L'algorithme s'arrête lorsque tous les clients sont affectés à une tournée, ou lorsque aucune insertion n'est possible.

Heuristiques de fusion : Une autre approche envisageable pour obtenir rapidement des solutions réalisables est la fusion de tournées. La plus classique, et donnant de très bons résultats pour le VRP (Laporte *et al.* (2000)), est l'algorithme de Clarke et Wright (1964).

— **L'heuristique de Clarke et Wright** (Clarke & W. (1964)) : Cette heuristique fonctionne de manière itérative. Il s'agit de partir de la solution la plus coûteuse consistant à visiter chaque client par une tournée dédiée puis d'effectuer des fusions de tournées pour améliorer le coût global. L'heuristique débute par une solution initiale où tous les clients sont visités individuellement sur une tournée aller-retour en formant ainsi une solution en forme de marguerite. À chaque étape, les deux tournées procurant la plus grande réduction du coût de la solution courante sont fusionnées. Cette heuristique permet à la fois de réduire le nombre de véhicules utilisés et le coût total, ce qui la rend efficace sur la plupart des instances.

Heuristiques appliquées en deux phases :

— **L'heuristique *Cluster First Route Second*** : Cette heuristique commence par partitionner l'ensemble des clients en groupes, puis décide de leurs séquencements, par résolution d'un TSP sur chaque groupe. L'exemple le plus simple de cette méthode est donnée par Gillett & Miller (1974) où un algorithme dit de balayage a été proposé. Ici, les clients à servir sont classés en fonctions de leurs coordonnées polaires dans le plan euclidien. Ainsi, la construction des secteurs (groupes de clients) se fait par balayage, i.e. les nœuds sont balayés dans le sens horaire ou inversement. A chaque étape, l'affectation d'un client à une tournée donnée se fait dans la mesure où celle-ci n'implique aucune violation (par rapport aux fenêtres de temps), sinon c'est un retour au dépôt. Une fois les groupes formés, l'algorithme procède à la résolution d'un TSP sur chacun d'entre eux.

— **L'heuristique *Route First Cluster Second*** : Contrairement à la technique précédente, celle-ci consiste à déterminer d'abord le séquencement des clients puis les partitionner en tournées. Ce paradigme a été proposé par Beasley (1983) pour un problème de TSP créant un tour géant. Ce tour est ensuite partitionné en tournées

de façon optimale par la résolution du problème du plus court chemin dans un graphe auxiliaire. Dans cette publication, aucun résultat numérique n'a été fourni. Cependant, Prins (2004) a été le premier à avoir résolu le VRPTW par l'application de cette technique.

Stratégies d'exploration Afin améliorer les résultats générés par une heuristique, plusieurs stratégies peuvent être envisagées :

— Exécuter plusieurs heuristiques et garder la meilleure solution.
— Randomiser une composante de l'heuristique, l'appliquer plusieurs fois et ainsi garder la meilleure solution obtenue.
— Appliquer une recherche locale à la solution initiale obtenue par l'heuristique constructive.
— Toute combinaison de ces dernières.

Dans une méthode de recherche locale, il s'agit de regarder d'abord dans le voisinage de la solution courante, s'il existe une meilleure solution. Pour éviter l'énumération implicite de l'ensemble des solutions possibles, le voisinage d'une solution est défini comme l'ensemble des solutions qui peuvent être obtenues en appliquant de simples transformations sur la solution en question. Cette transformation est appelée " *mouvement*". La solution obtenue après application de tels mouvements améliorant est appelée un minimum local (dans le cas d'une minimisation). Il existe de nombreux mouvements de recherche locales proposés dans la littérature pour les problèmes de tournées. Les plus simples sont les mouvements de déplacement (Relocate) et de permutation (Exchange). Ils consistent à déplacer ou permuter les positions de certains nœuds et peuvent impliquer une ou plusieurs tournées à la fois. Ils seront développés plus en détail par la suite, puisque nous les utilisons dans la recherche locale proposée pour résoudre les problèmes étudiés dans cette thèse.

D'autres mouvements plus compliqués sont généralement utilisés pour les problèmes de tournées, tels que le k-opt de Lin & Kernighan (1973) et Or-opt proposé par (Or (1976). Dans le premier mouvement, il s'agit d'enlever k arêtes non consécutives d'un cycle et à les remplacer par k autres arêtes dans le but de construire une nouvelle solution réalisable. En général, on prend $k = 2$ ou $k = 3$ car l'exploration du voisinage de k-opt se fait en $O(n^k)$. Dans le même genre, Or-opt consiste à déplacer 1, 2 ou 3 nœuds consécutifs vers d'autres tournées, tout en gardant le sens de direction. D'autres mouvements d'amélioration ont été développés par la suite, par exemple, les échanges $2 - opt^*$ de Potvin & Rousseau (1995), $4 - opt^*$ de Renaud *et al.* (1996) qui consistent à supprimer 2 (resp. 4) arêtes de tournées différentes et de les remplacer par 2 (resp. 4) nouvelles arêtes pour reconstruire la solution.

2.5.2.2 Métaheuristiques

Les métaheuristiques sont des méthodes génériques pouvant s'appliquer à des problèmes de nature complètement différentes. Elles utilisent des techniques permettant d'éviter les minimums locaux, ce qui les rend plus performantes et plus efficaces que les heuristiques, même lorsque ces dernières sont couplées avec des recherches locales. Elles peuvent être classées en deux grandes familles :

— Celles fondées sur l'exploration du voisinage qui manipulent une seule solution à la fois, les méthodes les plus connues de cette catégorie sont la méthode *GRASP* "*Greedy Randomized Adaptive Search Procedure*", le *Récuit Simulé* "*Simulated Annealing*" ou encore la *Recherche Taboue* "*Tabu Search*".

— Celles basées sur l'utilisation d'un ensemble de solutions, généralement appelée population. A chaque itération, l'approche fait évoluer cette population de solutions. Les méthodes les plus connues de cette catégorie sont principalement les *Algorithmes Génétiques* "*Genetic Algorithms*" et les *Algorithmes de Colonies de Fourmis* "*Ant Colony*".

Une description plus détaillée des principales métaheuristiques et leurs différentes applications au VRP peuvent être trouvé dans le livre de Labadie *et al.* (2016). Ici nous présentons seulement le principe de base des méthodes citées ci-dessus.

GRASP (introduit par Feo & Resende (1995)). L'idée principale est de créer à chaque itération une nouvelle solution réalisable et ainsi l'améliorer par des méthodes de recherches locales. C'est donc une métaheuristique composée de deux phases : une phase de construction permettant de générer une solution initiale au moyen d'heuristiques randomisées gloutonnes et une phase d'amélioration qui consiste à appliquer des techniques de recherche locale. Ces deux étapes sont répétées un nombre de fois prédéfini et la meilleure solution obtenue au cours des itérations est retenue.

Récuit Simulé (introduit par Kirkpatrick *et al.* (1983)). Cette méthode est inspirée d'un processus utilisé en métallurgie où des cycles de refroidissement lent et de réchauffage (recuit), sont alternés dans le but de minimiser l'énergie du matériau. Cette méthode a été transposée en optimisation pour trouver les "extremas" d'une fonction. Dans un problème d'optimisation combinatoire, l'énergie correspond à la fonction objectif qui est dépendante d'un paramètre fictif, noté $Temp$. Ce dernier est un nombre réel positif assimilé à la température du matériau.

Partant d'une solution de départ produite par une heuristique, à chaque itération une transformation à effectuer sur la solution courante est tirée au sort. Si cette transformation entraîne une variation de coût $\Delta_E < 0$ (en cas de minimisation), alors cette dernière est

appliquée à la solution courante. Sinon, elle est acceptée avec une probabilité $e^{-\frac{\Delta_E}{Temp}}$ et la valeur de $Temp$ est réduite. La valeur de $Temp$ diminue légèrement au fur et à mesure des itérations et le processus se poursuit jusqu'à ce que $Temp$ atteigne une certaine valeur proche de 0, ou après un certain nombre d'itérations. La meilleure solution trouvée est alors renvoyée.

Recherche Taboue (introduites par Glover (1989)). Il s'agit d'une métaheuristique déterministe (ce qui n'est pas le cas pour les deux précédentes), puisque aucun paramètre aléatoire n'est considéré. Cette méthode consiste à balayer entièrement un voisinage prédéfini de la solution courante (fournie par une heuristique), puis effectuer la meilleure transformation possible, même si cette dernière risque de détériorer la fonction objectif. Il est interdit de faire des retours en arrière. L'historique des solutions est stocké dans une file ($FIFO$) de taille limitée, cette liste est appelée : liste taboue. Ainsi, la méthode s'arrête après un nombre prédéfini d'itérations et la meilleure solution rencontrée au cours de son exécution est restituée.

En pratique, sauvegarder les solutions entières consomme trop de mémoire. Ainsi, seules les caractéristiques des dernières solutions visitées sont conservées, par exemple, les mouvements ayant permis de passer d'une solution à l'autre ou parfois, simplement, la valeur de la fonction objectif. Une transformation ne peut alors être exécutée que si elle conduit à une solution qui ne possède aucune des caractéristiques précédemment mémorisées.

Des détails peuvent être trouvés dans un livre récent portant sur les méthodes de recherche taboue de Glover & Laguna (2013) et des applications aux problèmes de tournées sont disponibles dans les articles de Gendreau *et al.* (1994), Taillard *et al.* (1997) et Cordeau *et al.* (2002).

Algorithmes Génétiques (introduits par Goldberg & Holland (1988)). Ces méthodes sont inspirées de la génétique et de l'évolution naturelle des espèces. Un processus fait évoluer une *population* d'individus (solutions obtenues par une heuristique gloutonne randomisée par exemple) représentés sous forme de chromosomes à l'aide de phénomènes de reproduction et de mutation.

Il s'agit de partir d'une telle population et d'effectuer à chaque itération des *croisements* de deux chromosomes. Généralement, les chromosomes (individus) sont *sélectionnés* en favorisant ceux étant les plus prometteurs selon un critère de sélection dit *fitness*. Un nouvel individu est alors créé en combinant les caractéristiques des deux parents. Ce dernier est appelé *enfant*.

Pour éviter une convergence prématurée de la population, des *mutations* sont appliquées

aux enfants à titre de diversification. Enfin, deux modes de gestion de la population peuvent intervenir : incrémental ou générationnel. La première consiste à insérer les enfants directement dans la population courante en éliminant les individus les moins performants en terme de fitness, et ceci pour garder une taille de population constante. La deuxième consiste à créer à partir d'une population initiale un nombre d'enfants égal à la taille de la précédente. Ces deux populations sont ensuite fusionnées pour ne garder que les meilleurs individus dans une population de taille égale aux précédentes.

Colonies de fourmis (introduites par Colorni *et al.* (1991)). Elles sont basées sur le comportement des fourmis qui cherchent la nourriture depuis leur nid. Dès qu'elles en trouvent, elles marquent le chemin y conduisant en laissant des quantités de phéromones dépendantes de la qualité de la source trouvée. Ainsi, les autres fourmis seront averties et attirées de l'endroit de la nourriture. Avec le temps, les parcours conduisant aux meilleures sources de nourriture sont de plus en plus fréquentés et les taux d'hormones augmentent par le passage de fourmis. En revanche, les chemins peu visités auront un taux de phéromones faible.

Sur ces observations, cette métaheuristique a été proposée, assimilant les trajets réalisés par les fourmis à des tournées, et la qualité de la source à la valeur de la fonction objectif. L'initialisation du trajet peut se faire en attribuant des taux de phéromones nuls sur les arcs. Les fourmis sont représentées par des agents construisant la solution. A chaque itération, ils avancent dans le graphe en effectuant des choix soumis à des probabilités. La construction de leur chemin est biaisée en favorisant les arcs fortement marqués de phéromones. Ensuite, selon la valeur de la fonction objectif obtenue, les taux de phéromones sont mis à jour. L'algorithme s'arrête après un nombre prédéfini d'itérations.

Plusieurs d'autres métaheuristiques existent et d'autres variantes et des hybrides (nouvelle tendance) peuvent être retrouvées dans la littérature. Plus de précisions seront données dans les chapitres suivants sur les méthodes développées dans le cadre de cette thèse.

2.6 Conclusion

Dans cette thèse nous nous intéressons à l'étude des problèmes de logistique liés au domaine de la santé. Plus particulièrement, l'organisation et la planification des structures d'hospitalisation à domicile, dans le but d'améliorer la qualité des services offerts par ces établissements. Ce sont des problématiques de décision pour lesquels les méthodes

d'optimisation combinatoire et de recherche opérationnelle semblent les mieux adaptées.

Le but de ce chapitre était de présenter les problèmes d'optimisation combinatoire de base de notre étude, principalement, les problèmes de logistique du transport. Ainsi que les principales méthodes de résolution classiquement étudiées.

Avant de proposer les modèles appropriés dédiés à la résolution du problème étudié dans cette thèse, il est nécessaire, voire important de faire un tour d'horizon ciblé sur les travaux de recherche concernant les problèmes de logistique appliqués dans le secteur de la santé, notamment les problèmes de tournées de véhicules dans le secteur des soins et des services à domicile. C'est le but du chapitre suivant. Nous identifierons ainsi les principaux problèmes de tournées déjà abordés dans la littérature, qui s'apparentent à ceux examinés dans cette thèse et ayant des contraintes similaires aux nôtres.

Chapitre 3

État de l'art

3.1 Introduction

Dans ce chapitre, nous allons voir un échantillon des travaux qui ont pu être réalisés dans les différents domaines ayant trait au sujet de cette thèse. Nous verrons tout d'abord les travaux qui se rapportent principalement au contexte de notre étude : la logistique dans le domaine des soins et des services à domicile. Puis, nous détaillerons les principaux travaux existants sur des problèmes de tournées qui présentent des similitudes avec les problèmes auxquels on s'intéresse. Les approches de résolution envisagées (exactes, approchées ou encore hybrides) fruits de l'utilisation de méthodes issues de l'optimisation combinatoire et de la recherche opérationnelle dont l'efficacité a été prouvée sur les problèmes de logistique du transport en général, feront l'objet de la troisième partie de ce chapitre.

Cette étude nous permet d'avoir un aperçu général des opportunités de recherche non encore explorées. De ce fait, nous exposons et ainsi analysons les problématiques traitées dans la littérature, pour identifier les nouvelles perspectives de recherche émergentes dans ce contexte.

3.2 Problèmes de logistique dans le contexte de la santé

Nous pouvons lier l'origine des travaux de logistique dans les systèmes de santé au problème de service de transport porte à porte des personnes âgées ou handicapées proposé par (Bodin & Sexton (1986) ou Desrosiers *et al.* (1986, 1995)). Dans les pays occidentaux, la raison pour laquelle les autorités locales ont pensé à mettre en place des services Dial-a-Ride *DARP* peut être attribuée en partie au vieillissement de la population qui a engendré une augmentation du nombre de malades souffrant d'incapacité fonctionnelle, mais aussi à une tendance vers le développement des services de soins de santé ambulatoires. Plus tard,

Rousseau *et al.* (2003, 2013) ont proposé une variante particulière du (DARP) où une équipe spéciale offrant des services spécifiques est envoyée à la résidence des patients avant l'arrivée du véhicule chargé de les transporter. Ce type de malades, ont généralement besoin d'aide avant d'être transportés (être aidé pour s'habiller, se laver, ...). Ceci nécessite la coordination entre les visites des aides soignants et les transporteurs. Pour cela, les auteurs discutent l'intérêt d'introduire des contraintes de synchronisation qui permettront de gérer les priorités entre les visites. Plus récemment, Coppi *et al.* (2013) ont introduit une nouvelle variante du DARP avec fenêtres de temps, dans laquelle l'objectif est de planifier les soins à l'hôpital. Néanmoins, les patients sont transportés depuis leur domicile à l'hôpital dans des conditions non urgentes. Le critère à optimiser concerne la minimisation des coûts engendrés par le transport des patients.

Les personnes âgées et/ou handicapées ont de plus en plus de nouveaux besoins nécessitant une prise en charge particulière. En effet, le DARP a été étendu au HHCP *Home Health Care Problem* par Cheng & Rich (1998), où les personnels de soins sont affectés aux domiciles des patients afin de fournir les soins demandés sur place. Ces derniers ont développé un modèle mathématique à variables mixtes entières (MILP) et ont développé une approche heuristique pour résoudre le HHCP. A ce stade, les résultats numériques sont donnés seulement pour 4 infirmières et 10 patients. Depuis, le nombre de publications traitant ce thème ne cesse d'augmenter et les outils d'optimisation utilisés se diversifient.

Dans le contexte du (HHC) *Home Health Care*, les problèmes logistiques considérés touchent des domaines variés. Nous distinguons deux catégories, détaillées dans ce qui suit, à savoir les problèmes d'affectation et de planification ainsi que les problèmes de ramassage et de livraison.

3.2.1 Problèmes d'affectation et de planification

De façon générale, cette famille aborde les problèmes d'affectation des visites aux personnels de soin et la planification des horaires de visites. Dans ce contexte, Bertels & Stefan (2006) ont introduit le problème d'affectation des infirmières aux domiciles des patients. Chaque infirmière a des compétences particulières et une charge horaire spécifique. Ainsi, des contraintes dites dures relatives à l'exigence de qualification et de fenêtres horaires des infirmières doivent être respectées. L'objectif de leur étude consistait à minimiser le coût total de déplacement des infirmières et à maximiser la satisfaction des patients/infirmières. Pour résoudre cette variante, les auteurs ont développé une méthode de programmation par contraintes combinée avec une méthode de recherche taboue qui a produit des solutions de bonne qualité dans des temps raisonnables. Eveborn *et al.* (2006) ont formulé ce problème comme un problème de partitionnement. Le problème est assez similaire au précédant, sauf

que les auteurs ont ajouté des contraintes de périodicité car les visites sont planifiées sur un horizon multipériode prédéfini (les patients doivent suivre des soins réguliers). Les auteurs introduisent des contraintes de régularité des soignants (un patient doit être visité par le même soignant préféré à chaque période), des fenêtres de temps sur le service, ...

Les problèmes de planification des soins à domicile sont généralement modélisés comme étant une variante du VRPTW. C'est le cas de Akjiratikarl *et al.* (2007) qui ont proposé une méthode approchée à base de population (PSO) *Particle Swarm Optimization* pour résoudre le VRPTW appliqué au soin à domicile. L'objectif est la minimisation de la distance totale parcourue par chaque soignant, en respectant les capacités des véhicules et les fenêtres de temps des visites. Les auteurs ont d'abord amélioré un modèle existant, puis généraliser la méthode PSO pour résoudre une telle variante. Pareillement, Trautsamwieser & Hirsch (2011) ont proposé une formulation mathématique ainsi qu'une approche de type méta-heuristique basée sur la recherche à voisinage variable pour optimiser la planification quotidienne des services de soin à domicile en Autriche. L'objectif est de minimiser le temps relatif aux déplacements des infirmières et le niveau d'insatisfaction des patients et des infirmières. Doerner *et al.* (2007) se sont intéressés à leur tour au sujet et plus particulièrement au problème de localisation/routage dans la planification des soins à domicile pour lequel, une formulation mathématique multi-objectif a été proposée. Les visites sont évaluées selon un critère coût qui est lié à la durée de visite et à la durée de déplacement du soignant ou encore la distance moyenne qu'un soignant doit parcourir pour arriver au plus proche arrêt. Deux méta-heuristiques à base de population à savoir une méthode de type colonie de fourmis et un algorithme génétique multi-objectif ont été proposés et comparés.

Bredström & Rönnqvist (2007, 2008) ont définit une variante assez particulière du VRPTW pour les problèmes de soins à domicile. Dans ce cas, des contraintes de coordination des visites ont été considérées. Ici, un ou plusieurs soignants peuvent être demandés simultanément pour réaliser un service de soin chez un client. Ce cas peut avoir lieu lorsqu'une personne âgée par exemple a besoin de recevoir de l'aide à l'hygiène par deux personnels en raison de son poids. Chaque patient (respectivement, personnel de soins) exige une fenêtre de temps indiquant sa disponibilité à domicile (respectivement au dépôt qui correspond à l'établissement d'hospitalisation à domicile). Pour gérer le fait qu'un service soit réalisée par plusieurs soignants à la fois, des contraintes dites de synchronisation ont été considérées, permettant ainsi de coordonner les instants de visites des soignants affectés au patient. L'objectif à optimiser étant la minimisation des coûts de déplacement des soignants ainsi que la minimisation des non préférences des patients envers les soignants. Les auteurs ont proposé une méthode exacte mono-objectif de type *Branch-and-Price*. Rasmussen *et al.* (2012) ont proposé une nouvelle formulation mathématique intégrant

à la fois des contraintes de préférence des clients envers les soignants et des contraintes de coordination en étendant les synchronisations simultanées proposées précédemment à des contraintes de type précédence. Un exemple concret de ce type de précédence est le cas où il faut prélever le taux de glucose à un malade avant de lui donner à manger, sachant que ces deux tâches ne sont pas réalisées par la même personne. Ces dernières contraintes sont définies comme une coordination prioritaire des activités de soins, où plusieurs soins peuvent être demandés par un patient avec une certaine exigence sur l'ordre de réalisation (i.e. une durée spécifique doit séparer les instants de visites des soignants chez un même client, cette durée est nulle dans le cas d'une synchronisation simultanée). Pour résoudre ce problème, une méthode exacte de type *Branch-and-Price* a été également développé. Dans le même esprit, Redjem *et al.* (2012) ont proposé un programme linéaire à variables mixtes entières pour lequel un critère d'optimisation important dans l'amélioration de la qualité de services des structures de soins à domicile a été introduit. En plus de la minimisation de la distance parcourue, ils se sont intéressés à la minimisation des temps d'attente des patients, notamment ceux ayant demandés plusieurs visites synchronisées. Plus récemment, ces mêmes auteurs proposent dans (Redjem & Marcon (In press)), une heuristique à deux phases où un ensemble de routes est d'abord généré en négligeant les contraintes de coordination, puis ces dernières sont introduites pour obtenir des solutions réalisables. Dans le même genre, Afifi *et al.* (2013) ont proposé un algorithme de recuit simulé. La méta-heuristique atteint des solutions optimales connues et améliore la plupart des meilleures solutions disponibles dans la littérature. Dans ce qui précède, les patients à synchroniser ont été modélisés par duplication du nœud correspondant, ce qui augmente la taille du graphe initial. Labadie *et al.* (2014) ont élaboré une nouvelle modélisation mathématique permettant d'éviter la duplication des nœuds (plus de détails sur cette publication seront donnés dans le chapitre suivant. Une méthode approchée de type *Iterated Local Search* (ILS) a été proposée et testée sur des instances de taille réduite.

Dans le même contexte, Issaoui *et al.* (2015) ont proposé une métaheuristique itérative basée sur la recherche à voisinage variable, minimisant les coûts de déplacement et maximisant la satisfaction des patients.

3.2.2 Problèmes de ramassage et de livraison

Cette catégorie considère les problèmes pour lesquels des patients et/ou des produits sont à récupérer et/ou à déposer d'un endroit initial à une destination (Rousseau *et al.* (2003) et Coppi *et al.* (2013)). Dans ce même contexte, Bräysy *et al.* (2009) ont combiné plusieurs variantes des problèmes communs rencontrés dans le système de santé, à savoir la planification des soins à domicile, le transport des personnes âgées et la livraison des

repas à domicile. Comme tous les problèmes précédemment cités, les personnels de soin disposent également d'un nombre maximal d'heures de travail par jour et différents niveaux de compétences pour exécuter les différents services. Les personnels sont regroupés par équipe et les clients par des zones. En effet, chaque patient d'une zone donnée est visité par un membre de l'équipe correspondante. Les clients peuvent demander plusieurs visites dans un intervalle de temps donné. L'objectif est de maximiser les heures de visites réalisés et les préférences des personnels tout en tenant compte de la réglementation des pauses, charge journalière, ... Les auteurs ont partagé le problème en deux :

- Le problème de transport des personnes âgées pour lequel deux modèles ont été proposés :
  - Un modèle statique, où les patients sont transportés vers les hôpitaux, ainsi la même route est parcourue plusieurs fois par jour.
  - Un modèle dynamique, où les patients sont répartis en zone. Chaque zone contient un point de contrôle où les véhicules affectés à cette zone peuvent attendre l'heure de début d'une visite, en raison des fenêtres de temps.
- Le problème de livraison des repas dans lequel des fenêtres temps sont considérées.

Liu *et al.* (2013) se sont intéressés à un problème similaire pour lequel deux services principaux sont introduits. La livraison des médicaments et des dispositifs médicaux depuis les pharmacies aux domiciles des patients ainsi que le ramassage des échantillons biologiques et/ou les dispositifs médicaux non utilisés par les patients. Ces services sont planifiées simultanément et sont soumis à des contraintes de fenêtres de temps. Selon les origines et les destinations, les services de ramassage et de livraison des demandes sont divisés en sous-classes :

- Ramassage des dispositifs médicaux depuis les domiciles des patients vers les laboratoires (par exemple, les échantillons biologiques).
- Ramassage des dispositifs médicaux depuis les domiciles des patients vers le dépôt (par exemple, les déchets médicaux).
- Livraison des produits pharmaceutiques depuis les pharmacies vers les domiciles des patients.
- Livraison des dispositifs médicaux spécifiques depuis l'hôpital vers les domiciles des patients (par exemple, des médicaments spéciaux pour le traitement du cancer).

Pour résoudre ce problème, deux modèles mathématiques, un algorithme génétique et une méthode de recherche taboue ont été proposés. Rousseau *et al.* (2013) ont décrit une variante dynamique du DARP (en temps réel), intégrant des contraintes de synchronisation. Ces dernières apparaissent lorsque certains malades ont besoin d'aide afin de se préparer pour être transporter à l'hôpital. Dans ce cas, une équipe est envoyée à la résidence du client quelques

minutes avant que le véhicule chargé du transport ne soit arrivé. Les résultats sur les instances contenant jusqu'à 1000 requêtes montrent que les solutions sont sensibles au nombre de contraintes de synchronisation ainsi que le taux d'arrivée des demandes. Pour terminer, nous pouvons citer les travaux de Ceselli *et al.* (2014) qui se sont intéressés à la distribution des vaccins et des médicaments à travers l'utilisation de centres de distribution et des véhicules coordonnés. Ce secteur se caractérise par une grande dépendance au comportement humain qui est imprévisible, la nécessité de l'équité dans l'attribution des services de soins ainsi que l'absence de données historiques fiables. Deux façons ont été envisagées pour atteindre un patient :

— Livraison aux domiciles des patients par une flotte de véhicules hétérogènes.
— Création de centres de distribution où les citoyens vont par leur propre moyen récupérer par exemple des médicaments.

L'objectif de cette étude est de minimiser des demandes non satisfaites. Pour résoudre ce problème, un algorithme exact basé sur la génération de colonnes a été élaboré.

3.3 Classification des problèmes de tournées en HHC

Dans la section précédente, les travaux concernant les problèmes de tournées en général appliqués au secteur des soins à domicile ont été exposés. Nous nous sommes concentrés dans cette revue de la littérature, particulièrement sur 17 articles que nous allons maintenant classifier selon 3 caractéristiques principales : (1) le critère d'optimisation, (2) les contraintes, (3) les méthodes de résolution proposées. Ce récapitulatif permet d'identifier les points communs et ainsi situer la problématique étudiée dans cette thèse par rapport à la littérature. Le premier critère de comparaison est présenté dans le tableau 3.1, proposant une dénomination par objectif à optimiser. Différentes fonctions-objectifs ont été traitées dans les articles examinés et résumées dans le tableau 3.2, où la colonne 1 correspond à la référence du document, tandis que les colonnes 2-7 correspondent à chacun des objectifs énumérés dans tableau 3.1. Lorsqu'un article considère un des objectifs proposés en colonnes, un symbole ($\times$) est affiché dans la cellule correspondante.

Concernant cette première classification, on peut constater de façon générale que les temps, les coûts ainsi que les préférences sont les objectifs les plus courants dans la littérature. Ainsi, tous les auteurs adoptent une résolution mono-objectif (par agrégation de plusieurs critères d'optimisation) sauf Doerner *et al.* (2007) qui proposent une vraie approche multi-critère basée sur le principe de dominance.

Le deuxième critère de comparaison est présenté dans le tableau 3.3, proposant une énumération des contraintes les plus courantes considérées en HHCP. Plusieurs contraintes

TABLE 3.1 – Critères d'optimisation communs

Abbr.	Description
T	Temps (Déplacement, Attente, Heures Supplémentaires, etc.)
C	Coûts (Déplacement, Attente, Affectation, etc.)
D	Distance parcourue
PP	Préférence des patients
PS	Préférence des soignants
SR	Services réalisés

ont été traitées dans les articles examinés, et elles sont résumées dans le tableau 3.4, où la colonne 1 correspond à la référence du document, et les colonnes 2-7 correspondent à chacune des contraintes énumérées dans le tableau 3.3.

D'après cette revue de littérature, les contraintes temporelles ne sont pas liées seulement aux fenêtres de temps associées aux patients et/ou aux véhicules, mais aussi à l'interdépendance des tournées réalisées par les soignants. Cette interdépendance a été définie par Drexl (2012) comme étant la synchronisation des opérations de tournées de véhicules en général. Plus précisément, dans le cadre des soins à domicile, Labadie *et al.* (2014) ont distingué deux types de contraintes de synchronisation :

— **Synchronisation simultanée** : quand un patient peut demander la présence de plusieurs soignants à la fois. C'est le cas lorsqu'il s'agit de charges lourdes du service demandé (par exemple le bain). On parle alors de synchronisation simultanée dans le sens où les personnels doivent réaliser simultanément un service donnée.

— **Synchronisation prioritaire ou Précédence** : un patient peut également demander des visites ordonnées. C'est le cas lorsqu'un patient a besoin par exemple d'être préparé pour une activité de soin ou avant d'être transporté à l'hôpital par exemple. Il s'agit d'une synchronisation prioritaire pour laquelle les activités des soignants devraient être ordonnées (enchaînées).

La majorité des auteurs ont généralement porté un intérêt à la qualité de service par l'utilisation des contraintes temporelles telles que le respect des fenêtres de temps. Cependant, certaines caractéristiques des services qui sont communes dans les problèmes réels (telle que la synchronisation prioritaire) n'ont pas reçu un niveau d'attention élevée. A notre connaissance, seuls les travaux de Rasmussen *et al.* (2012) et ceux de Liu *et al.* (2013) ont combiné les deux types de contraintes de synchronisation dans le même modèle. A ce stade, un patient demandant deux services synchronisés est représenté par deux sommets.

Le troisième critère de comparaison considéré est donné dans le tableau 3.5 présentant

TABLE 3.2 – Critère d'optimisation de la littérature

	T	*C*	*D*	*PP*	*PS*	*SR*
Cheng & Rich (1998)	×					
Bertels & Stefan (2006)		×		×	×	
Eveborn *et al.* (2006)		×				
Akjiratikarl *et al.* (2007)			×			
Bredström & Rönnqvist (2007, 2008)			×	×		
Doerner *et al.* (2007)	×		×			
Bräysy *et al.* (2009)					×	×
Trautsamwieser & Hirsch (2011)	×			×	×	
Rasmussen *et al.* (2012)		×			×	×
Redjem *et al.* (2012)	×					
Rousseau *et al.* (2013)		×				×
Afifi *et al.* (2013)		×				
Liu *et al.* (2013)		×				
Coppi *et al.* (2013)		×				
Ceselli *et al.* (2014)		×				×
Labadie *et al.* (2014)	×					
Redjem & Marcon (In press)	×					
Issaoui *et al.* (2015)			×	×		

une classification des approches de résolution utilisées par les auteurs dans les documents examinés. Ce dernier critère est de trois types : résolution du modèle mathématique à l'aide d'un solveur linéaire (colonne 2), de méthodes exactes (colonne 3) et approchées (colonne 4). On peut constater que la plupart des contributions se trouvent dans la catégorie des méta-heuristiques avec parfois une résolution exacte du modèle par un solveur linéaire.

Dans le cadre de cette thèse, nous nous intéressons particulièrement à coordonner les interventions des soignants chez les patients demandant plusieurs services (contraintes type QS), avec des contraintes temporelles (FT, Simu, Prio). Cela regroupe l'ensemble des types de contraintes classiquement rencontrées en HAD et n'a jamais été étudié auparavant. De nombreux chercheurs ont introduit des contraintes de synchronisation et de précédence afin d'améliorer la qualité de services. Jusque là, nous avons présenté seulement les travaux de recherche rencontrés dans le secteur de santé. Cependant, les contraintes temporelles et/ou de coordination existent dans différentes thématiques liées au transport (routier, aérien, transport de marchandises, etc.), c'est pourquoi nous exposons dans la section suivante les problèmes de tournées en général pour lesquelles des contraintes de synchronisation ont été considérées.

TABLE 3.3 – Contraintes communes

Abbr.	Description
FT	Fenêtres de temps
$Simu$	Synchronisation simultanée
$Prio$	Synchronisation prioritaire
$Pref$	Préférences des patients et/ou soignants
QS	Qualification des soignants

3.4 Problèmes de tournées avec synchronisation

Il existe une multitude de travaux de tournées intégrant des contraintes de synchronisation et/ou de précédence. Nous allons ciblé notre recherche sur ceux qui nous semblent les plus proches du problème auquel on s'intéresse. Par exemple, dans le domaine aérien, Ioachim *et al.* (1999) ont décrit un problème de routage et d'affectation des avions avec contraintes de synchronisation simultanée. Ces dernières exigent un même temps de départ pour tout vol régulier devant partir chaque jour pendant une semaine. C'est une variante périodique du problème de tournées de véhicules. Les critères d'optimisation concernent la minimisation du coût total associé aux vols. Pour résoudre le problème, les auteurs ont proposé une formulation du problème de multi-flots non-miscibles et ont développé une approche de type *Branch-and-Price*.

Dans le contexte de ramassage et de livraison, Del Pia & Flippi (2006) ont également introduit des contraintes de synchronisation dans des problèmes de collecte de déchets ménagers. Dans cette étude, deux types camions sont utilisés : *petits* (*satellites*) et qui sont autorisés à voyager dans tout type de rue et un deuxième type de camions, appelés *grands* (*compacteurs*) qui sont interdits à la circulation dans des rues étroites en raison de leur largeur. Toutefois, les transbordements sont autorisés à partir des satellites aux compacteurs utilisés comme des dépôts mobiles, lorsque le dépôt se trouve loin de la ville. Ainsi, la synchronisation se produit lorsqu'un satellite doit vider son contenu dans le compacteur, ces derniers doivent arriver simultanément. L'objectif considéré est de minimiser la durée totale des trajets. Pour résoudre le problème, les auteurs ont proposé une heuristique basée sur une procédure de recherche locale à descente variable "*Variable Neighborhood Descent*" (VND). Le critère d'optimisation considéré est la minimisation des coûts de déplacement des camions. Dans le même ordre d'idée, Drexl (2012) a introduit des contraintes de synchronisation dans un problème de *Vehicle Routing Problem With Trailer and Transhipement* (VRPTT), appliqué à la collecte de lait cru dans les fermes. Dans ce problème, deux types de véhicules ont été utilisé : des véhicules autonomes appelés aussi camions capables de se déplacer seuls et

TABLE 3.4 – Contraintes considérées

	FT	*Simu*	*Prio*	*Pref*	*QS*
Cheng & Rich (1998)	×				×
Bertels & Stefan (2006)				×	×
Eveborn *et al.* (2006)	×	×		×	×
Akjiratikarl *et al.* (2007)	×				
Bredström & Rönnqvist (2007, 2008)	×	×			
Doerner *et al.* (2007)	×				
Bräysy *et al.* (2009)	×	×			×
Trautsamwieser & Hirsch (2011)	×				
Rasmussen *et al.* (2012)	×		×		
Redjem *et al.* (2012)	×	×			
Rousseau *et al.* (2013)	×		×		×
Afifi *et al.* (2013)	×	×			
Liu *et al.* (2013)	×	×	×		
Coppi *et al.* (2013)	×				
Ceselli *et al.* (2014)	×				
Labadie *et al.* (2014)	×	×			×
Redjem & Marcon (In press)	×	×			×
Issaoui *et al.* (2015)	×			×	×

des véhicules non-autonomes appelés remorques qui doivent être déplacé à l'aide d'un camion. Ainsi, deux types de clients sont considérés : des clients "camion" qui peuvent être visités seulement par un camion sans remorque et des clients "remorque" qui peuvent être visités par des camions avec ou sans remorque. Dans ce problème, la remorque peut être considérée comme un véhicule de support utilisé comme un dépôt mobile où le camion est autorisé à transférer sa charge complètement ou partiellement. Une remorque peut être affectée à n'importe quel camion ainsi tous les véhicules sont stationnés au dépôt. L'objectif est de déterminer les itinéraires des camions et des remorques afin de réduire le coût total tout en satisfaisant un ensemble de contraintes (les demandes des clients, les capacités de chargement, les contraintes d'accessibilité, les fenêtres de temps et la synchronisation entre les routes des camions et celles des remorques). L'auteur propose une modélisation mathématique basée sur une représentation graphique pour le VRPTT. Toutefois, aucune méthode de résolution n'a été abordée dans cet article. Dans ce même contexte, Masson *et al.* (2013) ont étudié le problème de ramassage et livraison avec points de transferts et ont proposé des tests de faisabilité en temps constant pour les contraintes de type précédence.

Une autre application introduisant les contraintes de synchronisation a été étudiée par El Hachemi *et al.* (2013), elle est posée dans le domaine de l'industrie forestière. Ce problème combine plusieurs aspects tels que le ramassage et la livraison, les actions d'inventaire, etc.

TABLE 3.5 – Méthodes de résolution

	Solveurs Linéaires	Exactes	Approchées
Cheng & Rich (1998)	×		×
Bertels & Stefan (2006)			×
Eveborn *et al.* (2006)			×
Akjiratikarl *et al.* (2007)			×
Bredström & Rönnqvist (2007, 2008)	×	×	
Doerner *et al.* (2007)	×		×
Bräysy *et al.* (2009)	×		
Trautsamwieser & Hirsch (2011)	×		×
Rasmussen *et al.* (2012)		×	
Redjem *et al.* (2012)	×		×
Rousseau *et al.* (2013)		×	×
Afifi *et al.* (2013)	×		×
Liu *et al.* (2013)	×		×
Coppi *et al.* (2013)		×	
Ceselli *et al.* (2014)		×	
Labadie *et al.* (2014)	×		×
Redjem & Marcon (In press)		×	×
Issaoui *et al.* (2015)			×

Les principales contributions de cet article sont la synchronisation des camions chargés de transporter le bois coupé avec les chargeurs de bois disponibles pour préparer l'expédition. Pour résoudre le problème, les auteurs ont développé une approche de décomposition à deux phases. Dans la première phase, ils utilisent un solveur linéaire pour résoudre le modèle au niveau tactique déterminant la destination des charges depuis la zone forestière jusqu'au moulin à scie. Dans la deuxième phase, ils utilisent deux méthodes différentes pour générer les routes et établir un planning de transport journalier : la première méthode consiste à utiliser une approche basée sur une procédure de recherche locale tandis que la seconde est une approche hybride impliquant un modèle basé sur la programmation par contraintes et une méthode basée sur la recherche locale.

Plusieurs recherches sur les problèmes de tournées sur arcs introduisant les contraintes de synchronisation peuvent être trouvées dans la littérature. Dans ce contexte, Amaya *et al.* (2007, 2010) ont décrit la synchronisation dans les opérations de marquage des routes. Deux types de véhicules ont été utilisé : un véhicule chargé de peindre et un véhicule-citerne servant de dépôt mobile, qui retourne au dépôt pour effectuer le rechargement en cas de manque. L'objectif était de déterminer les itinéraires des véhicules pour réduire au minimum le coût total des déplacements. Dans leur publication de 2007, les auteurs ont proposé un modèle mathématique à variables entières et ont développé une méthode de coupes appliquée à des

cas impliquant de 20 à 70 nœuds et de 50 à 595 arcs. En 2010, les auteurs ont amélioré leur modèle précédent et ont développé une méthode heuristique basée sur le principe "*Route-First-Cluster-Second*".

Salazar-Aguilar *et al.* (2013) ont également introduit des contraintes de synchronisation dans un problème de tournées sur arc modélisant les opérations de déneigement au Canada. Dans ce problème, une flotte de véhicules spécialisées dans le déneigement est disponible au dépôt initial, pour traiter un ensemble de routes prédéfinies. Chaque segment de route a une ou deux directions et pour chaque direction, le nombre de voies est entre 1 et 3. Les contraintes de synchronisation imposent que les segments des rues avec plus d'une voie soient dégagés simultanément par différents véhicules synchronisés. L'objectif est de déterminer l'ensemble de routes minimisant la durée de la plus longue route, tout en veillant à réaliser toutes les tâches de déneigement. Pour résoudre ce problème, les auteurs ont proposé un modèle de programmation non linéaire mixte en nombres entiers et une heuristique basée sur la métaheuristique *Adaptative Large Neighbourhood Search* (ALNS). L'heuristique construit un ensemble initial de routes possibles et des techniques destruction/réparation sont utilisées pour améliorer cette solution. Les contraintes de synchronisation sont prises en compte dans la procédure de construction. Les tests numériques ont été réalisées à la fois sur de grandes instances générées aléatoirement et également sur des instances réelles.

3.5 Conclusion

Dans ce chapitre, nous avons présenté un état de l'art général sur les problèmes de tournées appliqués au secteur de soins à domicile. Dans ce contexte, nous avons identifié 3 critères de classification des articles traités à savoir, l'objectif à optimiser, les contraintes et les méthodes de résolution. De façon générale, les coûts de déplacement et les préférences sont les critères les plus étudiés dans la littérature. En effet, ces derniers restent les plus importants dans le souci d'optimiser les frais des structures de soins à domicile et dans le but d'améliorer leur qualité de service. L'examen de la littérature montre également que les contraintes temporelles (fenêtre de temps, synchronisation, ...) sont très souvent considérées. Nous pouvons même constaté que ces dernières couvrent un large champs d'applications réelles, particulièrement en logistique de transport. Par conséquent, l'intérêt de les étudier est majeur. Enfin, les méthodes de type approchée semblent également les plus utilisées pour résoudre ce type de problèmes.

L'état de l'art présenté ici montre l'existence d'un grand nombre de travaux dans le domaine de la santé introduisant une variété de contraintes et de critères d'optimisation intéressants. Cependant, une variante combinant les deux types de contraintes de

synchronisation (simultanée et précédence) ainsi que la qualification des soignants, et dont l'objectif est de minimiser les coûts des tournées et de maximiser les préférences des clients n'a jamais été traitée, d'où l'originalité de notre étude.

De façon globale, les informations rassemblées et synthétisées dans ce chapitre et dans le chapitre précédent nous permettent d'avoir une vue globale sur le système étudié, à savoir l'optimisation de la logistique des établissements de services et de soins à domicile. Elles nous serviront de bases dans les chapitres suivants pour proposer des approches de résolution dédiées aux problèmes étudiés dans cette thèse.

Chapitre 4

Modèles et outils dédiés à la résolution du VRPTW-SP

4.1 Introduction

Avant de présenter les techniques de résolution que nous avons proposées pour le problème de tournées de véhicules avec fenêtres de temps et des contraintes de synchronisation et de précédence (VRPTW-SP), il est important de bien exposer le problème étudié et les hypothèses prises en compte. Pour cela, la section 4.2 explique l'intérêt et les raisons qui ont motivé notre étude et pour lesquelles nous étudions ce problème. Ensuite, une description formelle est donnée dans la section 4.3, permettant ainsi l'écriture du modèle mathématique. Les sections d'après sont consacrées aux outils dédiés à la résolution du problème de tournées avec synchronisation et fenêtres de temps. Il s'agit des heuristiques de construction (section 4.4) et de recherches locales (section 4.5). Les résultats obtenus par ces différentes techniques de résolution sont exposés dans la section 4.6. Enfin, le chapitre se conclut par la section 4.7.

4.2 Motivation

En raison de sa complexité, le système de santé cache un grand nombre et une grande diversité de processus de décisions interpellant les chercheurs du domaine de la Recherche Opérationnelle et d'Aide à la Décision. De nombreuses améliorations sont possibles, que ce soit au regard de l'objectif d'efficacité (évolution de la qualité des soins) ou de l'efficience (garantir la mission au moindre coût).

En particulier, les coûts liés aux HAD sont en grande partie dus au transport. Le but principal de ce travail de recherche est d'établir la planification des visites des personnels de soins en optimisant les coûts relatifs à leurs déplacements tout en garantissant une qualité de

service et un suivi médical équivalent à ceux donnés en établissements de santé traditionnels (hôpitaux). Cependant, plusieurs caractéristiques ont été identifiées dans les chapitres précédents, notamment, la présence des fenêtres de disponibilité des patients/soignants, la nécessité de recevoir un ou plusieurs soins spécifiques et qui peuvent être liés par des contraintes temporelles, la qualification du personnel (certains intervenants ne peuvent pas réaliser certaines catégories de soins).

Le nombre de travaux existant sur le sujet et intégrant ces différents aspects est peu élevé, en particulier pour les cas combinant les deux types de contraintes de synchronisation (simultanée et prioritaire). Néanmoins, comme il a été montré dans le chapitre précédent, certains auteurs se sont intéressés à la résolution d'un problème similaire intégrant ces deux catégories de contraintes, en proposant une modélisation mathématique en dupliquant les nœuds correspondant aux patients ayant demandés des services à synchroniser. Cette modélisation est intéressante mais reste peu envisageable d'un point de vue complexité d'autant plus que le graphe de départ se voit augmenter de taille. Imaginons que nous ayons n patients à visiter et que chacun demande m services. Le nombre de nœud dans le graphe résultant sera de $n \times m$. Labadie *et al.* (2014) ont proposé une modélisation permettant de garder la taille à n nœuds. Néanmoins, ils ne se sont intéressés qu'aux contraintes de type synchronisation simultanée. Ainsi, l'objectif principal de leur travaux consistait à minimiser les coûts de déplacement des soignants. A ce stade, les résultats numériques ont été donnés pour des instances de tailles relativement petites.

Dans ce chapitre, nous proposons d'abord de gérer les deux types de contraintes de synchronisation au sein d'un modèle unique, sans duplication des nœuds et d'introduire un critère d'optimisation important dans l'amélioration de la qualité de service des établissements de soins à domicile, et qui consiste à maximiser la satisfaction des patients. Nous proposerons également de nouveaux tests de faisabilité permettant d'intégrer les contraintes temporelles dans les mouvements de recherche locale. Ces techniques sont une généralisation des tests de faisabilité proposés par Kindervater & Savelsbergh (1997), adaptés efficacement au problème étudié dans cette thèse et qui restent applicables en $O(1)$.

De façon générale, la résolution du problème de VRP avec synchronisation représente donc un intérêt non négligeable. La section suivante donne les notations et une formulation mathématique du problème étudié.

4.3 Description du problème

4.3.1 Énumération des données et des contraintes

Le problème auquel on s'intéresse consiste à planifier les visites de personnels soignants aux domiciles de patients disponibles dans des fenêtres de temps spécifiques, et répartis sur une zone géographique donnée. En associant les patients aux clients et les soignants aux véhicules, le problème introduit est défini comme une variante du problème de tournées de véhicules où les soins demandés doivent être réalisés à des horaires spécifiques nécessitant l'intervention d'un ou plusieurs soignants. Cette variante est appelée VRPTW-SP pour *Vehicle Routing Problem With Time Windows, Synchronization and Precedence Constraints* où certains patients demandent plusieurs services simultanément (Synchronisation simultanée) ou dans un ordre de priorité (Synchronisation prioritaire ou Précédence). Formellement, le VRPTW-SP est défini sur un graphe orienté $G = (V, E)$ où $V = N \cup D$ est l'ensemble des nœuds, où $N = \{1, .., n\}$ est l'ensemble des clients et $D = \{0, n+1\}$ représentent respectivement le dépôt initial (d'où les véhicules commencent leurs tournées) et le dépôt final (que les véhicules rejoignent à la fin de leurs tournées), $E = \{(i, j) : i, j \in V,\ i \neq j\}$ est l'ensemble des arcs reliant les clients entre eux, le dépôt initial et les clients et vise versa (les clients et le dépôt final). Chaque arc $e = (i, j) \in E$ est pondéré par une valeur positive T_e correspondant à la durée relative au déplacement sur cet arc, ainsi qu'un coût de déplacement C_e ($C_{0\ n+1} = +\infty$ et $T_{0\ n+1} = 0$). $S = \{1, ..s\}$ est ensemble de services (soins) disponibles. Un patient peut demander un ou plusieurs services différents. Pour cela, nous définissons S_i comme étant l'ensemble de services demandés par le client i tel que $S_i = \{s \in S : m_{is} = 1\}$ où m_{is} est un paramètre donné égale à 1 si le client i demande le service s, 0 sinon. Chaque client $i \in N$ possède également une fenêtre de temps $[a_i, b_i]$ représentant sa disponibilité à domicile et une durée de visite D_{is} pour chaque service s demandé par i ($D_{0s} = D_{n+1\ s} = 0$).

Une flotte de véhicules limitée est disponible dans le dépôt initial (0), chacun correspond à un soignant donné. Soit K cet ensemble de véhicules, offrant chacun un service spécifique. Chaque véhicule $k \in K$ est associé à une fenêtre de temps $[\alpha_k, \beta_k]$, où α_k représente l'heure à laquelle le véhicule k peut quitter le dépôt initial "0" et β_k correspond à l'heure à laquelle le véhicule k doit retourner au dépôt final "n+1". Un coefficient de non préférence $Pref_{ik}$ est attribué par chaque client i pour évaluer le soignant k qualifié pour réaliser le service demandé. Dans cette étude, nous supposons que chaque véhicule (soignant) $k \in K$ est capable d'effectuer un seul type de service s, spécifié par le paramètre o_{ks} qui est égal à 1 lorsque le véhicule k offre le service s, ainsi nous définissions par K_s l'ensemble de véhicules offrant le service s comme suit : $K_s = \{k \in K : o_{ks} = 1\}$. Pour simplifier, on note $k \in K_s$ si

$o_{ks} = 1$ et $s \in S_i$ si $m_{is} = 1$.

Le problème consiste à déterminer l'ensemble des tournées réalisées par l'ensemble de véhicules, depuis le dépôt initial en servant chacun un sous ensemble de clients demandant le service correspondant à sa qualification, avant de se rendre au dépôt final. La route de chaque véhicule ne doit pas excéder la durée imposée par sa fenêtre de disponibilité, ainsi les fenêtres relatives aux clients doivent être respectées.

Comme nous l'avons déjà expliqué, la particularité de ce problème réside dans le fait qu'un client peut demander plusieurs services simultanément ou dans un ordre donné. Pour cela, on définit gap_{isr} comme étant la durée qui sépare le début des services s et r chez le client i. En effet, cet écart est strictement positif dans le cas d'une synchronisation prioritaire tel que $gap_{isr} = gap_{irs}$. En revanche, gap_{isr} est nul quand s et r doivent être réalisés en même temps et ceci dans le cas d'une synchronisation simultanée. Un tel client sera donc appelé client synchronisé.

Les contraintes de ce problème sont les suivantes :

— Chaque véhicule k doit commencer sa tournée depuis le dépôt initial "0" après α_k et doit impérativement retourner au dépôt final "n+1" au plus tard à l'instant β_k.

— Les demandes de tous les clients doivent être réalisées.

— Le début de service chez tout client i doit avoir lieu par le véhicule qualifié et dans $[a_i, b_i]$.

— L'écart souhaité entre les services à synchroniser doit être respecté.

Le but est de répondre à la question suivante : à quelle moment les clients sont servis ? et par quels véhicules ? de sorte à minimiser les coûts de déplacement et à maximiser la satisfaction des clients. Ce dernier se traduit par la minimisation des non préférences des clients envers les véhicules.

Avant d'exposer le modèle que nous avons proposé pour le VRPTW-SP, les contraintes de synchronisation telles que modélisées dans la littérature (travaux les plus proches à notre variante), sont présentées :

Notations :

— $Ecart_{ij}$ est le temps minimum séparant les débuts de service chez les patients i et j.

— y_i est une variable binaire égale à 1 si le patient $i \in N$ n'est pas visité, 0 sinon.

Bredström & Rönnqvist (2008) et Rasmussen *et al.* (2012) ont proposé un programme linéaire basé sur un graphe augmenté dans lequel une visite synchronisée correspond à un nœud par soignant demandé. C'est à dire, pour chaque patient i nécessitant n_i visites, $(n_i - 1)$ nœuds sont ajoutés dans le graphe, induisant un nombre total de nœuds égal à $(\sum_{i=1}^{n} n_i)$.

En effet, Bredström & Rönnqvist (2008) ont désigné l'ensemble des paires de visites à

synchroniser par $P^{sync} \subset N \times N$ pour lequel un client j est virtuel dans une paire $(i, j) \in P^{sync}$ lorsque i et j correspondent au même patient. Ainsi, les contraintes de synchronisation simultanées sont définies comme suit :

$$\forall (i,j) \in P^{sync} \quad \sum_{k \in K} t_{ik} = \sum_{k \in K} t_{jk} \tag{4.1}$$

Rasmussen *et al.* (2012) ont défini l'ensemble des couples de synchronisation $P \subset N \times N$. Ainsi, les contraintes temporelles (à la fois synchronisation simultanée et prioritaire) sont définies comme suit :

$$\forall (i,j) \in P \quad a_i.y_i + \sum_{k \in K} t_{ik} + Ecart_{ij} \leq \sum_{k \in K} t_{jk} + b_i.y_i$$

où ($Ecart_{ij} = Ecart_{ji}$) et lorsque $Ecart_{ij} = 0$ on est dans un cas d'une synchronisation simultanée.

En revanche, Labadie *et al.* (2014) ont donné une nouvelle formulation pour éviter la duplication des nœuds (relatifs aux patients) synchronisés, en considérant un ensemble de services demandés pour chaque patient. Ainsi, la synchronisation simultanée a été formulée comme ceci :

$$\forall i \in N, \quad \forall p_1, p_2 = 1, ..|S| \quad v_{i,p_1}.v_{i,p_2} \sum_{k \in Kp_1} t_{ik} = v_{i,p_1}.v_{i,p_2} \sum_{k \in Kp_2} t_{ik}$$

où v_{i,p_1} et v_{i,p_2} sont des données, qui prennent la valeur 1 si le patient i demande les services p_1 (resp. p_2), 0 sinon.

4.3.2 Modèle mathématique

On peut formuler le VRPTW-SP sous forme d'un programme linéaire à variables mixtes entières (PLME), impliquant deux types de variables : des variables de routage binaires notées x_{ijk} et des variables d'ordonnancement notées t_{ik}.

Variables de routage :

— $x_{ijk} = 1$ si le véhicule $k \in K$ traverse l'arc $(i, j) \in E$, 0 sinon.

Variables d'ordonnancement :

— $t_{ik} \geq 0$ désigne l'heure à laquelle le véhicule k commence la réalisation du service demandé par le client i.

Le problème peut s'écrire sous sa formulation mathématique comme suit :

$$\min \sum_{e\in E}\sum_{k\in K} C_e.x_{ek} + \sum_{i\in N}\sum_{j\in V\setminus\{0\}}\sum_{k\in K} Pref_{ik}.x_{ijk} \tag{4.2}$$

Sous les contraintes :

$$\sum_{j\in N} x_{0jk} = 1 \qquad \forall k \in K \tag{4.3}$$

$$\sum_{i\in N} x_{in+1k} = 1 \qquad \forall k \in K \tag{4.4}$$

$$\sum_{i\in V\setminus\{n+1\}} x_{ihk} = \sum_{j\in V\setminus\{0\}} x_{hjk} \qquad \forall h \in N, \forall k \in K \tag{4.5}$$

$$\sum_{j\in V\setminus\{0\}}\sum_{k\in K_s} x_{ijk} = m_{is} \qquad \forall i \in N, \forall s \in S \tag{4.6}$$

$$\forall i,j \in V, \forall s \in S_i \cap S_j, \forall k \in K_s$$

$$t_{ik} + (T_{ij} + D_{is}).x_{ijk} \leq t_{jk} + b_i.(1 - x_{ijk}) \tag{4.7}$$

$$a_i.\sum_{j\in N} x_{ijk} \leq t_{ik} \leq b_i.\sum_{j\in N} x_{ijk} \qquad \forall i \in N, \forall s \in S_i, \forall k \in K_s \tag{4.8}$$

$$\alpha_k \leq t_{0k} \leq \beta_k \qquad \forall k \in K \tag{4.9}$$

$$\alpha_k \leq t_{n+1k} \leq \beta_k \qquad \forall k \in K \tag{4.10}$$

$$\sum_{k\in K_s} t_{ik} - \sum_{k\in K_r} t_{ik} \leq gap_{isr} \qquad \forall i \in N, \forall s,r \in S_i : r \neq s \tag{4.11}$$

$$\sum_{k\in K_s} t_{ik} - \sum_{k\in K_r} t_{ik} \geq -gap_{isr} \qquad \forall i \in N, \forall s,r \in S_i : r \neq s \tag{4.12}$$

$$\sum_{k\in K_s} t_{ik} - \sum_{k\in K_r} t_{ik} \geq gap_{isr} - M.z_i \qquad \forall i \in N, \forall s,r \in S_i : r \neq s \tag{4.13}$$

$$\sum_{k\in K_s} t_{ik} - \sum_{k\in K_r} t_{ik} \leq -gap_{isr} + M.(1 - z_i) \qquad \forall i \in N, \forall s,r \in S_i : r \neq s \tag{4.14}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i,j \in V, \forall k \in K \tag{4.15}$$

$$t_{ik} \geq 0 \qquad \forall i \in V, \forall k \in K \tag{4.16}$$

$$z_i \in \{0,1\} \qquad \forall i \in N \tag{4.17}$$

Sous cette formulation, la fonction objectif (4.2) reprend les coûts définis auparavant. Les contraintes (4.3 - 4.6) sont dites de routage, où les contraintes (4.3) (resp. (4.4)) spécifient que chaque véhicule quitte (resp. retourne au) le dépôt exactement une fois. Les contraintes de conservation de flot sont données en (4.5). Les contraintes (4.6) assurent que les demandes de tous les clients soient réalisées. Les contraintes (4.7) sont dites d'ordonnancement et permettent la cohérence des instants de visites. Les contraintes (4.8 - 4.10) sont des contraintes de ponctualité permettant de vérifier que les fenêtres de temps des clients

(resp. véhicules) sont respectées. Les contraintes (4.11 - 4.14) sont des contraintes de synchronisation assurant la coordination des instants de visites des différents véhicules devant servir un client i demandant plus d'un service. Cette dernière est la normalisation de la contrainte non linéaire suivante : $\forall i \in N, \forall s, r \in S_i : r \neq s \quad |\sum_{k \in K_s} t_{ik} - \sum_{k \in K_r} t_{ik}| = gap_{isr}$, par l'introduction de la variable binaire z_i qui est égale à 0 si $\sum_{k \in K_s} t_{ik} \geq \sum_{k \in K_r} t_{ik}$ et une grande valeur M que nous initialisons à $(max_{k \in K} \beta_k + max_{i \in N, s, r \in S : s \neq r} gap_{isr} + 1)$. Enfin, les contraintes (4.15 - 4.17) fixent la nature des variables de décision.

4.4 Heuristiques constructives

Après avoir présenté en détail le VRPTW-SP à l'aide d'une formulation mathématique, cette section est consacrée à la première méthode de résolution développée pour résoudre des problèmes de taille réelle. Il s'agit des heuristiques constructives dans le but de fournir des solutions de bonne qualité dans un temps de calcul raisonnable. La construction d'une telle solution résulte en général d'une suite de décisions élémentaires.

Dans cette thèse, deux heuristiques gloutonnes sont proposées, suivant chacune une stratégie d'insertion particulière. La première est une approche séquentielle (section 4.4.3) consistant à construire les tournées les unes après les autres. La seconde est une approche parallèle (section 4.4.4) consistant à construire les tournées simultanément. Dans les deux versions, les tournées sont élaborées sur le principe de l'algorithme du plus proche voisin (PPV) et un ensemble de règles de priorité est appliqué pour choisir le meilleur client à insérer dans une tournée.

Dans les deux cas, le choix du véhicule à utiliser est déterministe, contrairement à la sélection du client à insérer, qui adopte une démarche randomisée. La randomisation est basée sur une liste de clients "candidats" pouvant être insérés dans la tournée en cours de construction, et permet la génération de plusieurs solutions en cas d'appels successifs.

Avant de détailler les deux stratégies, nous présentons les parties communes, à savoir la construction de la liste des clients candidats ainsi que les critères de sélection utilisés pour choisir le client à insérer.

4.4.1 Construction de la liste des candidats

Listons d'abord les notations utilisées dans la suite de ce chapitre. Soit D_{is} la durée du service s demandé par le client i, T_{ij} la durée de déplacement entre les clients i et j, a_i (resp. b_i) correspond à l'heure de début (resp. de fin) de la fenêtre de temps du client i et β_k correspond à l'heure de fin de disponibilité du véhicule k, $time_{is}$ correspond à l'heure de

début de service s chez le client i. Dans ce qui suit, notons par d le dépôt initial et par f le dépôt final.

Pour construire la liste des clients L_i candidats à l'insertion après le client i (à la première itération i est le dépôt initial 0, et nous avons $D_{is} = 0$) dans la tournée du véhicule k, on calcule d'abord pour tous les clients j non encore affectés à une tournée et demandant le service s : une date au plus tôt $Edat_{ij}$ qui correspond à l'heure à laquelle le client j peut être atteint depuis i, une date au plus tard $Ldat_{ij}$ qui correspond à l'heure maximale à laquelle le client j peut être inséré après i, ainsi qu'une borne inférieure BI_{js} et une borne supérieure BS_{js}, calculée comme suit :

— $Edat_{ij} = time_{is} + D_{is} + T_{ij}$.
— $Ldat_{ij} = \beta_k - T_{jf} - D_{js}$.
— $BI_{js} = max\{a_j, Edat_{ij}\}$.
— $BS_{js} = min\{b_j, Ldat_{ij}\}$.

Un client j est ainsi considéré comme étant atteignable depuis i et intègre la liste L_i, si ce dernier vérifie les trois conditions suivantes :

— $Edat_{ij} \leq b_j$.
— $Ldat_{ij} \geq Edat_{ij}$.
— $BI_{js} \leq time_{js} \leq BS_{js}$.

4.4.2 Critères de tri de la liste de sélection :

Trois stratégies sont envisagées pour trier la liste L_i :

1. C_1 : Ordre croissant des BI_{js}.

2. C_2 : Ordre croissant des BS_{js}.

3. C_3 : Ordre croissant des $BS_{js}.(BS_{js} - BI_{js})$.

Un seul critère de tri est considéré à la fois, nous obtenons ainsi 3 versions différentes associées à chaque stratégie. Les résultats obtenus seront présentés dans la section 4.6. Les noms de ces versions auront le format suivant : $A-B$ où $A \in \{(P)$ pour une stratégie parallèle et (S) pour une stratégie séquentielle$\}$ et $B \in \{C_1, C_2, C_3\}$ correspondant au critère de tri sélectionné.

L'algorithme 4.1 décrit la construction de la liste L_i :

Algorithme 4.1 ConstructionCandidats *(i, k, s) : CL*

1: $CltRest$: Ensemble des clients non encore insérés
2: $L_i = \emptyset$
3: **Pour** chaque $j \in CltRest$ **Faire**
4: $Edat_{ij} = time_{is} + D_{is} + T_{ij}$
5: $Ldat_{ij} = \beta_k - T_{jf} - D_{js}$
6: $BI_{js} = max\{a_j, Edat_{ij}\}$
7: $BS_{js} = min\{b_j, Ldat_{ij}\}$
8: **Si** $(Edat_{ij} \leq b_j)$ *et* $(Ldat_{ij} \geq Edat_{ij})$ *et* $(BI_{js} \leq time_{js} \leq BS_{js})$ **Alors**
9: $L_i = L_i \cup \{j\}$
10: $CltRest = CltRest \setminus \{j\}$
11: **Fin Si**
12: **Fin Pour**
13: **Retourne** L_i

4.4.3 Construction séquentielle :

La structure générale de cette heuristique est donnée dans l'algorithme (4.2). Dans cette première approche, les tournées sont construites entièrement les unes par les autres pour chaque véhicule disponible au dépôt initial, tant qu'aucune des conditions d'arrêts suivantes n'est vérifiée :

— Tous les clients sont insérés ;
— Toutes les tournées des véhicules sont saturées : en raison des contraintes temporelles.

4.4.4 Construction parallèle :

Contrairement à la version précédente, en démarche parallèle, une tournée est initialisée pour chaque véhicule depuis le dépôt initial. Ensuite, la liste des clients atteignables depuis le dernier client inséré (ou le dépôt initial s'il s'agit d'une première insertion) dans la tournée courante est construite comme introduit précédemment (dans le cas d'une construction séquentielle). L'insertion poursuit de façon **parallèle**, tant qu'aucun des critères d'arrêt cités précédemment n'est vérifié. L'idée est de construire les tournées simultanément pour éviter de privilégier un véhicule par rapport à un autre. La structure générale de cette heuristique est donnée dans l'algorithme (4.3).

Algorithme 4.2 Approche séquentielle : HCS

$Tour$: Tournées à construire.
$NbClients$: Nombre de clients
$NbVehicules$: Nombre de véhicules
$NbServices$: Nombre de services
$NbClientsRes = NbClients$ *//Nombre de clients non encore insérés*
$NbVehiclesRes = NbVehicules$ *//Nombre de véhicules disponibles*
Tant que $(NbVehiculesRes \neq 0)$ *et* $(NbClientsRes \neq 0)$ **Faire**
  Choisir un véhicule k disponible
  $Tour_k \leftarrow \emptyset$
  Marquer k comme non disponible
  $NbVehiculesRes = NbVehiculesRes - 1$
  $tdepart_k = \alpha_k$
  $tretour_k = 0$
  $pos_k = d$ *// le dépôt initial*
  $inserer(d, Tour_k)$
  $s =$ service offert par k
  Tant que $(CL \neq 0)$ *et* $(NbClientsRes \neq 0)$ **Faire**
    $CL = ConstructionCandidats(pos_k, k, s)$
    $i = Random(CL, n_p)$
    //Choix aléatoire parmi les n_p premiers candidats de CL triée selon l'un des critères
    Si $(i \neq 0)$ **Alors**
      $time_{is} = BI_{is}$
      $pos_k = i$
      Si (i est synchronisé) **Alors**
        $\forall (s' \in S_i) \quad time_{is'} = time_{is} + gap_{iss'}$
      Fin Si
      $inserer(i, Tour_k)$
      $NbClientsRes = NbClientsRes - 1$
    Fin Si
  Fin Tant que
  $tretour_k = time_{pos_k s} + D_{pos_k s} + T_{pos_k f}$
  $inserer(f, Tour_k)$
Fin Tant que

Algorithme 4.3 Approche parallèle : HCP

$Tour$: Tournées à construire.
$NbClients$: Nombre de clients
$NbVehicules$: Nombre de véhicules
$NbServices$: Nombre de services
$NbClientsRes = NbClients$ *//Nombre de clients non encore insérés*
$NbVehiclesRes = NbVehicules$ *//Nombre de véhicules disponibles*
$SatureT = faux$
Tant que $(NbClientsRes \neq 0)$ *ou* $(SatureT \neq vrai)$ **Faire**
  Pour $k = 1$ to NbVehicules **Faire**
    Si (k est disponible) **Alors**
      $Tour_k \leftarrow \emptyset$
      $tdepart_k = \alpha_k$
      $pos_k = d$ *// le dépôt initial*
      $Tour_k \leftarrow Tour_k \cup \{d\}$
      Marquer k comme non disponible
      $NbVehiculesRes = NbVehiculesRes - 1$
    Fin Si
    $s =$ service offert par k
    $CL = ConstructionCandidats(pos_k, k, s)$
    $i = Random(CL, n_p)$
    //Choix aléatoire parmi les n_p premiers candidats de CL triée selon l'un des critères
    Si $(i \neq 0)$ **Alors**
      $time_{is} = BI_{is}$
      $pos_k = i$
      $Tour_k \leftarrow Tour_k \cup \{i\}$
      $NbClientsRes = NbClientsRes - 1$
      Si (i est synchronisé) **Alors**
        $\forall (s' \in S_i) \quad time_{is'} = time_{is} + gap_{iss'}$
      Fin Si
      $tretour_k = time_{pos_k s} + D_{pos_k s} + T_{pos_k f}$
    Sinon
      $SatureT = vrai$
      $Tour_k \leftarrow Tour_k \cup \{f\}$
    Fin Si
  Fin Pour
Fin Tant que

4.4.5 Procédure de réparation

Les expérimentations préliminaires des heuristiques de construction (4.2 et 4.3) font apparaître souvent des demandes non satisfaites (clients non insérés), le critère d'arrêt n'est pas seulement (NbClientsRes = 0) à cause des fenêtres de temps. Cependant, une procédure de réparation peut être appliquée sur la solution obtenue, afin de compléter l'insertion des clients restants.

Soit R une solution obtenue par l'une des heuristiques proposées, NL représente la liste des clients n'appartenant pas à R. Soit $i \in NL$ un client demandant le service s. Le processus de réparation consiste dans un premier temps, à parcourir les tournées construites par les véhicules offrants le service s et à insérer ces clients dans la première position possible. Afin de maximiser les chances d'insertion de chacun, ces clients sont considérés dans l'ordre croissant de l'heure de début de la fenêtre de temps.

Cependant, si la solution n'est toujours pas réalisable à la fin de cette étape, une deuxième démarche de réparation est alors entamée sur une solution voisine. Cette dernière est obtenue en appliquant une perturbation à la solution courante. Il s'agit de supprimer un certain nombre c_{max} de clients choisis aléatoirement et de les insérer dans d'autres positions possibles afin d'obtenir une solution "voisine" à la précédente. Ainsi, nous retentons la première partie de réparation sur la solution obtenue. Ici, un client est supprimé à la fois, i.e. une fois un client est choisi aléatoirement, ce dernier est supprimé de sa position actuelle et réinsérée dans une autre position si c'est possible. Sinon, il est considéré comme non inséré.

Après des tests préliminaires, il s'est avéré que le processus composé de "construction + réparation" n'aboutit pas parfois à une solution réalisable. C'est pourquoi, dans les heuristiques nous proposons de répéter le processus composé de (construction + réparation), jusqu'à ce qu'une solution faisable soit construite. Par conséquent, afin de produire des solutions différentes à chaque lancement, ces heuristiques ont été randomisées. La randomisation est basée sur la liste de clients candidats et elle consiste à choisir aléatoirement un client parmi les n_p premiers candidats de la liste (CL) triée selon l'un des critères cités avant. Cette liste est donc restreinte et sera appelée dans la suite (RCL) pour *Restricted Condidate List*.

4.5 Recherche locale

4.5.1 Principe

Pour les problèmes de tournées de véhicules, les procédures de recherche locale ou les stratégies d'exploration consistent à améliorer les tournées générées à l'aide d'une heuristique

constructive, en effectuant des échanges d'arcs et/ou de nœuds (appelés mouvements), afin d'obtenir des tournées de meilleure qualité. La solution obtenue après application d'une telle transformation à une solution *Sol* est dite voisine de cette dernière. Les mouvements les plus connus pour les problèmes de tournées concernent les déplacements et/ou les permutations de nœuds, les échanges de séquences, etc.

4.5.2 Mouvements utilisés dans la recherche locale

Pour améliorer la qualité des solutions obtenues par les heuristiques proposées pour résoudre le VRPTW-SP, cinq mouvements sont développés ici. Ces derniers impliquent un ou plusieurs clients et/ou tournées. Bien entendu, chaque mouvement n'est accepté que lorsqu'il conduit à une solution réalisable et de moindre coût.

- $S\_Relocate$: déplace un client de sa position courante à une autre position dans la même tournée.
- $S\_Exchange$: permute les positions de deux clients appartenant à la même tournée.
- $M\_Relocate$: déplace un client de sa position courante à une autre position d'une autre tournée.
- $M\_Exchange$: permute les positions de deux clients appartenant respectivement à deux tournées différentes.
- $2Opt^*$: deux arcs de deux tournées différentes sont supprimés et ainsi remplacés par deux nouveaux arcs (le cas mono-tournée n'est pas considéré car après des tests préliminaires, s'est montré non prometteur pour le VRPTW-SP).

Chaque type de mouvement k défini un voisinage N_k. Les figures de (4.2) à (4.5) donnent des exemples d'utilisation des mouvements proposés. Ces mouvements de recherche locale peuvent suivre différentes stratégies d'exploration. Par exemple, lorsqu'un mouvement donné est examiné, on peut choisir de garder le premier améliorant la solution courante ("*Première amélioration*") ou bien de générer tout le voisinage et garder le meilleur voisin rencontré ("*Meilleure amélioration*"). La question qui se pose est de trouver un bon compromis entre la taille du voisinage et la qualité des solutions rencontrées.

Les mouvements de recherche locale que nous proposons, proviennent des problèmes de tournées classiques, leur adaptation au VRPTW-SP nécessite cependant une évaluation en terme de coût et de faisabilité qui peut se révéler gourmande en terme de temps. De plus, et en particulier pour ce problème en question, certains mouvements ne sont pas prometteurs et conduisent à des solutions non réalisables et devraient être rejetés rapidement à cause des contraintes de synchronisation et des fenêtres de temps. Ainsi, des tests préliminaires ont montré que la recherche du meilleur voisinage demande beaucoup plus de temps, sans une réelle amélioration globale de la solution par rapport à une stratégie retenant la première

amélioration. Cette dernière est donc choisie.

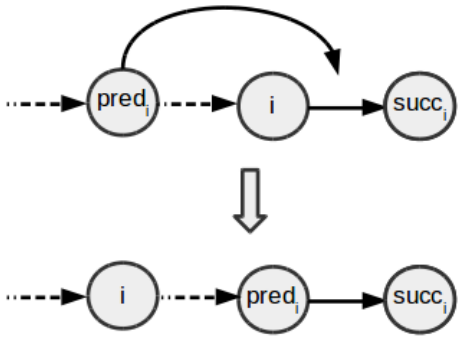

FIGURE 4.1 – Exemple du mouvement $S\_Relocate$

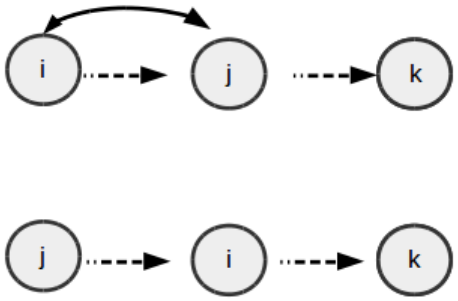

FIGURE 4.2 – Exemple du mouvement $S\_Exchange$

4.5.3 Stratégies de descente

Une méthode de recherche locale consiste à explorer le voisinage de la solution courante pour trouver une meilleure solution. Ce processus est répété au sein d'une stratégie de descente spécifique. Nous proposons dans cette sous-section deux stratégies de descente (stratégie de changement de mouvements).

4.5.3.1 Descente à voisinage aléatoire :

Itérativement, l'espace de solutions est parcouru en visitant les différents voisinages prédéfinis dans un ordre aléatoire, à la recherche d'une solution de meilleure qualité. Avant que l'exploration ne soit lancée, une liste de Θ_{max} mouvements est générée aléatoirement (Θ_{max} correspond au nombre de mouvements proposés, à savoir 5). Cet ordre est conservé tout au long de la procédure. Chaque voisinage est entièrement exploré l'un après l'autre et la recherche locale s'arrête lorsqu'aucune amélioration n'est possible (voir Algorithme 4.4).

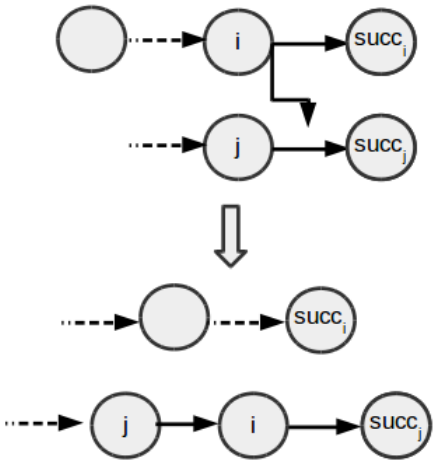

FIGURE 4.3 – Exemple du mouvement $M\_Relocate$

Algorithme 4.4 Descente à voisinage aléatoire : DVA

Sol : Solution initiale
$List$: liste des index de voisinages générée de façon aléatoire
no_Improve = faux
Tant que $(no\_Improve \neq vrai)$ **Faire**
  no_Improve = vrai
  $i = 1$
  Tant que $(i < \Theta_{max})$ **Faire**
    $\delta = List[i]$
    $Sol^* \leftarrow N_\delta(Sol)$
    Si $(F(Sol^*) < F(Sol))$ **Alors**
      $Sol \leftarrow S^*$
      $no\_Improve \leftarrow faux$
    Sinon
      $i \leftarrow i + 1$
    Fin Si
  Fin Tant que
Fin Tant que

4.5.3.2 Descente à voisinage variable :

Un mouvement donné est répété autant de fois que possible (tant qu'une amélioration est obtenue), le mouvement suivant est introduit dans le cas contraire. Il est supposé qu'une amélioration pendant l'exploration de N_δ (désignant le voisinage obtenu en appliquant

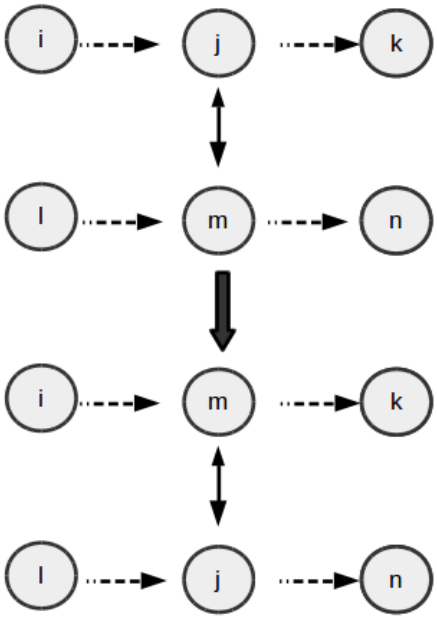

FIGURE 4.4 – Exemple du mouvement $M\_Exchange$

le mouvement δ) peut créer de nouveaux changements dans $N_1, ..., N_{\delta-1}$, c'est pour quoi après toute amélioration, i est réinitialisé à 1. Cette stratégie trouve son intérêt lorsque les mouvements les moins gourmands en temps (car moins complexes) sont introduits en premier. En effet, dans cette approche, les mouvements sont considérés dans l'ordre donné en section 4.5.2 (voir Algorithme 4.5).

Algorithme 4.5 Descente à voisinage variable : VND

S : Solution initiale
$\delta \leftarrow 1$
Tant que $(\delta < \Theta_{max})$ **Faire**
  $Sol^* \leftarrow N_\theta(Sol)$
  Si $(F(Sol^*) < F(Sol))$ **Alors**
    $Sol \leftarrow Sol^*$
    $\delta \leftarrow 1$
  Sinon
    $\delta \leftarrow \delta + 1$
  Fin Si
Fin Tant que

L'évaluation de chacun de ces mouvements doit être précédée par un test de faisabilité des fenêtres horaires, des contraintes de synchronisation, ... En effet l'insertion d'un client

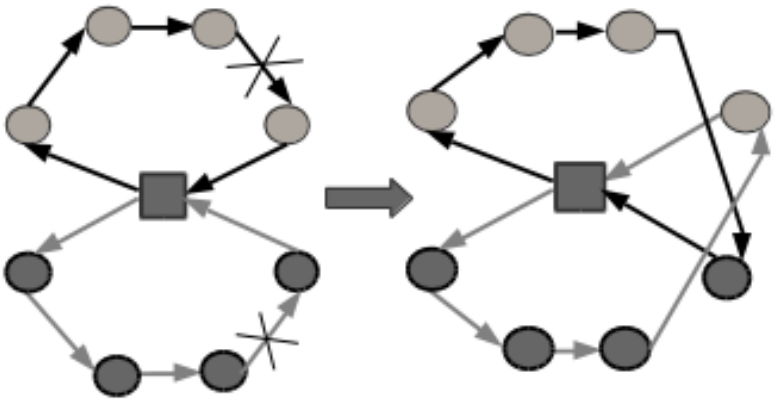

FIGURE 4.5 – Exemple du mouvement $2Opt^*$

dans une tournée à n clients entraîne le décalage des horaires de début de service des clients suivants, ce qui génère une complexité en $O(n)$ dans le pire des cas, si aucune technique n'est utilisée pour vérifier efficacement la faisabilité de l'insertion. Nous proposons dans ce qui suit une généralisation des techniques décrites par Kindervater & Savelsbergh (1997), qui ont été efficacement appliquées au problèmes de tournées avec fenêtres de temps.

4.5.4 Test de faisabilité

Des tests de faisabilité inspirés des techniques proposées par Kindervater & Savelsbergh (1997) sont réalisés, et ceci dans le but de vérifier particulièrement le respect des contraintes temporelles dans la nouvelle solution, avant que le mouvement testé ne soit appliqué. Pour le VRPTW, ces tests sont basés sur l'idée suivante : pour qu'un client puisse être inséré dans une tournée donnée, il faut vérifier que les nouveaux temps de début de service des clients qui suivent soient compatibles avec les fenêtres horaires correspondantes. Nous ajoutons aussi une compatibilité avec les contraintes de synchronisation.

La particularité réside dans l'existence d'une interdépendance entre les tournées visitant des clients synchronisés, augmentant ainsi la complexité du problème. En effet, quand un client synchronisé nécessite deux services, cela signifie que le même client appartient à deux tournées différentes. Par conséquent, l'insertion d'un client dans une tournée visitant au moins un client synchronisé, peut impacter toutes les autres tournées où ce même client est visité pour un autre service qui lui même peut impacter d'autres clients synchronisés visités après lui dans la tournée correspondante. Par conséquent, la possibilité de retarder ou avancer un clients synchronisé dépend des autres visites du même client.

Notons que cette particularité a été soulignée par Afifi *et al.* (2013) pour un cas de synchronisation simultanée. Nous proposons d'étendre cette contribution au cas étudié ici,

en évitant la duplication des nœuds et en considérant la synchronisation de type précédence. Pour cela, nous avons mis en place des techniques complétant celles de Kindervater & Savelsbergh (1997) en gardant des informations supplémentaires en mémoire, de sorte que la synchronisation/précédence soient prises en compte, tout en gardant une complexité en O(1) pour réaliser les tests de faisabilité.

Considérons une route $Tour_k = (d, R_1, ..., R_i, ..., R_r, f)$ visitant r clients effectués par le véhicule v et soit $Ttôt_{R_is}$ (resp. $Ttard_{R_is}$) l'heure de départ au plut tôt (resp. au plus tard) d'un client R_i demandant le service s, a_{R_i} (resp. b_{R_i}) représente l'heure de début (resp. fin) de la fenêtre horaire du client i. Dans ce qui suit, $T_{R_iR_j}$ correspond à la durée séparant le client R_i du client R_j et D_{R_is} correspond à la durée de réalisation du service s demandé par le client R_i, où R_{i+1} (resp. R_{i-1}) désigne le successeur (resp. le prédécesseur) du client R_i dans cette route.

Si le véhicule débute le service du client R_i à l'heure $times^{new}_{R_is}$ au lieu de $times_{R_is}$, avec $times^{new}_{R_is} \geq times_{R_is}$ et $times^{new}_{R_is} \leq b_{R_i}$, alors l'heure de départ de chaque client R_j (avec $i < j \leq r$) peut être retardée de $MT_{R_js} = min(MU_{R_j}, MT_{R_{j+1}s})$, avec $MU_{R_is} = Ttard_{R_is} - Ttôt_{R_is}$ et $MT_{fs} = MU_{fs}$. Il est clair que si $MT_{R_js} \geq times^{new}_{R_j} - times_{R_js}$ la fenêtre horaire relative au client R_j sera violée.

A chaque insertion, ces données sont calculées :

— $Ttôt_{R_is} = min(a_{R_i}, Ttôt_{R_{i-1}s} + D_{R_{i-1}s} + T_{R_{i-1}R_i})$.

— $Ttard_{R_is} = max(b_{R_i}, Ttard_{R_{i+1}s} - T_{R_i\ R_{i+1}} - D_{R_is})$.

Comme nous l'avons évoqué précédemment, il existe une dépendance entre les tournées. Soit R_i un client synchronisé demandant deux services différents s et s'. R_i appartient à deux tournées, soient k et k' deux véhicules offrant respectivement le service s et s'. Le calcul des marges minimales pour les clients synchronisés se fait comme suit :

— $MU_{R_is} = min(MU_{R_is}, MU_{R_is'})$.

— $MT_{R_is} = min(MU_{R_is}, MT_{R_{i+1}s}, MT_{R_is'})$.

Ensuite, un client synchronisé demandant un service s peut être inséré dans la tournée du véhicule k, sachant que ce client est déjà visité par le véhicule k' pour un autre service s', si ces marges ne sont pas excédées suite à une telle insertion. Notons, que les tests classiques pour le VRPTW sont également maintenus pour tous les clients.

Lorsque une insertion possible est effectuée, la mise à jour est propagée sur les tournées impliquées. Cette mise à jour peut boucler infiniment si les insertions croisées ne sont pas interdites. Donc, pour éviter l'insertion de clients synchronisés dans des positions incompatibles, quelques tests supplémentaires sont nécessaires. Un tel cas apparaître lorsqu'un client synchronisé est candidat à une insertion dans une tournée contenant au moins un client synchronisé (voir les Figures 4.6). Dans ce cas, les conditions de faisabilité doivent

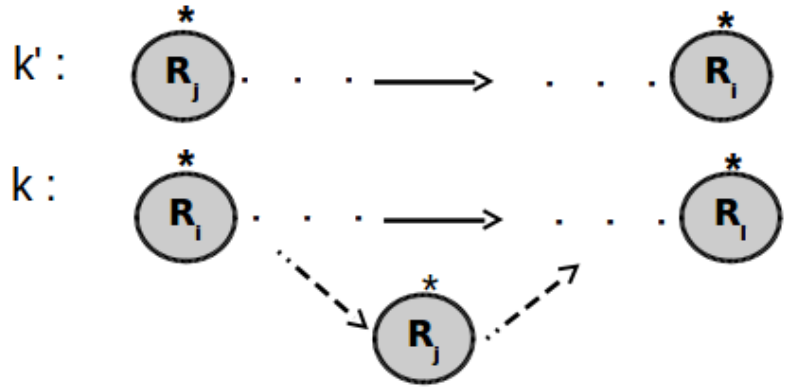

FIGURE 4.6 – Exemple d'une insertion entre deux clients synchronisés

être respectées. Les nœuds mentionnés par $*$ signifie qu'il s'agit d'un client synchronisé.

Notons qu'ici, on parle d'un client synchronisé lorsque ce dernier demande plusieurs visites et qu'au moins deux visites sont déjà affectées, incluant celle que l'on souhaite insérer. Ainsi, de façon générale, deux cas peuvent avoir lieu :

— Soit une insertion avant le premier client synchronisé de la tournée (voir figure 4.7). Il s'agit d'insérer le client R_j dans la tournée du véhicule k à une position avant le premier client synchronisé (R_i) de cette tournée.

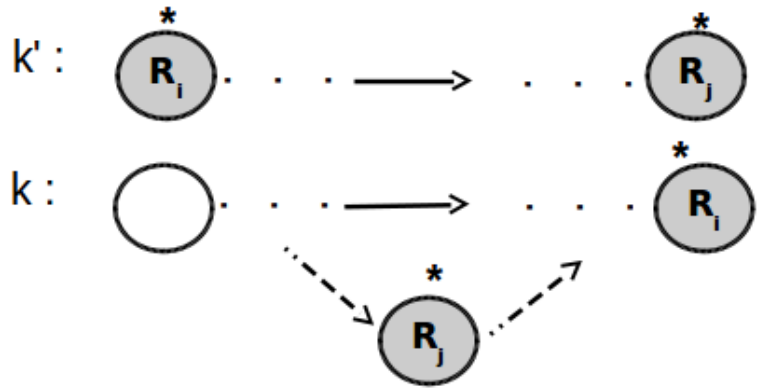

FIGURE 4.7 – Exemple d'une insertion avant le premier client synchronisé

— Soit une insertion à une position après au moins un client synchronisé (voir figure 4.6, 4.8). Il s'agit d'insérer le client R_j dans la tournée du véhicule k à une position située après au moins un client synchronisé (soit R_i dans la figure 4.8) ou bien à une position entre deux clients synchronisés (R_i et R_l dans la figure 4.6).

Comme illustré dans la figure 4.6, soit R_j un client synchronisé candidat pour l'insertion dans la tournée du véhicule k, son insertion dans cette tournée avant R_i devrait être interdite.

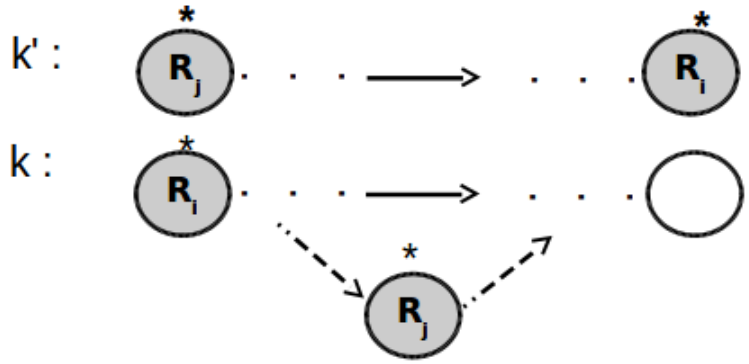

FIGURE 4.8 – Exemple d'une insertion après un client synchronisé

Pour empêcher une elle insertion, nous introduisons les informations suivantes :

Pour chaque client R_i visité dans une tournée du véhicule k : PPS_{R_ik} désigne le premier prédécesseur synchronisé de R_i dans la tournée de k, PSS_{R_ik} désigne le premier successeur synchronisé de R_i dans la tournée de k et PS_k correspond au premier client synchronisé inséré dans la tournée du véhicule k. Enfin, Pos_{ik} correspond à la position du client R_i dans la tournée effectuée par le véhicule k.

Rappelons qu'ici nous ne considérons un client comme étant synchronisé, que lorsque ce dernier est inséré dans aux moins deux tournées.

Ainsi, insérer le client R_j après un certain prédécesseur qu'on appelle ici *predj* dans la tournée du véhicule k, implique les calculs suivants et les tests correspondants pour chaque cas de figure :

— En fonction de la position où l'on souhaite insérer, un des cas suivants est nécessairement retrouvé, ainsi les valeurs calculées ici sont indispensables pour la vérification des tests :

1. Soit *predj* est le dépôt initial ou $PPS_{predj\ k} = 0$ (i.e. qu'aucun client synchronisé n'a été visité avant cette position, voir la figure 4.7) :
   — Dans ce cas, une position temporaire appelée *pos* est initialisée à 0 ;
2. *predj* peut être le premier client synchronisé de la tournée du véhicule k (i.e. $predj = PS_k$), dans ce cas :
   — $PPS_{predj\ k} = PS_k$;
   — Et une position temporaire appelée *pos* est initialisée à $Pos_{predj\ k}$;
3. Enfin, le dernier cas concerne l'insertion d'un client à une position où au moins un client synchronisé précède *predj* (i.e. $PPS_{predj\ k} \neq 0$, voir la figure 4.8) :
   — Dans ce cas, *pos* est initialisée à $Pos_{PPS_{predj\ k}\ k}$;

Ensuite, la position potentielle du client à insérer est calculée dans cet ordre :

— $Pos^{new}_{R_j\ k} = pos + 1$;
— $Pos^{new}_{R_j\ k'} = max(Pos^{new}_{R_j\ k}; Pos_{R_j\ k'})$;
— $Pos^{new}_{R_j\ k} = Pos^{new}_{R_j\ k'}$;

Enfin, une fois le pré-calcul terminé, les tests de faisabilité sont alors entamés. En effet, l'insertion du client R_j n'est acceptée que si les conditions suivantes sont vérifiées :

1. La première condition doit être vérifiée quelque soit la position d'insertion dans la tournée k : soit k' un autre véhicule servant également le client R_j, alors la nouvelle position de R_j dans la tournée de k' ne doit pas excéder la position de son premier successeur synchronisé. Cette condition est traduite ainsi :

$$Pos^{new}_{R_j\ k'} \leq Pos_{PSS_{R_j\ k'}\ k'}; \tag{4.18}$$

2. La deuxième condition à vérifier, implique la tournée du véhicule k et est considérée si au moins un client synchronisé figure dans cette tournée. Ainsi, deux cas sont possibles :
   (a) Soit R_j est candidat pour une insertion avant le premier client synchronisé. C'est le cas présenté dans la figure 4.7. Cependant, l'insertion de R_j est autorisée, si en plus de la condition 4.18, la condition suivante est vérifiée :

$$Pos^{new}_{R_j k} \leq Pos_{PS_k\ k}; \tag{4.19}$$

   (b) Soit R_j est candidat à une insertion après au moins un client synchronisé. C'est le cas représenté par les figures 4.6 et 4.8. Ainsi, l'insertion de R_j est autorisée, si en plus de la condition 4.18, la condition suivante est vérifiée :

$$Pos^{new}_{R_j\ k} \leq Pos_{PSS_{pred_j\ k}\ k}; \tag{4.20}$$

Dans cette thèse, les instances traitées impliquent uniquement deux services. Cependant, dans le cas où plus de deux visites sont demandées par un client R_j appartenant à la tournée servie par le véhicule k et impliquée par ces tests, k' est choisi comme suit :

$$k' \leftarrow \min_{v \in K : v \neq k \ \ \& \ R_j \in Tour_v \ \& \ PSS_{R_j\ v} \neq 0} \{PSS_{R_j\ v}\}. \tag{4.21}$$

Ces techniques gèrent efficacement et en temps constant (O(1)), l'empêchement des insertions contradictoires qui peuvent se produire en raison de contraintes de synchronisation, même lorsque plusieurs services sont demandés.

Concrètement, toutes les informations introduites dans cette sous-section sont calculées en même temps qu'une solution est construite par une heuristique par exemple. Les tests de faisabilité sont appelés à chaque insertion potentielle et sont utilisés dans les procédures de recherches locales et la réparation. Notons qu'en phase de construction les heures de visite des clients synchronisés sont fixées.

4.6 Évaluations numériques

4.6.1 Implémentation et instances

Le modèle linéaire de la section 4.3.2 est traduit en langage de modélisation Opl Studio et résolu en utilisant le solveur linéaire Cplex 12.5. Les heuristiques et recherches locales proposées dans ce chapitre ont été codées en C et exécutées sur un ordinateur avec un processeur Core i5 de 8 Go de RAM fonctionnant sous Linux.

L'évaluation numérique a été réalisée sur un ensemble de 37 instances initialement conçues par Bredström & Rönnqvist (2007) pour les problèmes de tournées de soins à domicile avec synchronisation simultanée. Ces instances comprennent la plupart des caractéristiques particulières du VRPTW-SP : fenêtres de temps, synchronisation, préférences, ... mais sont étendues pour répondre aux spécifications de notre problème notamment les types de services (demandés par les clients et/ou offerts par les véhicules) qui sont générés aléatoirement. Ainsi, dans notre cas, deux types de synchronisation sont étudiés (simultanée et précédence) et sont donc générés de manière aléatoire.

Le benchmark est partitionné en 3 groupes de tailles différentes, distingués par le nombre de clients et de véhicules utilisés. Le premier groupe (G_1) contient 14 instances à 18 clients et 4 véhicules, le deuxième groupe (G_2) contient 13 instances à 45 clients et 10 véhicules et le troisième groupe (G_3) contient 10 instances à 73 clients et 16 véhicules. Dans chaque groupe, le nombre de clients à synchroniser varie de deux à cinq (ce qui représente environ 10% du nombre total de clients, ce qui est cohérent avec le taux de synchronisation rencontrée généralement dans les structures de soins à domicile). La taille des fenêtres de temps relatives aux clients varient également : Elles peuvent être réduites (r), moyennes (m) ou larges (l). Chaque instance est nommée comme suit : $n - m - t - nb$ où n indique le nombre de clients ($n \in \{18, 45, 73\}$), m le nombre de véhicules disponibles ($m \in \{4, 10, 16\}$), t désigne la largeur de la fenêtre de temps relative aux clients ($t \in \{r, m, l\}$) et le dernier indice nb indique le nombre de clients à synchroniser ($nb \in \{2, .., 5\}$).

Les résultats obtenus par Cplex sont d'abord présentés, ensuite sont exposés ceux obtenus par les deux heuristiques (chacune est testée pour les 3 critères de sélection de la liste des candidats), améliorés par les recherches locales. Les solutions des heuristiques + recherches locales sont comparées entre elles pour distinguer la meilleure méthode qui sera ensuite sélectionnée dans la conception des métaheuristiques présentées dans les chapitres suivants.

Nous montrons dans ce qui suit les résultats de 12 évaluations notées ainsi : $A - B - C$ où $A \in \{P, S\}$ ("P" pour l'heuristique parallèle et "S" pour l'heuristique séquentielle), B correspond au type de descente réaliser ($B \in \{DVA, VND\}$ ("DVA" pour une descente à voisinage aléatoire et "VND" pour une descente à voisinage variable). Enfin, C ($C \in$

$\{1, 2, 3\}$) correspondant aux critères de sélection cités dans la section 4.4.2.

Les tableaux récapitulatifs sont exposés dans cette section et les tableaux détaillés sont donnés à la fin de ce chapitre. Le but de ces tests est de mesurer le gain apporté par chaque recherche locale sur les heuristiques proposées, pour choisir la meilleure. Pour évaluer la stabilité des méthodes testées, il est intéressant d'effectuer plusieurs lancements et de rapporter la meilleure valeur obtenue. Pour cela, toutes les évaluations ont été lancées sur 10 exécutions indépendantes.

Les paramètres des heuristiques utilisés pour obtenir les résultats présentés sont les suivants : $n_p = 3$ pour la taille de la liste des clients candidats et $c_{max} = 2$ pour le nombre de clients à supprimer dans la phase de réparation. Ils ont été choisis après des tests préliminaires, afin de réaliser un bon compromis entre le temps de calcul et la qualité des solutions. Enfin, le temps de résolution du solveur linéaire Cplex a été fixé à 3600 secondes.

4.6.2 Résultats et interprétations

Le tableau 4.1 donne les résultats détaillés des solutions obtenues par Cplex sur les 37 instances. La colonne 1 correspond à l'instance, la colonne UB correspond à la borne supérieure (la valeur de la fonction objectif) donnée par le solveur Cplex (le symbole $*$ indique que la solution correspondante est optimale), $Depl$ et $Pref$ correspondent à la valeur de chaque terme de la fonction objectif optimisée ($Depl$ pour le critère de déplacement et $Pref$ pour le critère de non préférence), LB donne la borne inférieure donnée par Cplex, la colonne Tps reporte en secondes le temps de calcul nécessaire pour obtenir cette solution, enfin la colonne E donne l'écart relatif en pourcentage entre la borne supérieure et inférieure. Pour chaque groupe, la moyenne des écarts et du cpu, la moyenne sur toutes instances résolues par Cplex et enfin, la moyenne sur toutes les instances testées sont reportées.

TABLE 4.1 – Résultats de Cplex pour les 37 instances

Instance	PLME					
	UB	Depl	Pref	LB	Cpu	E
18-4-s-2a	40,13*	108,30	-68,17	40,13*	0,14	0,00
18-4-s-2b	81,15*	130,50	-49,35	81,15*	0,22	0,00
18-4-s-2c	34,78*	104,10	-69,32	34,78*	0,25	0,00
18-4-s-2d	55,26*	123,00	-67,74	55,26*	1,58	0,00
18-4-m-2a	57,42*	108,90	-51,48	57,42*	0,43	0,00
18-4-m-2b	21,31*	97,80	-76,49	21,31 *	0,17	0,00
18-4-m-2c	49,68*	128,40	-78,72	49,68*	17,38	0,00
18-4-m-2d	51,27*	128,70	-77,43	51,27*	1,47	0,00
18-4-m-2e	36,39*	102,90	-66,51	36,39*	0,61	0,00
18-4-l-2a	18,14*	86,40	-68,26	18,14*	10,67	0,00
18-4-l-2b	4,91*	92,70	-87,79	4,91*	2,16	0,00
18-4-l-2c	27,39*	107,10	-79,71	27,39*	529,34	0,00
18-4-l-2d	44,91*	96,60	-51,69	44,91*	629,50	0,00
18-4-l-2e	9,78*	78,00	-68,22	9,78*	0,34	0,00
Moy_{G_1}	-	-	-	-	85,31	0,00
45-10-s-3a	-82,01*	268,20	-350,21	-82,01*	17,07	0,00
45-10-s-2a	-37,79*	231,30	-269,09	-37,79*	16,79	0,00
45-10-s-3b	-35,74*	241,20	-276,94	-35,74*	15,38	0,00
45-10-s-2b	-97,42*	256,50	-353,92	-97,42*	1512,76	0,00
45-10-s-3c	-97,13*	264,30	-361,43	-97,13*	2433,61	0,00
45-10-m-4	-96,97*	270,60	-367,57	-96,97*	2658,89	0,00
45-10-m-2a	-43,10*	243,30	-286,40	-43,10 *	1066,90	0,00
45-10-m-2b	-120,65*	249,00	-369,65	-120,65*	2265,34	0,00
45-10-m-3	-124,06*	234,60	-358,66	-124,06*	1588,07	0,00
45-10-l-2a	-105,33	213,9	-319,23	-115,01	3601,11	8,42
45-10-l-2b	-147,14	212,4	-359,54	-157,3	3602,23	6,46
45-10-l-3	-150,53	220,2	-370,73	-162,18	3600,29	7,18
45-10-l-4	-152,48	221,4	-373,88	-165,55	3639,09	7,89
Moy_{G_2}	-	-	-	-	2001,35	2,30
73-16-s-2a	-195,82	369,3	-565,19	-231,36	3601,00	15,36
73-16-s-3	-187,45	351,9	-539,35	-234,52	3600,22	20,07
73-16-s-2b	-195,82	369,3	-565,197962	-231,36	3601,37	15,36
73-16-m-3a	-187,45	351,9	-539,35	-234,57	3600,81	20,09
73-16-m-2	-	-	-	-255,89	3600,60	-
73-16-m-3b	-145,38	386,1	-531,48	-258,94	3601,16	43,86
73-16-l-2	-	-	-	-319,76	3600,00	-
73-16-l-3	-	-	-	-325,19	3600,00	-
73-16-l-4	-	-	-	-333,82	3600,00	-
73-16-l-5	-	-	-	-339,51	3600,00	-
Moy_{G_3}	-	-	-	-	3600,86	22,95
$Moy_{G_{BS}}$	-	-	-	-	1413,01	4,52
Moyenne	-	-	-	-	1479,30	4,52

Opl studio trouve 23 optimas sur 37 instances, dont toutes les instances du premier groupe (G_1), 9 instances du deuxième groupe (G_2) et 0 instance du troisième groupe (G_3). Pour ce dernier ensemble, seulement 5 instances parmi les 10 ont été résolues par Cplex.

En moyenne et par rapport aux bornes inférieures, Cplex obtient un écart de 0% sur les petites instances (G_1) pour un temps de calcul acceptable (85,31 secondes) et un écart de 2,31 % pour les moyennes instances (G_2) dans un temps de calcul assez élevé (soit 2001,35 secondes). Et enfin, pour le dernier groupe d'instances, Cplex atteint un écart de 22,95 % en 3600 secondes. En moyenne sur toutes les instances ayant pu être résolues, Cplex obtient un écart de 4,52 % en 1413 secondes et éventuellement le même écart sur toutes les d'instances en 1479,30 secondes.

En raison de la nature du problème étudié dans cette thèse (NP-Difficile), les solveurs linéaires tels que Cplex utilisant l'algorithme de *Branch-and-Bound*, ne peuvent pas résoudre ce problème pour des instances à 73 clients dans un temps raisonnable, surtout lorsque les fenêtres de temps sont larges. D'ailleurs, on remarque clairement que le temps de calcul augmente considérablement en fonction de la largeur des fenêtres de temps exigées, ainsi que par rapport au nombre de clients à synchroniser.

Pour les 37 instances testées, les tableaux (4.2 - 4.5) récapitulent les résultats obtenus pour chacune des évaluations introduites précédemment (un tableau de résumé pour les 3 critères de sélection). Sur les 10 exécutions réalisées, la meilleure solution en terme de coût est retenue. Les résultats évalués sont ceux obtenus à la fin de la recherche locale, néanmoins dans les tableaux détaillés, les résultats heuristiques sont également présentés. Les indicateurs de performance sélectionnés sont (en moyenne sur toutes les instances) : le pourcentage de l'écart à la borne inférieure (E_{LB}) et à la borne supérieure (E_{UB}) de Cplex, le pourcentage de l'écart de chaque critère séparément (E_{Depl} et E_{Pref}) et enfin le temps de calcul Tps est reporté en secondes. Les solutions moyennes pour chaque groupe d'instance sont données dans Moy_{G1}, Moy_{G2} et Moy_{G3}. Ensuite, $Moy_{G_{BS}}$ correspond pour chaque méthode à la moyenne sur les instances résolues par Cplex (i.e. pour lesquelles Cplex a obtenu une borne supérieure). Enfin, *Moyenne* donne la moyenne sur toutes les instances testées.

TABLE 4.2 – Récapitulatif des résultats de l'évaluation P-DVA

	$P-DVA-1$					$P-DVA-2$					$P-DVA-3$				
	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps
Moy_{G_1}	28,88	28,88	4,09	5,05	0,01	35,85	35,85	4,92	7,79	0,01	46,12	46,12	6,48	6,60	0,01
Moy_{G_2}	40,54	38,45	1,74	5,76	0,19	42,28	40,23	0,43	7,16	0,21	47,28	45,19	3,54	6,38	0,14
Moy_{G_3}	29,70	15,91	4,68	2,63	1,34	28,79	15,19	7,07	0,95	1,33	29,92	14,66	0,89	4,90	1,06
$Moy_{G_{BS}}$	34,91	30,74	3,23	4,96	0,34	38,42	34,40	3,43	6,47	0,37	44,66	40,82	4,41	6,24	0,29
Moyenne	33,20	30,74	3,23	4,96	0,43	36,20	34,40	3,43	6,47	0,44	42,15	40,82	4,41	6,24	0,34

Pour le premier et le deuxième groupe d'instances, toutes les approches mettent quelques fractions de secondes pour construire une solution réalisable et l'améliorer par une recherche locale. Toutefois, la recherche locale met un peu plus de temps pour améliorer la solution construite (2,43 secondes au maximum) pour le troisième groupe d'instances. Nous pouvons

remarquer également que la recherche locale aléatoire est plus rapide que la recherche locale variable (2 fois plus de temps) et ceci pour toutes les évaluations.

TABLE 4.3 – Récapitulatif des résultats de l'évaluation P-VND

	$P-VND-1$					$P-VND-2$					$P-VND-3$				
	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps
Moy_{G_1}	100,50	100,50	10,00	14,52	0,01	81,83	81,83	11,54	11,16	0,01	87,60	87,60	7,62	14,72	0,01
Moy_{G_2}	67,94	66,03	1,91	12,52	0,15	65,12	63,31	7,36	8,04	0,16	95,32	93,33	5,78	13,36	0,14
Moy_{G_3}	39,25	26,35	5,68	5,55	2,33	35,79	23,12	3,85	5,52	2,02	34,52	10,65	1,72	2,01	1,99
$Moy_{G_{BS}}$	77,54	74,91	6,04	12,30	0,67	67,97	65,13	8,64	9,01	0,60	82,26	77,28	6,15	12,24	0,54
Moyenne	72,51	74,91	6,04	12,30	0,69	63,51	65,13	8,64	9,01	0,61	75,97	68,82	5,38	10,81	0,59

Le premier élément remarquable au vu de ces résultats est l'importance de la stratégie de construction de la solution initiale. En effet, l'heuristique de construction parallèle semble meilleure que l'heuristique séquentielle avec un écart moyen de 33 % par rapport à la borne inférieure.

Ainsi, nous pouvons distinguer le critère de sélection C_1 relatif à la priorité considérée dans le choix du client à insérer en phase de construction (ordre croissant des heures au plus tôt), comme meilleur critère de sélection.

TABLE 4.4 – Récapitulatif des résultats de l'évaluation S-DVA

	$S-DVA-1$					$S-DVA-2$					$S-DVA-3$				
	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps
Moy_{G_1}	33,41	33,41	3,35	6,03	0,01	28,84	28,84	4,61	6,11	0,01	42,11	49,64	7,05	4,56	0,01
Moy_{G_2}	48,18	46,09	1,70	7,49	0,15	49,66	47,60	3,70	6,11	0,16	41,49	39,40	-0,70	8,68	0,14
Moy_{G_3}	33,34	24,54	5,98	4,75	1,15	30,74	16,26	3,22	3,70	1,08	30,77	16,34	4,52	3,09	1,32
$Moy_{G_{BS}}$	40,64	37,18	3,09	6,42	0,30	38,55	34,49	4,02	5,73	0,28	41,10	40,28	3,50	6,00	0,36
Moyenne	38,58	37,18	3,09	6,42	0,37	36,67	34,49	4,02	5,73	0,35	38,83	40,28	3,50	6,00	0,41

La deuxième remarque à tirer concerne la stratégie d'exploration des mouvements de recherches locales. En effet, pour les deux heuristiques, la descente à voisinage aléatoire trouve plus d'intérêt sur ces instances que la descente à voisinage variable. Enfin, nous pouvons déduire de façon générale que les meilleures performances sont obtenues par l'heuristique de construction parallèle lorsque celle-ci est améliorée par une recherche locale suivant une descente aléatoire. Cette conclusion est justifiée en terme d'écart moyen à la borne inférieure.

Bien que la recherche locale aléatoire semble avoir des résultats meilleurs lorsqu'elle est appliquée sur des solutions construites séquentiellement, l'écart le moins faible qu'elle obtient est de 41,42 (S-VND-3). Ce dernier reste plus élevé que le gap obtenu par P-DVA-1.

De plus, à partir des résultats détaillés à la fin de ce chapitre, nous remarquons également que la plupart des évaluations permettent d'améliorer une borne supérieure. Le tableau 4.8 (resp.4.9) montre l'obtention de deux (resp. une) solutions optimales par les versions

$S-DVA-1$ et $S-VND-1$. Néanmoins, les écarts très élevés en moyenne dévoilent son instabilité.

TABLE 4.5 – Récapitulatif des résultats de l'évaluation S-VND

	$S-VND-1$					$S-VND-2$					$S-VND-3$				
	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	Tps
Moy_{G_1}	31,74	31,74	4,74	5,26	0,01	44,27	44,27	6,67	5,73	0,01	28,74	28,74	3,19	8,34	0,01
Moy_{G_2}	60,67	58,85	3,12	10,12	0,17	66,11	64,43	3,10	11,06	0,23	58,75	56,92	3,81	9,14	0,23
Moy_{G_3}	30,80	12,59	3,26	2,64	1,94	36,04	27,94	5,12	6,47	2,27	36,63	24,35	4,30	5,67	2,43
$Moy_{G_{BS}}$	43,86	39,76	3,85	6,82	0,51	53,10	49,91	4,98	8,01	0,65	43,14	39,50	3,62	8,25	0,71
Moyenne	41,65	39,76	3,85	6,82	0,59	49,72	49,91	4,98	8,01	0,70	41,42	39,50	3,62	8,25	0,74

Les tableaux détaillés sont présentés à la fin de ce chapitre, les meilleures solutions trouvées sur les 10 exécutions réalisées sont données dans la deuxième partie de chaque tableau (Obj_{RL} pour le coût obtenu après la recherche locale, $Depl_{RL}$ et $Pref_{RL}$ pour chaque critère de la fonction objectif séparément et enfin Tps_{RL} pour le temps d'exécution). La première partie quant à elle, illustre la solution initiale qui a conduit à la meilleure solution sur les 10 exécutions réalisées. Dans cette partie, les colonnes correspondantes sont Obj_H pour le coût obtenu par l'heuristique, $Depl_H$ et $Pref_H$ pour chaque critère séparément, Tps_H pour le temps d'exécution et enfin la colonne Max_{It} donne le nombre d'itérations réalisées pour construire une solution initiale réalisable. Cette information nous permet de discerner la difficulté du problème ; ainsi la première remarque que nous pouvons tirer est que le nombre d'itérations à effectuer dépend de la taille de l'instance ainsi que de la largeur des fenêtres de temps. En effet, le nombre d'itérations nécessaires augmente lorsque les fenêtres de temps relatives aux clients sont réduites ou quand la taille de l'instance est relativement grande. La troisième partie est consacrée à quelques indicateurs avec la colonne E_{LB} (resp. E_{UB}) correspond à l'écart en pourcentage de la meilleure solution présentée en partie 2, à la borne inférieure (resp. supérieure) donnée par Cplex. Les écarts de chaque terme de la fonction objectif sont également donnés (E_{Depl} et E_{Pref}).

D'après les résultats récapitulatifs, le meilleur critère obtenu est C_1, ainsi nous ne présentons que les tableaux détaillés impliquant C_1 (P-DVA-1, P-VND-1, S-DVA-1 et SVND-1).

4.7 Conclusion

Dans ce chapitre, une formulation mathématique du VRPTW-SP étudié dans cette thèse est proposée. Deux heuristiques de construction exécutées chacune pour trois critères de sélection différents et basées sur le principe du plus proche voisin, ainsi que deux stratégies

de descente sont développées. Plusieurs combinaisons des méthodes introduites ont été comparées et la meilleure configuration a été distinguée.

Le VRPTW-SP étudié ici est NP-Difficile, car c'est une généralisation du VRPTW. Lorsque la taille de la flotte de véhicules est limitée (tel est le cas dans cette thèse), le problème de trouver une solution réalisable est NP-Difficile. Par ailleurs, les contraintes de synchronisation/précédence génèrent une interdépendance entre plusieurs tournées. Ces caractéristiques temporelles nécessitent des adaptations non triviales des heuristiques et recherches locales connues pour les problèmes de tournées de véhicules.

Le modèle linéaire résultant semble approprié aux spécifications du problème proposé, résolvant optimalement 23 instances sur 37. Cependant, le temps de calcul augmente considérablement en fonction de la taille des instances, la largeur des fenêtres de temps ainsi que le nombre de clients soumis aux contraintes de synchronisation. Ainsi, l'idée d'éviter la duplication des nœuds à synchroniser s'avère efficace, par rapport aux résultats obtenus dans le même contexte, par Bredström & Rönnqvist (2007) et Afifi *et al.* (2016), où le modèle ne résout optimalement que les petites instances.

Les résultats obtenus par les différentes méthodes approchées permettent de discerner la difficulté du problème. Pour obtenir de bons résultats, la stratégie de construction des tournées est cruciale et impacte à la fois la faisabilité et la qualité des solutions construites. L'heuristique parallèle améliorée par une recherche locale suivant une descente à voisinage aléatoire a donné le plus faible écart par rapport à la meilleure solution connue (donnée par Cplex).

Le point-clé de l'efficacité de ces méthodes réside à notre avis dans l'adaptation des mouvements de recherche locale aux contraintes de synchronisation : applications des mouvements pour tout type de clients (synchronisés ou pas), contrairement à la plupart des problèmes traités dans la littérature, où seuls les mouvements impliquant des clients non synchronisés sont généralement testés. En effet, en général, la recherche locale n'est pas réalisée sur les clients synchronisés ou bien lorsqu'une heure de visite d'un premier service est attribuée, l'heure de visite du deuxième service est systématiquement fixée. Dans notre cas, les tests de faisabilité nous permettent de modifier l'heure de visite des clients synchronisés librement tout en respectant les contraintes de synchronisation. Les meilleurs résultats ont été obtenus par l'heuristique parallèle suivie d'une recherche locale à descente aléatoire. Cette version sera majoritairement réutilisée par la suite.

Les méthodes proposées dans ce chapitre permettent de résoudre le problème proposé en quelques secondes, ce qui est très important pour ce type de problème, particulièrement complexe. Néanmoins, les observations faites sur les solutions obtenues par les heuristiques suivies d'une recherche locale dévoilent les limites de ces méthodes de descente simple. Les

algorithmes développés sont performants mais l'écart important par rapport aux bornes inférieures de Cplex laisse espérer une marge de progression si on se tourne vers des méthodes plus sophistiquées.

Le modèle mathématique proposé dans ce chapitre a fait l'objet d'une présentation en conférence internationale francophone, Conférence Francophone de Modélisation et Simulation - MOSIM'14 (Ait Haddadene *et al.* (2014b)).

Les heuristiques de construction ainsi que les recherches locales ont été présentées pour la première fois lors d'une conférence internationale, *The international workshop on e-Health Pervasive Wireless Applications and Services e-HPWAS'14 (in conjunction with the 10th IEEE WiMob conference)* (Ait Haddadene *et al.* (2014a)) puis dans la revue internationale "*Expert Systems with Applications*" (Ait Haddadene *et al.* (2016a)).

TABLE 4.6 – Résultats détaillés de l'évaluation P-DVA-1 sur 37 instances

	Heuristique parallèle					Descente à voisinage aléatoire				Écart en % à Cplex			
Instance	Obj_H	$Depl_H$	$Pref_H$	Tps_H	Max_{It}	Obj_{RL}	$Depl_{RL}$	$Pref_{RL}$	Tps_{RL}	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}
18-4-s-2a	60,67	117,90	-57,23	0,00	6,00	52,14	114,00	-61,86	0,01	30,05	30,05	5,29	9,28
18-4-s-2b	138,78	151,20	-12,42	0,00	11,00	89,92	135,30	-45,38	0,01	10,84	10,84	3,69	8,10
18-4-s-2c	105,68	120,00	-14,32	0,00	3,00	42,09	102,60	-60,51	0,01	21,53	21,53	-1,43	12,77
18-4-s-2d	96,77	131,10	-34,33	0,00	4,00	61,53	123,60	-62,07	0,01	11,42	11,42	0,49	8,46
18-4-m-2a	170,36	128,40	41,96	0,00	4,00	61,40	114,60	-53,20	0,01	6,98	6,98	5,28	-3,39
18-4-m-2b	109,61	131,10	-21,49	0,00	5,00	23,71	100,20	-76,49	0,01	11,48	11,48	2,47	0,00
18-4-m-2c	138,17	147,30	-9,13	0,00	3,00	50,91	125,10	-74,19	0,01	2,51	2,51	-2,58	5,79
18-4-m-2d	106,95	129,90	-22,95	0,00	5,00	67,09	120,30	-53,21	0,01	31,02	31,02	-6,56	31,44
18-4-m-2e	136,60	129,90	6,70	0,00	11,00	43,14	112,20	-69,06	0,01	18,78	18,78	9,12	-3,86
18-4-l-2a	106,63	120,00	-13,37	0,00	26,00	27,91	103,20	-75,29	0,01	54,33	54,33	19,53	-10,34
18-4-l-2b	148,89	159,60	-10,71	0,00	2,00	8,13	99,00	-90,87	0,00	80,50	80,50	6,85	-3,55
18-4-l-2c	121,90	136,20	-14,30	0,01	68,00	40,76	105,60	-64,84	0,02	49,56	49,56	-1,40	18,82
18-4-l-2d	136,18	163,50	-27,32	0,01	20,00	53,12	106,20	-53,08	0,01	18,66	18,66	10,00	-2,75
18-4-l-2e	118,02	120,00	-1,98	0,00	14,00	14,88	83,10	-68,22	0,01	56,67	56,67	6,54	-0,01
Moy_{G_1}	-	-	-	0,00	13,00	-	-	-	0,01	28,88	28,88	4,09	5,05
45-10-s-3a	267,32	348,60	-81,28	0,00	4,00	-63,75	277,50	-341,25	0,17	22,27	22,27	3,47	2,56
45-10-s-2a	400,96	369,90	31,06	0,00	4,00	0,22	257,10	-256,88	0,15	102,73	102,73	11,17	4,54
45-10-s-3b	391,48	387,60	3,88	0,00	4,00	22,36	263,70	-241,34	0,17	166,00	166,00	9,34	12,90
45-10-s-2b	311,45	353,10	-41,65	0,00	3,00	-75,60	260,70	-336,30	0,13	22,48	22,48	1,64	4,99
45-10-s-3c	251,68	325,80	-74,12	0,00	3,00	-68,22	252,60	-320,82	0,13	29,80	29,80	-4,43	11,25
45-10-m-4	280,87	369,90	-89,03	0,00	3,00	-62,27	278,10	-340,37	0,23	36,15	36,15	2,78	7,41
45-10-m-2a	346,33	360,90	-14,57	0,00	3,00	-17,65	256,20	-273,85	0,13	59,19	59,19	5,31	4,39
45-10-m-2b	313,12	341,70	-28,58	0,00	2,00	-112,61	235,80	-348,41	0,16	6,71	6,70	-5,30	5,76
45-10-m-3	291,89	330,30	-38,41	0,00	2,00	-102,02	238,20	-340,22	0,24	17,78	17,77	1,54	5,15
45-10-l-2a	341,78	354,30	-12,52	0,00	2,00	-95,01	215,70	-310,71	0,17	17,39	9,83	0,85	2,67
45-10-l-2b	294,70	330,30	-35,60	0,00	2,00	-136,09	215,40	-351,49	0,14	13,51	7,52	1,42	2,24
45-10-l-3	235,89	342,00	-106,11	0,00	3,00	-144,13	216,00	-360,13	0,14	11,14	4,27	-1,91	2,86
45-10-l-4	243,87	308,70	-64,83	0,02	12,00	-129,40	214,20	-343,60	0,47	21,91	15,18	-3,26	8,12
Moy_{G_2}	-	-	-	0,00	3,62	-	-	-	0,19	40,54	38,45	1,74	5,76
73-16-s-2a	248,07	600,90	-352,83	0,01	7,00	-155,60	362,40	-518,00	1,88	32,80	20,63	-1,87	8,35
73-16-s-3	289,41	650,10	-360,69	0,02	18,00	-143,94	404,40	-548,34	1,49	38,71	23,27	14,96	-1,67
73-16-s-2b	248,07	600,90	-352,83	0,01	7,00	-155,60	362,40	-518,00	1,31	32,80	20,63	-1,87	8,35
73-16-m-3a	289,41	650,10	-360,69	0,02	18,00	-143,94	404,40	-548,34	1,74	38,73	23,27	14,96	-1,67
73-16-m-2	226,28	607,20	-380,92	0,01	5,00	-158,74	375,00	-533,74	1,10	38,10	-	-	-
73-16-m-3b	237,77	571,50	-333,73	0,02	14,00	-157,31	375,30	-532,61	1,03	39,39	-8,23	-2,80	-0,21
73-16-l-2	246,63	582,60	-335,97	0,00	2,00	-265,63	315,30	-580,93	1,16	16,97	-	-	-
73-16-l-3	258,18	596,40	-338,22	0,01	2,00	-268,25	318,30	-586,55	1,09	17,52	-	-	-
73-16-l-4	233,01	598,20	-365,19	0,01	2,00	-269,52	320,40	-589,92	1,64	19,31	-	-	-
73-16-l-5	223,65	596,70	-373,05	0,06	41,00	-262,47	329,70	-592,17	0,93	22,73	-	-	-
Moy_{G_3}	-	-	-	0,02	11,60	-	-	-	1,34	29,70	15,91	4,68	2,63
$Moy_{G_{BS}}$	-	-	-	0,00	9,03	-	-	-	0,34	34,91	30,74	3,23	4,96
Moyenne	-	-	-	0,01	9,32	-	-	-	0,43	33,20	30,74	3,23	4,96

TABLE 4.7 – Résultats détaillés de l'évaluation P-VND-1 sur 37 instances

	Heuristique parallèle					Descente à voisinage variable				Écart en % à Cplex			
Instance	Obj_H	$Depl_H$	$Pref_H$	Tps_H	Max_{It}	Obj_{RL}	$Depl_{RL}$	$Pref_{RL}$	Tps_{RL}	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}
18-4-s-2a	164,67	142,20	22,47	0,00	2,00	92,68	126,90	-34,22	0,01	131,40	131,40	17,23	49,93
18-4-s-2b	110,34	149,70	-39,36	0,00	9,00	95,65	139,80	-44,15	0,01	17,91	17,91	7,15	10,61
18-4-s-2c	158,79	128,70	30,09	0,00	2,00	48,07	128,40	-80,33	0,02	39,12	39,12	23,37	-15,96
18-4-s-2d	129,00	147,60	-18,60	0,00	6,00	62,47	122,70	-60,23	0,01	13,13	13,13	-0,24	11,21
18-4-m-2a	170,36	128,40	41,96	0,00	4,00	66,41	102,60	-36,19	0,01	15,77	15,77	-5,83	29,96
18-4-m-2b	152,20	145,20	7,00	0,00	2,00	52,61	109,80	-57,19	0,01	149,10	149,10	12,37	25,39
18-4-m-2c	122,31	131,40	-9,09	0,00	2,00	50,91	125,10	-74,19	0,01	2,51	2,51	-2,58	5,79
18-4-m-2d	94,17	125,70	-31,53	0,00	5,00	74,37	123,60	-49,23	0,01	45,29	45,29	-3,98	36,61
18-4-m-2e	156,37	145,20	11,17	0,00	2,00	68,29	106,80	-38,51	0,00	88,64	88,64	3,82	42,42
18-4-l-2a	143,23	111,60	31,63	0,00	2,00	24,97	103,80	-78,83	0,01	38,00	38,00	20,23	-15,54
18-4-l-2b	128,71	117,60	11,11	0,00	2,00	23,46	110,40	-86,94	0,01	463,75	463,75	19,24	0,97
18-4-l-2c	127,22	125,70	1,52	0,00	2,00	54,82	108,60	-53,78	0,01	101,63	101,63	1,40	32,82
18-4-l-2d	122,25	142,50	-20,25	0,00	2,00	56,68	116,70	-60,02	0,01	26,75	26,75	20,94	-16,35
18-4-l-2e	112,31	135,30	-22,99	0,00	2,00	34,44	99,00	-64,56	0,01	274,00	274,00	26,92	5,37
Moy_{G_1}	-	-	-	0,00	3,14	-	-	-	0,01	100,50	100,50	10,00	14,52
45-10-s-3a	215,96	333,90	-117,94	0,03	55,00	-37,08	264,00	-301,08	0,32	54,79	54,79	-1,57	14,04
45-10-s-2a	303,41	315,90	-12,49	0,00	5,00	9,37	230,10	-220,73	0,19	127,46	127,46	-0,52	17,98
45-10-s-3b	385,17	387,90	-2,73	0,00	3,00	36,58	257,70	-221,12	0,23	206,63	206,63	6,85	20,22
45-10-s-2b	355,88	350,10	5,78	0,00	4,00	-82,49	252,60	-335,09	0,16	15,38	15,38	-1,52	5,33
45-10-s-3c	279,58	315,90	-36,32	0,01	2,00	-37,91	254,40	-292,31	0,18	61,05	61,05	-3,75	19,15
45-10-m-4	337,35	320,70	16,65	0,00	2,00	7,89	288,90	-281,01	0,17	109,23	109,23	6,78	23,59
45-10-m-2a	291,98	316,50	-24,52	0,00	2,00	9,06	270,00	-260,94	0,14	121,30	121,30	10,99	8,90
45-10-m-2b	178,71	275,10	-96,39	0,01	2,00	-88,32	238,20	-326,52	0,15	26,95	26,94	-4,34	11,69
45-10-m-3	182,15	287,10	-104,95	0,00	2,00	-38,89	248,70	-287,59	0,14	68,69	68,69	6,03	19,85
45-10-l-2a	280,58	269,70	10,88	0,00	2,00	-88,36	227,40	-315,76	0,07	23,17	16,16	6,34	1,09
45-10-l-2b	157,31	258,60	-101,29	0,00	2,00	-126,30	213,60	-339,90	0,07	19,75	14,18	0,57	5,47
45-10-l-3	194,97	263,40	-68,43	0,00	2,00	-130,60	213,30	-343,90	0,09	19,49	13,29	-3,14	7,25
45-10-l-4	223,22	285,90	-62,68	0,00	2,00	-117,13	226,20	-343,33	0,09	29,35	23,26	2,17	8,19
Moy_{G_2}	-	-	-	0,00	6,54	-	-	-	0,15	67,94	66,03	1,91	12,52
73-16-s-2a	219,41	565,50	-346,09	0,28	257,00	-139,03	380,10	-519,13	2,80	39,97	29,12	2,93	8,15
73-16-s-3	215,80	552,90	-337,10	0,09	79,00	-112,27	394,50	-506,77	4,39	52,24	40,20	12,14	6,04
73-16-s-2b	219,41	565,50	-346,09	0,30	257,00	-139,03	380,10	-519,13	2,86	39,97	29,12	2,93	8,15
73-16-m-3a	215,80	552,90	-337,10	0,09	79,00	-112,27	394,50	-506,77	4,15	52,26	40,20	12,14	6,04
73-16-m-2	190,70	542,40	-351,70	0,05	34,00	-143,85	357,30	-501,15	2,48	43,94	-	-	-
73-16-m-3b	200,58	538,80	-338,22	0,03	17,00	-155,36	379,50	-534,86	3,33	40,15	-6,88	-1,71	-0,64
73-16-l-2	59,27	388,50	-329,23	0,01	2,00	-216,62	330,60	-547,22	0,48	32,33	-	-	-
73-16-l-3	73,07	402,30	-329,23	0,00	2,00	-197,06	337,80	-534,86	0,67	39,42	-	-	-
73-16-l-4	84,78	423,00	-338,22	0,00	2,00	-225,17	324,30	-549,47	1,03	32,63	-	-	-
73-16-l-5	136,49	507,30	-370,81	0,01	3,00	-273,20	311,10	-584,30	1,11	19,56	-	-	-
Moy_{G_3}	-	-	-	0,09	73,20	-	-	-	2,33	39,25	26,35	5,68	5,55
$Moy_{G_{BS}}$	-	-	-	0,03	25,82	-	-	-	0,67	77,54	74,91	6,04	12,30
Moyenne	-	-	-	0,02	23,27	-	-	-	0,69	72,51	74,91	6,04	12,30

TABLE 4.8 – Résultats détaillés de l'évaluation S-DVA-1 sur 37 instances

	Heuristique séquentielle					Descente à voisinage aléatoire				Écart en % à Cplex			
Instance	Obj_H	$Depl_H$	$Pref_H$	Tps_H	Max_{It}	Obj_{RL}	$Depl_{RL}$	$Pref_{RL}$	Tps_{RL}	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}
18-4-s-2a	83,72	129,90	-46,18	0,00	7,00	48,33	119,70	-71,37	0,01	20,53	20,53	10,56	-4,71
18-4-s-2b	103,37	129,30	-25,93	0,01	26,00	84,81	131,40	-46,59	0,01	4,53	4,53	0,69	5,63
18-4-s-2c	128,58	119,10	9,48	0,00	3,00	42,09	102,60	-60,51	0,01	21,53	21,53	-1,43	12,77
18-4-s-2d	97,99	129,90	-31,91	0,00	7,00	55,26	123,00	-67,74	0,01	0,00	0,00	0,00	0,00
18-4-m-2a	156,17	132,90	23,27	0,00	23,00	66,41	102,60	-36,19	0,01	15,77	15,77	-5,83	29,96
18-4-m-2b	110,70	119,10	-8,40	0,00	4,00	28,62	96,30	-67,68	0,01	34,86	34,86	-1,55	11,59
18-4-m-2c	132,81	133,50	-0,69	0,00	5,00	52,40	115,50	-63,10	0,01	5,55	5,55	-10,08	20,01
18-4-m-2d	112,04	144,00	-31,96	0,00	6,00	75,97	131,10	-55,13	0,01	48,43	48,43	1,88	28,95
18-4-m-2e	123,30	130,20	-6,90	0,00	6,00	45,37	114,30	-68,93	0,01	24,97	24,97	11,18	-3,67
18-4-l-2a	106,01	124,50	-18,49	0,00	8,00	28,21	103,50	-75,29	0,00	56,00	56,00	19,88	-10,34
18-4-l-2b	128,71	117,60	11,11	0,00	2,00	12,28	102,30	-90,02	0,00	184,25	184,25	10,43	-2,57
18-4-l-2c	128,23	139,80	-11,57	0,00	4,00	37,58	93,30	-55,72	0,01	37,78	37,78	-12,90	30,37
18-4-l-2d	136,24	142,50	-6,26	0,01	92,00	50,88	119,70	-68,82	0,02	13,57	13,57	24,06	-33,61
18-4-l-2e	114,28	118,80	-4,52	0,00	22,00	9,78	78,00	-68,22	0,01	0,00	0,00	0,00	0,00
Moy_{G_1}	-	-	-	0,00	15,36	-	-	-	0,01	33,41	33,41	3,35	6,03
45-10-s-3a	286,11	346,20	-60,09	0,02	56,00	-68,15	251,70	-319,85	0,23	16,90	16,90	-6,16	8,67
45-10-s-2a	276,28	337,80	-61,52	0,00	7,00	22,66	257,70	-235,04	0,11	163,38	163,38	11,43	12,66
45-10-s-3b	345,36	357,00	-11,64	0,02	46,00	21,14	268,80	-247,66	0,14	162,51	162,51	11,45	10,61
45-10-s-2b	207,42	239,70	-32,28	0,00	2,00	-59,98	246,90	-306,88	0,15	38,59	38,59	-3,75	13,32
45-10-s-3c	257,51	323,70	-66,19	0,00	4,00	-58,82	249,60	-308,42	0,20	39,49	39,49	-5,57	14,68
45-10-m-4	211,06	267,00	-55,94	0,00	2,00	-63,63	252,90	-316,53	0,20	34,73	34,73	-6,56	13,91
45-10-m-2a	346,32	346,20	0,12	0,00	5,00	-16,96	262,80	-279,76	0,13	60,79	60,79	8,02	2,32
45-10-m-2b	193,16	228,00	-34,84	0,00	2,00	-86,61	245,10	-331,71	0,10	28,38	28,37	-1,57	10,28
45-10-m-3	216,03	291,60	-75,57	0,00	3,00	-102,14	240,90	-343,04	0,14	17,69	17,68	2,69	4,36
45-10-l-2a	237,51	206,40	31,11	0,00	2,00	-89,25	214,80	-304,05	0,08	22,40	15,31	0,42	4,76
45-10-l-2b	121,81	190,20	-68,39	0,00	2,00	-141,58	228,30	-369,88	0,11	10,01	3,78	7,50	-2,88
45-10-l-3	113,45	197,70	-84,25	0,00	2,00	-138,20	223,20	-361,40	0,13	14,80	8,22	1,36	2,52
45-10-l-4	123,12	209,40	-86,28	0,00	2,00	-138,11	227,70	-365,81	0,19	16,63	9,45	2,85	2,16
Moy_{G_2}	-	-	-	0,00	10,38	-	-	-	0,15	48,18	46,09	1,70	7,49
73-16-s-2a	185,22	535,80	-350,58	0,10	113,00	-152,58	383,40	-535,98	1,00	34,10	22,17	3,82	5,17
73-16-s-3	210,71	556,80	-346,09	0,15	161,00	-112,70	405,30	-518,00	1,70	52,06	39,97	15,21	3,96
73-16-s-2b	185,22	535,80	-350,58	0,11	113,00	-152,58	383,40	-535,98	1,04	34,10	22,17	3,82	5,17
73-16-m-3a	210,71	556,80	-346,09	0,18	161,00	-112,70	405,30	-518,00	1,98	52,08	39,97	15,21	3,96
73-16-m-2	161,63	483,00	-321,37	0,12	149,00	-175,84	357,90	-533,74	1,02	31,39	-	-	-
73-16-m-3b	139,09	481,80	-342,71	0,21	220,00	-147,67	354,60	-502,27	1,25	43,13	-1,58	-8,16	5,50
73-16-l-2	89,51	369,30	-279,79	0,00	2,00	-257,10	308,10	-565,20	0,77	19,64	-	-	-
73-16-l-3	69,85	400,20	-330,35	0,01	3,00	-262,19	307,50	-569,69	0,74	19,38	-	-	-
73-16-l-4	93,25	423,60	-330,35	0,01	8,00	-250,74	302,10	-552,84	1,04	24,95	-	-	-
73-16-l-5	187,43	504,30	-316,87	0,05	29,00	-263,15	318,90	-582,05	0,99	22,53	-	-	-
Moy_{G_3}	-	-	-	0,09	95,90	-	-	-	1,15	33,34	24,54	5,98	4,75
$Moy_{G_{BS}}$	-	-	-	0,03	38,39	-	-	-	0,30	40,64	37,18	3,09	6,42
Moyenne	-	-	-	0,03	35,38	-	-	-	0,37	38,58	37,18	3,09	6,42

TABLE 4.9 – Résultats détaillés de l'évaluation S-VND-1 sur 37 instances

	Heuristique séquentielle					Descente à voisinage variable				Écart en % à Cplex			
Instance	Obj_H	$Depl_H$	$Pref_H$	Tps_H	Max_{It}	Obj_{RL}	$Depl_{RL}$	$Pref_{RL}$	Tps_{RL}	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}
18-4-s-2a	83,72	129,90	-46,18	0,00	7,00	48,33	119,70	-71,37	0,01	20,53	20,53	10,56	-4,71
18-4-s-2b	103,37	129,30	-25,93	0,01	26,00	84,81	131,40	-46,59	0,01	4,53	4,53	0,69	5,63
18-4-s-2c	128,58	119,10	9,48	0,00	3,00	42,09	102,60	-60,51	0,01	21,53	21,53	-1,43	12,77
18-4-s-2d	97,99	129,90	-31,91	0,00	7,00	55,26	123,00	-67,74	0,01	0,02	0,02	0,00	0,00
18-4-m-2a	129,64	134,10	-4,46	0,00	19,00	68,37	115,80	-47,43	0,01	19,21	19,21	6,39	7,92
18-4-m-2b	110,70	119,10	-8,40	0,00	4,00	23,71	100,20	-76,49	0,01	11,48	11,48	2,47	0,00
18-4-m-2c	132,81	133,50	-0,69	0,00	5,00	53,91	128,10	-74,19	0,01	8,63	8,63	-0,23	5,79
18-4-m-2d	112,04	144,00	-31,96	0,01	6,00	75,97	131,10	-55,13	0,02	48,43	48,43	1,88	28,95
18-4-m-2e	134,67	131,40	3,27	0,01	5,00	52,24	91,80	-39,56	0,01	44,06	44,06	-10,88	40,83
18-4-l-2a	106,01	124,50	-18,49	0,00	8,00	28,21	103,50	-75,29	0,01	56,00	56,00	19,88	-10,34
18-4-l-2b	148,89	159,60	-10,71	0,00	2,00	8,13	99,00	-90,87	0,00	80,50	80,50	6,85	-3,55
18-4-l-2c	125,27	145,20	-19,93	0,01	61,00	43,55	105,60	-62,05	0,01	59,89	59,89	-1,40	22,35
18-4-l-2d	136,24	142,50	-6,26	0,01	92,00	50,66	115,50	-64,84	0,02	13,07	13,07	19,69	-25,80
18-4-l-2e	114,10	132,00	-17,90	0,00	16,00	14,86	87,30	-72,44	0,01	56,44	56,44	11,92	-6,22
Moy_{G_1}	-	-	-	0,00	18,64	-	-	-	0,01	31,74	31,74	4,74	5,26
45-10-s-3a	332,50	362,10	-29,60	0,01	10,00	-49,54	267,00	-316,54	0,39	39,60	39,60	-0,45	9,62
45-10-s-2a	395,91	330,00	65,91	0,01	11,00	22,42	273,60	-251,18	0,24	162,73	162,73	18,31	6,66
45-10-s-3b	345,36	357,00	-11,64	0,02	46,00	31,38	279,60	-248,22	0,19	191,77	191,77	15,93	10,41
45-10-s-2b	207,42	239,70	-32,28	0,00	2,00	-32,14	248,10	-280,24	0,17	67,29	67,29	-3,28	20,87
45-10-s-3c	243,89	306,60	-62,71	0,00	3,00	-48,52	250,80	-299,32	0,21	50,11	50,11	-5,11	17,20
45-10-m-4	211,94	307,50	-95,56	0,02	29,00	-53,15	260,70	-313,85	0,19	45,65	45,65	-3,67	14,64
45-10-m-2a	346,32	346,20	0,12	0,00	5,00	-24,40	252,60	-277,00	0,13	43,49	43,49	3,83	3,29
45-10-m-2b	193,16	228,00	-34,84	0,00	2,00	-52,34	244,50	-296,84	0,14	56,93	56,92	-1,81	19,73
45-10-m-3	216,03	291,60	-75,57	0,00	3,00	-89,44	250,20	-339,64	0,14	27,93	27,92	6,67	5,31
45-10-l-2a	237,51	206,40	31,11	0,00	2,00	-65,99	201,90	-267,89	0,09	42,63	37,47	-5,63	16,09
45-10-l-2b	121,81	190,20	-68,39	0,00	2,00	-142,25	224,40	-366,65	0,15	9,59	3,33	5,66	-1,98
45-10-l-3	113,45	197,70	-84,25	0,00	2,00	-112,16	246,00	-358,16	0,08	30,88	25,58	11,73	3,40
45-10-l-4	123,12	209,40	-86,28	0,00	2,00	-132,43	217,80	-350,23	0,08	20,07	13,19	-1,63	6,34
Moy_{G_2}	-	-	-	0,00	9,15	-	-	-	0,17	60,67	58,85	3,12	10,12
73-16-s-2a	162,36	525,30	-362,94	0,09	94,00	-159,71	372,90	-532,61	1,60	31,02	18,52	0,98	5,77
73-16-s-3	210,71	556,80	-346,09	0,15	161,00	-143,59	387,90	-531,49	3,88	38,86	23,45	10,26	1,46
73-16-s-2b	162,36	525,30	-362,94	0,10	94,00	-159,71	372,90	-532,61	1,76	31,02	18,52	0,98	5,77
73-16-m-3a	210,71	556,80	-346,09	0,17	161,00	-143,59	387,90	-531,49	3,73	38,88	23,45	10,26	1,46
73-16-m-2	169,01	510,60	-341,59	0,02	14,00	-147,35	377,40	-524,75	1,52	42,56	-	-	-
73-16-m-3b	152,06	491,40	-339,34	0,09	100,00	-175,83	362,40	-538,23	1,87	32,21	-21,00	-6,14	-1,27
73-16-l-2	89,51	369,30	-279,79	0,00	2,00	-235,44	312,90	-548,34	1,17	26,43	-	-	-
73-16-l-3	127,84	415,50	-287,66	0,01	4,00	-245,18	314,40	-559,58	0,85	24,62	-	-	-
73-16-l-4	126,50	416,40	-289,90	0,00	2,00	-250,41	324,90	-575,31	1,93	25,05	-	-	-
73-16-l-5	77,09	443,40	-366,31	0,01	4,00	-280,85	305,70	-586,55	1,04	17,30	-	-	-
Moy_{G_3}	-	-	-	0,06	63,60	-	-	-	1,94	30,80	12,59	3,26	2,64
$Moy_{G_{BS}}$	-	-	-	0,02	30,42	-	-	-	0,51	43,86	39,76	3,85	6,82
Moyenne	-	-	-	0,02	27,46	-	-	-	0,59	41,65	39,76	3,85	6,82

Chapitre 5

Métaheuristiques pour la résolution mono-objectif du VRPTW-SP

5.1 Introduction

Comme nous venons de voir dans le chapitre précédent, le VRPTW-SP est un problème très complexe. Les solveurs de programmation linéaire atteignent rapidement leurs limites et leur utilisation n'est pas toujours envisageable dans un contexte opérationnel. De plus, nous avons constaté que le recours à des algorithmes de construction de solution initiale améliorée par des recherches locales sophistiquées, bien que plus rapides, présentent des écarts non négligeables par rapport à l'optimum. Ceci laisse espérer une possibilité d'amélioration dans la qualité des solutions, en développant des approches plus puissantes. C'est la motivation qui est à la base des recherches exposées dans ce chapitre.

Les métaheuristiques sont en général plus puissantes que des heuristiques constructives suivies d'une recherche locale car elles incluent des mécanismes permettant de sortir d'un minimum local. Il en existe deux grandes familles, comme décrit au chapitre 2 : celles fondées sur l'exploration d'un voisinage (GRASP, TS, ...), et celles basées sur l'utilisation d'une population (GA, ACO, ...). De façon générale, les métaheuristiques sont des méthodes d'amélioration de haut niveau, combinant des heuristiques de construction (plus souvent randomisées), des mouvements de recherches locales et des stratégies permettant de sortir d'un minimum local. Elles peuvent donner d'excellents résultats sur des problèmes de grande taille, en un temps raisonnable, lorsque les solveurs linéaires qui sont connus pour être très performants, demandent un temps prohibitif.

Dans ce chapitre, nous proposons de résoudre le VRPTW-SP par trois métaheuristiques

basées sur l'exploration du voisinage : *GRASP Greedy Randomized Adaptive Procedure*, *ILS Iterated Local Search* et $GRASP \times ILS$ (hybridation de ces deux dernières). Notre choix s'est porté sur ces méthodes car elles ont déjà donné de bons résultats sur différentes variantes du VRP. De plus, ce sont des méthodes offrant un bon compromis entre efficacité et simplicité d'implémentation. Le principe général de ces méthodes est donné dans les sections suivantes (section 5.2 à 5.4). La section 5.5 est dédiée aux résultats obtenus et à l'analyse des contributions apportées par l'hybridation de ces deux métaheuristiques. Enfin, une conclusion termine ce chapitre en section 5.6.

5.2 GRASP

La métaheuristique GRASP (*Greedy randomized adaptive search procedure*) a été introduite pour la première fois par Feo & Resende (1995) pour un problème de recouvrement d'ensembles (*set covering problem*). Cette méthode a déjà fait ses preuves pour des problèmes de construction de tournées, depuis, Kontoravdis & Bard (1995) l'ont utilisé pour résoudre un problème de conception de tournées avec fenêtres de temps où le but est de minimiser le nombre de véhicules utilisés.

C'est une méthode itérative qui produit à chaque itération une nouvelle solution en deux phases :

— La construction d'une solution réalisable par une heuristique gloutonne randomisée : elle consiste à générer une solution par une suite de décisions élémentaires.

— L'amélioration de celle-ci en utilisant une recherche locale : exploration itérative de l'espace des solutions visant à améliorer la solution courante.

Les solutions des différentes itérations sont totalement indépendantes et la meilleure solution trouvée est restituée à la fin de la procédure par modification élémentaire de cette dernière.

Dans la première phase, l'heuristique gloutonne choisie est randomisée afin de générer des solutions différentes. Nous avons déjà développé de telles heuristiques au chapitre précédent où la randomisation est basée sur la liste des candidats (RCL) "Restricted Candidate List" de notre liste de patients pouvant être insérés à chaque étape de la procédure d'insertion.

La partie la plus difficile dans la conception d'un GRASP consiste à réussir une randomisation suffisante d'une heuristique gloutonne afin de diversifier la recherche tout en préservant la qualité des solutions résultantes. En effet, si la taille de RCL est très réduite, l'heuristique se rapproche d'une heuristique déterministe. Par conséquent, la recherche locale peut trouver le même optimum local à chaque itération du GRASP. En revanche, l'utilisation d'une RCL de taille importante peut donner des solutions très éloignées les unes des autres, mais risque de rendre le GRASP similaire à un échantillonnage aléatoire de l'espace des solutions faisables.

L'algorithme (5.1) donne la structure général d'un GRASP de base. La méthode commence par initialiser le coût de la meilleure solution F^* à l'infini et le nombre d'itérations $iter$ à 0. Chaque itération commence par construire une solution Sol à l'aide d'une heuristique gloutonne randomisée (HGR), cette dernière est améliorée en appliquant une recherche locale (RL). La meilleure solution Sol^* ainsi que son coût F^* sont mis à jour si le coût de la nouvelle solution est meilleur que celui de Sol^*. L'algorithme s'arrête lorsque le nombre d'itérations $iter$ atteint une valeur maximale $iterMax$ et retourne la meilleure solution Sol^*.

Algorithme 5.1 GRASP de base

1: $F^* \leftarrow +\infty$
2: $iter \leftarrow 0$
3: **Tant que** (iter $<$ iterMax) **Faire**
4: $\quad iter \leftarrow iter + 1$
5: $\quad Sol \leftarrow HGR()$
6: $\quad Sol \leftarrow RL(Sol)$
7: $\quad$ **Si** $(F(Sol) < F^*)$ **Alors**
8: $\quad\quad Sol^* \leftarrow Sol$
9: $\quad\quad F^* \leftarrow F(Sol)$
10: $\quad$ **Fin Si**
11: **Fin Tant que**
12: Retourner Sol^*

Pour la résolution de notre problème grâce à la métaheuristique GRASP, nous choisissons deux types d'approche. Tout d'abord, nous proposons de combiner les techniques proposées dans le chapitre 4, à savoir les heuristiques de construction et les recherches locales. Ensuite, nous proposons une version hybride où la partie recherche locale est remplacée par une métaheuristique ILS (voir la section 5.4).

L'algorithme (5.2) donne la structure générale du GRASP proposé pour résoudre le VRPTW-SP. Une version randomisée de l'heuristique de construction parallèle décrite dans la section 4.4.4 est utilisée dans cette méthode. A chaque itération du GRASP, une solution Sol est construite et réparée en cas de non faisabilité, grâce aux méthodes de réparation introduites dans le chapitre précédent (HCP). La solution résultante est ensuite améliorée en utilisant la recherche locale organisée sous forme d'une descente à voisinage aléatoire (DVA). L'algorithme s'arrête après un nombre maximum d'itérations $iterMax$ ou quand la borne inférieure LB est atteinte. A la fin, la meilleure solution Sol^* et son coût F^* sont renvoyés.

La conception d'un GRASP repose sur des principes simples. C'est un algorithme

d'échantillonnage de l'espace des solutions, qui peut manquer d'efficacité pour garantir de très bons résultats à cause de ces itérations complètement indépendantes. Plusieurs techniques d'amélioration du GRASP peuvent être envisagées, notamment, la combinaison avec l'utilisation d'une recherche locale plus agressive à la place d'une recherche locale classique.

Algorithme 5.2 GRASP pour le VRPTW-SP

```
1: F* ← +∞
2: iter ← 0
3: Tant que (iter < iterMax) et (F* > LB) Faire
4:    Sol ← HCP()
5:    Si (Sol est réalisable) Alors
6:       iter ← iter + 1
7:       Sol ← DVA(Sol)
8:       Si (F(Sol) < F*) Alors
9:          Sol* ← Sol
10:         F* ← F(Sol)
11:      Fin Si
12:   Fin Si
13: Fin Tant que
14: Retourner Sol*
```

5.3 ILS

La recherche locale itérée (ILS) est une autre métaheuristique de la famille fondée sur l'exploration d'un voisinage qui est basé sur le principe de proximité des optimums locaux. Selon Glover & Laguna (2013), les optimums locaux son regroupés en grappes dans l'espace de recherche.

L'ILS initie sa recherche à partir d'une solution de départ de bonne qualité. Cette solution est d'abord construite à l'aide d'une heuristique gloutonne, puis améliorée au moyen d'une recherche locale de manière à obtenir un premier minimum local. Ensuite, contrairement au GRASP, chaque itération part de la solution courante et applique une perturbation dans l'espoir de sortir de ce bassin d'attraction. La solution une fois perturbée, est améliorée par la recherche locale pour atteindre un autre minimum local, proche du précédent. La solution courante ne sera alors remplacée par la nouvelle solution, que si cette dernière est meilleure. Le cycle composé de (Perturbation + Recherche locale) est répété un certain nombre de fois jusqu'à ce qu'un critère d'arrêt soit atteint.

Algorithme 5.3 ILS de base

1: $iter = 0$
2: $Sol \leftarrow HG()$
3: $Sol \leftarrow RL(Sol)$
4: $F^* \leftarrow F(Sol)$
5: **Tant que** $(iter < iterMax)$ **Faire**
6: $\quad \overline{Sol} \leftarrow PP(Sol)$
7: $\quad \overline{Sol} \leftarrow RL(\overline{Sol})$
8: $\quad$ **Si** $(F(\overline{Sol}) < F^*)$ **Alors**
9: $\quad\quad Sol \leftarrow \overline{Sol}$
10: $\quad\quad F^* \leftarrow F(\overline{Sol})$
11: $\quad$ **Fin Si**
12: $\quad iter = iter + 1$
13: **Fin Tant que**
14: Retourner Sol

Un chapitre récent du livre de Lourenço *et al.* (2010) est dédié entièrement à la méthode ILS, avec une longue liste de références citées. L'algorithme (5.3) donne la structure générale d'une ILS de base. La méthode commence par initialiser le coût de la meilleure solution F^* à l'infini et le nombre d'itérations $iter$ à 0. Puis une solution initiale Sol est construite à l'aide d'une heuristique gloutonne (HG), cette dernière est améliorée en appliquant une recherche locale (RL). Ensuite, à chaque itération la solution courante (Sol) subit le processus composé de (Procédure de Perturbation (PP) + Recherche Locale (RL)). La solution résultante ($\overline{Sol}$) est évaluée (ligne 8) et la meilleure solution Sol^*, son coût F^* et la solution courante (Sol) sont mis à jour si le coût de la nouvelle solution est meilleur que celui de la précédente. L'algorithme s'arrête lorsque le nombre d'itérations $iter$ atteint une valeur maximale $iterMax$ et retourne la meilleure solution Sol^*.

Pour résoudre le VRPTW-SP introduit dans cette thèse, nous avons choisi de tester la performance de cette méthode avant de la combiner avec la méthode GRASP. L'algorithme (5.4) détaille sa structure générale. L'heuristique de construction parallèle (HCP) a été utilisée pour construire la solution de départ de l'ILS. Cette heuristique est employée dans sa version déterministe. C'est à dire le choix du client à insérer pendant la construction correspond au meilleur candidat selon le critère C_1 (voir section 4.4.2). Ainsi, afin d'obtenir des résultats comparables, la recherche locale (DVA) adoptée pour le GRASP a été encore une fois utilisée. Cette méthode s'arrête lorsque le nombre d'itérations $iter$ atteint une valeur maximale $iterMax$ ou quand la borne inférieure LB est retrouvée.

Algorithme 5.4 ILS pour le VRPTW-SP

1: $F^* \leftarrow +\infty$
2: $nbIter = 0$
3: $Sol \leftarrow HCP()$
4: $Sol \leftarrow DVA(Sol)$
5: $F^* \leftarrow F(Sol)$
6: **Tant que** $(nbIter < iterMax)$ *et* $(F^* > LB)$ **Faire**
7: $\quad \overline{Sol} \leftarrow PP(Sol)$
8: $\quad$ **Si** (Sol est réalisable) **Alors**
9: $\quad\quad \overline{Sol} \leftarrow DVA(\overline{Sol})$
10: $\quad\quad$ **Si** $(F(\overline{Sol}) < F^*)$ **Alors**
11: $\quad\quad\quad Sol \leftarrow \overline{Sol}$
12: $\quad\quad\quad F^* \leftarrow F(\overline{Sol})$
13: $\quad\quad$ **Fin Si**
14: $\quad\quad nbIter = nbIter + 1$
15: $\quad$ **Fin Si**
16: **Fin Tant que**
17: **Retourner** Sol

L'élément clé se situe au niveau de la procédure de perturbation (PP). En effet, cette métaheuristique est basée sur une idée simple : au lieu de l'application répétée d'une procédure de recherche locale à partir de solutions générées aléatoirement, l'ILS génère la solution de départ pour la prochaine itération en appliquant une perturbation sur l'optimum local trouvé à l'itération courante. Le mécanisme de perturbation est crucial : d'une part, une perturbation trop faible peut ne pas être suffisante pour échapper à l'optimum local actuel ; d'autre part, une perturbation trop forte aura tendance à générer des solutions indépendantes les unes des autres à l'instar de la méthode GRASP.

Pour sauter d'un bassin d'attraction à un autre, la perturbation devrait introduire des composants non déterministes. Plusieurs techniques de perturbations peuvent être envisageables, la plus fréquente pour les problèmes de tournées consiste à échanger les positions d'un certain nombre de clients choisis aléatoirement. Néanmoins, ce mécanisme risque d'être insuffisant dans le sens où la recherche locale appliquée par la suite pourrait facilement revenir à la même solution de départ par l'application de mouvements inverses. Cependant, après des tests préliminaires, nous avons opté pour une procédure de perturbation de type "destruction/construction". Elle consiste à supprimer un client choisi aléatoirement et de le réinsérer dans la première position possible d'une autre tournée. Ce

processus est répété un certain nombre prédéfini (cp_{max}) de fois. À la fin de cette procédure, si la solution n'est pas réalisable, le mécanisme de réparation (décrit dans la sous section 4.4.5) utilisé pour réparer l'infaisabilité des solutions construites est appliqué afin d'obtenir une solution réalisable.

5.4 GRASP × ILS pour le VRPTW-SP

Comme toute méthode d'optimisation, l'ILS possède certaines limites : l'ILS peut se trouver enfermée dans une grappe de bassins d'attraction ou dans un bassin très large, il est donc plus fructueux de redémarrer la recherche dans des espaces non explorés au lieu de prendre du temps dans des itérations inutiles. Cette stratégie à démarrage multiple peut être obtenue en combinant les deux métaheuristiques détaillées ci-dessus, i.e. en remplaçant la recherche locale de la méthode GRASP par l'ILS.

La structure générale de la méthode issue de l'hybridation GRASP × ILS, proposée pour résoudre le VRPTW-SP est donnée dans l'algorithme 5.5. Malgré sa simplicité, cette méthode a déjà fait preuve de succès sur plusieurs problèmes combinatoires, par exemple, le problème de localisation-routage à deux échelons traité dans Nguyen *et al.* (2012) ou encore le problème de tournées de véhicules périodiques avec fenêtre temporelles introduit par Michallet *et al.* (2014).

La boucle principale (lignes 3-32) décrit la méthode hybride GRASP × ILS composée d'une phase de construction et une phase d'amélioration réalisée par une recherche locale itérée (ILS). La première solution $\overline{Sol}$ est généré à l'aide de l'heuristique constructive randomisée parallèle (HCP). Chaque itération de l'ILS (lignes 10-25) appelle la procédure de perturbation (PP) décrite dans la section précédente, la recherche locale aléatoire détaillée précédemment (DVA) et met à jour la dernière solution enregistrée $\overline{Sol}$ lorsque celle-ci est améliorée. Les critères d'arrêts considérés sont les suivants :

— La construction d'une nouvelle solution (redémarrage de la méthode) s'arrête après un nombre d'itérations maximum $iterMax_G$ ou quand la borne inférieure LB est atteinte.

— La procédure ILS quant à elle s'arrête soit après un nombre $iterMax_L$ ou après un certain nombre d'itérations sans amélioration max_{Echec}, comptabilisé par la variable $notImp$, cette dernière est incrémentée lorsque la solution perturbée n'est pas réalisable même après la réparation et/ou si l'itération en cours de l'ILS n'est pas productive (i.e. n'aboutit pas à une solution meilleure).

Algorithme 5.5 $GRASP \times ILS$

1: $F^* \leftarrow +\infty$
2: $Iter_G = 0$
3: **Tant que** $((Iter_G < iterMax_G)$ *et* $(F^* > LB))$ **Faire**
4: $\quad \overline{Sol} \leftarrow HCP(\overline{Sol})$
5: $\quad$ **Si** ($\overline{Sol}$ faisable) **Alors**
6: $\quad\quad Iter_G = Iter_G + 1$
7: $\quad\quad Iter_L = 0$
8: $\quad\quad notImp = 0$
9: $\quad\quad \overline{Sol} \leftarrow DVA(\overline{Sol})$
10: $\quad\quad$ **Tant que** $(Iter_L < iterMax_L)$ et $(notImp < max_{Echec}))$ **Faire**
11: $\quad\quad\quad Sol \leftarrow \overline{Sol}$
12: $\quad\quad\quad Sol \leftarrow PP(Sol)$
13: $\quad\quad\quad$ **Si** (Sol faisable) **Alors**
14: $\quad\quad\quad\quad Iter_L = Iter_L + 1$
15: $\quad\quad\quad\quad Sol \leftarrow DVA(Sol)$
16: $\quad\quad\quad\quad$ **Si** $(F(Sol) < F(\overline{Sol}))$ **Alors**
17: $\quad\quad\quad\quad\quad \overline{Sol} \leftarrow Sol$
18: $\quad\quad\quad\quad\quad Iter_L \leftarrow 0$
19: $\quad\quad\quad\quad$ **Sinon**
20: $\quad\quad\quad\quad\quad notImp = notImp + 1$
21: $\quad\quad\quad\quad$ **Fin Si**
22: $\quad\quad\quad$ **Sinon**
23: $\quad\quad\quad\quad notImp = notImp + 1$
24: $\quad\quad\quad$ **Fin Si**
25: $\quad\quad$ **Fin Tant que**
26: $\quad\quad$ **Si** $(F(\overline{Sol}) < F^*)$ **Alors**
27: $\quad\quad\quad S^* \leftarrow \overline{Sol}$
28: $\quad\quad\quad F^* \leftarrow F(\overline{Sol})$
29: $\quad\quad\quad Iter_G \leftarrow 0$
30: $\quad\quad$ **Fin Si**
31: $\quad$ **Fin Si**
32: **Fin Tant que**
33: **Retourner** Sol^*

Rappelons que $F(Sol)$ correspond à l'évaluation du coût de la solution Sol décrit dans

la section (4.3.2).

5.5 Évaluations numériques

5.5.1 Implémentation et instances

Les métaheuristiques proposées dans ce chapitre ont été codées en C et exécutées sur un ordinateur avec un processeur Corei5 de 8 Go de RAM fonctionnant sous Linux. L'évaluation numérique a été réalisée sur l'ensemble d'instances décrit dans le chapitre précédent.

5.5.2 Paramétrage des algorithmes

Le réglage des paramètres d'une métaheuristique est critique. L'objectif est d'obtenir les meilleures performances en testant un nombre limité de combinaisons de paramètres. Un plan d'expérience a donc été conçu pour fixer les paramètres des méthodes proposées. Pour se faire, plusieurs façons peuvent être envisagées. Naturellement, on peut penser à paramétrer chaque métaheuristique séparément et de les comparer. Néanmoins, cette stratégie n'assure pas la cohérence des résultats obtenus, dans le sens où le nombre d'appels à la recherche locale n'est plus le même.

Nous avons proposé de régler les paramètres de la méthode hybride (GRASP $\times$ ILS), à savoir le nombre de redémarrages $iterMax_G$, le nombre d'appels à l'ILS $iterMax_L$, le nombre d'itérations sans succès max_{Echec} et enfin le paramètre de la procédure de perturbation cp_{max} relatif au nombre de clients à supprimer. Ensuite, le nombre de recherches locales moyen effectué pour obtenir la meilleure solution pour la version hybride est restitué et sera donné comme critère d'arrêts pour le GRASP et l'ILS. Ceci est donné en ajoutant une variable de comptage qui sera incrémentée à chaque appel à la procédure de recherche locale (DVA). Notons que toutes les métaheuristiques proposées dans ce chapitre, ont été exécutées 10 fois et la meilleure solution ainsi que la solution moyenne sont données pour la comparaison.

Finalement, plusieurs paramètres ont été évalués, la colonne 3 du tableau 5.1 correspond à l'intervalle des valeurs testées. Les meilleures valeurs permettant d'obtenir un bon compromis entre la qualité de la solution et le temps de calcul sont indiquées dans la colonne 4. Quant aux colonnes 1 et 2, elles détaillent le paramètre en question ainsi que sa désignation (notation) dans les algorithmes.

TABLE 5.1 – Paramètres fixés pour le $GRASP \times ILS$

Paramètre	Notation	Intervalle des valeurs testées	Valeur
Nombre de redémarrages	$iterMax_G$	[20 .. 80]	40
Nombre d'itérations de l'ILS	$iterMax_L$	[20 .. 80]	40
Nombre de solutions sans amélioration	max_{Echec}	[10 .. 40]	30
Niveau de perturbation	cp_{max}	[2 .. 5]	3

5.5.3 Résultats et interprétations

Les résultats récapitulatifs sont donnés dans les tableaux 5.2 et 5.3. Les résultats détaillés pour les 37 instances de chaque métaheuristique sont présentés à la fin de ce chapitre. Afin de mieux analyser les résultats obtenus, nous proposons deux comparaisons. Une fois, en comparant les résultats de la métaheuristique GRASP × ILS aux résultats donnés par Cplex (correspondant à la résolution du modèle mathématique proposé dans le chapitre précédent) et ensuite, cette dernière est considérée comme une méthode de référence, ainsi les résultats obtenus par GRASP et ILS seront comparés d'abord entre eux, puis aux résultats de la méthode de référence. Les résultats sont donnés pour chaque groupe d'instances séparément (soient G_1, G_2 et G_3), pour les instances pour lesquelles Cplex a retrouvé une solution réalisable (G_{BS}) et enfin pour toutes les instances (G_{All}).

Dans le tableau 5.2, la partie 1 rappelle les résultats obtenus par le solveur Cplex. Ensuite, la partie 2 du même tableau ainsi que les deux parties du tableau récapitulatif 5.3 exposent les résultats des méthodes proposées. Les indicateurs de performance utilisés pour comparer ces différents résultats sont : le pourcentage de l'écart moyen de la meilleure solution du GRASP × ILS sur toutes les exécutions, à la borne inférieure (E_{LB}) et à la borne supérieure (E_{UB}) ainsi qu'à chaque critère séparément (E_{Depl} et E_{Pref}). Nous exposons également en secondes, les temps nécessaires pour obtenir les meilleurs résultats (Tps_{min}). Ensuite, la solution moyenne sur les différentes exécutions est comparée à la meilleure solution obtenue par la méthode prise comme référence (donné par $Dev_{Obj_{Ref}}$ $Dev_{Depl_{Ref}}$ et $Dev_{Pref_{Ref}}$). Enfin, nous exposons les temps de calcul moyen en secondes, nécessaires pour obtenir les solutions moyennes, notés Tps_{moy} et les temps de calcul nécessaires pour obtenir les solutions optimales, notés Tps_{opt} (Ref = GRASP × ILS, Z = coût total de la fonction objectif).

Notons que Tps_{opt} n'est pas reporté dans le tableau 5.3 car les méthodes comparées n'ont pas retrouvé les mêmes solutions optimales.

Dans les tableaux récapitulatifs, la colonne G_{BS} montre les résultats obtenus par Cplex, n'incluant pas les solutions pour lesquelles Cplex ne parvenait pas à obtenir une solution réalisable en un temps limité à 3600 secondes. Toutefois, les solutions obtenues par les

méthodes proposées seraient considérées meilleure. Ainsi, les tableaux détaillés montrent que toutes les métaheuristiques résolvent efficacement les instances non résolues par Cplex, en des temps raisonnables, alors que Cplex n'arrive pas à atteindre une solution faisable après une heure d'exécution sur certaines d'entre elles.

5.5.3.1 Résultats du GRASP × ILS

TABLE 5.2 – Indicateurs de performance pour les 37 instances - partie 1

Indicateurs	$MILP$					$GRASP \times ILS$				
	G_1	G_2	G_3	G_{BS}	G_{All}	G_1	G_2	G_3	G_{BS}	G_{All}
E_{LB}	0,00	2,30	22,95	4,52	4,52	0,00	2,06	11,92	3,24	3,95
E_{UB}	-	-	-	-	-	0,00	-0,26	-12,45	-2,05	-2,05
E_{Depl}	-	-	-	-	-	0,00	-0,43	3,08	0,31	0,31
E_{Pref}	-	-	-	-	-	0,00	0,20	-5,60	-0,79	-0,79
Tps_{min}	85,31	2001,35	3600,86	1413,01	1479,30	0,28	64,91	1184,93	160,74	343,17
$Dev_{Z_{Ref}}$	-	-	-	-	-	0,04	3,08	2,78	1,98	1,85
$Dev_{Depl_{Ref}}$	-	-	-	-	-	0,05	-0,33	0,54	0,05	0,05
$Dev_{Pref_{Ref}}$	-	-	-	-	-	-0,06	0,62	0,69	0,37	0,38
Tps_{moy}	-	-	-	-	-	0,37	101,31	980,86	164,42	300,83
Tps_{opt}	85,31	1286,09	-	608,03	608,03	0,28	56,89	-	22,43	22,43

Rappelons que Opl Studio a permis de récupérer 23 solutions optimales sur les 37 instances testées. La méthode hybride a permis de récupérer toutes ces solutions en un temps beaucoup plus réduit (**22,43** secondes pour le GRASP × ILS, v.s **608,03** secondes pour Cplex). Ainsi, toutes les bornes supérieures ont été améliorées (voir tableau détaillé 5.6).

De façon générale, les résultats obtenus montrent que la méthode hybride proposée ici améliore les solutions de Cplex avec un gain moyen de **2,05 %**. Ainsi, cette méthode atteint un écart moyen de **3,24 %** à la borne inférieure en **160,74** secondes. Cet écart est réduit de plus de **1 %** par rapport aux résultats obtenus par Cplex (**4,52 %** d'écart en **1413,01** secondes) et en temps de calcul considérablement plus petit (soit 10 fois plus rapide).

L'efficacité de cette méthode hybride est confirmée surtout en observant les résultats du troisième groupe (G_3). En effet, GRASP × ILS obtient une amélioration de **12,45 %** en **1184,93** secondes, comparés aux bornes supérieures de Cplex où le gap est de 22 % en **3600** secondes.

En ce qui concerne les écarts relatifs de chaque critère de la fonction objectif, les résultats montrent clairement la nature multi-objectif du problème : pour certains cas, par exemple l'instance 45-10-l-2a (voir le tableau détaillé 5.6)), le critère de déplacement s'est dégradé

de **4,92%** alors que le critères de préférence s'est amélioré de **1,56 %**. Bien que les deux critères soient contradictoire, les résultats expriment que les mouvements de recherche locale proposés sont adaptés aux deux critères de décision. D'ailleurs, dans la littérature, les auteurs ayant traités les instances initiales de Bredström & Rönnqvist (2007) ont souvent donné les résultats pour les deux critères séparément.

En d'autres termes, les solutions de la méthode hybride GRASP × ILS sont meilleures que la borne supérieure donnée par le solveur Cplex après 3600 secondes de calcul, même si ce dernier utilise une méthode exacte de type *Branch and Bound* pour résoudre le modèle MILP ; et cela dans des temps nettement plus court. De plus, par sa nature NP-Difficile, les solveurs linéaires tels que Cplex n'assurent pas l'obtention de solutions faisables pour des instances de plus de 45 clients. En revanche, GRASP × ILS fonctionne efficacement sur des instances plus grandes.

TABLE 5.3 – Indicateurs de performance pour les 37 instances -partie 2

Indicateurs	ILS					$GRASP$				
	G_1	G_2	G_3	G_{BS}	G_{All}	G_1	G_2	G_3	G_{BS}	G_{All}
E_{LB}	0,34	6,14	9,78	4,69	4,93	0,00	4,89	17,23	5,48	6,38
E_{UB}	0,34	3,82	-15,38	-0,70	-0,70	0,00	2,63	-3,40	0,54	0,54
E_{Depl}	0,18	-0,13	-4,74	-0,72	-0,72	0,00	1,17	-1,58	0,23	0,23
E_{Pref}	-0,07	0,62	-1,28	0,02	0,02	0,00	-0,34	0,65	-0,04	-0,04
$E_{Z_{Ref}}$	0,34	4,08	-2,44	1,39	0,90	0,00	2,88	6,12	2,46	2,67
$E_{Depl_{Ref}}$	0,18	0,28	-0,62	-1,00	0,00	0,00	1,63	2,68	-0,05	1,30
$E_{Pref_{Ref}}$	-0,07	0,42	-0,11	0,77	0,10	0,00	-0,55	1,50	0,69	0,21
Tps_{min}	0,13	62,83	463,82	105,44	147,48	0,16	258,99	1360,41	171,96	458,74
$Dev_{Z_{Ref}}$	19,73	14,22	2,12	14,88	13,03	0,92	8,68	8,25	5,71	5,63
$Dev_{Depl_{Ref}}$	3,00	0,07	0,97	0,62	1,42	0,10	0,84	3,98	-0,09	1,41
$Dev_{Pref_{Ref}}$	1,71	2,33	7,17	4,34	3,40	0,40	1,10	1,57	1,55	0,96
Tps_{moy}	0,22	73,80	478,46	112,35	155,33	0,23	280,61	1330,44	180,03	458,26

5.5.3.2 Évaluation de l'efficacité de l'hybridation

Pour évaluer l'efficacité de la méthode, la métaheuristique GRASP × ILS est comparée à la méthode :

— GRASP pur utilisant la recherche locale DVA,

— ILS pur obtenue en éliminant la boucle de génération des solutions initiales.

Comme nous l'avons précisé précédemment, le nombre moyen d'appels à la recherche locale réalisés dans le GRASP × ILS est donné comme critère d'arrêt au GRASP et à l'ILS. Les résultats sont donnés dans le tableau 5.3.

En comparant les deux méthodes pures (GRASP et ILS) aux résultats donnés par Cplex, nous concluons que l'ILS est meilleure que le GRASP (**4,69 %** d'écart à la borne inférieure en **105,44** seconde pour l'ILS, v.s **5,48 %** d'écart en **171,96** secondes pour GRASP). Ainsi, l'ILS permet une amélioration de **0,70 %** de la borne supérieure de Cplex par contre, le GRASP obtient plutôt un gap de **0,54 %**.

En revanche, en considérant les meilleures solutions du GRASP × ILS comme solutions de référence, les résultats montrent que la métaheuristique GRASP est moins efficace avec un écart moyen d'environ **2,46 %**. Par conséquent, l'ILS semble plus performante que cette dernière avec d'environ **1,39 %** d'écart à la méthode de référence. D'ailleurs, nous pouvons même dire que la différence avec la méthode hybride est peu significative (0,90 % d'écart en considérant toutes les instances).

En regardant les solutions moyennes, GRASP serait moins performant mais plus stable que l'ILS avec une déviation moyenne de **5,71 %** comparé à **14,88 %**. ILS est donc moins stable, néanmoins, son introduction dans une boucle à redémarrage la rend plus stable (**14,88 %** d'écart pour l'ILS pur, v.s **1,98 %** d'écart pour le GRASP × ILS sur la moyenne des 10 exécutions)

En résumé, les résultats nous permettent de déduire que la version hybride est plus performante et plus stable que la version standard de l'ILS, qui à son tour est meilleure que le GRASP, pour un nombre équivalent d'appels à la recherche locale. Par conséquent, nous pouvons ressortir l'inégalité suivante : GRASP × ILS < ILS < GRASP. Cependant, nous allons justifier nos résultats statistiquement en utilisant le test de Friedman afin d'évaluer les résultats obtenus.

5.5.3.3 Tests statistiques

L'écart relatif absolu à l'optimal d'une métaheuristique suit rarement une distribution normale. La méthode ne peut pas trouver moins de 0% (et cet écart peut être fréquemment atteint). Une comparaison de la qualité des solutions basée seulement sur des valeurs moyennes et/ou des temps d'exécution n'est pas toujours suffisante. Des performances exceptionnelles sur un échantillon d'instances peut influencer la performance globale d'une méthode observée. C'est pourquoi une évaluation statistique est nécessaire afin de sélectionner la meilleure méthode. Conover (1980) recommande plusieurs tests non paramétriques. Nous proposons réaliser le test de Friedman, qui permet d'analyser nos résultats et qui ne nécessitent aucune hypothèse sur les distributions de probabilités.

Le test est réalisé sur $k = 3$ métaheuristiques randomisées à comparer (GRASP × ILS, ILS et GRASP), sur un échantillon d'instances $b = 18$. Les instances sélectionnées pour le test statistique appartiennent à G_2 et G_3.

Soit X_{ij} la meilleure solution obtenue sur les 10 exécutions réalisées par la métaheuristique i sur l'instance j. Tout d'abord, nous commençons par calculer les rangs R_{ij} de X_{ij} en attribuant la valeur de 1 à la meilleure méthode et k à la pire. En cas d'égalité, la valeur de 1,5 pour deux heuristiques ayant le même coût sera affectée. Ensuite, nous calculons pour chaque métaheuristique j la somme des rangs comme suit : $S_j = \sum_{i=1}^{b} R_{ij}$, la moyenne de ces sommes au carré $B_2 = \frac{1}{b}(\sum_{j=1}^{k} S_j^2)$ et la somme des rangs au carré $A_2 = \sum_{i=1}^{b} \sum_{j=1}^{k} R_{ij}^2$.

Si l'hypothèse nulle (que toutes les méthodes sont équivalentes) est vraie, la distribution des rangs pour chaque métaheuristique sera due à la chance (instances choisies), et les différents rangs apparaîtront avec la même fréquence. Le total des rangs pour chaque méthode sera donc aléatoire. Dans le cas contraire, le total des rangs par méthode devrait varier. Le test de Friedman vérifie si les totaux des rangs diffèrent significativement. La valeur statistique du test χ^2 est T_2. Cette valeur se calcule de la façon suivante :

$$T_2 = \frac{(b-1)[B_2 - bk(k+1)^2/4]}{(A_2 - B_2)} \tag{5.1}$$

Comme la distribution T_2 est une approximation de la distribution du khi^2, si la valeur statistique T_2 calculée par la formule (5.1) est supérieure ou égale à la distribution $F_{1-\alpha,k_1,k_2}$ pour un niveau de signification $(1-\alpha)$ et un degré de liberté donnés (k_1, k_2) avec $k_1 = k-1$ et $k_2 = (k-1)(b-1)$, les sommes des rangs des trois méthodes diffèrent significativement et l'hypothèse nulle peut être rejetée ($\alpha = 0,01$).

Les résultats sont donnés dans le tableau 5.4. Dans notre cas, nous obtenons $T_2 = 14,28 > F_{0,99;2;34} = 5,39$. Nous pouvons donc affirmer que les méthodes comparées ne sont pas équivalentes avec un niveau de confiance de 99 %.

TABLE 5.4 – Résultats du test de Friedman sur 18 instances - Partie 1

	$GRASP \times ILS$	ILS	$GRASP$
Rang moyen	1,75	1,5	2,75
Somme des rangs S_j	31,5	27	49,5
Somme des rangs au carré	59,75	50,5	140,25
Indicateurs	A_2	B_2	T_2
Valeurs	250,5	231,75	**14,28**

Par conséquent, un test sur chaque paire de méthodes devrait être effectué, pour distinguer la performance entre chaque deux métaheuristiques proposées. Pour cela et pour chaque paire de méthodes (i, j), nous calculons la différence absolue des sommes des rangs de i et j. Ainsi, i et j seraient deux méthodes différentes lorsque la différence ente eux dépasse une valeur critique notée CV et définie par l'équation (5.2). Où $t_{1-\frac{\alpha}{2}}$ désigne $1 - \frac{\alpha}{2}$ est une approximation de la loi de *Student* avec $(b-1)(k-1)$ comme degré de liberté ($\alpha = 0,05$).

Dans ce cas, sur chaque paire de métaheuristiques, la méthode correspondante à la plus petite somme de rangs est considérée comme meilleure.

$$|S_i - S_j| > CV = t_{1-\frac{\alpha}{2}} \sqrt{\frac{2b(A_2 - B_2)}{(b-1)(k-1)}} \tag{5.2}$$

Le tableau (5.5) donne les résultats obtenus en appliquant ce test de comparaison entre les paires de méthodes proposées. Pour nos instances, $CV = 12,25$. Les conclusions remarquables (soulignées) sont : (a) $GRASP \times ILS$ est meilleure que $GRASP$. (b) $GRASP$ est dominée par les deux autres méthodes, (c) pas de différence significative entre les deux métaheuristiques $GRASP \times ILS$ et ILS. Néanmoins, les résultats donnés dans le tableau (5.3) montraient l'instabilité de la méthode ILS, d'où l'intérêt de l'hybridation.

TABLE 5.5 – Résultats du test de Friedman sur 18 instances - Partie 2

$\lvert S_i - S_j \rvert$	12,25	
	ILS	$GRASP$
$GRASP \times ILS$	4,5	<u>18</u>
ILS		<u>22,5</u>

5.6 Conclusion

Dans ce chapitre, trois métaheuristiques pour le VRPTW-SP ont été proposées. Une méthode GRASP et une méthode ILS qui utilisent chacune une heuristique constructive inspirée de la méthode du plus proche voisin et une recherche locale explorant des voisinages organisée sous forme d'une descente aléatoire.

Ces deux métaheuristiques sont performantes puisqu'elles permettent d'améliorer largement les solutions obtenues par les meilleures méthodes présentées dans le chapitre précédent, qui utilisent déjà une recherche locale. Néanmoins, comparée aux bornes inférieures données par Cplex, la méthode GRASP apparaît moins efficace que la méthode ILS, ceci s'explique par l'indépendance des itérations réalisées dans le GRASP. Ainsi, les temps de calcul sont plus longs comparés à l'ILS, mais restent raisonnables pour un tel problème, puisqu'ils ne dépassent pas quelques minutes en moyenne. L'ILS est donc plus performante que GRASP mais très peu stable, d'où l'avantage de la méthode GRASP × ILS issue de la combinaison de ces deux dernières.

Ces conclusions ont été prouvées par la réalisation du test statistique de Friedman, permettant de distinguer la performance des méthodes proposées. Le GRASP × ILS apporte donc des résultats intéressants. Les performances les plus remarquables sont atteintes surtout pour les instances du troisième groupe, pour lesquelles les solveurs linéaires tels que Cplex ne

parvenaient pas à trouver une solution réalisable. Cette méthode s'est montrée stable, efficace avec un bon compromis entre la qualité de la solution et le temps de calcul. Cependant, il arrive que la satisfaction d'un patient obtenue en lui affectant son soignant préféré nécessite un grand détour augmentant ainsi le coût de déplacement, sans trop gagner en terme de mesure de préférence totale. Pour éviter ce problème, une approche multicritère pourrait être intéressante. Ce sera l'objectif du chapitre suivant où nous présenterons des méthodes basées sur l'optimalité au sens de Pareto et leur adaptation au VRPTW-SP dans le but d'optimiser les coûts de déplacement et la satisfaction des patients simultanément.

Les métaheuristiques proposées dans ce chapitre ont été soumises à la revue internationale "*Expert Systems with Applications*" (Ait Haddadene *et al.* (2016a)).

TABLE 5.6 – Résultats détaillés des meilleures solutions du GRASP × ILS

Instance	$GRASP \times ILS$							
	Obj_{min}	$Depl_{min}$	$Pref_{min}$	Tps_{min}	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}
18-4-s-2a	40,13	108,30	-68,17	1,77	0,00	0,00	0,00	0,00
18-4-s-2b	81,15	130,50	-49,35	0,64	0,00	0,00	0,00	0,00
18-4-s-2c	34,78	104,10	-69,32	0,17	0,00	0,00	0,00	0,00
18-4-s-2d	55,26	123,00	-67,74	0,04	0,00	0,00	0,00	0,00
18-4-m-2a	57,42	108,90	-51,48	0,07	0,00	0,00	0,00	0,00
18-4-m-2b	21,31	97,80	-76,49	0,11	0,00	0,00	0,00	0,00
18-4-m-2c	49,68	128,40	-78,72	0,04	0,00	0,00	0,00	0,00
18-4-m-2d	51,27	128,70	-77,43	0,04	0,00	0,00	0,00	0,00
18-4-m-2e	36,39	102,90	-66,51	0,02	0,00	0,00	0,00	0,00
18-4-l-2a	18,14	86,40	-68,26	0,14	0,00	0,00	0,00	0,00
18-4-l-2b	4,91	92,70	-87,79	0,71	0,00	0,00	0,00	0,00
18-4-l-2c	27,39	107,10	-79,71	0,10	0,00	0,00	0,00	0,00
18-4-l-2d	44,91	96,60	-51,69	0,05	0,00	0,00	0,00	0,00
18-4-l-2e	9,78	78,00	-68,22	0,03	0,00	0,00	0,00	0,00
Moy_{G_1}	-	-	-	0,28	0,00	0,00	0,00	0,00
45-10-s-3a	-82,01	268,20	-350,21	24,27	0,00	0,00	0,00	0,00
45-10-s-2a	-37,79	231,30	-269,09	50,15	0,00	0,00	0,00	0,00
45-10-s-3b	-35,74	241,20	-276,94	75,12	0,00	0,00	0,00	0,00
45-10-s-2b	-97,42	256,50	-353,92	43,40	0,00	0,00	0,00	0,00
45-10-s-3c	-97,13	264,30	-361,43	16,89	0,00	0,00	0,00	0,00
45-10-m-4	-96,97	270,60	-367,57	57,26	0,00	0,00	0,00	0,00
45-10-m-2a	-43,10	243,30	-286,40	0,86	0,00	0,00	0,00	0,00
45-10-m-2b	-120,65	249,00	-369,65	85,41	0,00	0,00	0,00	0,00
45-10-m-3	-124,06	234,60	-358,66	158,65	0,00	0,00	0,00	0,00
45-10-l-2a	-107,94	200,90	-308,84	60,75	6,15	-2,48	-6,08	3,25
45-10-l-2b	-147,27	212,40	-359,67	57,19	6,38	-0,09	0,00	-0,04
45-10-l-3	-150,64	220,20	-370,84	122,09	7,12	-0,07	0,00	-0,03
45-10-l-4	-153,63	222,40	-376,03	91,84	7,20	-0,75	0,45	-0,58
Moy_{G_2}	-	-	-	64,91	2,06	-0,26	-0,43	0,20
73-16-s-2a	-196,14	367,30	-563,44	838,96	15,22	-0,16	-0,54	0,31
73-16-s-3	-198,99	380,90	-579,89	456,21	15,15	-6,16	8,24	-7,52
73-16-s-2b	-196,14	367,30	-563,44	835,01	15,22	-0,16	-0,54	0,31
73-16-m-3a	-198,99	380,90	-579,89	446,93	15,17	-6,16	8,24	-7,52
73-16-m-2	-211,80	324,40	-536,20	1835,26	17,23	-	-	-
73-16-m-3b	-217,51	386,1	-603,61	1718,78	16,00	-49,61	0,00	-13,57
73-16-l-2	-309,94	289,20	-599,14	1939,42	3,07	-	-	-
73-16-l-3	-306,19	248,100006	-554,29015	1359,57	5,84	-	-	-
73-16-l-4	-309,93	289,20	-599,13	1779,87	7,16	-	-	-
73-16-l-5	-308,57	268,100006	-576,670015	639,29	9,11	-	-	-
Moy_{G_3}	-	-	-	1184,93	11,92	-12,45	3,08	-5,60
$Moy_{G_{BS}}$	-	-	-	160,74	3,24	-2,05	0,31	-0,79
Moyenne	-	-	-	343,17	3,95	-2,05	0,31	-0,79

TABLE 5.7 – Résultats détaillés des solutions moyennes du GRASP × ILS

Instance	$GRASP \times ILS$										
	Obj_{moy}	$Depl_{moy}$	$Pref_{moy}$	Tps_{moy}	Dev_{LB}	Dev_{UB}	Dev_{Depl}	Dev_{Pref}	$Dev_{Obj_{Ref}}$	$Dev_{Depl_{Ref}}$	$Dev_{Pref_{Ref}}$
18-4-s-2a	40,16	108,30	-68,14	1,71	0,07	0,07	0,00	0,04	0,07	0,00	0,04
18-4-s-2b	81,15	130,50	-49,35	0,34	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-s-2c	34,93	104,85	-69,92	0,96	0,44	0,44	0,72	-0,86	0,44	0,72	-0,86
18-4-s-2d	55,26	123,00	-67,74	0,05	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2a	57,42	108,90	-51,48	0,25	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2b	21,31	97,80	-76,49	0,12	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2c	49,68	128,40	-78,72	0,17	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2d	51,27	128,70	-77,43	0,59	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2e	36,39	102,90	-66,51	0,03	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2a	18,14	86,40	-68,26	0,17	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2b	4,91	92,70	-87,79	0,56	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2c	27,39	107,10	-79,71	0,08	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2d	44,91	96,60	-51,69	0,13	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2e	9,78	78,00	-68,22	0,04	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Moy_{G_1}	-	-	-	0,37	0,04	0,04	0,05	-0,06	0,04	0,05	-0,06
45-10-s-3a	-79,32	271,67	-350,99	135,59	3,28	3,28	1,29	-0,22	3,28	1,29	-0,22
45-10-s-2a	-31,72	248,56	-280,28	113,86	16,06	16,06	7,46	-4,16	16,06	7,46	-4,16
45-10-s-3b	-31,43	247,49	-278,92	117,82	12,05	12,05	2,61	-0,71	12,05	2,61	-0,71
45-10-s-2b	-96,24	244,53	-340,77	98,20	1,21	1,21	-4,67	3,71	1,21	-4,67	3,71
45-10-s-3c	-95,80	249,20	-346,00	104,67	1,37	1,37	-5,71	4,27	1,37	-5,71	4,27
45-10-m-4	-95,02	250,90	-345,92	100,62	2,01	2,01	-7,28	5,89	2,01	-7,28	5,89
45-10-m-2a	-42,51	247,14	-289,65	108,94	1,36	1,36	1,58	-1,14	1,36	1,58	-1,14
45-10-m-2b	-120,65	249,00	-369,65	83,98	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-m-3	-122,44	234,60	-357,04	127,78	1,31	1,31	0,00	0,45	1,31	0,00	0,45
45-10-l-2a	-107,06	200,90	-308,84	73,67	6,91	-1,64	-6,08	3,25	0,82	0,00	0,00
45-10-l-2b	-147,27	210,30	-357,57	73,03	6,38	-0,09	-0,99	0,55	0,00	-0,99	0,58
45-10-l-3	-149,85	222,87	-372,71	91,56	7,60	0,45	1,21	-0,54	0,52	1,21	-0,51
45-10-l-4	-153,52	222,90	-376,42	87,25	7,27	-0,68	0,68	-0,68	0,07	0,22	-0,10
Moy_{G_2}	-	-	-	101,31	5,14	2,82	-0,76	0,82	3,08	-0,33	0,62
73-16-s-2a	-190,65	373,30	-563,95	1004,25	17,59	2,64	1,08	0,22	2,80	1,63	-0,09
73-16-s-3	-185,26	393,90	-579,16	384,67	21,00	1,17	11,94	-7,38	6,90	3,41	0,13
73-16-s-2b	-190,65	367,30	-557,95	1000,85	17,59	2,64	-0,54	1,28	2,80	0,00	0,97
73-16-m-3a	-185,26	380,91	-566,17	385,15	21,02	1,17	8,24	-4,97	6,90	0,00	2,37
73-16-m-2	-208,46	324,50	-532,96	1539,99	18,53	-	-	-	1,58	0,03	0,61
73-16-m-3b	-209,98	386,1	-596,08	1164,40	18,91	-44,44	0,00	-12,15	3,46	0,00	1,25
73-16-l-2	-303,02	290,00	-593,02	1072,70	5,24	-	-	-	2,23	0,28	1,02
73-16-l-3	-304,68	248,100006	-552,78015	1295,40	6,31	-	-	-	0,49	0,00	0,27
73-16-l-4	-308,77	289,20	-597,97	1213,79	7,50	-	-	-	0,37	0,00	0,19
73-16-l-5	-307,63	268,100006	-575,73015	747,41	9,39	-	-	-	0,30	0,00	0,16
Moy_{G_3}	-	-	-	980,86	14,31	-7,36	4,14	-4,60	2,78	0,54	0,69
$Moy_{G_{BS}}$	-	-	-	164,42	5,11	0,01	0,36	-0,41	1,98	0,05	0,37
Moyenne	-	-	-	300,83	5,69	0,01	0,36	-0,41	1,85	0,05	0,38

TABLE 5.8 – Résultats détaillés des meilleures solutions de l'ILS

Instance	ILS										
	Obj_{min}	$Depl_{min}$	$Pref_{min}$	Tps_{min}	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	$E_{Obj_{Ref}}$	$E_{Depl_{Ref}}$	$E_{Pref_{Ref}}$
18-4-s-2a	40,28	103,20	-62,92	0,83	0,37	0,37	-4,71	7,70	0,37	-4,71	7,70
18-4-s-2b	81,15	130,50	-49,35	0,03	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-s-2c	36,31	111,60	-75,29	0,60	4,40	4,40	7,20	-8,61	4,40	7,20	-8,61
18-4-s-2d	55,26	123,00	-67,74	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2a	57,42	108,90	-51,48	0,09	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2b	21,31	97,80	-76,49	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2c	49,68	128,40	-78,72	0,04	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2d	51,27	128,70	-77,43	0,04	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2e	36,39	102,90	-66,51	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2a	18,14	86,40	-68,26	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2b	4,91	92,70	-87,79	0,05	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2c	27,39	107,10	-79,71	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2d	44,91	96,60	-51,69	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2e	9,78	78,00	-68,22	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Moy_{G_1}	-	-	-	0,13	0,34	0,34	0,18	-0,07	0,34	0,18	-0,07
45-10-s-3a	-78,09	273,90	-351,99	80,27	4,78	4,78	2,13	-0,51	4,78	2,13	-0,51
45-10-s-2a	-24,73	244,50	-269,23	60,02	34,56	34,56	5,71	-0,05	34,56	5,71	-0,05
45-10-s-3b	-31,71	229,50	-261,21	92,48	11,28	11,28	-4,85	5,68	11,28	-4,85	5,68
45-10-s-2b	-97,42	256,50	-353,92	5,61	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-s-3c	-97,13	264,30	-361,43	40,78	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-m-4	-96,97	270,60	-367,57	8,50	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-m-2a	-42,55	242,70	-285,25	108,51	1,28	1,28	-0,25	0,40	1,28	-0,25	0,40
45-10-m-2b	-120,65	249,00	-369,65	58,11	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-m-3	-124,06	234,60	-358,66	96,87	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-l-2a	-106,12	194,70	-300,82	65,77	7,73	-0,75	-8,98	5,77	1,69	-3,09	2,60
45-10-l-2b	-147,27	220,50	-367,77	59,27	6,38	-0,09	3,81	-2,29	0,00	3,81	-2,25
45-10-l-3	-150,64	222,60	-373,24	69,52	7,12	-0,07	1,09	-0,68	0,00	1,09	-0,65
45-10-l-4	-154,41	220,50	-374,91	71,11	6,73	-1,27	-0,41	-0,28	-0,51	-0,85	0,30
Moy_{G_2}	-	-	-	62,83	6,14	3,82	-0,13	0,62	4,08	0,28	0,42
73-16-s-2a	-203,87	345,60	-549,47	561,98	11,88	-4,11	-6,42	2,78	-3,94	-5,91	2,48
73-16-s-3	-203,33	354,00	-557,33	553,50	13,30	-8,47	0,60	-3,33	-2,18	-7,06	3,89
73-16-s-2b	-203,87	345,60	-549,47	564,11	11,88	-4,11	-6,42	2,78	-3,94	-5,91	2,48
73-16-m-3a	-202,06	356,40	-558,46	215,51	13,86	-7,79	1,28	-3,54	-1,54	-6,43	3,70
73-16-m-2	-215,03	346,80	-561,83	524,35	15,97	-	-	-	-1,53	6,91	-4,78
73-16-m-3b	-221,56	336,90	-558,46	660,26	14,44	-52,40	-12,74	-5,08	-1,86	-12,74	7,48
73-16-l-2	-310,24	280,80	-591,04	430,55	2,98	-	-	-	-0,10	-2,90	1,35
73-16-l-3	-311,50	291,90	-603,40	438,34	4,21	-	-	-	-1,73	17,65	-8,86
73-16-l-4	-315,24	294,90	-610,14	320,60	5,57	-	-	-	-1,71	1,97	-1,84
73-16-l-5	-326,79	290,10	-616,89	369,01	3,75	-	-	-	-5,90	8,21	-6,97
Moy_{G_3}	-	-	-	463,82	9,78	-15,38	-4,74	-1,28	-2,44	-0,62	-0,11
$Moy_{G_{BS}}$	-	-	-	105,44	4,69	-0,70	-0,72	0,02	1,39	-1,00	0,77
Moyenne	-	-	-	147,48	4,93	-0,70	-0,72	0,02	0,90	0,00	0,10

TABLE 5.9 – Résultats détaillés des solutions moyennes de l'ILS

Instance	*ILS*										
	Obj_{moy}	$Depl_{moy}$	$Pref_{moy}$	Tps_{moy}	Dev_{LB}	Dev_{UB}	Dev_{Depl}	Dev_{Pref}	$Dev_{Obj_{Ref}}$	$Dev_{Depl_{Ref}}$	$Dev_{Pref_{Ref}}$
18-4-s-2a	47,70	114,12	-66,42	1,10	18,87	18,87	5,37	2,57	18,87	5,37	2,57
18-4-s-2b	88,23	134,01	-45,78	0,17	8,73	8,73	2,69	7,24	8,73	2,69	7,24
18-4-s-2c	36,31	111,60	-75,29	0,60	4,40	4,40	7,20	-8,61	4,40	7,20	-8,61
18-4-s-2d	60,07	121,50	-61,43	0,03	8,71	8,71	-1,22	9,32	8,71	-1,22	9,32
18-4-m-2a	62,50	110,91	-48,41	0,12	8,85	8,85	1,85	5,97	8,85	1,85	5,97
18-4-m-2b	30,32	104,46	-74,14	0,08	42,28	42,28	6,81	3,07	42,28	6,81	3,07
18-4-m-2c	51,83	125,25	-73,42	0,09	4,34	4,34	-2,45	6,74	4,34	-2,45	6,74
18-4-m-2d	55,16	129,51	-74,35	0,33	7,59	7,59	0,63	3,98	7,59	0,63	3,98
18-4-m-2e	42,08	105,48	-63,41	0,02	15,62	15,62	2,51	4,67	15,62	2,51	4,67
18-4-l-2a	18,56	86,82	-68,26	0,11	2,32	2,32	0,49	0,00	2,32	0,49	0,00
18-4-l-2b	10,99	98,16	-87,17	0,36	123,75	123,75	5,89	0,70	123,75	5,89	0,70
18-4-l-2c	27,82	106,27	-78,44	0,04	1,58	1,58	-0,78	1,59	1,58	-0,78	1,59
18-4-l-2d	48,32	107,46	-59,14	0,08	7,60	7,60	11,24	-14,41	7,60	11,24	-14,41
18-4-l-2e	11,89	79,38	-67,49	0,02	21,62	21,62	1,77	1,08	21,62	1,77	1,08
Moy_{G_1}	-	-	-	0,22	19,73	19,73	3,00	1,71	19,73	3,00	1,71
45-10-s-3a	-73,57	270,75	-344,32	87,10	10,29	10,29	0,95	1,68	10,29	0,95	1,68
45-10-s-2a	-12,44	245,64	-258,08	75,62	67,09	67,09	6,20	4,09	67,09	6,20	4,09
45-10-s-3b	-14,25	248,58	-262,84	102,96	60,11	60,11	3,06	5,09	60,11	3,06	5,09
45-10-s-2b	-92,35	248,73	-341,08	44,65	5,20	5,20	-3,03	3,63	5,20	-3,03	3,63
45-10-s-3c	-90,04	258,36	-348,40	60,80	7,30	7,30	-2,25	3,60	7,30	-2,25	3,60
45-10-m-4	-89,08	264,78	-353,86	63,11	8,14	8,14	-2,15	3,73	8,14	-2,15	3,73
45-10-m-2a	-38,65	245,91	-284,56	104,41	10,33	10,33	1,07	0,64	10,33	1,07	0,64
45-10-m-2b	-117,01	240,24	-357,25	61,23	3,02	3,02	-3,52	3,36	3,02	-3,52	3,36
45-10-m-3	-118,19	237,72	-355,91	85,57	4,73	4,73	1,33	0,77	4,73	1,33	0,77
45-10-l-2a	-102,93	202,38	-305,31	65,30	10,51	2,28	-5,39	4,36	4,65	0,74	1,14
45-10-l-2b	-146,34	217,47	-363,81	61,99	6,97	0,55	2,39	-1,19	0,63	2,39	-1,15
45-10-l-3	-148,14	215,52	-363,65	76,98	8,66	1,59	-2,13	1,91	1,66	-2,13	1,94
45-10-l-4	-151,03	218,61	-369,64	69,66	8,77	0,95	-1,26	1,13	1,69	-1,70	1,70
Moy_{G_2}	-	-	-	73,80	16,24	13,97	-0,36	2,52	14,22	0,07	2,33
73-16-s-2a	-188,67	361,47	-550,14	601,94	18,45	3,65	-2,12	2,66	3,81	-1,59	2,36
73-16-s-3	-199,93	356,60	-181,83	544,06	14,75	-6,66	1,34	66,29	-0,47	-6,38	68,64
73-16-s-2b	-188,67	361,47	-550,14	603,94	18,45	3,65	-2,12	2,66	3,81	-1,59	2,36
73-16-m-3a	-186,24	369,75	-555,98	221,10	20,61	0,65	5,07	-3,08	6,41	-2,93	4,12
73-16-m-2	-205,47	347,37	-552,84	566,21	19,70	-	-	-	2,99	7,08	-3,10
73-16-m-3b	-214,23	345,12	-559,35	661,67	17,27	-47,36	-10,61	-5,24	1,51	-10,61	7,33
73-16-l-2	-303,66	283,56	-587,22	443,29	5,03	-	-	-	2,03	-1,95	1,99
73-16-l-3	-302,23	287,01	-589,24	428,27	7,06	-	-	-	1,29	15,68	-6,31
73-16-l-4	-305,66	294,60	-600,26	330,59	8,44	-	-	-	1,38	1,87	-0,19
73-16-l-5	-313,46	295,11	-608,57	383,51	7,67	-	-	-	-1,59	10,07	-5,53
Moy_{G_3}	-	-	-	478,46	13,74	-9,21	-1,69	12,66	2,12	0,97	7,17
$Moy_{G_{BS}}$	-	-	-	112,35	18,03	12,87	0,90	3,75	14,88	0,62	4,34
Moyenne	-	-	-	155,33	16,89	12,87	0,90	3,75	13,03	1,42	3,40

TABLE 5.10 – Résultats détaillés des meilleures solutions du GRASP

Instance	*GRASP*										
	Obj_{min}	$Depl_{min}$	$Pref_{min}$	Tps_{min}	E_{LB}	E_{UB}	E_{Depl}	E_{Pref}	$E_{Obj_{Ref}}$	$E_{Depl_{Ref}}$	$E_{Pref_{Ref}}$
18-4-s-2a	40,13	108,30	-68,17	0,91	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-s-2b	81,15	130,50	-49,35	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-s-2c	34,78	104,10	-69,32	0,33	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-s-2d	55,26	123,00	-67,74	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2a	57,42	108,90	-51,48	0,10	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2b	21,31	97,80	-76,49	0,09	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2c	49,68	128,40	-78,72	0,23	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2d	51,27	128,70	-77,43	0,40	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2e	36,39	102,90	-66,51	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2a	18,14	86,40	-68,26	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2b	4,91	92,70	-87,79	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2c	27,39	107,10	-79,71	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2d	44,91	96,60	-51,69	0,09	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2e	9,78	78,00	-68,22	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Moy_{G_1}	-	-	-	0,16	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-s-3a	-82,01	268,20	-350,21	305,58	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-s-2a	-31,48	235,50	-266,98	323,06	16,70	16,70	1,82	0,78	16,70	1,82	0,78
45-10-s-3b	-33,57	242,10	-275,67	482,88	6,07	6,07	0,37	0,46	6,07	0,37	0,46
45-10-s-2b	-97,42	256,50	-353,92	115,92	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-s-3c	-97,13	264,30	-361,43	223,73	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-m-4	-95,70	269,70	-365,40	288,54	1,31	1,31	-0,33	0,59	1,31	-0,33	0,59
45-10-m-2a	-42,07	256,80	-298,87	411,69	2,39	2,39	5,55	-4,35	2,39	5,55	-4,35
45-10-m-2b	-120,65	249,00	-369,65	223,63	0,00	0,00	0,00	0,00	0,00	0,00	0,00
45-10-m-3	-123,44	248,40	-371,84	343,91	0,50	0,50	5,88	-3,67	0,50	5,88	-3,67
45-10-l-2a	-105,33	213,90	-319,23	220,86	8,42	0,00	0,00	0,00	2,42	6,47	-3,36
45-10-l-2b	-147,27	220,50	-367,77	184,27	6,38	-0,09	3,81	-2,29	0,00	3,81	-2,25
45-10-l-3	-148,28	221,40	-369,68	98,59	8,57	1,49	0,54	0,28	1,57	0,54	0,31
45-10-l-4	-143,66	216,00	-359,66	144,26	13,22	5,78	-2,44	3,80	6,49	-2,88	4,35
Moy_{G_2}	-	-	-	258,99	4,89	2,63	1,17	-0,34	2,88	1,63	-0,55
73-16-s-2a	-180,56	354,30	-534,86	294,69	21,96	7,79	-4,06	5,37	7,94	-3,54	5,07
73-16-s-3	-174,47	370,50	-544,97	388,54	25,61	6,92	5,29	-1,04	12,32	-2,73	6,02
73-16-s-2b	-186,70	354,90	-541,60	148,30	19,30	4,66	-3,90	4,18	4,81	-3,38	3,88
73-16-m-3a	-176,04	372,30	-548,34	210,17	24,95	6,09	5,80	-1,67	11,53	-2,26	5,44
73-16-m-2	-200,12	347,10	-547,22	2053,96	21,79	-	-	-	5,51	7,00	-2,05
73-16-m-3b	-207,09	343,50	-550,59	1091,93	20,02	-42,45	-11,03	-3,60	4,79	-11,03	8,78
73-16-l-2	-293,75	292,80	-586,55	3134,99	8,13	-	-	-	5,22	1,24	2,10
73-16-l-3	-294,65	291,90	-586,55	3598,28	9,39	-	-	-	3,77	17,65	-5,82
73-16-l-4	-300,77	310,50	-611,27	2245,90	9,90	-	-	-	2,96	7,37	-2,03
73-16-l-5	-301,22	312,30	-613,52	437,30	11,28	-	-	-	2,38	16,49	-6,39
Moy_{G_3}	-	-	-	1360,41	17,23	-3,40	-1,58	0,65	6,12	2,68	1,50
$Moy_{G_{BS}}$	-	-	-	171,96	5,48	0,54	0,23	-0,04	2,46	-0,05	0,69
Moyenne	-	-	-	458,74	6,38	0,54	0,23	-0,04	2,67	1,30	0,21

TABLE 5.11 – Résultats détaillés des solutions moyennes du GRASP

Instance	*GRASP*										
	Obj_{moy}	$Depl_{moy}$	$Pref_{moy}$	Tps_{moy}	Dev_{LB}	Dev_{UB}	Dev_{Depl}	Dev_{Pref}	$Dev_{Obj_{Ref}}$	$Dev_{Depl_{Ref}}$	$Dev_{Pref_{Ref}}$
18-4-s-2a	40,13	108,30	-68,17	0,59	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-s-2b	81,15	130,50	-49,35	0,20	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-s-2c	34,93	104,85	-69,92	0,83	0,44	0,44	0,72	-0,86	0,44	0,72	-0,86
18-4-s-2d	55,92	121,56	-65,64	0,04	1,19	1,19	-1,17	3,10	1,19	-1,17	3,10
18-4-m-2a	57,42	108,90	-51,48	0,14	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-m-2b	21,79	98,28	-76,49	0,13	2,25	2,25	0,49	0,00	2,25	0,49	0,00
18-4-m-2c	50,54	126,09	-75,55	0,29	1,73	1,73	-1,80	4,03	1,73	-1,80	4,03
18-4-m-2d	51,57	129,00	-77,43	0,48	0,59	0,59	0,23	0,00	0,59	0,23	0,00
18-4-m-2e	38,31	105,30	-66,99	0,03	5,29	5,29	2,33	-0,72	5,29	2,33	-0,72
18-4-l-2a	18,21	86,47	-68,26	0,16	0,37	0,37	0,08	0,00	0,37	0,08	0,00
18-4-l-2b	4,91	92,70	-87,79	0,07	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2c	27,39	107,10	-79,71	0,01	0,00	0,00	0,00	0,00	0,00	0,00	0,00
18-4-l-2d	45,39	97,08	-51,69	0,26	1,07	1,07	0,50	0,00	1,07	0,50	0,00
18-4-l-2e	9,78	78,00	-68,22	0,02	0,00	0,00	0,00	0,00	0,00	0,00	0,00
Moy_{G_1}	-	-	-	0,23	0,92	0,92	0,10	0,40	0,92	0,10	0,40
45-10-s-3a	-77,78	266,10	-343,88	409,06	5,16	5,16	-0,78	1,81	5,16	-0,78	1,81
45-10-s-2a	-28,66	237,00	-265,66	327,11	24,16	24,16	2,46	1,27	24,16	2,46	1,27
45-10-s-3b	-20,78	245,16	-265,94	482,63	41,86	41,86	1,64	3,97	41,86	1,64	3,97
45-10-s-2b	-93,95	252,87	-346,82	242,73	3,56	3,56	-1,42	2,01	3,56	-1,42	2,01
45-10-s-3c	-93,53	264,33	-357,86	271,49	3,71	3,71	0,01	0,99	3,71	0,01	0,99
45-10-m-4	-91,60	263,70	-355,30	288,62	5,54	5,54	-2,55	3,34	5,54	-2,55	3,34
45-10-m-2a	-39,54	250,08	-289,62	413,51	8,27	8,27	2,79	-1,12	8,27	2,79	-1,12
45-10-m-2b	-120,05	246,93	-366,98	222,95	0,50	0,50	-0,83	0,72	0,50	-0,83	0,72
45-10-m-3	-117,32	245,49	-362,81	344,09	5,43	5,43	4,64	-1,16	5,43	4,64	-1,16
45-10-l-2a	-102,85	211,42	-314,28	221,23	10,57	2,35	-1,16	1,55	4,71	5,24	-1,76
45-10-l-2b	-146,99	216,84	-363,83	183,64	6,55	0,10	2,09	-1,19	0,19	2,09	-1,16
45-10-l-3	-147,19	219,39	-366,58	96,15	9,24	2,22	-0,37	1,12	2,29	-0,37	1,15
45-10-l-4	-142,24	217,89	-360,13	144,71	14,08	6,72	-1,59	3,68	7,41	-2,03	4,23
Moy_{G_2}	-	-	-	280,61	10,66	8,43	0,38	1,31	8,68	0,84	1,10
73-16-s-2a	-173,49	362,30	-535,79	225,51	25,01	11,40	-1,90	5,20	11,55	-1,36	4,91
73-16-s-3	-169,58	385,50	-555,08	388,98	27,69	9,53	9,55	-2,92	14,78	1,21	4,28
73-16-s-2b	-179,18	356,80	-535,98	205,97	22,55	8,50	-3,38	5,17	8,65	-2,86	4,87
73-16-m-3a	-169,94	374,33	-544,27	197,55	27,55	9,34	6,37	-0,91	14,60	-1,73	6,14
73-16-m-2	-196,99	350,60	-547,59	2046,31	23,02	-	-	-	6,99	8,08	-2,12
73-16-m-3b	-201,16	345,50	-546,66	1091,85	22,32	-38,37	-10,52	-2,86	7,52	-10,52	9,44
73-16-l-2	-290,41	297,90	-588,31	3141,56	9,18	-	-	-	6,30	3,01	1,81
73-16-l-3	-294,02	303,20	-597,22	3615,46	9,58	-	-	-	3,97	22,21	-7,75
73-16-l-4	-297,35	306,05	-603,40	1953,48	10,92	-	-	-	4,06	5,83	-0,71
73-16-l-5	-295,81	310,80	-606,61	437,69	12,87	-	-	-	4,13	15,93	-5,19
Moy_{G_3}	-	-	-	1330,44	19,07	0,08	0,03	0,74	8,25	3,98	1,57
$Moy_{G_{BS}}$	-	-	-	180,03	8,65	3,84	0,20	0,82	5,71	-0,09	1,55
Moyenne	-	-	-	458,26	9,25	3,84	0,20	0,82	5,63	1,41	0,96

Chapitre 6

Métaheuristiques pour la résolution bi-objectif du VRPTW-SP

6.1 Introduction

Jusqu'à maintenant, nous avons mis en évidence la difficulté de résolution du VRPTW-SP introduit dans cette thèse. Nous avons également illustré la nature conflictuelle des critères de déplacement et de non-préférence, souvent considérés dans la littérature sous forme de combinaison convexe.

Dans les deux chapitres précédents, nous avons abordé le VRPTW-SP sous un angle mono-objectif que nous avons résolu à l'aide de méthodes approchées après avoir proposé et testé une formulation mathématique du problème. Les heuristiques constructives suivies de recherches locales simples manquent de puissance pour obtenir de bons résultats car elles ont une vision locale du problème, par conséquent orienter la recherche vers un minimum global n'est pas évident. En revanche, les métaheuristiques amènent des résultats plus satisfaisants.

Cependant, nous avons souligné un grand nombre de problèmes d'optimisation réels intégrant plusieurs objectifs souvent contradictoires. Dans le problème étudié ici, se déplacer pour apporter des soins spécifiques à des patients nécessite la minimisation des coûts liés au déplacement tout en maximisant la satisfaction du patient. Le fait d'optimiser un de ces objectifs pourra avoir tendance à détériorer le second. C'est d'ailleurs le cas dans la plupart des problèmes d'hospitalisation à domicile.

Pourtant, comme nous l'avons signalé précédemment, cette caractéristique est souvent négligée et les différents objectifs sont alors abordés comme une somme d'agrégation. Dans ce cas l'optimisation est décrite sous forme d'un problème de minimisation (ou maximisation) d'une fonction-objectif définie par une somme pondérée des critères.

D'après la classification proposée dans la section 3.3, plusieurs travaux ont traité

des problèmes similaires aux nôtres considérant plusieurs objectifs gérés par agrégation. Cependant, l'utilisation de ce type de fonctions objectifs présente certains inconvénients en particulier trouver le jeu de points appropriés n'est pas trivial. En utilisant et adaptant les instances de la littérature proposées par Bredström & Rönnqvist (2008), la somme des objectifs tend à s'approcher d'une valeur proche de zéro (Déplacement > 0, Préférence < 0). Il est alors plus difficile de mesurer la performance des méthodes développées, un petit écart sur le résultat global pouvant engendrer un grand écart en pourcentage des critères séparément.

D'un point de vu multi-objectif, Braekers *et al.* (2016) ont étudié les relations de compromis entre la minimisation des coûts de déplacement et la maximisation du niveau de satisfaction des patients en structures de soins à domicile, en modélisant le problème de tournées et de planification des soins comme une variante multi-objectif. Une métaheuristique à base de recherche locale multi-directionnelle a été proposée, construisant un front de solutions. Ce dernier a été comparée au front de Pareto optimal obtenu à l'aide de la méthode ϵ-*Constraints*. A notre connaissance, ces travaux sont les seuls ayant introduit une vraie approche multicritère du problème de tournées de véhicules dans le cadre des soins à domicile.

Dans ce qui précède, nous nous sommes intéressés à la résolution mono-objectif du VRPTW-SP où deux objectifs sont considérés et pondérée dans un seul critère comme une somme d'agrégation. Ce chapitre sera consacré principalement à la résolution bi-objectif (au sens de Pareto) du VRPTW-SP à l'aide de méthodes appartenant à la deuxième grande famille des méthodes approchées, à savoir, celles basées sur une population de solutions (décrites au chapitre 2). Les méthodes que nous proposons ici, en plus des composants des algorithmes génétiques multi-objectif, intègrent différentes stratégies de diversification par l'introduction de nouveaux mécanismes d'évolution à l'algorithme *NSGAII* "*Non-Dominated Sorting Genetic Algorithm*". Cette dernière méthode a été introduite pour la première fois dans (Deb et al., 2002). Il s'agit d'un des algorithmes multi-objectif les plus utilisés, notamment à cause de ses bonnes performances, et de sa facilité d'adaptation à la plupart des problèmes combinatoires. Il reprend le schéma d'un algorithme génétique classique, à savoir un cycle de sélection et un cycle de reproduction. Le choix de cette approche s'appuie sur les bons résultats qu'elle apporte selon la littérature sur l'optimisation des problèmes combinatoires multi-objectif en général et sur les problèmes de tournées de véhicules, en particulier : aux problèmes de tournées de véhicules avec équilibrage des durées par Jozefowiez *et al.* (2006), aux problèmes de tournées sur arcs, présentés dans Lacomme *et al.* (2006) où plusieurs procédures de recherches locales ont été testées et comparées. Plus récemment, Velasco *et al.* (2012) ont également adapté ces mêmes techniques pour résoudre

le problème de collecte et de livraison à deux objectifs, soit, minimiser les coûts de transport et maximiser la satisfaction des demandes les plus urgentes. Plus de détails sur les problèmes de tournées de véhicules multi-objectif peuvent être trouvés dans Jozefowiez *et al.* (2008) et Labadie & Prodhon (2014).

Le reste de ce chapitre se décompose comme suit : la définition du problème étudié est rappelée en section 6.2. Les sections (6.3) et (6.4) sont consacrées au développement des méthodes multicritères proposées, adoptant chacune une stratégie d'évolution spécifique, enrichies par l'ajout de différentes recherches locales. Plusieurs emplacements possibles de ces dernières dans la structure de chaque métaheuristique sont évaluées. Les pseudos codes des différentes versions sont détaillés dans la section (6.5). Les performances de chaque méthode sont mesurées à travers les résultats expérimentaux présentés en section (6.6), le chapitre se termine par quelques remarques et conclusions en section (6.7).

6.2 Rappel du problème et des notions de base

Rappelons que le problème étudié dans cette thèse (VRPTW-SP) consiste à déterminer l'ensemble des tournées réalisées par l'ensemble de véhicules (soignants) disponibles, depuis le dépôt initial en servant chacun un sous ensemble de clients (patients). Chaque client demande un ou plusieurs services, réalisés par des soignants qualifiés. Enfin, chaque soignant doit retourner au dépôt final à la fin de sa tournée. La route de chaque véhicule ne doit pas excéder la durée imposée par sa fenêtre de disponibilité, et les fenêtres relatives aux clients doivent être respectées. Comme nous l'avons déjà mentionné, la particularité de ce problème réside dans le fait qu'un client peut demander plusieurs services simultanément ou dans un ordre de priorité.

Les contraintes de ce problème sont les suivantes :

— Chaque véhicule k (soignant) doit commencer sa tournée depuis le dépôt initial, après α_k et doit impérativement retourner au dépôt final avant β_k.
— Les demandes de tous les clients doivent être satisfaites.
— Chaque client i doit être servi entre $[a_i, b_i]$ et par le véhicule qualifié pour effectuer le service demandé.
— L'écart imposé entre les services à synchroniser doit être respecté.

L'objectif de cette optimisation est de minimiser les coûts de déplacement des soignants tout en maximisant la satisfaction des patients, se traduisant par la minimisation des non-préférences des patients envers les soignants. Ici, la formulation mathématique de la fonction objectif considérée est la suivante :

$$\min \quad Z = (Z_1, Z_2) \tag{6.1}$$

où,

$$Z_1 = \min \sum_{e \in E} \sum_{k \in K} C_e.x_{ek} \tag{6.2}$$

$$Z_2 = \sum_{i \in N} \sum_{j \in V \setminus \{0\}} \sum_{k \in K} Pref_{ik}.x_{ijk} \tag{6.3}$$

La fonction $Z = (Z_1, Z_2)$ est donc considérée comme un vecteur à deux dimensions, où les deux objectifs sont à optimiser simultanément (i.e. avec le même degré d'importance). Si la comparaison de deux solutions est naturelle dans le cadre de l'optimisation mono-objectif, elle est toutefois plus complexe lorsque plusieurs objectifs sont considérés. Pour cela, des règles dites de dominance sont mises en place. La plus courante est certainement la dominance au sens de Pareto.

Soit Ens l'ensemble des solutions réalisables d'un problème donné, u et v sont deux solutions appartenant à Ens, et $f = (f_1, f_2, ..., f_c) : Ens \rightarrow R^c$ une fonction vectorielle à optimiser.

Les définitions suivantes sont données pour expliquer la notion de dominance au sens de Pareto.

Définition 1 : La solution u est dominée par v au sens strict ($u \prec v$) si et seulement si : $\forall i \leq c, f_i(v) < f_i(u)$.

Définition 2 : La solution u est dominée par v au sens faible ($u \preceq v$) si et seulement si : $\forall i \leq c, f_i(v) \leq f_i(u)$ et $\exists j \neq i \leq c, f_j(v) < f_j(u)$.

Définition 3 : Les solutions u et v sont dites incomparables ou non-dominées ($u \sim v$) si et seulement si : $\forall i \leq c$, si $f_i(v) < f_i(u)$ alors $\exists j \neq i : f_j(v) \geq f_j(u)$

Ces règles permettent donc de définir trois zones de l'espace des objectifs relatives à une solution u : (1) la zone de dominance contenant l'ensemble des solutions dominant u, (2) la zone de préférence contenant l'ensemble des solutions dominées par u et enfin (3) la zone des solutions incomparables à u.

Définition 4 : La solution u est dite optimale si et seulement si $\nexists v \in Ens, u \preceq v$. On dit alors que la solution u est non-dominée ou Pareto-optimale.

Définition 5 : L'ensemble $Ens^* \subseteq Ens$ est l'ensemble des solutions non-dominées, $\forall v \in Ens^*, \nexists u \in Ens, v \preceq u$. Les solutions appartenant à Ens^* forment le front de Pareto optimal.

L'optimisation multi-objectif se basant sur la dominance de Pareto se fait à l'aide d'algorithmes d'optimisation semblables à ceux utilisés dans l'optimisation mono-objectif. On retrouve des algorithmes évolutionnaires tels que les algorithmes génétiques ou des recherches locales. Cependant, dans un contexte multicritère, le but d'un algorithme multi-objectif n'est pas de fournir une seule solution de bonne qualité, mais plutôt de s'approcher le plus possible du front Pareto optimal. Il existe de nombreux algorithmes dédiés à l'optimisation multi-objectif, les plus courants étant les algorithmes évolutionnaires. Leurs performances varient selon le nombre d'objectifs à traiter ainsi que la nature de l'espace des solutions.

Dans cette thèse, nous nous intéresserons à deux algorithmes évolutionnaires à savoir : l'algorithme génétique *NSGAII* et l'algorithme génétique à redémarrage multiples, appelée *MS-NSGAII* pour *Multi-Start-NSGAII*, qui est une extension du premier. Une méthode de voisinage multi-objectif appelé *NSILS* pour *Non-Dominated Sorting Iterated Local Search* est également proposée. Ces algorithmes sont décrits dans les sections suivantes.

6.3 Algorithme génétique multi-objectif

L'algorithme génétique *NSGAII* a été introduit pour la première fois par Deb *et al.* (2002). Ces principales caractéristiques sont les suivantes :

- Une phase de décomposition de la population en fronts de Pareto successifs. Chaque front est donc composé d'un ensemble d'individus de même rang, ils appartiennent à la zone des solutions incomparables entre elles du point de vue de la dominance de Pareto. Le premier front contient tous les individus non-dominés de la population courante, le second contient les individus uniquement dominés par des individus du premier front, etc. De façon générale, les individus d'un front i seront dominés par des individus contenus dans les fronts précédents. Cette décomposition interviendra dans la sélection des individus pour la reproduction, qui permettra la transition à la prochaine itération de l'algorithme.
- La sélection pour la reproduction consiste en l'insertion successive des fronts de la population courante dans la prochaine population jusqu'à atteindre exactement la taille requise. Il est courant que le dernier front à insérer contienne plus d'individus qu'il n'en faut, et par conséquent il faut procéder à une sélection au sein du front en question. Deb *et al.* (2002) proposent l'utilisation d'une mesure de distance dite

crowding pour procéder au choix des individus à insérer. Les priorités sont alors mises sur les deux points extrêmes du front, puis sur les individus classés selon leur espacement dans l'espace des objectifs. L'objectif de cet opérateur de *crowding* est d'avoir une idée de comment les solutions sont proches les unes des autres au sein d'un même front.

Le *NSGAII* travaille sur une population de solutions, il fonctionne de la façon suivante : sa population R_t obtenue à la fin de l'itération t est composée de deux sous-populations P_t et Q_t de taille NP. La taille de R_t est donc $2NP$. Elle est obtenue en combinant la population P_t qui correspond aux parents avec la population Q_t qui correspond aux enfants.

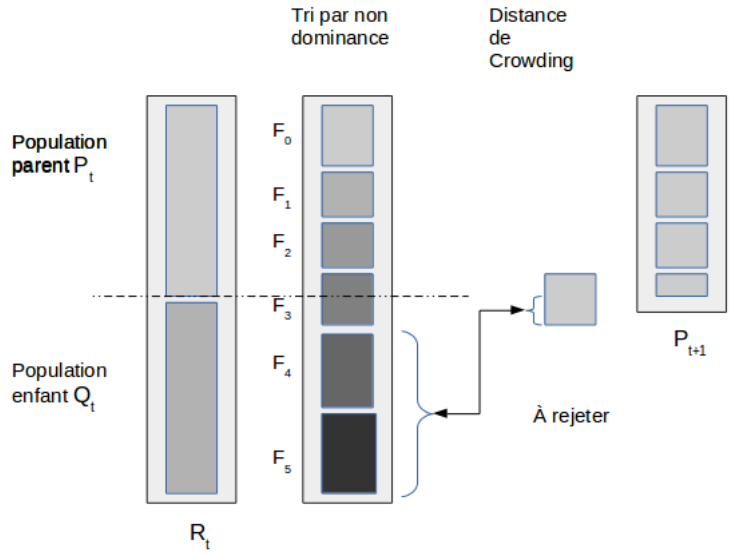

FIGURE 6.1 – Mise à jour de la population

La figure 6.1 conçue par Deb *et al.* (2002), illustre le déroulement d'une génération. Soit t l'indice d'une population, R_t est donc triée selon le critère de non-dominance. Les solutions du front 0 sont évidemment d'une meilleure qualité et doivent être conservées dans la population suivante (P_{t+1}). Toutefois, si le nombre de solutions dans le front 0 est plus grand que NP, seule une partie du front est maintenue. En revanche, dans le cas où le nombre de solutions dans le front 0 est strictement inférieur à NP, la population P_{t+1} doit être complétée par les solutions des fronts suivants. Dans les deux cas, le choix se base sur la mesure de distance dite *crowding* prise dans l'ordre décroissant. Considérons nos deux objectifs pour le VRPTW-SP et un front F, la distance de *crowding* de la solution k se calcul comme suit :

$$Crowding\ distance(F(k)) = \frac{Z_1^{pred(k)} - Z_1^{suc(k)}}{Z_1^{max} - Z_1^{min}} + \frac{Z_2^{suc(k)} - Z_2^{pred(k)}}{Z_2^{max} - Z_2^{min}} \tag{6.4}$$

$suc(k)$ et $pred(k)$ représentent respectivement le successeur et le prédécesseur de la k^{eme} solution du front trié (voir la figure 6.2). Z_c^{max} (respectivement Z_c^{min}) correspondent à la pire (respectivement meilleure) valeur de l'objectif c, où $c = 1, 2$ dans notre cas. La distance de *crowding* des points extrêmes dans chaque front est fixée à l'infini.

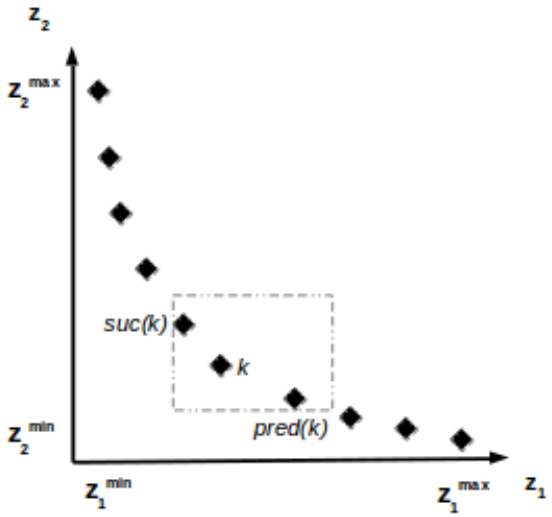

FIGURE 6.2 – Distance de *crowding*

La structure générale du *NSGAII* est la suivante. A chaque génération, une population initiale (P_t) composée de NP individus est générée et triées selon le critère de dominance. A chaque itération t, une population enfant (Q_t) contenant également NP individus est reproduite. Pour obtenir cette population (Q_t), des paires de parents P_1, P_2 sont sélectionnés par la méthode du tournoi binaire. Ensuite, un opérateur de croisement est appliqué aux deux parents choisis pour obtenir deux enfants qui sont ajoutés à (Q_t). Enfin, les populations P_t et Q_t sont fusionnées pour obtenir une population mixte (R_t) de taille $2NP$ $(R_t = P_t \cup Q_t)$. Cette dernière est triée par la suite (selon le critère de dominance) et seules les premières NP solutions sont retenues dans P_{t+1} pour la prochaine itération. Ce processus est répété NG fois.

6.4 Les principales composantes des algorithmes proposés

6.4.1 Codage en chromosomes

Le codage des solutions en chromosomes est une tâche particulièrement délicate dans la conception d'un algorithme génétique en général. En effet, le codage doit être représentatif pour toutes les solutions faisables du problème étudié. Dans notre *NSGAII*, un chromosome est codé comme un vecteur de tournées où chaque tournées contient les clients apparaissant dans l'ordre dans lequel ils sont visités par le véhicule associé. Les délimiteurs sont utilisés pour identifier les tournées.

Le tableau 6.1 représente un exemple d'un chromosome à trois tournées. La première est composée du client 1 suivi par le client 3 ; la deuxième tournée est composée du client 4, suivi par le client 7 qui est suivi par le client 2 et enfin la dernière tournée est composée du client 6 suivi du client 5 (0 correspond au délimiteur de tournées). Étant Donné que le nombre de véhicules utilisés dans le VRPTW-SP est fixe, chaque vecteur a une taille de $(|K + N| + 1)$ où K correspond à la flotte de véhicules disponibles et N à l'ensemble de clients à visiter.

0	1	3	0	4	7	2	0	6	5	0

TABLE 6.1 – Exemple d'un chromosome

6.4.2 Population initiale

La génération de la population initiale se fait grâce à l'adaptation de l'heuristique constructive parallèle randomisée (HCP) proposée dans le chapitre 4. Rappelons que cette méthode utilise une stratégie de construction parallèle pour générer une solution de départ. Ici, NP solutions sont générées en phase de création de la population parent. Dans ce chapitre, la première fois que l'heuristique (HCP) est appelée, la liste des clients candidats est restreinte à 1, i.e. pendant cette phase de construction, la première solution construite est obtenue de façon déterministe. Autrement dit, le choix du client à insérer correspond au meilleur selon l'ordre croissant des dates d'arrivées au plus-tôt. Ensuite, à chaque génération (de 2 à NG), la taille de la liste des clients candidats est fixée à $n_p = 4$, afin d'obtenir des solutions différentes.

Amélioration de la population initiale Pour le VRPTW-SP, nous proposons de démarrer l'évolution à partir d'une population initiale de bonne qualité. Pour cela, la recherche locale suivant une stratégie de descente aléatoire a été utilisée. Une fois la

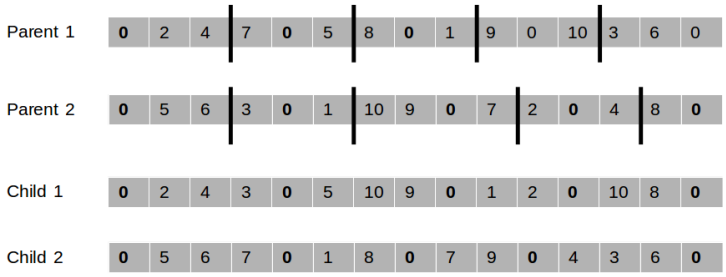

FIGURE 6.3 – Exemple de croisement

population initiale est générée, toutes les solutions sont soumises à la recherche locale. A ce stade, le concept de dominance de Pareto n'est pas encore exploité. La recherche locale (DVA) présentée dans le chapitre 4 est donc employée pour améliorer la population parent avec comme objectif de minimiser la fonction $(Z_1 + Z_2)$. Après recherche locale, le tri par non-dominance est effectué pour attribuer les rangs aux solutions obtenues.

6.4.3 Processus évolutionnaire

6.4.3.1 Sélection et diversification par croisement dans les GA

Dans les algorithmes génétiques (GA) proposés ici, la sélection des parents pour la reproduction se fait par tournoi binaire. Cette méthode consiste à choisir aléatoirement deux parents dans la population et à garder le meilleur des deux comme premier parent (selon le critère de dominance). Autrement dit, la solution appartenant au plus petit front est retenue. Cependant, lorsque les deux solutions appartiennent au même front, la meilleure solution serait celle correspondante à la plus grande valeur du *crowding*. Cette opération est répétée pour choisir le second parent. Les deux parents sont ensuite croisés en utilisant un opérateur de croisement. Diverses techniques de croisement existent pour le problème de tournées de véhicules en général. Néanmoins, la structure complexe de notre problème nécessite la conception d'un croisement spécifique.

Le croisement ordonné (OX) pour *Ordered Crossover* est adapté à notre problème comme suit : un point de coupure est choisi aléatoirement pour chaque tournée de chaque chromosome sélectionné. Ensuite, les deux chromosomes sont combinés tournée par tournée pour obtenir deux enfants. La figure 6.3 illustre un exemple de croisement. Les traits verticaux correspondent aux points de coupures.

Le fait de croiser les individus par tournée permet de conserver la contrainte liée aux services (qualification des véhicules). Les enfants issus de cette reproduction ne respectent pas forcément toutes les contraintes exigées (fenêtres de temps, synchronisation, etc). Par

conséquent, une procédure de réparation est nécessaire afin de re-basculer sur l'espace des solutions réalisables. Cette procédure de réparation réalisée en trois étapes est employée sur l'enfant une fois la combinaison des deux parents est terminée :

- Suppression des doublons : certains clients peuvent appartenir à deux tournées différentes. Dans ce cas, l'une des deux copies est retenue aléatoirement.
- Mise à jour des horaires de visite : la violation des fenêtres de temps et des contraintes de synchronisation est interdite. Cependant, lorsque l'horaire de visite d'un client donné ne respecte pas sa fenêtre de disponibilité, ce dernier est tout simplement retiré de la solution. D'autres cas d'insatisfaction des fenêtres temporelles peuvent être identifiés. Par exemple, un véhicule peut se retrouver avec une surcharge, i.e. sa durée de travail maximale est excédée. Dans ce cas, les clients provoquant la violation de cette contrainte sont éliminés et ceci en partant du dernier client de la tournée correspondante et en remontant dans l'ordre anti-chronologique, et en supprimant à chaque fois le dernier client s'il cause un excès.
- Réparation de la synchronisation : les horaires de visite des clients à synchroniser doivent également être mis à jour. Pour cela, on vérifie deux conditions : (1) que les services synchronisés ne se trouvent pas dans des positions contradictoires ; (2) que les horaires de visite sont cohérents par rapport à l'écart exigé (gap). Dans le cas où ces deux conditions ne sont pas vérifiées, les clients correspondants sont supprimés et les horaires de services des clients affectés par cette suppression sont mises à jour.

Notons que les tests de faisabilité proposées en chapitre 4 sont adaptées aux spécificités de ce chapitre. À la fin de ces étapes de suppression, les solutions non réalisables sont corrigées par l'application des méthodes de réparation utilisées dans les chapitres précédents, qui consistent à insérer les clients non affectés (voir chapitre 4).

Remarque. Étant donné que le nombre de tournées existantes est lié au nombre de véhicules disponibles (qui est limité à |K|), le nombre de points de croisement est égal au nombre de véhicules utilisés. Dans le cas où une tournée donnée est vide (ce qui signifie que la tournée n'est pas ouverte pour le véhicule correspondant), la tournée de l'enfant issu d'une telle recombinaison sera une copie de la tournée du parent (i.e. une tournée vide).

6.4.3.2 Diversification par ILS

En raison de la complexité des contraintes du VRPTW-SP étudié dans cette thèse, nous proposons également une autre méthode de diversification de la population parent sans avoir recours au croisement et par conséquent, la procédure de réparation n'est plus nécessaire.

Une fois un parent choisi pour la reproduction (avec une probabilité donnée), une

métaheuristique de type ILS est appliquée de la manière suivante : une recherche locale est d'abord appliquée (cette procédure sera détaillée dans la section 6.4.4) dans le but d'obtenir un premier optimum local. Ensuite, le cycle constitué de perturbation et de recherche locale est répété jusqu'à ce qu'un nombre d'itérations (max_P) soit atteint. A chaque itération, un critère d'acceptation de la solution (voir la sous-section 6.4.4.1) est appliqué et la meilleure solution est mise à jour dans le cas où une amélioration est trouvée. A la fin, la solution résultante est considérée comme une solution enfant. Cette méthode nous permet une diversification et intensification suffisante de la population parent tout en maintenant la faisabilité des solutions grâce à la perturbation utilisée.

Cette dernière repose sur une procédure de permutation aléatoire. Un nombre fixe (C_{max2}) de clients sélectionnés aléatoirement sont déplacés de leurs positions actuelles à d'autres positions réalisables.

Précisons que, la perturbation proposée dans le chapitre 5 pour résoudre le problème mono-objectif au sein de la métaheuristique ILS, n'est pas envisageable, car le but de l'introduction de l'ILS dans le schéma de l'algorithme génétique multi-objectif est de maintenir la faisabilité des individus. Or, cette dernière consistait à détruire une partie de la solution par la suppression d'un certain nombre de clients, puis la réparer en les réinsérant ailleurs. C'est pourquoi, nous choisissons de réaliser uniquement des permutations réalisables.

6.4.4 Recherches locales

Dans les algorithmes génétiques, la recherche locale est appliquée avec une probabilité donnée sur les solutions générées par croisement (cette notion est détaillée en section 6.4.4.3), généralement, on parle alors d'algorithmes mémétiques. La recherche locale est exécutée ici directement sur les chromosomes. Chaque itération de la recherche locale teste les 4 mouvements suivants : $S\_Relocate$, $S\_Exchange$, $M\_Relocate$, $M\_Exchange$ proposés dans le chapitre 4 et adaptés au cas multicritères. Notons que la stratégie d'exploration de ces voisinages est encore une fois la descente à voisinage aléatoire. Les mouvements de recherche locale peuvent être élaborés en utilisant le concept de "*première*" ou encore "*meilleure*" amélioration. Dans cette étude, les tests préliminaires ont montré que la recherche du meilleur mouvement possible nécessite beaucoup plus de temps avec un gain limité en comparaison avec une stratégie qui consiste à retenir la *"première amélioration"*. C'est pourquoi, cette dernière a été sélectionnée. L'exploration des mouvements s'arrête lorsque tous les mouvements sont explorés sans qu'aucune amélioration ne soit possible.

6.4.4.1 Critères d'acceptation

Dans cette section, nous définissons deux critères d'acceptation ($LS1$ et $LS2$) où $LS2$ est utilisée comme post-optimisation à la fin de l'application de $LS1$.

Dans ce qui suit, $Z_1(Sol)$ et $Z_2(Sol)$ représentent respectivement le coût de déplacement et la valeur de non-préférence d'une solution Sol et w est un nombre réel compris entre $[0, 1]$.

Les deux stratégies de comparaison de solutions sont définies ainsi :

$LS1$: c'est la dominance au sens de Pareto : ont dit que Sol' est meilleure que Sol si

$(Z_1(Sol') - Z_1(Sol) \leq 0)$ *et* $(Z_2(Sol') - Z_2(Sol) < 0)$ ou

$(Z_1(Sol') - Z_1(Sol) < 0)$ *et* $(Z_2(Sol') - Z_2(Sol) \leq 0)$.

$LS2$: c'est un critère basé sur la combinaison linéaire convexe des fonctions-objectifs. Elle est appliquée uniquement sur les solutions du premier front. Un mouvement est alors accepté lorsque son application permet d'améliorer la fonction suivante :

$w.(Z_1(Sol') - Z_1(Sol)) + (1 - w).(Z_2(Sol') - Z_2(Sol)) < 0$ où $w \in [0, 1]$.

Les poids doivent être indiqués pour pouvoir appliquer $LS2$. Pour cela, nous considérons la technique proposée par Murata *et al.* (2003), où les auteurs ont utilisé l'équation 6.5 dans laquelle Z_c^{min} et Z_c^{max} sont respectivement les valeurs minimales et maximales correspondants au critère c (avec $c = 1, 2$).

$$\boldsymbol{w} = \frac{\frac{Z_1(Sol) - Z_1^{min}}{Z_1^{max} - Z_1^{min}}}{\frac{Z_1(Sol) - Z_1^{min}}{Z_1^{max} - Z_1^{min}} + \frac{Z_2(Sol) - Z_2^{min}}{Z_2^{max} - Z_2^{min}}} \tag{6.5}$$

La figure 6.4 conçue par Murata *et al.* (2003) illustre les directions divergentes des solutions dans le premier front d'une population donnée suite à l'application de $LS2$.

6.4.4.2 Stratégies de recherche locale

Un moyen classique pour ajouter une recherche locale dans un algorithme génétique consiste à l'appliquer sur certaines solutions de la population enfant avec une certaine probabilité donnée. Différents emplacements pour la recherche locale pourraient être choisis. L'option la plus facile est d'ajouter la procédure de recherche locale après l'opérateur de croisement. Dans ce qui suit, cet emplacement de la recherche locale est retenu pour $LS1$.

Une deuxième option est également envisagée dans ce chapitre pour les méthodes basées sur le GA (*NSGAII* et *MS-NSGAII*). Elle consiste à ajouter un appel à $LS2$ pour les solutions du premier front, une fois que la condition d'arrêt de l'algorithme est remplie (i.e. utiliser $LS2$ comme une post-optimisation sur les solutions du premier front de la dernière et meilleure population obtenue). En d'autres termes, au cours de toutes les itérations de

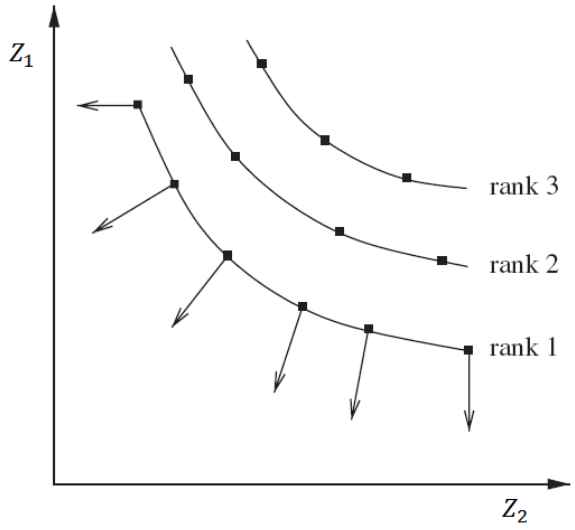

FIGURE 6.4 – Exemple de direction de descente de $LS2$

l'algorithme génétique $LS1$ est appelée (avec une certaine probabilité), puis, une procédure de recherche locale basée sur $LS2$ est appliquée sur les solutions du premier front de la population finale.

Dans cette étude, d'autres combinaisons possibles ont été testées. Par exemple, ajouter $LS2$ après la phase de croisement, puis considérer $LS1$ comme post-optimisation, ... Les différents tests démontrent que le critère d'acceptation basé sur la dominance au sens de Pareto est plus efficace que celui basé sur la combinaison convexe des fonctions objectifs. Ainsi, l'option d'inverser les positions de ces deux critères n'est pas prometteuse. Ceci justifie le choix fait pour le GA.

Concernant l'utilisation de l'ILS comme critère de diversification, nous avons fais le choix, après des tests préliminaires, de conserver $LS1$ comme stratégie d'exploration du voisinage au cours des différentes itérations de l'ILS. En effet, l'impact de l'application de $LS2$ comme post optimisation n'a pas été remarquable.

6.4.4.3 Fréquence d'appels à la recherche locale

La procédure de recherche locale peut être introduite à différents endroits dans l'algorithme génétique multi-objectif. Le tableau 6.2 résume les versions utilisées dans nos différents algorithmes proposés dans la section suivante.

Dans les algorithmes génétiques, la recherche locale n'est pas appliquée sur toutes les solutions de la population enfant, mais uniquement sur certaines d'entre elles. La structure générale de nos algorithmes est conçue de telle sorte qu'une itération (génération) permet de produire une population enfant de NP individus. Une fois cette population est créée, la recherche locale peut être introduite selon deux fréquences : la première est systématique, i.e.

à chaque itération, la recherche locale est employée. Néanmoins, seulement un pourcentage d'individus sur les NP solutions de la population enfant subissent cette procédure. La seconde est périodique (seulement pour le GA), i.e. pour chaque nb itérations, la recherche locale est appelée. De même, un pourcentage des individus sur les NP solutions de la population enfant sont soumis à la recherche locale.

Le dernier point important concerne le choix de ces individus. Nous pourrons penser que le fait de sélectionner les meilleurs individus permettent une meilleure convergence. Néanmoins, le critère principal des algorithmes évolutionnaires concerne la diversification, ainsi des individus choisis aléatoirement pourraient bien être prometteurs. Pour cela, nous proposons deux critères de choix de solutions à améliorer : le premier est le choix aléatoire et le second consiste à trier la population enfant concernée selon le critère de dominance au sens de Pareto et les solutions sont donc choisies selon cet ordre. Dans l'algorithme NSILS, la procédure d'ILS est appliquée systématiquement sur un nombre d'individus donnés choisis aléatoirement.

Il est important de noter que pour le génétique, la recherche locale n'est pas appliquée à toutes les nouvelles solutions, pour deux raisons : a) la recherche locale est très coûteuse en temps de calcul, b) comme le croisement utilisé est capable de changer le bassin d'attraction, la reproduction de nouvelles solutions (de qualité moyenne) peut apporter une autre source de diversification.

9 versions sont élaborées pour les deux approches basées sur le principe génétique (*NSGAII-v* et *MS-NSGAII-v* avec v correspondant au numéro de la version considérées (soit $v \in \{1, ..., 9\}$)). Une version est également conçue pour la méthode *NSILS*. Notons que, les deux dernières colonnes sont relatives à l'application de $LS1$. LS2 est toujours appliquée sur F_0 et à la fin de la procédure.

TABLE 6.2 – Versions proposées pour GA

Méthode	Version	RL	Post-Opt	Fréquence	Choix
NSGAII et MS-NSGAII	1	-	-	-	-
NSGAII et MS-NSGAII	2	LS1	-	Périodique - nb = 10	50% Aléatoire
NSGAII et MS-NSGAII	3	LS1	LS2	Périodique - nb = 10	50% Aléatoire
NSGAII et MS-NSGAII	4	LS1	-	Périodique - nb = 10	50% Dominance
NSGAII et MS-NSGAII	5	LS1	LS2	Périodique - nb = 10	50% Dominance
NSGAII et MS-NSGAII	6	LS1	-	Systématique	10% Aléatoire
NSGAII et MS-NSGAII	7	LS1	LS2	Systématique	10% Aléatoire
NSGAII et MS-NSGAII	8	LS1	-	Systématique	10% Dominance
NSGAII et MS-NSGAII	9	LS1	LS2	Systématique	10% Dominance
NSILS		LS1	-	Systématique	10% Aléatoire

6.5 Algorithmes

Maintenant que tous les composants nécessaires sont détaillés, cette section est consacrée à la description algorithmique des trois métaheuristiques utilisant les différentes idées proposées précédemment.

6.5.1 NSGAII

Le premier est l'algorithme génétique multi-objectif *NSGAII*, auquel une procédure de recherche locale est ajoutée afin d'améliorer sa performance. Sa structure générale est donnée dans l'algorithme (6.1).

Algorithme 6.1 Non-Dominated Sorting Genetic Algorithm : NSGAII

$P''_0 \leftarrow$ Population initiale de NP solutions
$P'_0 \leftarrow DVA(P''_0)$
$P_0 \leftarrow$ Tri_Rang(P'_0)
$i = 1$
Tant que ($i \leq NG$) **Faire**
  $Q_i \leftarrow \emptyset$
  $R_i \leftarrow \emptyset$
  Pour (j = 1 to NP/2) **Faire**
    $Select(P_1, P_2)$
    $(E_1, E_2) \leftarrow Croisement(P_1, P_2)$
    $Q_i \leftarrow Q_i \cup (E_1, E_2)$
  Fin Pour
  Si (Appel périodique) **Alors**
    Si $((i\ mod\ nb) = 1)$ ou $(i = 1)$) **Alors**
      $LS1(\%Q_i)$ // Appel périodique
    Fin Si
  Sinon
    $LS1(Q_i)$ // Appel systématique
  Fin Si
  $R_i \leftarrow Fusion(P_0, Q_i)$
  $P_0 \leftarrow$ Tri_Rang(R_i)
  $i = i + 1$
Fin Tant que
$LS2(\%F_0(P_0))$ // post-optimisation (cas des versions 3, 5, 7 et 9)

6.5.2 MS-NSGAII

Pour diversifier la recherche, une idée est de redémarrer le *NSGAII* à partir d'une autre population initiale au lieu de perdre du temps dans des générations improductives, donnant ainsi ce que nous appelons *NSGAII* à démarrage multiples (*MS-NSGAII*) pour *Multi-Start NSGAII*. Sa structure générale est donnée dans l'algorithme (6.2).

Algorithme 6.2 Multi-Start Non-Dominated Sorting Genetic Algorithm : MS-NSGAII

1: $j = 1$
2: **Tant que** $(j \leq nbstart)$ **Faire**
3: $P_0'' \leftarrow$ Population initiale de NP solutions
4: $P_0' \leftarrow DVA(P_0'')$
5: $P_j \leftarrow$ Tri_Rang(P_0')
6: $BestPop \leftarrow \emptyset$
7: $P_j \leftarrow NSGAII$
8: **Si** $(j = 1)$ **Alors**
9: $BestPop \leftarrow P_j$
10: **Sinon**
11: $BestPop \leftarrow Fusion(P_j, BestPop)$
12: **Fin Si**
13: $BestPop \leftarrow$ Tri_Rang$(BestPop)$
14: $j = j + 1$
15: **Fin Tant que**
16: $LS2(F_0(BestPop))$ // post-optimisation (cas des versions 3, 5, 7 et 9)

Les lignes de 1 à 22 de l'algorithme (6.1) sont répétées "*nbstart*" fois. C'est donc une généralisation de l'Algorithme 6.2. A la fin de chaque démarrage, la meilleure population est mise à jour en fusionnant la population courante avec la meilleure population sauvegardée dans *BestPop* (ligne 11). La population résultante est ensuite triée (selon le critère de non-dominance) et seule les NP premières solutions sont conservées (ligne 13). Pour tester le même nombre d'itérations que dans l'algorithme (6.1), une itération du *NSGAII* dans le *MS-NSGAII* se compose de $NG_1 = \frac{NG}{nbstart}$ générations.

6.5.3 NSILS

La troisième méthode est présentée par l'algorithme 6.3. Elle est obtenue en remplaçant la phase de (croisement + recherche locale) de l'algorithme 6.1 par la métaheuristique itérative (ILS) détaillée dans la sous-section 6.4.3.2.

Algorithme 6.3 Non-Dominated Sorting Iterated Local Search Algorithm : NSILS

$P_0'' \leftarrow$ Population initiale de NP solutions
$P_0' \leftarrow DVA(P_0'')$ //
$P_0 \leftarrow$ Tri_Rang(P_0')
$i = 1$
Tant que ($i \leq NG$) **Faire**
  $Q_i \leftarrow \emptyset$
  $R_i \leftarrow \emptyset$
  Pour (j = 1 to NP) **Faire**
    $E_j \leftarrow LS1(\%P_j)$
    Pour (k = 1 to max_P) **Faire**
      $E_j' \leftarrow Perturb(E_j, C_{max2})$
      $E_j' \leftarrow LS1(\%E_j')$
      Si ($E_j' < E_j$) **Alors**
        $E_j \leftarrow E_j'$
      Fin Si
    Fin Pour
    $Q_i \leftarrow Q_i \cup (E_j)$
  Fin Pour
  $R_i \leftarrow Fusion(P_0, Q_i)$
  $P_0 \leftarrow$ Tri_Rang(R_i)
  $i = i + 1$
Fin Tant que

6.6 Évaluations numériques

6.6.1 Implémentation et instances

Les techniques proposées dans ce chapitre ont été codées en C et exécutées sur un ordinateur avec un processeur Corei5 de 8 Go de RAM fonctionnant sous Linux. L'évaluation numérique a été réalisée sur l'ensemble d'instances décrit dans le chapitre 4.

6.6.2 Paramétrages des algorithmes

Des tests préliminaires nous ont permis de sélectionner la combinaison suivante de valeurs de paramètres pour les algorithmes génétiques : toutes les méthodes disposent d'une population de 100 individus. Le nombre de générations NG est fixé à 500 pour le *NSGAII*,

à 100 pour le *MS-NSGAII* avec 5 redémarrages (5 $\times$100 = 500 générations au total). Pour la méthode *NSILS*, le nombre de générations est égale à 50, le nombre d'itérations de la méthode ILS est fixé par $max_P = 20$. Enfin, le paramètre de la procédure de perturbation C_{max2} relatif au nombre de clients à échanger est fixé à 5. Le tableau 6.3 résume ces divers paramètres.

TABLE 6.3 – Paramètres fixés

Paramètre	Méthode	Nom	Valeur
Taille de la population	Commun	NP	100
Nombre de générations	NSGAII	NG	500
Nombre de générations	MS-NSGAII	NG_1	100
Nombre de redémarrages	MS-NSGAII	$nbstart$	5
Nombre de générations	NSILS	NG	50
Nombre d'itérations d'ILS	NSILS	max_P	20
Degré de perturbation	NSILS	C_{max2}	5

6.6.3 Métriques d'évaluation des performances

De nombreuses mesures permettant d'évaluer l'efficacité des méthodes d'optimisation multicritères existent dans la littérature. Dans ce chapitre, quatre métriques de comparaison sont utilisées pour comparer nos trois algorithmes multi-objectifs.

Deux catégories de métriques peuvent être identifiées. Celles mesurant la diversité et celles mesurant la convergence.

6.6.3.1 Mesures évaluant la diversité

1. La première est très courante, elle correspond aux nombres de solutions non-dominées, soit le nombre de solutions dans le premier front ($|F_0|$).
2. La deuxième est la métrique (*Spacing SP*), proposée par Schott (1995). Elle évalue la distance relative entre deux solutions consécutives obtenues dans l'ensemble des solutions non-dominées, comme suit :

$$SP = \sqrt{\frac{1}{|F_0|}\sum_{i=1}^{|F_0|}(d_i - \bar{d})^2} \tag{6.6}$$

Où $d_i = \min_{j \in F_0 : j \neq i} |[Z_1(i) - Z_1(j)]| + |[Z_2(i) - Z_2(j)]|$ et $\bar{d}$ est la valeur moyenne de la distance précédente $\bar{d} = \frac{\sum_{i=1}^{F_0} d_i}{|F_0|}$. Ainsi, la distance d_i est la valeur minimale de la

somme des différences absolues des valeurs des fonctions-objectifs entre la i^{me} solution et toutes les autres solutions de l'ensemble des solutions non-dominées. Il est à noter que cette distance calcule les écarts-types des différentes valeurs de d_i. Ainsi, si les solutions sont uniformément espacées, la distance correspondante sera faible. Donc, un algorithme pour lequel le front des solutions non dominées trouve un faible espacement est meilleur.

Il est à noter que les objectifs à comparer ici ne sont pas de même amplitude. Donc, pour ce calcul, les objectifs sont normalisés sur l'intervalle $[0, 1]$ avant le calcul. Ainsi, le résultat obtenu par cette métrique sera compris entre 0 et 1.

6.6.3.2 Mesures évaluant la convergence

1. La métrique d'hyper-volume (HV), introduite par Zitzler *et al.* (2003) est également utilisée pour évaluer nos résultats. Celle-ci calcule l'aire de la région multi-dimensionnelle séparant le front à évaluer et un point de référence (Rf). Les solutions à comparer ici, sont normalisées sur l'intervalle $[0, 1]$ avant le calcul. Le résultat de cette mesure est donc ramenée entre 0 et 1, ainsi une grande valeur implique une meilleure convergence.

   Cette mesure est calculée comme suit : pour chaque solution $i \in F_0$, un hyper-rectangle v_i est construit par rapport à un point de référence. Ce dernier est défini par les pires valeurs de Z_1 et Z_2 obtenues par toutes les versions testées. Ainsi, v_i correspond à l'aire séparant le point de référence et l'angle oblique de la i^{me} solution se trouvant dans F_0 (voir la Figure 6.5). L'union de tous les hyper-rectangles est calculée donnant ainsi la valeur de l'hyper-volume.

   Cette mesure nécessite le choix d'une borne supérieure (point de référence) de la région dans laquelle se trouve les points réalisables. Cependant, ce choix peut avoir un impact sur la valeur de l'hyper-volume.

2. La dernière métrique que nous proposons est une sorte de déviation à l'optimum, notée DO. Elle mesure l'écart relatif absolu par rapport aux solutions optimales trouvées par le solveur Cplex utilisé pour résoudre le modèle mono-objectif présenté dans le chapitre 4.

   L'écart est évalué pour une seule solution p dans F_0 qui est la plus proche de la solution optimale mono-objectif, en terme de distance euclidienne. Cette déviation est donnée en pourcentage pour chaque critère séparément et est définie par les équations suivantes (le symbole $(^*)$ désigne la solution optimale) :

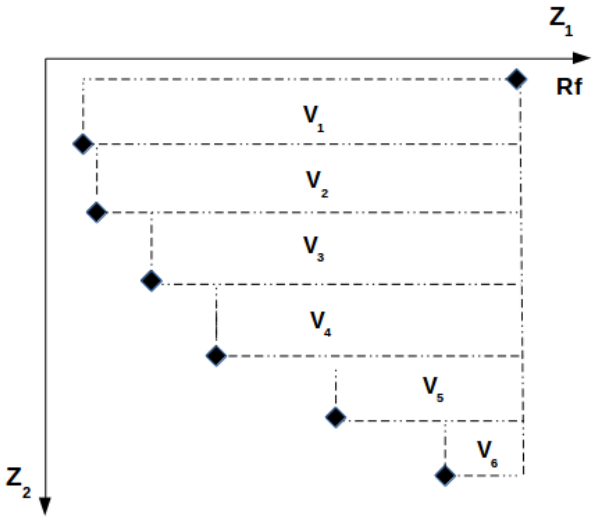

FIGURE 6.5 – Hyper-volume pour Z_1 et Z_2 du VRPTW-SP

$$DO_1(p) = ((Z_1^p - Z_1^*)/|Z_1^*|).100 \tag{6.7}$$

$$DO_2(p) = ((Z_2^p - Z_2^*)/|Z_2^*|).100 \tag{6.8}$$

Notons que cette métrique est évaluée seulement pour les petites instances et pour certaines des instances de taille moyenne, pour lesquelles la solution optimale a été trouvée (G_1 et quelques instances du G_2).

6.6.4 Résultats et interprétations

Dans cette section, nous présentons les résultats récapitulatifs obtenus par chaque algorithme pour les différents indicateurs de performance, sur l'ensemble des instances disponibles, sont fournis dans le tableau 6.4. Chaque instance a été résolue 5 fois pour chaque version et les résultats des différentes métriques ainsi que les temps de calcul Tps donnés en secondes sont reportés dans ces tableaux, où l'on peut trouver les valeurs minimales (min), maximales (max) et moyennes (moy) de chaque indicateur pour chaque algorithme. Nous présentons d'abord le récapitulatif de la solution de départ (qui est commune pour toutes les versions, puisque la taille de la population est la fixe. Cette dernière nous permet de mesurer l'impact du processus d'évolution des algorithmes proposés. Puis, les résultats relatifs aux algorithmes 6.1, 6.2 et 6.3 sont donnés dans cet ordre.

Le premier point remarquable en vue de ces résultats est relatif à l'impact du processus évolutionnaire. En effet, l'amélioration est largement visible sur toutes les versions et pour

TABLE 6.4 – Résultats récapitulatifs du *NSGAII* sur les 37 instances

	Version	$\|F_0\|$			*HV*			*SP*			DO_1			DO_2			*Tps*		
		min	max	moy	min	max	moy	min	max	moy	min	max	moy	min	max	moy	min	max	moy
Initiale		3,30	7,52	5,38	0,69	0,89	0,79	0,09	0,37	0,21	18,50	35,36	26,45	36,33	57,32	46,85	3,04	3,42	3,19
NSGAII	1	20,54	34,10	27,23	0,90	0,94	0,93	0,03	0,09	0,06	1,90	5,99	3,93	4,88	10,87	7,90	55,75	57,47	56,66
	2	18,51	29,09	23,41	0,93	0,95	0,94	0,04	0,10	0,07	0,05	2,98	1,14	1,70	5,29	3,40	144,81	164,97	154,95
	3	20,63	31,69	26,26	0,93	0,95	0,94	0,05	0,09	0,07	-0,12	2,31	1,17	1,89	6,13	3,77	145,15	165,36	155,31
	4	18,96	28,97	24,06	0,93	0,95	0,94	0,05	0,10	0,07	-0,08	2,20	0,86	1,88	4,81	3,25	139,31	160,45	149,50
	5	21,43	31,89	26,91	0,93	0,95	0,94	0,04	0,09	0,06	0,01	3,14	1,40	1,65	5,42	3,35	139,67	160,86	149,86
	6	20,52	29,66	25,18	0,93	0,95	0,94	0,05	0,09	0,07	-0,19	1,70	0,60	1,89	4,42	3,01	220,89	262,04	240,09
	7	22,74	33,82	28,28	0,93	0,95	0,94	0,04	0,09	0,06	0,21	2,02	1,19	2,04	4,88	3,18	221,25	262,47	240,49
	8	21,17	31,43	25,92	0,93	0,95	0,94	0,05	0,09	0,06	-0,37	2,15	0,65	1,15	3,77	2,49	209,95	247,86	230,58
	9	23,30	32,73	28,07	0,93	0,95	0,94	0,04	0,09	0,06	0,25	2,63	1,20	1,61	4,11	2,70	210,36	248,20	230,94
$MS-NSGAII$	1	12,72	23,54	18,16	0,88	0,95	0,92	0,05	0,12	0,08	2,30	8,03	5,39	7,11	15,03	10,91	69,01	70,12	69,53
	2	11,09	17,79	14,37	0,93	0,95	0,94	0,07	0,17	0,11	0,16	4,17	1,92	2,31	6,52	4,69	177,50	184,83	181,30
	3	10,77	18,94	14,68	0,93	0,96	0,94	0,06	0,17	0,11	0,09	3,40	1,49	3,29	8,03	5,52	177,73	185,12	181,57
	4	11,48	18,08	14,44	0,93	0,95	0,94	0,06	0,15	0,11	0,18	4,17	2,08	1,47	6,09	3,76	166,23	176,34	172,14
	5	11,71	20,17	15,30	0,93	0,96	0,94	0,07	0,16	0,11	-0,27	3,32	1,50	2,78	7,38	4,89	166,47	176,58	172,39
	6	11,43	19,34	14,92	0,92	0,96	0,94	0,06	0,16	0,11	-0,41	3,57	1,47	1,95	7,41	4,23	258,57	272,63	265,95
	7	11,97	21,29	16,18	0,93	0,96	0,95	0,05	0,17	0,10	-0,77	2,68	1,04	2,70	8,23	5,20	258,80	272,83	266,18
	8	12,44	19,23	15,57	0,93	0,95	0,94	0,06	0,14	0,10	-0,38	2,72	1,09	1,84	5,95	4,12	207,32	221,24	215,66
	9	13,05	21,32	16,91	0,93	0,96	0,94	0,06	0,15	0,10	-0,37	2,75	1,09	2,33	6,63	4,50	207,43	221,34	215,77
NSILS	10	8,67	13,98	11,15	0,93	0,96	0,94	0,07	0,16	0,11	-1,05	3,72	0,90	5,94	10,19	8,02	110,89	121,60	117,53

tous les indicateurs de performance.

La deuxième remarque à tirer concerne l'impact de la recherche locale sur les résultats finaux. En effet, toutes les méthodes utilisant la recherche locale sont au moins aussi performantes que les versions 1 n'incluant pas cette dernière. Ceci est justifié principalement par les résultats de la métrique DO, où l'on voit clairement que les versions 1 ne convergent pas assez. Néanmoins, ces versions sans recherche locale, semblent très rapides (ce qui est logique, car la recherche locale est coûteuse en temps), et offrent au décideur un ensemble de solutions avec un bon espacement et un hyper-volume acceptable. En outre, l'avantage des versions sans recherche locale est remarquable pour F_0 et SP. Néanmoins, leur convergence en terme de DO n'est pas garantie, d'où l'intérêt d'introduire la recherche locale.

En terme de cardinalité de l'ensemble des solutions non-dominées $|F_0|$, globalement *NSGAII* obtient le plus de solutions dans le premier front pour pratiquement toutes les versions. Les valeurs moyennes de cette métrique sont comprises entre 11,15 pour *NSILS* et 28,28 pour $NSGAII - 7$. La meilleure performance est donc obtenue par cette dernière lorsque LS1 est employée systématiquement sur 10% des individus sélectionnés selon l'ordre de non-dominance. Pour cette version, $LS2$ est aussi considérée en post-optimisation.

La méthode à redémarrage multiples *MS-NSGAII* construit des fronts de taille moins importante, ce qui est logique puisque nous n'effectuons que 100 générations pour chaque démarrage, contrairement aux deux autres algorithmes pour lesquels le nombre de générations est de 500. Ceci nous conduit à la conclusion suivante : le concept de reproduction a un intérêt positif sur la taille de l'ensemble des solutions non-dominées. Plus d'itérations fournissent plus de solutions de bonne qualité. Ainsi, les solutions dans F_0 sont plus proches et ceci est en adéquation avec le nombre de solutions non dominées. Les temps de calcul sont légèrement plus important, néanmoins, cette méthode obtient des hyper-volumes au moins plus importants que les différentes versions du *NSGAII*. Ainsi, la métrique DO affirme également la bonne convergence du *MS-NSGAII*.

A son tour, *NSILS* construit des fronts encore plus petits et par conséquent l'espacement est moins bon. Ceci est dû en partie au fait que la recherche locale n'est appliquée que sur 10 % de la population. Néanmoins, cette méthode obtient un hyper-volume meilleur comparé aux autres versions en des temps beaucoup moins élevé, à savoir 110,89 secondes en moyenne pour NSILS v.s. 139,31 secondes pour NSGAII-4 (qui correspond à la version la plus rapide incluant la recherche locale après NSILS). Ainsi, on remarque que cette méthode a une meilleure convergence sur le critère de déplacement par rapport à toutes les autres méthodes, mais l'écart sur le critère de non préférence est détérioré.

De façon plus générale, nous pouvons dire que toutes les versions proposées ici offrent au décideur un ensemble de solutions avec un bon compromis entre les coûts de déplacement

et les valeurs de non-préférences. Ainsi, toutes les versions ont une bonne distribution dans l'ensemble des solutions non-dominées. Les meilleures performances sont obtenues par les versions *NSGAII-1* avec un espacement de 0,03 puis *NSGAII-2*, *NSGAII-5*, *NSGAII-7* et *NSGAII-9* avec un espacement de 0,04 et la pire performance est obtenue par *MS-NSGAII-2*, *MS-NSGAII-5* et *NSILS* avec une valeur d'espacement de 0,07. Par conséquent, par rapport à ces deux mesures de performance, l'ordre suivant peut être conclut : $NSGAII < MS\text{-}NSGAII \leq NSILS$.

Concernant la métrique HV, les meilleures performances sont comme signalées avant, obtenues majoritairement par *MS-NSGAII* et *NSILS* avec une valeur de 0,96, ainsi la pire évaluation est donnée par la version *NSGAII-1* et qui est égale à 0,94. Mais de façon globale, toutes les versions ont un hyper-volume acceptable et l'ordre suivant peut être déduit pour cette métrique : $NSILS \leq MS\text{-}NSGAII < NSGAII$.

Regardons à présent la dernière métrique proposée ici (à savoir DO), elle permet l'évaluation de la convergence de nos algorithmes par rapport aux solutions optimales. D'un côté, on remarque que toutes les versions avec recherche locale dominent complètement les deux versions n'incluant pas cette procédure. De l'autre côté, en comparant les différentes versions de chaque algorithme, on remarque que la plupart des versions (en particulier, les versions périodiques) considérant $LS2$ en post optimisation améliorent l'un des critères sans détériorer excessivement le second (qui reste acceptable) par rapport aux versions utilisant $LS1$ seulement. Puisque $LS2$ est basée sur la combinaison convexe des critères à optimiser, cela montre que les mouvements de recherches locales proposées sont bien adaptés aux deux critères à optimiser.

Concernant cette dernière métrique, nous pouvons proposer un ordre pour chaque critère. Globalement, *NSILS* obtient l'écart le plus faible par rapport au critère de déplacement. Aussi, *MS-NSGAII* a une meilleure convergence sur DO_1, comparée à *NSGAII*, lorsque la recherche locale est introduite périodiquement. En revanche, *NSGAII* est meilleure que *MS-NSGAII* sur DO_1, quand la recherche locale est appliquée systématiquement. Concernant à présent DO_2, *NSGAII* semble légèrement meilleure que *MS-NSGAII* qui est meilleure que *NSILS* pour toutes les versions.

Ainsi, l'ordre suivant peut être déduit : $NSILS < MS\text{-}NSGAII \leq NSGAII$ par rapport au critère de déplacement. Par contre, $NSGAII < MS\text{-}NSGAII < NSILS$ pour le critère des non préférences.

Jusque là, nous avons discuté l'impact du processus d'évolution de la population utilisée et l'impact de la recherche locale sur la diversité et la convergence des solutions. Maintenant, nous allons évaluer l'impact de la fréquence d'appel de la recherche locale sur les solutions obtenues. Pour toutes les versions, l'introduction de la recherche locale systématiquement

semble plus efficace en regardant le cardinal de F_0. Cet impact est visible aussi bien sur les versions avec ou sans post-optimisation. Par exemple, *NSGAII-2* obtient en moyenne les valeurs suivantes : $F_0 = 23{,}14$; $HV = 0{,}94$; $SP = 0{,}07$; $DO_1 = 1{,}14$ et $DO_2 = 3{,}4$. *NSGAII-6* obtient en moyenne les valeurs suivantes : $F_0 = \mathbf{25{,}18}$; $HV = 0{,}94$; $SP = 0{,}07$; $DO_1 = \mathbf{0{,}6}$ et $DO_2 = \mathbf{3{,}01}$. Les valeurs en gras désignent celles qui ont été améliorées. Ainsi, nous remarquons la même tendance sur pratiquement toutes les autres versions. De ce fait, on conclut que la recherche locale incluse systématiquement, bien qu'elle soit appliquée sur une population réduite (10 % au lieu de 50 % en périodique) reste plus performante.

Le dernier paramètre des algorithmes et qui mérite d'être discuté concerne le critère de sélection des individus qui subissent la recherche locale. Nous en avons proposé deux : aléatoire ou dominance. Les résultats obtenus permettent de distinguer dans certains cas le deuxième critère comme étant plus prometteur. Cette conclusion ne se généralise pas pour toutes les versions ni pour tous les indicateurs de performance, sauf pour la métrique $|F_0|$.

Enfin, afin de différencier qu'elle est la meilleure version, nous considérons tous les algorithmes à la fois et nous allons explorer les résultats de la métrique (DO). Nous avons comparer les différentes versions de sorte à déterminer celles qui ne sont dominées par aucune autre (i.e. celles qu'aucune autre version n'est meilleure qu'elle sur DO_1 et DO_2). Ainsi, nous avons déduit que *NSGAII-8*, *MS-NSGAII-6*, *MS-NSGAII-8* et *NSILS* ne sont dominées par aucune version.

Nous déduisons cependant, que la version 8 est la plus prometteuse quel que soit le processus d'évolution adopté, que la version 6 trouve sont intérêt en processus à redémarrage multiple et que *NSILS* est aussi efficace que le processus génétique.

Le dernier point à aborder en vue de cette comparaison concerne les temps d'exécution (Tps). Ce dernier dépend fortement des versions testées. Globalement, ces algorithmes nécessitent un temps de calcul relativement faible et se révèlent efficaces pour la résolution de la variante bi-objectif du VRPTW-SP. En effet, le classement suivant peut être considéré : *NSILS* est plus rapide que *NSGAII* qui est à son tour plus rapide que *MS-NSGAII*.

Évaluation graphique Les représentations graphiques sont bien adaptées pour visualiser le comportement des méthodes d'optimisation multi-objectif. L'impact de la procédure de recherche locale, peut être illustré pour certaines instances en comparant l'ensemble des solutions non-dominées donné par les versions *NSGAII-1* et *MS-NSGAII-1* (sans recherche locale) aux versions qui ont été identifiées comme les plus performantes (6 et 8).

La figure 6.6 illustre les résultats sur une moyenne instance à 45 clients et 10 véhicules pour *NSGAII-1*, *NSGAII-6*, *MS-NSGAII-1*, *MS-NSGAII-6* et *NSILS*. On remarque une grande proximité des fronts fournis par *NSGAII-6*, *MS-NSGAII-6* et *NSILS*, mais également

une nette distinction du front fourni par *MS-NSGAII-1*, qui est beaucoup plus loin. Notons que pour cet exemple, *NSGAII-1* se démarque positivement par rapport à *MS-NSGAII-1*. Enfin, comparés à la solutions initiales, la performance des différentes versions est nettement visible.

La figure 6.7 exprime les résultats sur la même instance à 45 clients et 10 véhicules pour *NSGAII-1*, *NSGAII-8*, *MS-NSGAII-1*, *MS-NSGAII-8* et *NSILS*. La même conclusion pour *MS-NSGAII-1* et *NSGAII-1* est maintenue. Ainsi, la dominance de *NSGAII-8* devient remarquable, suivie de *NSILS*, puis *MS-NSGAII-8*. Ensuite, les figures qui suivent comparent pour chaque algorithme, une fois les versions périodiques puis systématiques sur une autre instance de même taille.

La figure 6.8 illustre les fronts obtenus par *NSGAII-1*, *NSGAII-2*, *NSGAII-3*, *NSGAII-4* et *NSILS-5*. Pour cet exemple, l'ordre entre les versions incluant la recherche locale est le suivant : *NSGAII-4* et *NSGAII-5* semblent équivalentes. De même, *NSGAII-2* et *NSGAII-3* semblent aussi très proches. Néanmoins, les deux premières sont visiblement meilleures que les deux dernières. Le point commun entre les quatre versions est la fréquence d'appels à la recherche locale qui est introduite périodiquement. Ainsi, entre chaque paire, le point commun et le critère de sélection des individus à améliorer. Par conséquent, nous pouvons déduire que pour cet exemple le critère de sélection selon l'ordre de non dominance est plus performant qu'un simple choix aléatoire.

La figure 6.9 illustre les fronts obtenus par *NSGAII-1*, *NSGAII-6*, *NSGAII-7*, *NSGAII-8* et *NSILS-9*. Les fronts exposés dans cette figure s'écartent moins, néanmoins la même conclusion que la précédente peut être tirée. *NSGAII-6* et *NSGAII-7* se voient équivalentes. De même pour *NSGAII-8* et *NSGAII-9*.

La figure 6.10 illustre les fronts obtenus par *MS-NSGAII-1*, *MS-NSGAII-2*, *MS-NSGAII-3*, *MS-NSGAII-4* et *MS-NSILS-5*. Dans cette figure, *MS-NSGAII-2* et *MS-NSILS-3* se montrent proches. C'est le cas aussi de *MS-NSGAII-4* et *MS-NSGAII-5*. Ainsi, les deux premières ont une convergence remarquable par rapport aux deux dernières. Ainsi, sur cette instance, c'est le choix aléatoire qui est plus prometteur dans ce cas.

La figure 6.11 illustre les fronts obtenus par *MS-NSGAII-1*, *MS-NSGAII-6*, *MS-NSGAII-7*, *MS-NSGAII-8* et *NSILS-9*. Pareil qu'avant, les fronts sont peu distingués mais la même conclusion sur l'équivalence des versions *MS-NSGAII-6* et *MS-NSGAII-7*. Par contre *MS-NSGAII-8* semble légèrement meilleure que *MS-NSGAII-9* dans ce cas.

Enfin, les différentes figures viennent confirmer les précédentes remarques. L'impact de la recherche locale est démontré. Ainsi, la différence entre les différentes versions incluant la recherche locale reste faible, il semble d'ailleurs difficile d'identifier la meilleure stratégie. Néanmoins, la dominance et la convergence de toutes les versions sont très visibles.

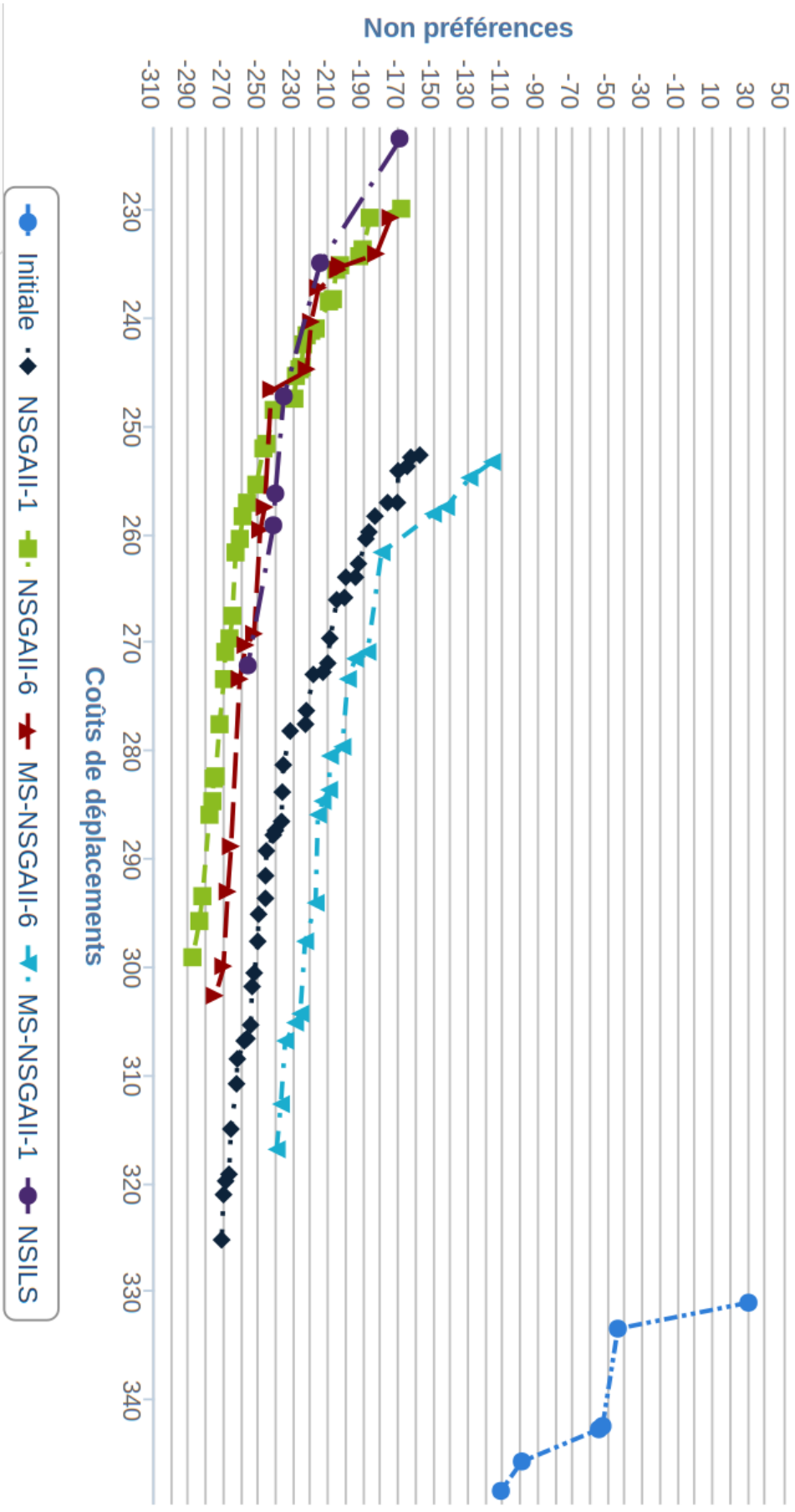

FIGURE 6.6 – Fronts des solutions non-dominées pour l'instance 18 - partie 1

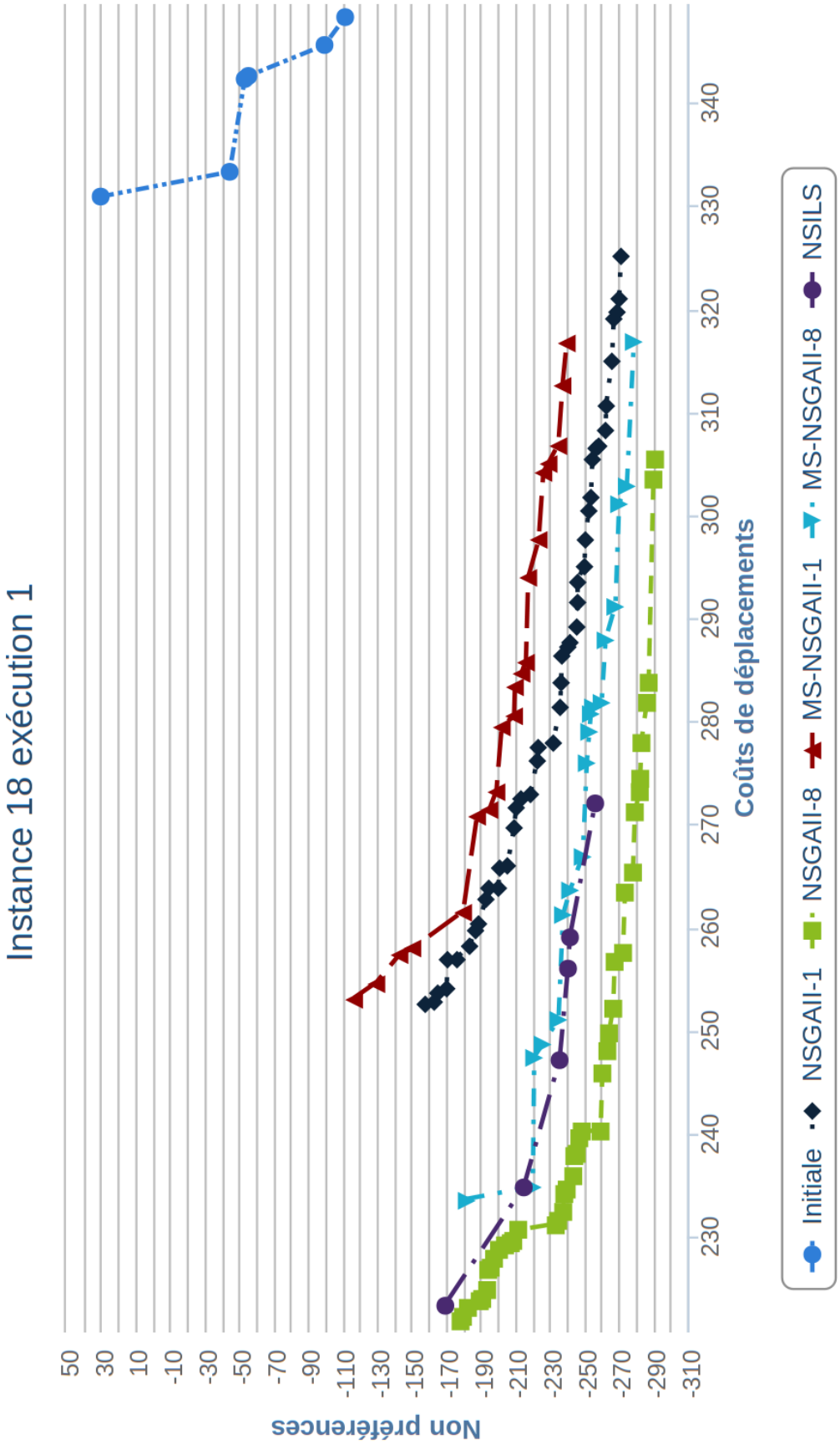

FIGURE 6.7 – Fronts des solutions non-dominées pour l'instance 18 - partie 2

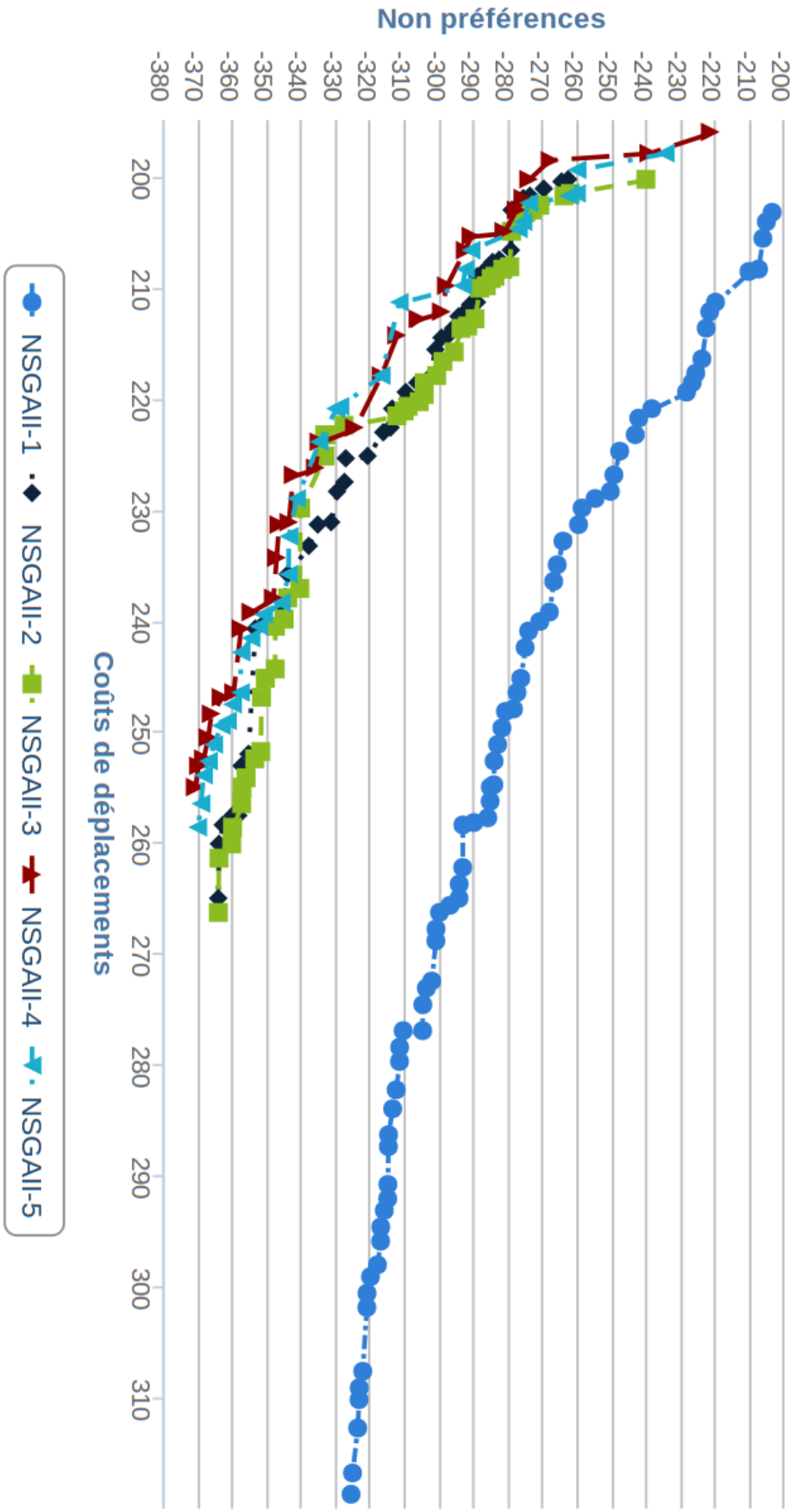

FIGURE 6.8 – Fronts des solutions non-dominées pour l'instance 25 - partie 1

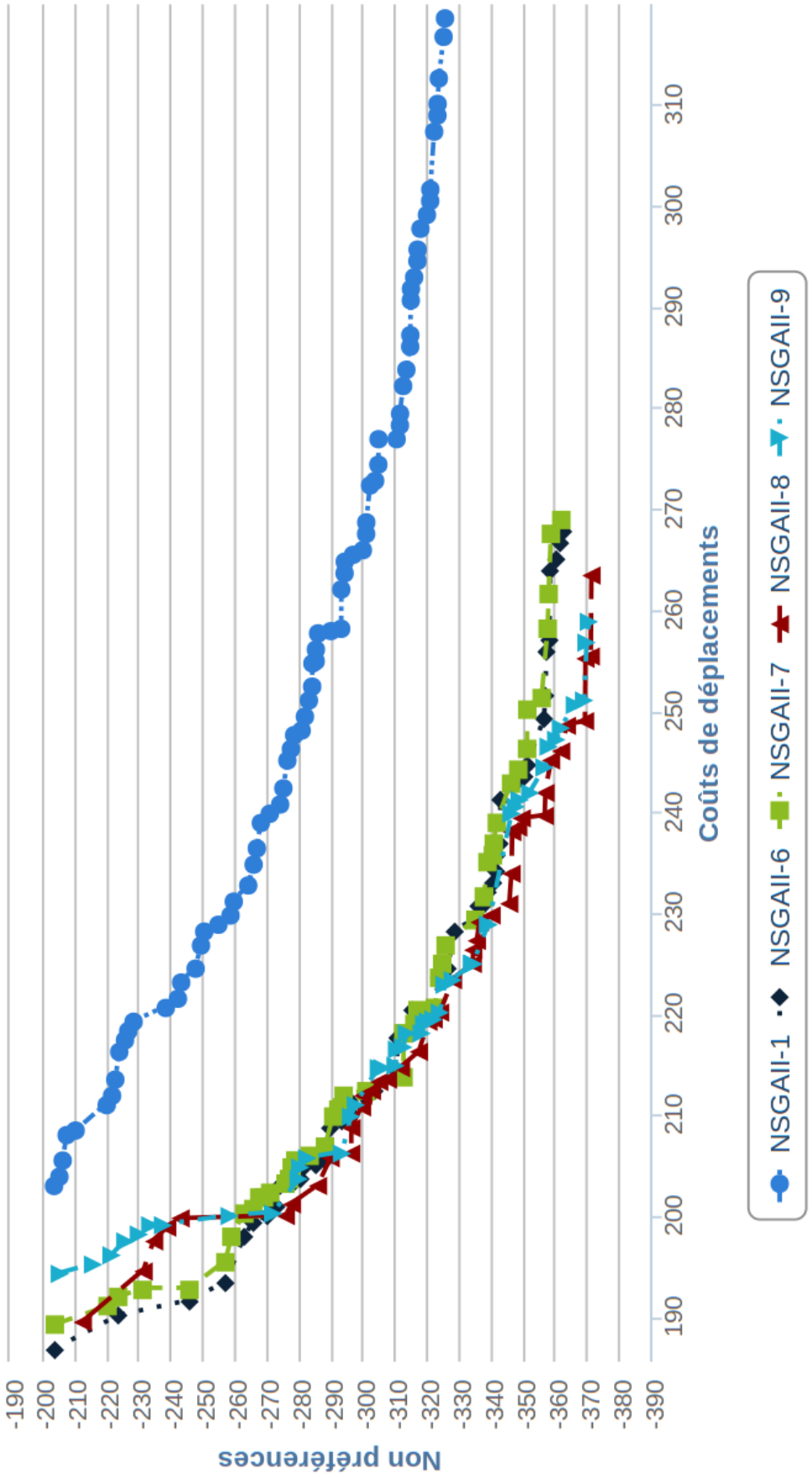

FIGURE 6.9 – Fronts des solutions non-dominées pour l'instance 25 - partie 1

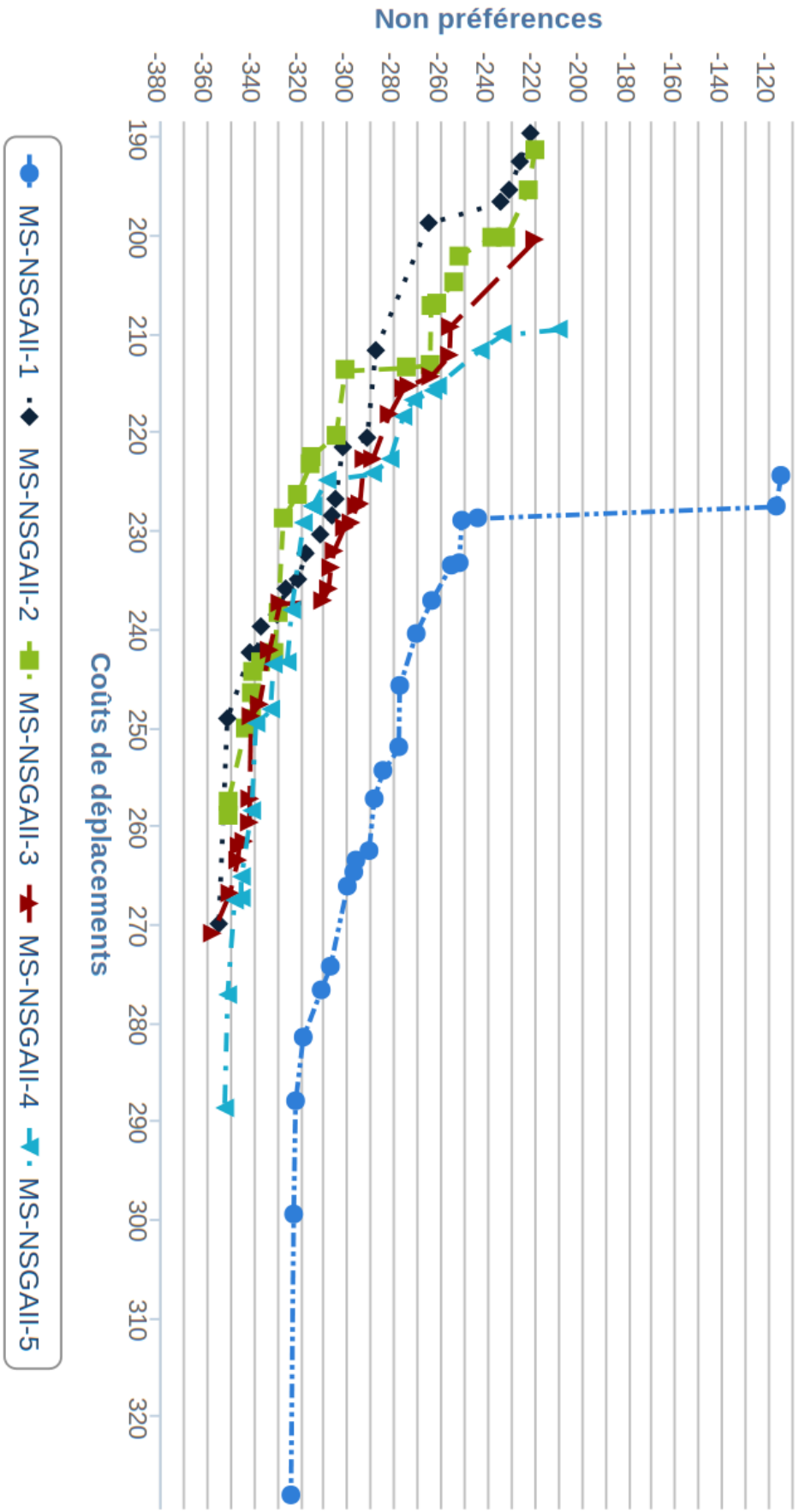

FIGURE 6.10 – Fronts des solutions non-dominées pour l'instance 25 - partie 1

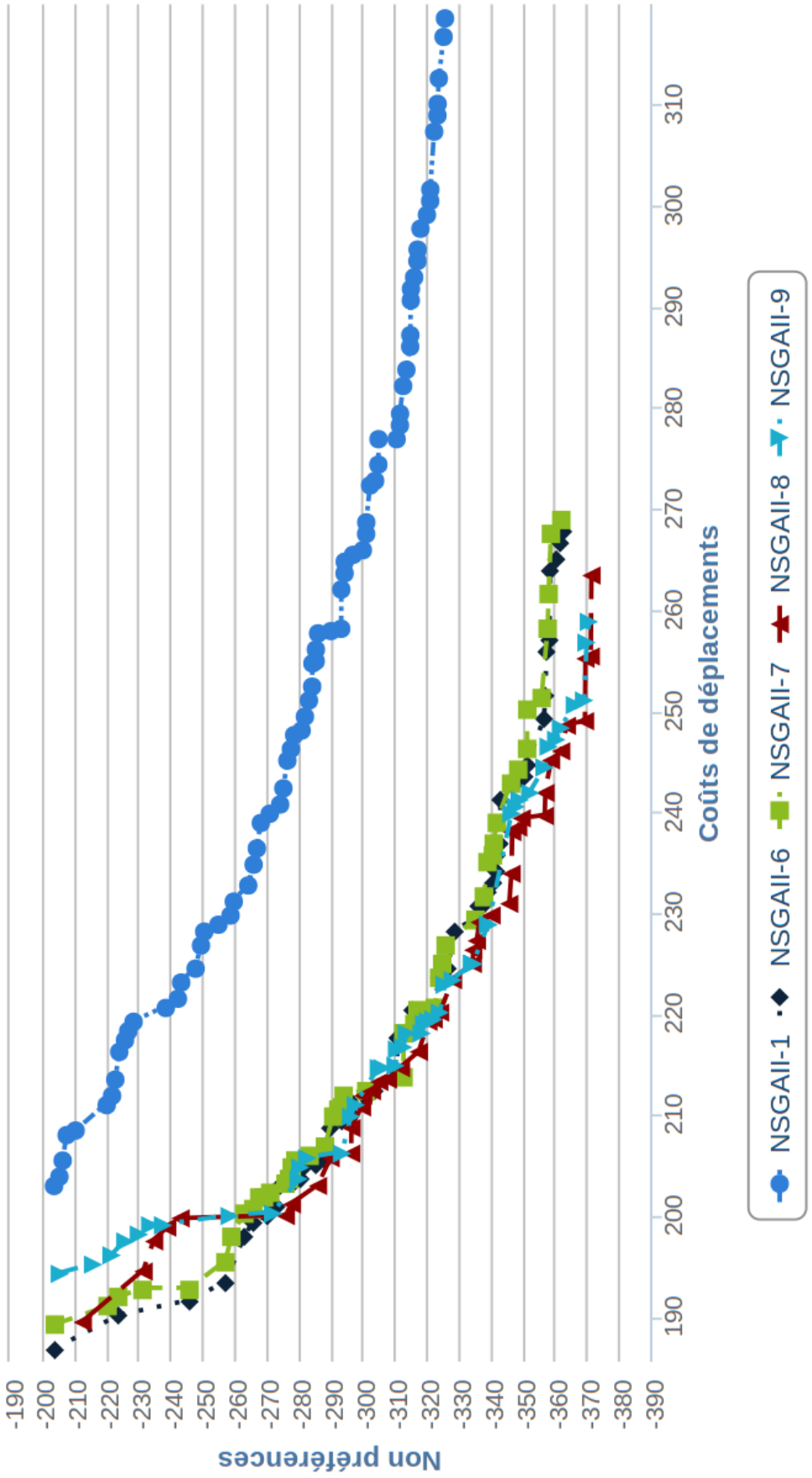

FIGURE 6.11 – Fronts des solutions non-dominées pour l'instance 25 - partie 1

6.7 Conclusion

Dans ce chapitre, des méthodes évolutionnaires basées sur le concept de non-dominance de Pareto sont proposées pour résoudre le VRPTW-SP. Plusieurs stratégies et fréquences d'appels à la recherche locale ont été discutées. Le critère d'optimisation du VRPTW-SP consiste à minimiser les coûts de déplacement des soignants (Z_1) utilisés pour visiter un ensemble de clients ayant des préférences à chaque soignant. Les valeurs relatives à leur non-satisfaction doivent également être minimisées (Z_2).

Différentes techniques d'évaluation de ces deux critères d'optimisation ont été comparées entre elles. Un nouveau mécanisme de reproduction a été également adapté, remplaçant le processus de croisement+recherche locale par une métaheuristique de type ILS. Le but était d'éviter de générer des enfants non réalisables, c'est pourquoi des techniques de perturbation garantissant la faisabilité des enfants ont été utilisées comme facteur de diversification. Nous avons également testé une généralisation du *NSGAII*, appelée *MS-NSGAII* qui consiste à redémarrer la reproduction à partir de populations parents indépendantes.

Les tests expérimentaux ont été effectués sur les 37 instances adaptées de celles proposées par Bredström & Rönnqvist (2008). Pour évaluer la performance des algorithmes, nous avons utilisés plusieurs métriques connues, qui comparent principalement les solutions non-dominées obtenues. En effet, l'efficacité des algorithmes proposés a été confirmée pour le VRPTW-SP, l'intérêt de la procédure de recherche locale est évident, et les temps de calcul restent raisonnables. Enfin, les méthodes proposées résolvent efficacement le bi-objectif VRPTW-SP, offrant au décideur un large choix de solutions de bonne qualité.

Les différentes approches proposées ici ont été présentées d'abord lors de la conférence internationale "*The 8th IFAC Conference on Manufacturing Modelling, Management and Control (MIM 2016)*" (Ait Haddadene *et al.* (2016b)), puis une version détaillée a été soumise à une revue internationale, qui est en cours d'évaluation.

Chapitre 7

Extension périodique du VRPTW-SP

7.1 Introducion

Dans les chapitres précédents, nous nous sommes intéressés à l'aspect mono-période du VRPTW-SP. Ce problème a été traité dans un premier temps, au moyen des métaheuristiques itératives mono-objectif et dans un second temps, à l'aide des métaheuristiques à base de population considérant le cas multicritère.

Cependant, dans un contexte d'HAD, le VRPTW-SP est souvent soulevé sur un horizon de planification. Le but de ce chapitre est donc d'étendre le problème de planification de tournées journalières sur un horizon de planification H. Le but serait par exemple d'élaborer un plan de visite des patients sur une semaine de travail ($H = 7$). D'ailleurs, lors de notre analyse de la littérature sur les problèmes de tournées dans le domaine de la santé, nous avons constaté que les contraintes liées à la planification sur un horizon étendu sont peu considérées. Avant de présenter le problème et les méthodes de résolution proposées, il est important de bien se situer par rapport aux travaux existants. Pour cela, la section 7.2 explique les raisons qui nous ont motivé à considérer cette variante du problème. Ensuite, une description formelle est donnée dans la section 7.3, induisant l'écriture du modèle mathématique. La section 7.4 est consacrée à l'approche de résolution proposée. Enfin, le chapitre se conclut par la section 7.5. Notons que ce chapitre propose de détailler une de nos perspectives.

7.2 Situation/Motivation

Dans cette section, nous proposons, de compléter la classification des travaux existants, proposée dans le chapitre 3 par les articles identifiés dans le contexte des soins à domicile, ayant étudié le cas périodique du problème. Pour cela, 6 références sont ajoutées, ainsi le tableau 7.1 rappelle les critères de comparaison considérés, auxquels s'ajoute le critère de

TABLE 7.1 – Rappel des critères de comparaison

Critères	Abbr.	Description
	T	Temps (Déplacement, Attente, Heures Supplémentaires, etc.)
	C	Coûts (Déplacement, Attente, Affectation, etc.)
	D	Distance parcourue
	PP	Préférence des patients
	PS	Préférence des soignants
	SR	Services réalisés
Contraintes		
	FT	Fenêtres de temps
	QS	Qualification des soignants
	Reg	Périodicité

TABLE 7.2 – Critère d'optimisation de la littérature

	T	C	D	PP	PS	NS
Steeg & Schröder (2008)	×	×		×		
Nickel *et al.* (2012)		×	×			×
Gamst & Jensen (2012)	×	×		×	×	
Maya Duque *et al.* (2015)				×		×
Allaoua *et al.* (2013)			×			
En-nahli *et al.* (2015)	×	×		×		

périodicité.

D'après cette classification (Tableau 7.2), on peut constater encore une fois que les temps, les coûts ainsi que les préférences sont les objectifs les plus considérés. Concernant les contraintes communes (Tableau 7.3), nous remarquons que tous les auteurs ont porté un intérêt aux contraintes temporelles par le respect des fenêtres de temps. Nous retrouvons également les contraintes liées aux qualifications des soignants (pour uniquement 2 articles). Cependant, à notre connaissance, les contraintes temporelles liées à la synchronisation simultanée et/ou la précédence qui sont communes dans les problèmes réels appliqués au domaine de la santé, n'ont pas été encore prises en compte. Ceci nous a donc motivé à aller dans cette direction et à considérer en plus des contraintes induites par la planification sur une journée (traitées dans les chapitres précédents), celles qui relèvent de la planification sur un horizon plus étendu.

Enfin, par rapport aux méthodes de résolution généralement utilisées dans la littérature, on peut constater (Tableau 7.4) que la plupart des contributions se situent dans la catégorie des méta-heuristiques. L'objectif ici serait de résoudre une combinaison de deux problèmes

TABLE 7.3 – Contraintes considérées

	FT	*QS*
Steeg & Schröder (2008)	×	
Nickel *et al.* (2012)	×	
Gamst & Jensen (2012)	×	×
Maya Duque *et al.* (2015)	×	
Allaoua *et al.* (2013)	×	×
En-nahli *et al.* (2015)	×	

TABLE 7.4 – Méthodes de résolution

	Solveurs Linéaires	Exactes	Approchées
Steeg & Schröder (2008)			×
Nickel *et al.* (2012)			×
Gamst & Jensen (2012)			×
Maya Duque *et al.* (2015)	×	×	
Allaoua *et al.* (2013)		×	×
En-nahli *et al.* (2015)	×		

connus pour être NP-difficiles, à savoir le VRPTW-SP auquel s'ajoute le problème de planification sur un horizon étendu.

Pour résoudre cette variante, nous proposons une approche qui se distingue des autres méthodes utilisées dans la littérature, en combinant une des méta-heuristique proposée précédemment pour le cas mono-période et la programmation mathématique.

7.3 Description et formulation mathématique

Dans cette section, nous proposons d'introduire une formulation mathématique pour le problème que nous appelons "P-VRPTW-SP pour *Periodic Vehicle Routing Problem with Time Window, Synchronization and Precedence constraints*". Plus précisément, le P-VRPTW-SP est défini sur un horizon H constitué de périodes (jours). Reprenons donc la définition du VRPTW-SP proposée dans le chapitre 4 en section 4.3, et introduisons des données complémentaires dans le tableau 7.5 permettant de proposer une formulation mathématique.

Certaines données méritent d'être détaillées, car elles sont spécifiques aux problèmes périodiques, notamment les combinaisons de jours et les fréquences de visites. Dans la réalité, ces informations sont fournies par un niveau de décision plus élevé, par exemple en fonction de l'état des patients dans le contexte des soins à domicile, lors de la visite du diagnostic. Des contraintes d'espacement entre les jours de visites sont alors spécifiées. Ici,

TABLE 7.5 – Données du problème

Nom	Définition
P	Horizon de planification
C	Nombre de combinaisons possibles
syn_i	Égale à 1 si le client $i \in N$ est un client synchronisé, 0 sinon
aff_{cp}	Égale à 1 si la période $p \in P$ est affecté à la combinaison $c \in C$, 0 sinon
$Freq_{is}$	Égale au nombre de visites pour le service $s \in S$ souhaitées par le client $i \in N$
$Combi_{is}$	Est l'ensemble de combinaisons de jours de visites possibles du client $i \in N$ demandant le service $s \in S$
Ps_i	Est l'ensemble de combinaisons de jours de synchronisations possibles pour le client $i \in N$ synchronisé
$MaxPref_i$	Le maximum sur les non-préférences du client $i \in N$

nous présentons ces espacements sous forme de combinaisons $Combi_{is}$ autorisées pour chaque service s demandé par chaque client i. Une combinaison est alors un ensemble constitué de $Freq_{is}$ de jours distincts. Par exemple, supposons que $Freq_{is} = 2$, que $C = 3$ et que $H = 7$ alors $Comb_{is}$ peut être un élément de l'ensemble suivant $\{\{1,3\},\{2,4\},\{5,7\}\}$.

Les instances étudiées dans cette thèse incluent 2 types de services, cependant, nous définissons les jours de synchronisation comme également un ensemble de combinaisons possibles Ps_i. Par exemple, soit i un client synchronisé demandant deux services différents soumis à une contrainte de synchronisation (à savoir, simultanée ou prioritaire), supposons que $C = 3$, $Freq_{i1} = 3$, $Freq_{i2} = 2$, $H = 7$, $Comb_{i1} = \{\{1,3,4\},\{2,4,6\},\{2,5,7\}\}$ et $Comb_{i2} = \{\{1,3\},\{2,6\},\{5,7\}\}$ alors les combinaisons de jours de synchronisation autorisées peuvent être les suivantes : $Ps_i = \{\{1,3\},\{2,6\},\{5,7\}\}$.

Le but cette étude est de déterminer un ensemble de jours de visites pour chaque type de service et pour chaque client, et un ensemble de tournées vérifiant les contraintes du VRPTW-SP sur chaque jour, de sorte à minimiser les coûts totaux sur l'horizon liés aux déplacements des soignants et les coûts relatifs au non respect de leurs affectations aux soignants préférés. Ce dernier consiste à minimiser la différence des valeurs de préférence du pire soignant et la valeur de préférence du soignant affecté. Ainsi, les contraintes du VRPTW-SP sont bien évidemment étendues aux cas périodique, par exemple la contrainte de satisfaction des demandes des clients se traduisent comme étant la satisfaction de la fréquence de visite. D'autres contraintes spécifiques au cas périodique sont également introduites et sont les suivantes :

— Chaque client est affecté à une et une seule combinaison de jours.
— Chaque client est visité à chaque période de la combinaison de jours qui lui sera affectée.
— Chaque client synchronisé reçoit l'ensemble des services synchronisés à chaque période de la combinaison de synchronisation qui lui sera affectée.
— Un client non synchronisé peut également recevoir différentes visites à la même

période, cependant ces dernières ne doivent pas se chevaucher.

Il s'agit donc de répondre à la question suivante : quand et à quelle moment les clients sont-ils visités ? et par quels soignants ? Le problème peut être formulé à l'aide d'un programme linéaire à variables mixtes entières, impliquant les variables de décision présentées dans le tableau 7.6.

TABLE 7.6 – Variables du problème

Nom	Définition
y_{ikp}	égale à 1 si le client $i \in N$ est visité par le véhicule $k \in K$ en période $p \in P$, 0 sinon
l_{ip}	égale à 1 si le client $i \in N$ reçoit un service synchronisé en période $p \in P$, 0 sinon
u_{isc}	égale à 1 si le client $i \in N$ est affecté à la combinaison $c \in C$ pour le service $s \in S$, 0 sinon
x_{ijkp}	égale à 1 si le véhicule $k \in K$ traverse l'arc $(i,j) \in E$ en période $p \in P$, 0 sinon
z_{kp}	égale à 1 si le véhicule $k \in K$ est utilisé en période $p \in P$, 0 sinon
$tdeb_{ikp}$	désigne l'heure à laquelle le véhicule $k \in K$ commence la réalisation du service demandé par le client i en période $p \in P$.

Ainsi, le problème peut s'écrire sous sa formulation mathématique comme suit :

$$\min \sum_{i \in V \setminus \{n+1\}} \sum_{j \in V \setminus \{0\}} \sum_{k \in K} \sum_{p \in P} C_{ij}.x_{ijkp} + \sum_{i \in N} \sum_{k \in K} \sum_{p \in P} (MaxPref_i - Pref_{ik}).z_{kp} \quad (7.1)$$

Sous les contraintes :

$$\forall i \in N, \forall s \in S, \quad \sum_{k \in K_s} \sum_{p \in P} y_{ikp} = Freq_{is} \quad (7.2)$$

$$\forall i \in N, k \in K, p \in P, \quad \sum_{j \in V \setminus \{0\}} x_{ijkp} = y_{ikp} \quad (7.3)$$

$$\forall k \in K, \forall p \in P, \quad \sum_{j \in N} x_{0jkp} = z_{kp} \quad (7.4)$$

$$\forall k \in K, \forall p \in P, \quad \sum_{i \in N} x_{in+1kp} = z_{kp} \quad (7.5)$$

$$\forall h \in N, \forall k \in K, \forall p \in P, \sum_{i \in V \setminus \{n+1\}} x_{ihkp} = \sum_{i \in V \setminus \{0\}} x_{hikp} \quad (7.6)$$

$$\forall k \in K, \forall p \in P, \quad \sum_{(i,j) \in E} x_{ijkp} \leq |V|.z_{kp} \quad (7.7)$$

$$\forall i \in N, \forall s \in S_i, \quad \sum_{c \in Combi_{is}} u_{isc} = 1 \quad (7.8)$$

$$\forall i \in N, \forall s \in S_i, \forall p \in P, \quad \sum_{k \in K_s} y_{ikp} = \sum_{c \in Combi_{is}} aff_{cp}.u_{isc} \quad (7.9)$$

$$\forall i \in N : syn_i = 1, \forall s \in S_i, \forall p \in P, \quad l_{ip} = \sum_{c \in Ps_i} aff_{cp}.u_{isc} \quad (7.10)$$

$$\forall i, j \in V, \forall s \in S_i \cap S_j, \forall k \in K_s, \forall p \in P$$

$$tdeb_{ikp} + (T_{ij} + D_{is}).x_{ijkp} \leq tdeb_{jkp} + b_i.(1 - x_{ijkp}) \quad (7.11)$$

$$\begin{aligned}&\forall i \in N, \forall s, r \in S_i : r \neq s, \forall k \in K_s, \forall v \in K_r, \forall p \in P,\\ &tdeb_{ikp} - tdeb_{ivp} + \beta_k.(2 - y_{ikp} - y_{ivp}) \geq D_{is}.z_{kp} - L.l_{ip} \text{ ou}\\ &tdeb_{ivp} - tdeb_{ikp} + \beta_k.(2 - y_{ivp} - y_{ikp}) \geq D_{ir}.z_{vp} - L.l_{ip}\end{aligned} \tag{7.12}$$

$$\forall i \in N, \forall s \in S_i, \forall k \in K_s, \forall p \in P, \quad a_i.y_{ikp} \leq tdeb_{ikp} \leq b_i.y_{ikp} \tag{7.13}$$

$$\alpha_k.z_{kp} \leq tdeb_{0kp} \leq \beta_k.z_{kp} \quad \forall k \in K, \forall p \in P, \tag{7.14}$$

$$\alpha_k.z_{kp} \leq tdeb_{n+1kp} \leq \beta_k.z_{kp} \quad \forall k \in K, \forall p \in P, \tag{7.15}$$

$$\begin{aligned}&\forall i \in N : syn_i = 1, \forall s, r \in S_i : r \neq s, \forall p \in P,\\ &(\sum_{k \in K_s} tdeb_{ikp} - \sum_{k \in K_r} tdeb_{ikp}) - (b_i.(1 - l_{ip}) \leq gap_{isr}\end{aligned} \tag{7.16}$$

$$\begin{aligned}&\forall i \in N : syn_i = 1, \forall s, r \in S_i : r \neq s, \forall p \in P,\\ &(\sum_{k \in K_s} tdeb_{ikp} - \sum_{k \in K_r} tdeb_{ikp}) - (b_i.(1 - l_{ip}) \geq -gap_{isr}\end{aligned} \tag{7.17}$$

$$\begin{aligned}&\forall i \in N : syn_i = 1, \forall s, r \in S_i : r \neq s, \forall p \in P,\\ &(\sum_{k \in K_s} tdeb_{ikp} - \sum_{k \in K_r} tdeb_{ikp}) - (b_i.(1 - l_{ip}) \geq gap_{isr} - M.var_i\end{aligned} \tag{7.18}$$

$$\begin{aligned}&\forall i \in N : syn_i = 1, \forall s, r \in S_i : r \neq s, \forall p \in P,\\ &(\sum_{k \in K_s} tdeb_{ikp} - \sum_{k \in K_r} tdeb_{ikp}) - (b_i.(1 - l_{ip}) \leq -gap_{isr} + M.(1 - var_i)\end{aligned} \tag{7.19}$$

$$\forall i, j \in V, \forall k \in K, \forall p \in P, \quad x_{ijkp} \in \{0, 1\} \tag{7.20}$$

$$\forall i \in V, \forall k \in K, \forall p \in P, \quad tdeb_{ikp} \geq 0 \tag{7.21}$$

$$\forall i \in V, \forall k \in K, \forall p \in P, \quad z_{kp} \in \{0, 1\} \tag{7.22}$$

$$\forall i \in N, \quad var_i \in \{0, 1\} \tag{7.23}$$

Sous cette formulation, la fonction objectif (7.1) reprend les coûts définis auparavant. Les contraintes (7.2) assurent que chaque client reçoit le nombre de visites souhaitées. Les contraintes (7.3 - 7.7) sont les contraintes de routage adaptées au cas périodique. Les contraintes (7.8) permettent d'affecter une et une seule combinaison de jour à chaque client. Les contraintes (7.9) assurent l'affectation des clients à chaque période de la combinaison choisie. Les contraintes (7.10) assurent l'affectation des visites synchronisées à chaque

période de la combinaison de synchronisation. Les contraintes (7.11) permettent la cohérence des instants de visites à chaque période. Ensuite, les contraintes (7.12) empêchent le chevauchement de deux visites. Ici, L est initialisé à $(\beta_k + D_{is} + 1)$. Les contraintes de ponctualité sont exprimées par (7.14 - 7.16). Enfin, les contraintes de synchronisation dans le cas périodique s'écrivent à l'aide des contraintes (7.17 - 7.19). Le reste des contraintes (7.20 - 7.23) fixent la nature des variables de décision.

7.4 Approche de résolution proposée

Comme nous l'avons signalé auparavant, le VRPTW-SP périodique combine deux grands problèmes connus pour être NP-difficiles. Les chercheurs se sont généralement appuyés sur les méthodes de résolution approchées afin de résoudre de tels problèmes. Dans ce chapitre, nous proposons une approche qui se distingue des méthodes existantes dans la littérature pour la résolution du P-VRPTW-SP. Nous proposons de combiner la programmation mathématique et les métaheuristiques, donnant les matheuristiques.

La matheuristique proposée ici est basée sur la décomposition du problème en deux. Le premier problème représente le VRPTW-SP sur une période, tel que nous l'avons introduit précédemment. Le deuxième est un problème d'affectation des demandes des clients aux périodes, en respectant les combinaisons de jours autorisées, il représente la partie planification périodique de notre problème. La matheuristique commence par affecter les personnels aux combinaisons de jours autorisées de manière optimale (sous-section 7.4.1), puis résout le VRPTW-SP à l'aide de l'une des métaheuristoques proposée dans les chapitres précédents (sous-section 7.4.2), en adaptant la fonction-objectif (la minimisation des valeurs de non préférences est remplacée par la minimisation du regret des patients envers les soignants). Une partie de la solution obtenue durant la deuxième phase est alors conservée pour démarrer la résolution du programme linéaire pour l'itération suivante. Le processus composé de "résolution du modèle mathématique + résolution du VRPTW-SP" est répété un certain nombre de fois et la meilleure solution est retenue à la fin.

7.4.1 Problème d'affectation

A cette étape, la résolution devrait répondre à la question suivante : quel client est visité en quelle période et par quel véhicule ? Ce problème peut s'écrire sous sa formulation mathématique à l'aide d'un programme linéaire à variables mixtes entières, impliquant les trois premières variables de décision présentées dans le tableau 7.6.

$$\sum_{i\in N}\sum_{k\in K}\sum_{p\in P}(MaxPref_i - Pref_{ik}).y_{ikp} \tag{7.24}$$

Sous les contraintes :

$$\forall i \in N, \forall s \in S, \quad \sum_{k\in K_s}\sum_{p\in P} y_{ikp} = Freq_{is} \tag{7.25}$$

$$\forall i \in N, \forall s \in S_i, \quad \sum_{c\in Combi_{is}} u_{isc} = 1 \tag{7.26}$$

$$\forall i \in N, \forall s \in S_i, \forall p \in P, \quad \sum_{k\in K_s} y_{ikp} = \sum_{c\in Combi_{is}} aff_{cp}.u_{isc} \tag{7.27}$$

$$\forall i \in N : syn_i = 1, \forall s \in S_i, \forall p \in P, \quad l_{ip} = \sum_{c\in Ps_i} aff_{cp}.u_{isc} \tag{7.28}$$

$$\forall i, j \in N : i \neq j, \forall s \in S : s \in S_i \text{ et } s \in S_j, \forall k \in K_s, \forall p \in P,$$
$$a_i + D_{is} - \beta_k.(2 - y_{ikp} - y_{jkp}) \leq b_j \tag{7.29}$$

$$\forall i \in N, \forall s \in S, \forall c \in Combi_{is}, \quad u_{isc} \in \{0,1\} \tag{7.30}$$

$$\forall i \in N, \forall p \in P, \quad l_{ip} \in \{0,1\} \tag{7.31}$$

$$\forall i \in N, \forall k \in K, \forall p \in P, \quad y_{ikp} \in \{0,1\} \tag{7.32}$$

Sous cette formulation, la fonction objectif (7.24) minimise le critère du regret maximum des clients envers les véhicules (soignants). Les contraintes (7.25 - 7.27) assurent l'affectation des demandes des clients aux périodes autorisées, aux véhicules qualifiés pour chaque type de service. L'affectation des visites synchronisées est également assurée par les contraintes 7.28. Plusieurs bornes inférieures peuvent être envisagées, nous proposons d'introduire les contraintes 7.29 permettant de vérifier la compatibilité de chaque paire de clients à affecter à la même tournée. Autrement dit, si i et j sont tous les deux affectés à la tournée du véhicule k en période p, dans le meilleur cas, le service chez l'un des clients commence en début de fenêtre de temps. Ainsi, l'ajout de la durée nécessaire pour la réalisation du service ne doit pas excéder l'heure de fin de disponibilité du deuxième client. Enfin, le reste des contraintes fixent la nature des variables de décision.

7.4.2 Résolution du VRPTW-SP

Dans notre démarche, les résultats de la planification des horaires de visites des clients par les soignants forment des ensembles de tournées, où chacune représente une journée de travail. Pour construire les tournées, nous proposons d'appliquer une des métaheuristiques proposées dans le chapitre 5. Deux approches semblent envisageables (ILS et GRASP × ILS) et sont discutées dans ce qui suit.

Notons que ces métaheuristiques doivent être appliquées sur une solution réalisable, c'est pourquoi nous proposons d'adapter les techniques utilisées en chapitre 6 pour la reconstruction d'un chromosome après le croisement (voir la section 6.4.3.1) pour former les tournées. Par exemple, à la fin de la résolution du PLME, nous pouvons tenter de construire les tournées pour chaque période ouverte, tout en satisfaisant les contraintes relatives au VRPTW-SP, mais en raison des contraintes temporelles, la faisabilité n'est pas garantie. Cependant, les clients ne satisfont pas ces contraintes doivent être affectés à d'autres tournées (véhicules) et/ou périodes (combinaisons).

7.4.2.1 ILS pour le P-VRPTW-SP

Rappelons que l'ILS démarre sa résolution d'une solution de bonne qualité, ainsi dans ce cas, à la fin de la résolution du programme linéaire, la recherche locale proposée en chapitre 4 peut être appliquée sur les périodes ouvertes (ou sur certaines périodes choisies aléatoirement). Ensuite, le processus composé de perturbation + recherche locale est répété un certain nombre de fois. Enfin, la meilleure solution obtenue est restituée.

7.4.2.2 GRASP × ILS pour le P-VRPTW-SP

La méthode GRASP × ILS proposée en chapitre 5 pourrait également être envisageable. Cette dernière peut être introduite ainsi : à la fin de la résolution du programme linéaire, GRASP × ILS est appelée sur chaque période (ou sur certaines périodes choisies aléatoirement). Rappelons que cette dernière est composée d'une phase de construction et une phase d'amélioration par application de la métaheuristique ILS. La méthode de construction d'une solution indépendante qui pourrait bien s'adapter dans ce cas, serait la destruction de la solution en changeant la combinaison de jours affectée à un certain nombre de clients choisis aléatoirement. C'est une adaptation de la procédure de perturbation proposée dans le chapitre 5 au cas périodique. Sa structure est la suivante, un client est sélectionné aléatoirement, et est affecté à une autre combinaison de jours. Ce processus peut être répété pour un certain nombre de clients. Le degré de perturbation appliqué ici, devrait être suffisamment grand afin de redémarrer d'une solution complètement différente et ainsi

augmenter les chances de tomber sur un minimum global. Ensuite, la phase d'amélioration consisterai à appliquer l'ILS proposée dans la sous-section précédente sur la nouvelle solution construite.

7.4.2.3 Procédure de perturbation dans l'ILS

Dans les deux métaheuristiques précédentes, une procédure de perturbation est appelée au cours des itérations de l'ILS. Cette dernière permet d'échapper au minimums locaux, ainsi, la perturbation utilisée dans le cas mono-période (présenté dans le chapitre 6) pourrait être appliquée à chaque période ou seulement sur certaines périodes. Dans ce cas, le niveau de perturbation est crucial : d'une part, une perturbation trop faible peut ne pas être suffisante pour basculer d'un bassin d'attraction à un autre ; d'autre part, une perturbation trop forte aura tendance à générer des solutions complètement indépendantes les unes des autres. C'est pourquoi la perturbation proposée en chapitre 6 qui consiste à appliquer un certain nombre de permutations aléatoires pour certaines périodes est appliquée ici. Notons que le fait de ne perturber que certaines périodes permet de limiter le niveau de perturbation, ainsi un bon ajustement de ce paramètre (nombre de périodes et/ou de clients à perturber) serait important.

Afin de redémarrer le modèle mathématique d'une solution différente (au deuxième appel de Cplex), la perturbation appliquée au cours de l'ILS ne doit pas se limiter aux périodes seulement. Une procédure perturbant les combinaisons de jours affectées aux clients est nécessaire. C'est pourquoi, durant les itérations de l'ILS, la procédure de perturbation utilisée dans la méthode GRASP $\times$ ILS et discutée en sous-section 7.4.2.2, permettant l'obtention d'une solution indépendante doit être introduite.

Nous proposons d'alterner entre ces deux techniques de perturbation afin d'avoir une diversification suffisante de la solution. Autrement dit, nous pouvons introduire la perturbation type "destruction/réparation" pour une certaine probabilité, par exemple, sur 10% des itérations de l'ILS et la perturbation type "permutation" sur le reste des itérations.

Algorithme 7.1 Matheuristique

```
1: Pour (i = 1 to maxIter) Faire
2:   Si (i = 1) Alors
3:     Sol ← Cplex(PLM)
4:   Sinon
5:     Sol ← Cplex(PLM, %BSol)
6:   Fin Si
7:   Pour (p = 1 to |P|) Faire
8:     Construction (Sol (p))      // VRPTW-SP
9:     Réparation (Sol (p))
10:    BSol ← Métaheuristique (Sol (p))
11:  Fin Pour
12: Fin Pour
```

7.4.3 Schéma général de la matheuristique

La structure générale de la mataheuristique proposée dans ce chapitre est récapitulée dans l'algorithme 7.1. À la première itération, $\%BSol$ n'est pas considérée et le programme linéaire (PLM) présenté en sous-section 7.4.1 est résolu de façon optimale. À la deuxième itération, une partie de la meilleure solution obtenue au cours de la résolution du VRPTW-SP à l'aide de l'une des métaheuristiques proposées ci-dessous est conservée. Autrement dit, au deuxième appel à la résolution exacte sous Cplex, un pourcentage du nombre de clients est fixé aux périodes obtenues au cours de la métaheuristque dans $BSol$. Les deux phases "construction et réparation" consisteraient à construire les tournées en respectant les différentes contraintes du VRPTW-SP pour chaque période affectée.

7.5 Discussion et conclusion

Dans ce chapitre nous avons introduit une extension périodique du problème de tournées de véhicules avec fenêtres de temps et contraintes de synchronisation et de précédence appliqués au système de soins à domicile. Pour résoudre ce dernier, nous avons étendu le programme linéaire défini dans le chapitre 4 de manière à prendre en compte les contraintes relatives à la périodicité. Enfin, de la même manière, nous avons proposé une méthode de résolution basée sur la décomposition du problème en deux sous-problèmes à savoir le problème d'affectation périodique et le VRPTW-SP. C'est une matheuristique combinant la programmation linéaire et les métaheuristiques proposées dans le chapitre 5. Le solveur

linéaire tel que Cplex peut être utilisé pour résoudre le problème d'affectation des demandes des clients aux périodes autorisées. Ensuite, les métaheuristiques proposées précédemment pourront être réutilisées pour construire les tournées sur chaque période en respectant les contraintes du VRPTW-SP. Par ailleurs, les tests de faisabilité introduits dans le cas mono-période peuvent être maintenues.

Le choix de la méthode proposée dans ce chapitre est motivée par sa facilité d'adaptation mais aussi par sa performance pour la résolution des grands problèmes combinatoires. Notons que le modèle mathématique proposé dans ce chapitre a été validé par des tests expérimentaux, réalisés sur l'ensemble d'instances proposées précédemment auquel les informations relatives au cas périodique (fréquences de visites, combinaisons, ...) ont été complétées aléatoirement. L'horizon de planification considéré ici est d'une semaine, i.e. $H = 7$ et le nombre de combinaisons de jours proposées par chaque clients est de 5. Cette valeur a été fixée après des tests préliminaires, car il est clair que le temps de résolution augmente considérablement en fonction du nombre de combinaisons existantes à tester.

Le tableau 7.7 présente les résultats expérimentaux obtenus suite à la résolution du modèle mathématique proposée en section 7.3. Ces derniers confirment les limites des solveurs linéaires tels que Cplex. L'optimalité n'est prouvée que sur 6 instances du premier groupe parmi les 37 instances en des temps beaucoup plus importants que dans le cas mono-période. L'utilisation d'un solveur pour résoudre le programme linéaire complet se restreint donc aux instances de tailles réduite (certaines instances à 45 clients). Les instances de taille plus grande n'ont pas pu être résolues à cause des limites en terme de mémoire de ce type de solveur. Rappelons que dans le cas mono-période 22 sur 37 instances ont été résolues optimalement. Ceci nous force à se retourner vers la résolution approchées, d'où l'intérêt de la matheuristique proposée ici. Cependant, la validation de la méthode introduite dans ce chapitre est en cours de réalisation.

TABLE 7.7 – Résultats sur les petites et moyennes instances

Instance	PLME					
	UB	Depl	Pref	LB	Cpu	E
18-4-s-2a	611,98	368,40	243,58	611,98	469,607	0,00
18-4-s-2b	645,06	432,60	212,46	645,06	118,725	0,00
18-4-s-2c	598,90	359,40	239,50	598,90	409,869	0,00
18-4-s-2d	615,83	403,80	212,03	615,83	941,177	0,00
18-4-m-2a	487,07	378,30	108,77	487,07	2617,36	0,00
18-4-m-2b	591,07	351,90	239,17	591,07	1739,12	0,00
18-4-m-2c	615,45	395,10	220,35	609,32	3600,00	1,01
18-4-m-2d	590,10	370,80	219,30	581,44	3600,00	1,49
18-4-m-2e	469,00	355,80	113,20	461,58	3600,00	1,61
18-4-l-2a	611,29	369,30	241,99	553,21	3600,00	10,50
18-4-l-2b	608,63	396,60	212,03	585,64	3600,00	3,93
18-4-l-2c	584,70	365,40	219,30	553,46	3600,00	5,65
18-4-l-2d	669,11	435,90	233,21	623,96	3600,00	7,24
18-4-l-2e	458,50	349,50	109,00	425,74	3600,00	7,70
Moy_{G1}	-	-	-	-	2506,85	2,79
45-10-s-3a	1258,51	886,50	372,01	1138,41	3600,00	10,55
45-10-s-2a	-	-	-	1008,19	3600,00	-
45-10-s-3b	2602,25	1343,40	1258,85	1010,77	3600,00	157,45
45-10-s-2b	-	-	-	1011,66	3600,00	-
45-10-s-3c	-	-	-	1005,50	3600,00	-
45-10-m-4	-	-	-	1004,85	3600,00	-
45-10-m-2a	-	-	-	943,94	3600,00	-
45-10-m-2b	-	-	-	908,45	3600,00	-
45-10-m-3	-	-	-	903,04	3600,00	-
45-10-l-2a	-	-	-	862,95	3600,00	-
45-10-l-2b	-	-	-	846,87	3600,00	-
45-10-l-3	-	-	-	820,15	3600,00	-
45-10-l-4	-	-	-	810,30	3600,00	-
Moy_{G2}	-	-	-	-	3600,00	84,00
Moyenne	-	-	-	-	3033,18	12,94

Conclusions et Perspectives

En raison du vieillissement de la population et ainsi l'apparition de différentes maladies chroniques, de nouveaux besoins sont nés en termes d'accès aux soins et d'amélioration de la qualité de vie et de prise en charge des patients. Par ailleurs, les contraintes économiques auxquelles fait face le secteur de la santé, ont suscité la réorganisation des établissements de santé.

De ce fait, comme introduit dans le chapitre 1, les établissements d'Hospitalisation à Domicile (HAD) sont apparus dans le but de libérer les lits des hôpitaux, surtout lorsque les patients ne nécessitent qu'une simple visite qui ne mobilise pas un plateau technique de haut niveau. Une lourde prise en charge à l'hôpital n'est pas essentielle. Et ainsi réduire les dépenses liées à l'hospitalisation tout en maintenant satisfaisant le bien-être des patients à leur domicile. Cependant, pour atteindre ce second objectif, les structures d'HAD ont besoin de support pour leur organisation, qui peut être apporté par des approches et outils d'aide à la décision issus de la recherche opérationnelle. Les travaux de recherche dans ce secteur ont comme but de mettre en place une coordination fine et une planification optimisée des ressources humaines et matérielles pour assurer une prise en charge et un suivi de soin de qualité, tout en maîtrisant les coûts. Les éléments de base de recherche opérationnelle souvent utilisés pour résoudre de tels problèmes sont rappelés dans le chapitre 4.

Le but de cette thèse, est de traiter une nouvelle variante du problème de tournées des soignants dans le contexte d'hospitalisation à domicile, impliquant différentes contraintes temporelles réelles. Les différentes approches de résolution de ce type de problèmes combinatoires ont été passées en revue dans le chapitre 3. Le principal constat qui en a découlé est que le problème abordé dans cette thèse reste encore très peu abordé tel qu'il est introduit ici, ce qui nous a encouragé d'entamer des recherches dans cette voie.

Dans le quatrième chapitre, nous avons proposé un modèle linéaire à variables mixtes entières pour l'optimisation mono-objectif du VRPTW-SP. Deux critères d'optimisation ont été optimisés, en les combinant dans une même fonction-objectif. Le premier objectif pris en compte consiste en la minimisation des coûts liés aux déplacements des soignants. Le deuxième objectif est formulé comme étant la minimisation des non-préférences des patients

envers le soignant réalisant le service. Ce problème a été modélisé sous forme d'un programme linéaire à variables mixtes entières (PLME) et a été testé en utilisant le solveur linéaire Cplex. La première remarque à tirer en vue des résultats obtenus concerne l'adéquation du modèle proposé introduisant les différentes contraintes communes du domaine de l'HAD. Néanmoins, les limites de cette approche de résolution est vite constatée sur des instances à partir de 45 clients. Toutefois, dans le même chapitre, nous avons développés des algorithmes basés sur des heuristiques constructives sont majoritairement ré-utilisés par la suite pour initialiser des méthodes plus sophistiquées.

Des méthodes plus puissantes et plus performantes basées sur des métaheuristiques ont été proposées par la suite. Dans un premier temps, le chapitre 5 présente des méthodes de résolution approchées pour la résolution du problème mono-objectif ont été élaborées élaborées, à savoir : 1) un GRASP : il s'agit d'une méthode itérative basée sur une heuristique gloutonne et randomisée ; 2) une ILS : il s'agit d'une recherche locale itérée et 3) une version hybride dans laquelle nous avons remplacé la recherche locale du GRASP classique par la méthode ILS, donnant lieu à la méthode GRASP $\times$ ILS. Ainsi, les résultats des heuristiques ont été nettement améliorés par toutes ces méthodes. La dernière méthode domine particulièrement les deux autres.

Dans un second temps, nous nous sommes intéressés au chapitre 6 à la résolution du problème multicritère à l'aide de métaheuristiques à base de population. Nous avons appliqué une forme récente d'algorithmes génétiques multi-objectif. Tout d'abord, nous avons implémenté un NSGAII intégrant une recherche locale : il s'agit d'un algorithme génétique avec tri par non-dominance. Ensuite, nous avons proposé un MS-NSGAII : il s'agit d'une généralisation de ce dernier avec redémarrage. Enfin, nous avons mis en œuvre une NSILS : c'est une recherche locale itérée intégrant l'aspect de tri par non-dominance. Plusieurs fréquences d'appels à la recherche locale ainsi que différentes localisations de cette dernière ont été testées sur le même ensemble d'instances, donnant 19 versions au total. Ainsi, des analyses sur l'impact de chaque paramètre ont été discutées et les meilleures configurations ont été déterminées.

Enfin, le chapitre 7 détaille l'une de nos perspectives qui consiste à étendre le problème sur un horizon de planification. Cependant, comme nous l'avons signalé en chapitre 7, ce type d'études connaît peu d'attention pour l'instant et de tels axes de recherche sont importants afin de se rapprocher des problématiques réelles. Nous avons proposé dans ce chapitre une formulation mathématique que nous avons validée sur le même ensemble d'instances auquel les caractéristiques périodiques ont été ajoutées. Le modèle a été résolu en utilisant le solveur linéaire Cplex. Les limites de ce dernier sont vite dévoilées en raison de la complexité du problème. Ainsi, le recours à des méthodes approchées semble inévitable.

La validation des contraintes de synchronisation fait l'objet principal de cette étude. Le point-clé de l'efficacité de ces méthodes réside justement dans l'adaptation des mouvements de recherche locale aux contraintes de synchronisation : applications des mouvements pour tout type de clients (synchronisés ou pas). De plus, notons que la complexité du problème augmente considérablement par rapport au nombre de clients synchronisés disponibles. Ainsi, une réflexion approfondie sur l'efficacité des tests de faisabilité incluant les clients à synchroniser nous semblait indispensable. En fait, les tests de faisabilité que nous avons développé sont pertinents car ils nous ont permis de considérer librement les clients synchronisés tout en respectant leur contraintes de synchronisation.

Le point-clé de l'efficacité de ces méthodes réside justement dans l'adaptation des mouvements de recherche locale aux contraintes de synchronisation : applications des mouvements pour tout type de clients (synchronisés ou pas). Les tests de faisabilité que nous avons proposé, nous ont permis de considérer librement les clients synchronisés tout en respectant leur contraintes de synchronisation. De plus, notons que la complexité du problème augmente considérablement par rapport au nombre de clients synchronisés disponibles. Ainsi, une réflexion approfondie sur l'efficacité des tests de faisabilité incluant les clients à synchroniser nous semblait indispensable. Par conséquent, les techniques que nous avons proposées sont pertinentes et gèrent parfaitement ces contraintes.

De plus, soulignons que pour tester l'ensemble de nos méthodes et fournir les résultats proposés, un développement informatique pointu a été nécessaire. L'implémentation des approches exposées n'est pas triviale et requiert le développement de nombreuses procédures. Celles-ci ont été réalisées en langage C.

L'ensemble de ces recherches ont donné lieux à un article publié dans la revue internationale (Expert Systems with Applications) et à 4 articles présentées en conférences nationales et internationales. Notons de plus qu'un deuxième article est en cours d'évaluation dans une revue internationale.

Diverses conclusions résultent de l'ensemble de ces travaux. La première est que des études restent à faire concernant le VRPTW-SP, en particulier afin d'apporter des bornes inférieures plus serrées au problème, voire des solutions optimales sur les instances encore ouvertes.

Il pourrait également être intéressant de développer de nouveaux jeux d'instances avec plus de deux services demandés par les clients ou encore d'étendre le problème pour le cas où les soignants pourraient offrir plusieurs services. L'intégration de nouvelles contraintes à notre problème semble pertinente pour des applications pratiques. L'introduction des préférences des clients dans les contraintes, ou encore la considération de contraintes réglementaires relatives aux pauses, aux congés des soignants, ...

Dans le cadre de l'amélioration de la qualité de service des structures d'HAD, l'ajout d'un critère d'optimisation minimisant les temps d'attente des patients s'avère aussi important pour se rapprocher au mieux des problèmes réalistes. En effet, ce problème des temps d'attente n'a pas été observé dans les instances testées dans cette thèse. Cependant, nous supposons que sur des instances plus larges ou plus réalistes, nous pourrons identifier de tels problèmes et par conséquent une telle étude semble importante.

Enfin, nous poursuivons encore l'étude de la méthode proposée dans le chapitre 7. En effet, l'adaptation de la métaheuristique, par exemple l'ILS au sein de la matheuristique n'est pas triviale. Le niveau de perturbation est critique car pour avoir une bonne diversification, une perturbation affectant les jours de visite est nécessaire. Cependant, la réalisation des tests expérimentaux et l'ajustement des différents paramètres sont en cours d'étude.

Bibliographie

Afifi, S., Dang, D.C., & Moukrim, A. 2013. A simulated annealing algorithm for the vehicle routing problem with time windows and synchronization constraints. *Pages 259–265 of : Learning and Intelligent Optimization*. Springer.

Afifi, S., Dang, D.C., & Moukrim, A. 2016. Heuristic solutions for the vehicle routing problem with time windows and synchronized visits. *Optimization Letters*, **10**(3), 511–525.

Ait Haddadene, S.R., Labadie, N., & Prodhon, C. 2014a. GRASP for the Vehicle Routing Problem With Time Windows, Synchronization and Precedence Constraints. *Pages 72–76 of : IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE.

Ait Haddadene, S.R., Labadie, N., & Prodhon, C. 2014b. The VRP With Time Windows, Synchronization and Precedence Constraints : Application In Home Health Care Sector. *In : MOSIM 2014, 10ème Conférence Francophone de Modélisation, Optimisation et Simulation*.

Ait Haddadene, S.R., Labadie, N., & Prodhon, C. 2016a. A GRASP $\times$ ILS for the vehicle routing problem with time windows, synchronization and precedence constraints. *Expert Systems with Applications*, **66**, 274 – 294.

Ait Haddadene, S.R., Labadie, N., & Prodhon, C. 2016b. NSGAII enhanced with a local search for the vehicle routing problem with time windows and synchronization constraints. *IFAC-PapersOnLine*, **49**(12), 1198–1203.

Akjiratikarl, C., Yenradee, P., & Drake, P.R. 2007. PSO-based algorithm for home care worker scheduling in the UK. *Computers & Industrial Engineering*, **53**(4), 559–583.

Allaoua, H., Borne, S., Létocart, L., & Wolfler-Calvo, R. 2013. A matheuristic approach for solving a home health care problem. *Electronic Notes in Discrete Mathematics*, **41**, 471–478.

Amaya, A., Langevin, A., & Trépanier, M. 2007. The capacitated arc routing problem with refill points. *Operations Research Letters*, **35**(1), 45–53.

Amaya, A., Langevin, A., & Trépanier, M. 2010. A heuristic method for the capacitated arc routing problem with refill points and multiple loads. *Journal of the Operational Research Society*, **61**(7), 1095–1103.

Balas, E., & Toth, P. 1983. *Branch and bound methods for the traveling salesman problem.* Tech. rept. DTIC Document.

Baldacci, R., Christofides, N., & Mingozzi, A. 2008. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, **115**(2), 351–385.

Beasley, J.E. 1983. Route first—cluster second methods for vehicle routing. *Omega*, **11**(4), 403–408.

Beltrami, E.J., & Bodin, L.D. 1974. Networks and vehicle routing for municipal waste collection. *Networks*, **4**(1), 65–94.

Bertels, S., & Stefan, T. 2006. A hybrid setup for a hybrid scenario : combining heuristics for the home health care problem. *Computers & Operations Research*, **33**(10), 2866–2890.

Bodin, L.D., & Sexton, T. 1986. The multi-vehicle subscriber dial-a-ride problem. *TIMS studies in Management Science*, **2**, 73–86.

Braekers, K., Hartl, R.F., Parragh, S.N., & Tricoire, F. 2016. A bi-objective home care scheduling problem : Analyzing the trade-off between costs and client inconvenience. *European Journal of Operational Research*, **248**(2), 428–443.

Bräysy, O., & Gendreau, M. 2005. Vehicle routing problem with time windows, part I : route construction and local search algorithms. *Transportation Science*, **39**(1), 119–139.

Bräysy, O., Dullaert, W., & Nakari, P. 2009. The potential of optimization in communal routing problems : case studies from Finland. *Journal of Transport Geography*, **17**(6), 484–490.

Bräysy, O., Gendreau, M., & Tarantilis, C.D. 2010. Solving large-scale vehicle routing problems with time windows : the state-of-the-art. *Technical report.*

Bredström, D., & Rönnqvist, M. 2007. A branch and price algorithm for the combined vehicle routing and scheduling problem with synchronization constraints. *NHH Dept. of Finance & Management Science Discussion Paper.*

Bredström, D., & Rönnqvist, M. 2008. Combined vehicle routing and scheduling with temporal precedence and synchronization constraints. *European Journal of Operational Research*, **191**(1), 19–31.

Ceselli, A., Righini, G., & Tresoldi, E. 2014. Combined location and routing problems for drug distribution. *Discrete Applied Mathematics*, **165**, 130–145.

Cheng, E., & Rich, J.L. 1998. A home health care routing and scheduling problem. *Technical report CAAM TR98-04, Rice University.*

Chu, F., Labadi, N., & Prins, C. 2006. A scatter search for the periodic capacitated arc routing problem. *European Journal of Operational Research*, **169**(2), 586–605.

Clarke, G.U., & W., John W. 1964. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, **12**(4), 568–581.

Colorni, A., Dorigo, M., Maniezzo, V., Varela, F., & Bourgine, P. 1991. Distributed optimization by ant colonies. *Pages 134–142 of : Proceedings of the first European conference on artificial life*, vol. 142. Paris, France.

Conover, W.J. 1980. Practical nonparametric statistics.

Coppi, A., Detti, P., & Raffaelli, J. 2013. A planning and routing model for patient transportation in health care. *Electronic Notes in Discrete Mathematics*, **41**, 125–132.

Cordeau, J.F. 2006. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, **54**(3), 573–586.

Cordeau, J.F., Gendreau, M., Laporte, G., Potvin, J.Y., & Semet, F. 2002. A guide to vehicle routing heuristics. *Journal of the Operational Research society*, 512–522.

Dantzig, G.B., & Ramser, J.H. 1959a. The truck dispatching problem. *Managmenet Science*, **6**(1), 80–91.

Dantzig, G.B., & Ramser, J.H. 1959b. The truck dispatching problem. *Management science*, **6**(1), 80–91.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. 2002. A fast and elitist multiobjective genetic algorithm : NSGA-II. *Evolutionary Computation, IEEE Transactions on*, **6**(2), 182–197.

Del Pia, A., & Flippi, C. 2006. A variable neighbourhood descent algorithm from a real waste collection problem with mobile depots. *International Transactions in Operational Research*, **13**, 125–141.

Desrosiers, J., Dumas, Y., & Soumis, F. 1986. A dynamic programming solution of the large-scale single-vehicle dial-a-ride problem with time windows. *American Journal of Mathematical and Management Sciences*, **6**(3-4), 301–325.

Desrosiers, J., Dumas, Y., Solomon, M.M., & Soumis, F. 1995. Time constrained routing and scheduling. *Handbooks in operations research and management science*, **8**, 35–139.

Doerner, K., Focke, A., & Gutjahr, W.J. 2007. Multicriteria tour planning for mobile healthcare facilities in a developing country. *European Journal of Operational Research*, **179**(3), 1078–1096.

Drexl, M. 2012. Synchronization in vehicle routing-A survey of VRPs with multiple synchronization constraints. *Transportation Science*, **46**(3), 297–316.

El Hachemi, N., Gendreau, M., & Rousseau, L.M. 2013. A heuristic to solve the synchronized log-truck scheduling problem. *Computers & Operations Research*, **40**(3), 666–673.

En-nahli, L., Allaoui, H., & Nouaouri, I. 2015. A Multi-objective Modelling to Human Resource Assignment and Routing Problem for Home Health Care Services. *IFAC-PapersOnLine*, **48**(3), 698 – 703.

Eveborn, P., Flisberg, P., & Rönnqvist, M. 2006. Laps Care - an operational system for staff planning of home care. *European Journal of Operational Research*, **171**(3), 962–976.

Feo, T.A., & Resende, M.G.C. 1995. Greedy randomized adaptive search procedures. *Journal of global optimization*, **6**(2), 109–133.

Fisher, M.L., & Jaikumar, R. 1981. A generalized assignment heuristic for vehicle routing. *Networks*, **11**(2), 109–124.

Gamst, M., & Jensen, T. 2012. A branch-and-price algorithm for the long-term home care scheduling problem. *Pages 483–488 of : Operations Research Proceedings 2011*. Springer.

Garey, M.R., & Johnson, D.S. 2002. *Computers and intractability ; A Guide to the Theory of NP-Completeness*. Vol. 29. W.H. Freeman & Co., New York.

Gehring, H., & Homberger, J. 1999. A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. *Pages 57–64 of : Proceedings of EUROGEN99*, vol. 2. Springer Berlin.

Gendreau, M., Hertz, A., & Laporte, G. 1994. A tabu search heuristic for the vehicle routing problem. *Management science*, **40**(10), 1276–1290.

Gillett, B.E., & Miller, L.R. 1974. A heuristic algorithm for the vehicle-dispatch problem. *Operations research*, **22**(2), 340–349.

Glover, F. 1989. Tabu search-part I. *ORSA Journal of computing*, **1**(3), 190–206.

Glover, F., & Laguna, M. 2013. *Tabu Search*. Springer New York.

Goldberg, D.E., & Holland, J.H. 1988. Genetic algorithms and machine learning. *Machine learning*, **3**(2), 95–99.

Guinet, Alain. 2014. ORGANISATION DES SOINS A DOMICILE EN EUROPE ET EN AMERIQUE DU NORD. *In : MOSIM 2014, 10ème Conférence Francophone de Modélisation, Optimisation et Simulation*.

Hartl, R.F. 2008. A survey on pickup and delivery problems. Part I : transportation between customers and depot. *Journal Für Betriebswirtschaft*, **58**, 21–51.

Ioachim, I., Desrosiers, J., Soumis, F., & Bélanger, N. 1999. Fleet assignment and routing with schedule synchronization constraints. *European Journal of Operational Research*, **119**(1), 75–90.

Issaoui, B., Zidi, I., Marcon, E., & Ghedira, K. 2015. New Multi-Objective Approach for the Home Care Service Problem Based on Scheduling Algorithms and Variable Neighborhood Descent. *Electronic Notes in Discrete Mathematics*, **47**, 181 – 188.

Jozefowiez, N., Semet, F., & Talbi, E. 2006. Enhancements of NSGAII and its application to the vehicle routing problem with route balancing. *Pages 131–142 of : Artificial evolution*. Springer.

Jozefowiez, N., Semet, F., & Talbi, E. 2008. Multi-objective vehicle routing problems. *European journal of operational research*, **189**(2), 293–309.

Kindervater, G.A., & Savelsbergh, M.W. 1997. Vehicle routing : handling edge exchanges. *Local search in combinatorial optimization*, 337–360.

Kirkpatrick, S., Vecchi, M.P., *et al.* . 1983. Optimization by simmulated annealing. *science*, **220**(4598), 671–680.

Knight, K.W., & Hofer, J.P. 1968. Vehicle scheduling with timed and connected calls : A case study. *OR*, 299–310.

Kontoravdis, G., & Bard, J.F. 1995. A GRASP for the vehicle routing problem with time windows. *ORSA journal on Computing*, **7**(1), 10–23.

Labadi, N., Prins, C., & Reghioui, M. 2008. A memetic algorithm for the vehicle routing problem with time windows. *RAIRO-Operations research*, **42**(03), 415–431.

Labadie, N., & Prodhon, C. 2014. A Survey on Multi-criteria Analysis in Logistics : Focus on Vehicle Routing Problems. *Pages 3–29 of : Applications of Multi-Criteria and Game Theory Approaches*. Springer.

Labadie, N., Prins, C., & Yang, Y. 2014. Iterated local search for a vehicle routing problem with synchronization constraints. *Pages 257–263 of : ICORES 2014-Proceedings of the 3rd International Conference on Operations Research and Enterprise Systems, Angers, Loire Valley, France*.

Labadie, N., Prins, C., & Prodhon, C. 2016. *Metaheuristics for Vehicle Routing Problems*. John Wiley & Sons.

Lacomme, P., Prins, C., & Sevaux, M. 2003. *Algorithmes de graphes*. Vol. 28. Eyrolles Paris.

Lacomme, P., Prins, C., & Ramdane-Chérif, W. 2005. Evolutionary algorithms for periodic arc routing problems. *European Journal of Operational Research*, **165**(2), 535–553.

Lacomme, P., Prins, C., & Sevaux, M. 2006. A genetic algorithm for a bi-objective capacitated arc routing problem. *Computers & Operations Research*, **33**(12), 3473–3493.

Laporte, G., Gendreau, M., Potvin, J-Y., & Semet, F. 2000. Classical and modern heuristics for the vehicle routing problem. *International transactions in operational research*, **7**(4-5), 285–300.

Lin, S., & Kernighan, B.W. 1973. An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, **21**(2), 498–516.

Liu, R., Xie, X., Augusto, V., & Rodriguez, C. 2013. Heuristic algorithms for a vehicle routing problem with simultaneous delivery and pickup and time windows in home health care. *European Journal of Operational Research*, **230**(3), 475–486.

Lourenço, H.R., Martin, O.C., & Stützle, T. 2010. *Iterated Local Search : Framework and Applications*. International Series in Operations Research & Management Science, vol. 146. Springer US.

Lysgaard, J., Letchford, A.N., & Eglese, R.W. 2004. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, **100**(2), 423–445.

Masson, R., Lehuédé, F., & Péton, O. 2013. Efficient feasibility testing for request insertion in the Pickup and Delivery Problem with Transfers. *Operations Research Letters*, **41**(3), 211–215.

Maya Duque, P.A., Castro, M., Sörensen, K., & Goos, P. 2015. Home care service planning. The case of Landelijke Thuiszorg. *European Journal of Operational Research*, **243**(1), 292 – 301.

Michallet, J., Prins, C., Amodeo, L., Yalaoui, F., & Vitry, G. 2014. Multi-start iterated local search for the periodic vehicle routing problem with time windows and time spread constraints on services. *Computers & Operations Research*, **41**(0), 196–207.

Murata, T., Nozawa, H., Ishibuchi, H., & Gen, M. 2003. Modification of local search directions for non-dominated solutions in cellular multiobjective genetic algorithms for pattern classification problems. *Pages 593–607 of : Evolutionary Multi-Criterion Optimization*. Springer.

Nagata, Y. 2007. Efficient Evolutionary Algorithm for the Vehicle Routing Problem with Time Windows : Edge Assembly Crossover for the VRPTW. *Pages 1175–1182 of : 2007 IEEE Congress on Evolutionary Computation*. IEEE Press.

Nguyen, V.P., Prins, C., & Prodhon, C. 2012. A multi-start iterated local search with tabu list and path relinking for the two-echelon location-routing problem. *Engineering Applications of Artificial Intelligence*, **25**(1), 56–71.

Nickel, S., Schröder, M., & Steeg, J. 2012. Mid-term and short-term planning support for home health care services. *European Journal of Operational Research*, **219**(3), 574–587.

Or, I. 1976. *Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking*. Xerox University Microfilms.

Parragh, S.N., Doerner, K.F., & Hartl, R.F. 2008. A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, **58**(1), 21–51.

Potvin, J.Y., & Rousseau, J.M. 1995. An exchange heuristic for routeing problems with time windows. *Journal of the Operational Research Society*, **46**(12), 1433–1446.

Prins, C. 2004. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, **31**(12), 1985–2002.

Pullen, H.G.M., & Webb, M.H.J. 1967. A computer application to a transport scheduling problem. *The Computer Journal*, **10**(1), 10–13.

Rasmussen, M.S., Justesen, T., Dohn, A., & Larsen, J. 2012. The home care crew scheduling problem : Preference-based visit clustering and temporal dependencies. *European Journal of Operational Research*, **219**(3), 598–610.

Redjem, R., & Marcon, E. In press. Operations management in the home care services : a heuristic for the caregivers' routing problem. *Flexible Services and Manufacturing Journal*, 1–24.

Redjem, R., Kharraja, S., Xie, X., & Marcon, E. 2012. Routing and scheduling of caregivers in home health care with synchronized visits. *In : 9th International Conference on Modeling, Optimization & SIMulation.*

Renaud, J., Boctor, F.F., & Laporte, G. 1996. A fast composite heuristic for the symmetric traveling salesman problem. *INFORMS Journal on Computing*, **8**(2), 134–143.

Rousseau, L.M., Gendreau, M., & Pesant, G. 2003. *The synchronized vehicle dispatching problem.* Citeseer.

Rousseau, L.M., Gendreau, M., & Pesant, G. 2013. The Synchronized Dynamic Vehicle Dispatching Problem. *INFOR : Information Systems and Operational Research*, **51**(2), 76–83.

Salazar-Aguilar, M., Langevin, A., & Laporte, G. 2013. The synchronized arc and node routing problem : application to road marking. *Computers & Operations Research*, **40**(7), 1708–1715.

Schott, J.R. 1995. *Fault tolerant design using single and multicriteria genetic algorithm optimization.* M.Phil. thesis, Massachusetts Institute of Technology, Dept. of Aeronautics and Astronautics.

Solomon, M.M. 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, **2**(35), 254–265.

Steeg, J., & Schröder, M. 2008. A hybrid approach to solve the periodic home health care problem. *Pages 297–302 of : Operations Research Proceedings 2007.* Springer.

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

Toth, P., & Vigo, D. 2014. *Vehicle Routing : Problems, Methods, and Applications*. Vol. 18. SIAM.

Trautsamwieser, A., & Hirsch, P. 2011. Optimization of daily scheduling for home health care services. *Journal of Applied Operational Research*, **3**(3), 124–136.

Velasco, N., Dejax, P., Guéret, C., & Prins, C. 2012. A non-dominated sorting genetic algorithm for a bi-objective pick-up and delivery problem. *Engineering Optimization*, **44**(3), 305–325.

Vidal, T., Crainic, T.G., Gendreau, M., & Prins, C. 2013. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research*, **40**(1), 475–489.

Wolsey, L.A. 1998. *Integer programming*. Vol. 42. Wiley New York.

Wolsey, L.A., & Nemhauser, G.L. 2014. *Integer and combinatorial optimization*. John Wiley & Sons.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., & Da Fonseca, V.G. 2003. Performance assessment of multiobjective optimizers : an analysis and review. *IEEE Transactions on Evolutionary Computation*, **7**(2), 117–132.

Syrine Roufaida AIT HADDADENE

Doctorat : Optimisation et Sûreté des Systèmes

Année 2016

Modèles et méthodes pour la gestion logistique optimisée dans le domaine des services et de la santé

Cette thèse aborde le problème de tournées de véhicules (VRP) intégrant des contraintes temporelles : fenêtres de temps (TW), synchronisation (S) et précédence (P), appliqué au secteur de soins à domicile, donnant le VRPTW-SP. Il s'agit d'établir un plan de visite journalier des soignants, aux domiciles des patients ayant besoin d'un ou plusieurs services.

Tout d'abord, nous avons abordé ce problème sous angle mono-objectif. Ensuite, le cas bi-objectif est considéré. Pour la version mono-objectif, un Programme Linéaire à Variables Mixtes Entières (PLME), deux heuristiques constructives, deux procédures de recherches locales et trois métaheuristiques à base de voisinages sont proposés : une procédure de recherche constructive adaptative randomisée (GRASP), une recherche locale itérée (ILS) et une approche hybride (GRASP × ILS). Concernant le cas bi-objectif, différentes versions de métaheuristiques évolutionnaires multi-objectifs sont proposées, intégrant différentes recherches locales : l'algorithme génétique avec tri par non-dominance version 2 (NSGAII), une version généralisée de ce dernier avec démarrages multiples (MS-NSGAII) et une recherche locale itérée avec tri par non-dominance (NSILS). Ces algorithmes ont été testés et validés sur des instances adaptées de la littérature.

Enfin, nous avons étendu le VRPTW-SP sur un horizon de planification, donnant le VRPTW-SP multi-période. Pour résoudre cette extension, un PLME ainsi qu'une matheuristique sont proposés.

Mots clés : recherche opérationnelle - métaheuristiques - optimisation combinatoire - transport, planification - modèles mathématiques - décision multi-critère - soins à domicile – synchronisation.

Models and Optimization Approaches for Logistic Problems in Health Care Systems and Services Sector

This work addresses the vehicle routing problem (VRP) including timing constraints: time windows (TW), synchronization (S) and precedence (P), applied in Home Health Care sector; giving the VRPTW-SP. This problem consists in establishing a daily caregivers planning to patients' homes asking for one or several services.

We have started by considering the problem as a single objective case. Then, a bi-objective version of the problem is introduced. For solving the single-objective problem, a Mixed Integer Linear Program (MILP), two constructive heuristics, local search procedures and three local search based metaheuristics are proposed : a Greedy Randomized Adaptive Search procedure (GRASP), an Iterated Local Search (ILS) and a hybrid approach (GRASP × ILS). Regarding the bi-objective VRPTW-SP, different versions of multi-objective evolutionary algorithm, including various local research strategies are proposed: the Non-dominated Sorting Genetic Algorithm version 2 (NSGAII), a generalized version of this latter with multiple restarts (MS-NSGAII) and an Iterated Local Search combined with the Non-dominated Sorting concept (NSILS). All these algorithms have been tested and validated on appropriate instances adapted from the literature.

Finally, we extended the VRPTW-SP on a multi-period planning horizon and then proposed a MILP and a matheuristic approach.

Keywords: operations research - metaheuristics - combinatorial optimization - transportation, planning - mathematical models - multiple criteria decision making - home care services - synchronization.