Problèmes de tournées de véhicules robustes multi-objectifs

Hiba Bederina

▶ **To cite this version:**

Hiba Bederina. Problèmes de tournées de véhicules robustes multi-objectifs. Autre [cs.OH]. Université de Picardie Jules Verne, 2018. Français. NNT : 2018AMIE0030 . tel-03692170

HAL Id: tel-03692170
https://theses.hal.science/tel-03692170

Submitted on 9 Jun 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse de Doctorat

Mention : Informatique
Spécialité : Recherche opérationnelle et optimisation

présentée à ***l'Ecole Doctorale en Sciences Technologie et Santé (ED 585)***

de l'Université de Picardie Jules Verne

présentée par

Hiba BEDERINA

pour obtenir le grade de Docteur de l'Université de Picardie Jules Verne

Problèmes de tournées de véhicules robustes multi-objectifs

Soutenue le 14 mai 2018, après avis des rapporteurs, devant le jury d'examen :

M^{me}. JOURDAN Laetitia	**Professeure, Université de Lille**	**Rapporteur**
M^{me}. SANTOS Andréa-Cynthia	**MCF - HDR , Université de Troyes**	**Rapporteur**
M. HIFI Mhand	**Professeur, Université d'Amiens**	**Directeur**
M^{me}. CHU Feng	**Professeure, Université d'Evry**	**Examinateur**
M. COCHARD Gérard-Michel	**Professeur, Université d'Amiens**	**Examinateur**
M. KACEM Imed	**Professeur, Université de Metz**	**Examinateur**

Remerciements

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

Sommaire

1.1 Contexte général

Le problème de tournées de véhicules (VRP) représente l'un des problèmes les plus étudiés dans le domaine de l'optimisation combinatoire, de part sa capacité à résoudre des problématiques différentes liées au monde réel comme la logistique et le transport. Cette classe d'algorithmes offre aux entreprises des leviers pour rendre plus efficace leur flotte de véhicules et réduire les coûts. Le mode de fonctionnement d'un VRP consiste à établir un planning de routes pour permettre à des véhicules de servir un ensemble de clients. Le but étant bien entendu la minimisation des coûts de livraison. Comme tous les problèmes dits de recherche opérationnelle, l'optimisation des problèmes de VRP se décline en deux étapes principales à savoir : la modélisation ou la conception du problème à résoudre et le choix d'une méthode de résolution par la suite.
L'étape de la modélisation nécessite la collecte des informations exactes concernant les paramètres décisionnels, tels que les distances, les demandes ou l'emplacement des clients. Toutefois, dans les problèmes du monde réel, certains paramètres dépendent principalement de l'environnement, et cela empêche de définir de façon précise les paramètres du modèle. Un exemple courant pour ce cas de figure est celui lié à l'évolution récente des moyens d'aide à la navigation qui exploitent les systèmes de géolocalisation (GPS) : quand plusieurs conducteurs demandent la même destination, le GPS leur affecte la même route ou en partie selon leurs positions de départ ainsi que le temps prévu pour arriver. La prise du même chemin par les différents conducteurs peut causer un embouteillage et retarder le temps d'arrivée prévu initialement. Par conséquent, le temps de trajet effectué entre deux clients est de nature aléatoire ou incertaine. L'incertitude peut affecter plusieurs paramètres du problème, notamment les demandes, les temps de trajet et les clients. Dans ce contexte, plusieurs travaux de recherche se sont intéressés à la généralisation du problème de VRP afin de l'adapter à un large spectre d'applications réelles, en prenant en considération les incertitudes. L'approche d'optimisation définie doit prendre en compte cette incertitude dans le modèle afin de proposer des solutions robustes. Un

défi important dans le domaine de l'optimisation robuste est le choix d'un critère, appelé souvent critère de robustesse qui sera utilisé pour décrire la performance de la solution dans la méthode d'optimisation. Plusieurs critères ont été proposés dans l'état de l'art, et certains d'entre eux sont parfois critiqués pour leur aspect conservateur ou bien pour leur complexité de mise en œuvre. Par conséquent, un choix judicieux du critère de robustesse peut impacter directement la qualités des solutions robustes trouvées.

Un autre aspect important de l'étape de la modélisation est lié à la fonction objectif à optimiser. La majorité des travaux de la littérature ont considéré une optimisation mono-objectif du VRP, où il s'agit souvent d'optimiser un objectif qui peut varier selon la variante du VRP étudié. On peut citer à titre d'exemple le coût du trajet qui a été considéré dans plusieurs travaux [Baldacci 2011]. Cependant, comme c'est le cas pour la plupart des problématiques réelles, le VRP peut être confronté à d'autres enjeux économiques comme le nombre des véhicules mis en service qui peuvent avoir un impact environnemental à travers leurs émissions polluantes. Ces objectifs sont souvent de nature antagoniste, où l'amélioration d'un objectif s'accompagne souvent par la détérioration d'un autre. Pour résoudre ce type de problèmes, plusieurs travaux utilisent les méthodes d'agrégation basées sur la définition d'un seul objectif qui représente une sommes pondérée des autres objectifs. Cependant, ce type de méthodes requière une certaine subjectivité dans la définition des poids de pondération entre les objectifs, ce qui les rend inefficaces pour ce genre de problème. Dans ce contexte, l'optimisation multi-objectif basée sur la notion de la Pareto-dominance semble offrir un bon cadre pour la résolution des problèmes de type grâce au principe de dominance qui permet d'optimiser les objectifs de façon simultanée, offrant à la fin au décideur un panel de solutions qui représentent un compromis entre les objectifs considérés.

La seconde étape d'optimisation des VRP concerne le choix d'une méthode pour résoudre le modèle établi. Plusieurs travaux se sont intéressés à l'application des méthodes exactes pour traiter le problème de VRP avec incertitudes. Cependant, l'application de ces méthodes se heurte à la complexité du problème car le VRP appartient à la famille des problèmes NP-difficiles [Lenstra 1981]. Rajoutant à cela la prise en compte d'incertitudes qui va augmenter davantage cette complexité. C'est pour cette raison que plusieurs travaux se sont penchés sur l'utilisation des métaheuristiques qui permettent de trouver des solutions satisfaisantes à des problèmes de grande taille en un temps raisonnable. Dans la grande famille des métaheuristiques, nous nous sommes intéressés particulièrement aux algorithmes évolutionnaires multi-objectifs que nous avons appliqués à deux variantes des problèmes VRP à savoir : le VRP capacitaire avec inceritudes (CRVRP), et le problème de tournées de véhicules sélectives (TOP) avec ses deux variantes sans et avec incertitudes.

1.2 Contributions et structure du manuscrit

Les contributions présentées dans ce manuscrit de thèse s'articulent autour de la modélisation et la résolution des problèmes de tournées de véhicules incertains multi-objectifs. Nous nous sommes intéressés dans un premier temps à la modélisation des incertitudes dans les problèmes de tournées de véhicules avec des incertitudes sur les temps de trajet et ceci à travers la proposition d'un nouveau critère de robustesse. L'objectif du critère proposé est de trouver une solution robuste qui présente le meilleur comportement sur une majorité de scénarios, où chaque scénario représente une valeur potentielle du temps de trajet. Le critère de robustesse proposé a été comparé à quelques critères de robustesse classiques sur une méthode exacte et une méta-heuristique, et les résultats obtenus ont confirmé sa pertinence pour l'optimisation robuste.
Dans un deuxième temps, nous nous sommes focalisés sur la résolution d'une variante de VRP avec incertitudes qui est le VRP robuste avec contraintes sur les capacités des véhicules (RCVRP). Nous avons tout d'abord formulé une variante multi-objectif de ce problème en considérant deux objectifs antagonistes à optimiser simultanément à savoir : le coût de trajet et le nombre de véhicules de la flotte. Pour réaliser l'optimisation, nous avons proposé une approche d'optimisation évolutionnaire multi-objectifs hybride (HMOEA). Cette approche est basée sur un algorithme génétique multi-objectif couplé avec des opérateurs de recherche locale. L'application d'un algorithme génétique a nécessité l'adaptation des ses principaux opérateurs afin de bien représenter le problème VRP. Les résultats obtenus ont été comparés avec ceux obtenus avec une méthode exacte et avec deux autres approches de la littérature basées sur des métaheuristiques en utilisant un jeu d'instances tiré de l'état de l'art. L'analyse des résultats a montré que l'approche proposée permet d'obtenir des résultats très compétitifs par rapport aux autres approches.
La troisième contribution de cette thèse concerne une variante du VRP qui est le problème de tournées de véhicules sélectives (TOP). Ce dernier représente un problème de VRP avec profit. Pour cette variante, nous avons tout d'abord proposé une formulation multi-objectif (MO-TOP) en considérant en plus de la maximisation du profit classiquement considéré dans les approches mono-objectif de l'état de l'art, la minimisation du coût de trajet. A noter que le TOP multi-objectif reste un domaine faiblement exploré dans l'état de l'art, et les seuls travaux qui se sont intéressés à cette question, ont considéré plutôt plusieurs composantes du profit (classiquement optimisé avec les méthodes mono-objectifs) en tant que des objectifs à optimiser. Comme c'était le cas pour les travaux sur le RVRP, un travail d'adaptation des opérateurs de l'algorithme évolutionnaire a été nécessaire afin de résoudre le problème TOP. Dans un premier temps, un algorithme génétique multi-objectif hybridé avec des approches de recherche locale a été proposé (HMOEA). Les résultats obtenus sur une série d'instances de l'état de l'art ont permis tout d'abord de démontrer le caractère antagoniste des objectifs optimisés. L'approche a été par la suite comparée à trois approches mono-objectifs de l'état de l'art et les résultats obtenus ont permis d'améliorer les résultats sur un certains nombre d'instances, tout en main-

tenant des résultats similaires sur les autres instances. Dans un deuxième temps, nous avons considéré une variante du problème TOP multi-objectif qui prend en considération les incertitudes (MO-RTOP). Cette variante a été résolue en utilisant l'algorithme multi-objectif hybride proposé pour le TOP ordinaire. Comme il n'existe pas d'instances dans la littérature dédiée au TOP robuste, nous avons généré une série d'instances afin d'évaluer la méthode que nous avons proposé. La comparaison des résultats obtenus avec CPLEX a permis de démontrer la pertinence de notre approche MO-RTOP.

Le reste de ce manuscrit est structuré en 5 chapitres.
Le chapitre 2 propose de revisiter l'état de l'art en relation avec les thématiques évoquées dans le contexte général. Après un parcours rapide des principales méthodes de modélisation d'incertitudes, une attention particulière sera portée sur les méthodes d'optimisation robuste. Par la suite, les principales méthodes utilisées pour résoudre les problèmes de VRP robuste sont synthétisées avec un focus particulier sur les méthodes évolutionnaires multi-objectifs.
Le chapitre 3 est consacré au critères de robustesse dans l'optimisation du VRP robuste. Après une présentation du critère que nous proposons pour les VRPs avec incertitude sur le temps de trajet, la comparaison de ce dernier avec les principaux critères de l'état de l'art est illustrée, et enfin, les expérimentations réalisées sont présentées.
Le chapitre 4 introduit l'approche basée sur l'optimisation multi-objectif pour aborder le problème de tournées de véhicules capacitaire avec coûts de trajet incertains. L'algorithme évolutionnaire multi-objectif proposé est présenté ainsi que les différents opérateurs de recherche locale permettant de l'hybrider.
Dans le chapitre 5 nous présentons les travaux relatifs à la problématique TOP au travers de ses deux variantes : déterministe et robuste. Enfin, le dernier chapitre permet de conclure en rappelant les principales contributions ainsi que les perspectives ouvrant sur des travaux futures.

Chapitre 2

Etat de l'art

Sommaire

2.1 Introduction

La recherche opérationnelle est un domaine important des mathématiques appliquées. Telle que connue aujourd'hui, elle peut être considérée comme une discipline

récente datant de la deuxième guerre mondiale où elle fut appliquée sur des opérations militaires. Cependant, des travaux plus anciens datant du XVII siècle ont cherché à résoudre des problèmes de *décision dans l'incertain* [Faure 2014]. Son objectif était d'aider à la prise de la ou des meilleures décisions à l'aide d'un ensemble de méthodes et techniques d'analyse et de résolution des phénomènes d'organisation. Ces derniers représentent une catégorie de problèmes de la vie, où l'homme, la machine et un produit sont en relation active, à la différence des décisions prises de façon totalement subjective, qui induit souvent à des conclusions conservatrices ou optimistes. Trois types de problèmes liés à ces phénomènes d'organisation peuvent être distingués :

- les problèmes combinatoires caractérisés par une grande complexité temporelle grâce à la multiplicité des combinaisons.
- Les problèmes aléatoires où toutes les incidences du hasard doivent être prises en compte dans la formulation mathématique.
- Les problèmes concurrentiels qui représentent un mixte entre le combinatoire et l'aléatoire tel que la théorie des jeux.

2.2 Problèmes d'optimisation

L'optimisation joue un rôle important en recherche opérationnelle. L'optimisation d'un problème est réalisée en deux étapes. La première étape est connue en économie comme la perception, et elle se compose à son tour en deux sous étapes : l'identification du problème ainsi que sa modélisation. La deuxième étape consiste dans le choix de la solution. On peut citer comme exemple d'applications : la chaine logistique, la répartition de charge (liens réseau, ordinateurs, etc.), la planification des vols (emploi du temps), l'ordonnancement, le calcul de trajectoire,... etc. Durant l'optimisation de ces situations, le décideur peut être confronté à plusieurs difficultés comme par exemple l'impossibilité de modéliser le système complet, réduisant l'étude à l'identification d'un sous problème (sous-optimisation), et cela peut provoquer des perturbations dans les sous systèmes connexes.

Un modèle est indispensable à un décideur pour obtenir les meilleurs choix dans des circonstances déterminées. Les critères d'optimisation et les contraintes imposées par le système sont définis par l'entrepreneur. L'analyste en économie ou le conseiller en recherche opérationnelle intervient dans la mise en forme mathématique des problèmes. Pour cela, il faut définir les variables de décision, préciser les paramètres du modèle, puis formuler la fonction objectif ainsi que les contraintes du problème. Puis, il faut aider le décideur à comprendre les implications des différentes décisions qui peuvent être prises et d'apporter des méthodes raffinées de résolution du modèle établi pour effectuer un choix. La qualité des résultats et des prédictions dépend de la pertinence du modèle et de l'efficacité algorithmique lors de la résolution.

La formulation mathématique de n'importe quel problème d'optimisation P peut être décrite comme suit :

$$(P)\begin{cases} & (\min / \max) f(S) & (1) \\ s.c. & S \in \Omega & (2) \\ & S \subseteq \mathbb{R}^p/\mathbb{N}^p/E & (3) \end{cases}$$

Où, l'équation (1) représente la fonction objectif f, qui consiste à choisir une solution ou un sous ensemble de solutions, parmi un ensemble de solutions possibles S, qui optimise le critère considéré soit par une minimisation ou une maximisation selon la définition du problème étudié. Les contraintes (2) assurent que la ou les solutions obtenues sont réalisables. Le domaine de faisabilité est noté Ω. Le domaine de variation de l'ensemble des variables S est défini par la contrainte (3). Ce dernier peut être réel, entier, ou un ensemble E de valeurs prédéfinies.

2.2.1 Classification des problèmes d'optimisation

Les problèmes d'optimisation diffèrent les uns des autres selon la forme de la fonction objectif et celle des contraintes, ainsi que le domaine de variation des variables du problème étudié. Ces derniers peuvent être classés d'après Minoux [Minoux 1983] comme suit :

- Selon le domaine de variation des variables : entier ou continue.
- Selon la fonction objectif et les contraintes : linéaires, quadratiques, ou non linéaires.
- Selon la ou les fonctions objectifs considérées : mono-objectif ou multi-objectif.
- Selon l'espace de recherche dessiné par les contraintes : convexes ou non convexes.

Le tableau 2.1 illustre la classification des problèmes d'optimisation. Les colonnes 1 et 2 représentent les formes possibles des équations ou inégalités relatives à la fonction objectif et les contraintes respectivement. La colonne 3 affiche les domaines de variation possibles caractérisant l'ensemble des variables. Enfin, la nature du problème est désignée dans la colonne 4.

Fonction objectif	Contraintes	Domaine	Nature du problème
linéaire	linéaire	$\mathbb{R}^p$	programmation linéaire (Dantzig, 1949)
linéaire	linéaire	$\mathbb{N}^p$	programmation linéaire en nombre entiers (gomory, 1958)
linéaire	linéaire	$\mathbb{N}^p+\mathbb{R}^p$	"mixed integer linear programming"
quadratique	linéaire	$\mathbb{R}^p$	programmation quadratique
non linéaire	$\emptyset$	$\mathbb{R}^p$	optimisation continue sans contraintes
non linéaire	$\emptyset$	$\mathbb{N}^p$	optimisation discrète sans contraintes
non linéaire	$\forall$	$\mathbb{R}^p$	optimisation non linéaire (kuhn et trucker, 1951)
$\forall$	non linéaire	$\mathbb{R}^p$	optimisation non linéaire (kuhn et trucker, 1951)
convexe	convexe	$\mathbb{R}^p$	optimisation convexe
multivoque	$\forall$	$\mathbb{R}^p/\mathbb{N}^p$	optimisation multi-objectif

TABLE 2.1 – Classification des problèmes d'optimisation.

L'optimisation linéaire étudie le cas où la fonction objectif et les contraintes,

définissant le problème, sont linéaires.

L'optimisation linéaire en nombres entiers étudie le cas où certaines ou toutes les variables sont contraintes de prendre des valeurs entières.

L'optimisation quadratique étudie le cas où la fonction objectif est d'une forme quadratique avec contraintes linéaires.

L'optimisation non linéaire étudie le cas général dans lequel l'objectif ou les contraintes (ou les deux) contiennent des parties non linéaires, éventuellement non-convexes.

L'optimisation convexe étudie le cas où la fonction objectif et les contraintes sont convexes.

L'optimisation multi-objectifs consiste à optimiser deux ou plusieurs objectifs simultanément.

2.2.2 Optimisation combinatoire

Dans cette thèse, nous nous intéressons à une catégorie des problèmes d'optimisation linéaire en nombres entiers. Il s'agit des problèmes d'optimisation combinatoires. La particularité de ces problèmes est que le domaine des variables de décision est un ensemble discret de valeurs prédéfinies de taille N.

$$(P_{PC}) \begin{cases} Z = & (\min / \max) f(S) & (1) \\ s.c. & S \in R & (2) \\ & S \subseteq N & (3) \end{cases}$$

La difficulté de ce type de problème réside dans le nombre de sous-ensembles (solutions) qui doivent être explorés qui est égale à $N!$. Quand N devient grand, la complexité temporelle de l'exploration de l'espace de solutions devient exponentielle. Par conséquent, les problèmes combinatoires sont dits NP-Complets. Pour cette raison, l'optimisation combinatoire a reçu beaucoup d'intérêts auprès des chercheurs dans différents domaines et même auprès de l'industrie, dans le but de développer des modèles et des méthodes de résolution.
Parmi les catégories des problèmes d'optimisation combinatoire classiques, on peut citer : le problème du sac à dos ; les problèmes de réseau et de graphique ; les problèmes d'emplacement, d'ordonnancement et de routage.
C'est dans cette dernière catégorie que s'inscrit le travail effectué dans cette thèse. Nous nous intéressons particulièrement aux problèmes de tournées de véhicules que nous présentons brièvement dans la prochaine section.

2.3 Problèmes de tournées de véhicules (VRP)

2.3.1 Définitions

Un problème de tournées de véhicules (VRP) est défini à partir d'un ensemble de n villes ou clients et d'une flotte de m véhicules au maximum. Il s'agit de trouver une tournée pour chaque véhicule p afin de leur permettre de visiter l'ensemble des

villes. Chaque ville doit être visitée au plus une seule fois par un seul véhicule. Le problème peut être modélisé sous forme de graphe complet $G = (V, A)$, où l'ensemble des nœuds V représente les villes $V = \{0, 1, ..., n\}$ où 0 représente le dépôt, et l'ensemble des arcs E représente le trajet (i, j) qui lie deux villes i et j. m variables binaires x_{ijp} sont utilisées pour chaque arc (i, j) , elles sont égales à 1 si le trajet (i, j) est traversé par le véhicule p, sinon elles sont égales à 0. Une deuxième variable binaire y_{ip} est utilisée pour chaque ville $i \in V - \{0\}$ afin de savoir si le véhicule p l'a visitée, si c'est le cas y_{ip} a la valeur 1, sinon elle est égale à 0. Le modèle mathématique d'un problème de VRP classique est décrit comme suit :

$$
\begin{cases}
Z = & (\min / \max) F & (1) \\
s.c. & \sum_{p=1}^{m} \sum_{i=1}^{n-1} x_{0ip} \leq m & (2) \\
 & \sum_{p=1}^{m} \sum_{i=1}^{n-1} x_{i0p} \leq m & (3) \\
 & \sum_{p=1}^{m} y_{kp} \leq 1, \ \forall k = 1, ..., n & (4) \\
 & \sum_{j=1}^{n-1} x_{ijp} = y_{ip}, & \\
 & i = \{1, \ldots, n\} \ p = \{1, .., m\} & (5) \\
 & \sum_{j=1}^{n} x_{jip} = y_{ip}, & \\
 & i = \{1, \ldots, n\} \ p = \{1, .., m\} & (6) \\
 & x_{ijp}, y_{ip} \in \{0, 1\}, & \\
 & \forall i, j = \{1, ..., n\}, p = \{1, ..., m\} & (7)
\end{cases}
$$

Les contraintes (2) et (3) garantissent que le nombre de véhicules quittant le dépôt est le même que le nombre de véhicules entrant dans le dépôt. La contrainte (4) garantit que chaque ville de 1 à n est visitée au maximum par un seul véhicule. Les contraintes (5) et (6) représentent la conservation de flux pour chaque ville i et assurent que le nombre de véhicules traversant tous les arcs entrant $\{(j, i), \forall j \in A\}$ est égale au nombre de véhicules traversant les arcs sortant $\{(i, j), \forall j \in A\}$. Enfin, les variables binaires utilisées x_{ijp} et y_{ip} sont déclarées par la contrainte (7).

Le problème de tournées de véhicules est une extension classique du problème du voyageur de commerce. Tous les deux font partie de la classe des problèmes NP-complet. Il fait parti des problèmes d'optimisation pour lesquels on ne connait pas d'algorithme permettant de trouver une solution exacte rapidement (temps polynomial) dans tous les cas de figure.

2.3.2 Méthodes de résolution

Le problème de VRP a toujours suscité un grand intérêt à la fois théorique et pratique. De plus, en raison de la recherche intensive sur le VRP depuis plus de cinquante ans, des progrès considérables ont été réalisés sur de nombreuses approches

et méthodes [Laporte 2009]. Le VRP a été résolu en utilisant différentes techniques. Pour plus de détails, le lecteur peut se référer à [Gendreau 2005] et [Hao 1999], où ces techniques ont été classées en trois catégories : méthodes exactes, heuristiques et métaheuristiques. D'autres revues de littérature [Golden 2008, Toth 2014, Prins 2014] permettent de donner plus de détails sur les méthodes de résolution du VRP.

Au cours de la dernière décennie, l'algorithme Branch-and-cut (BC) proposé par [Lysgaard 2004] était l'algorithme exacte le plus dominant pour le problème de VRP. Cependant, certaines instances avec seulement 50 clients n'ont pas été résolues. Ensuite, [Fukasawa 2006] a proposé un "Branch-Cut-and-Price" (BCP), (combinaison de génération de colonnes et de coupe) pour le VRP. Ce dernier a résolu toutes les instances de la littérature jusqu'à 134 clients. En outre, [Baldacci 2008] a proposé un algorithme de génération de colonnes qui permet de réduire le coût de calcul des instances de la littérature. Néanmoins, certaines instances de grande taille n'ont pas été résolues. Ensuite, une amélioration a été apportée à [Baldacci 2011] pour accélérer encore plus le processus et résoudre aussi les problèmes avec un nombre plus important de clients. A noter que le "Branch-Cut-and-Price" le plus récent a été proposé par [Pecin 2017] et a permis d'améliorer tous les résultats.

Comme les algorithmes exacts sont limités aux instances de petites tailles, les métaheuristiques semblent être de bons candidats pour résoudre les problèmes d'une plus grande taille. Elles sont aujourd'hui considérées parmi les approches les plus utilisées pour résoudre tous les types des problèmes VRP. De l'année 1990 à 2000, la recherche tabous a été prouvée dans [Laporte 2000] comme étant la méthode la plus efficace comparée à certaines heuristiques classiques, comme les économies de Clarke et Wright [Clarke 1964] et les algorithmes de balayage [Gillett 1974], où cette méthode a été capable de résoudre des instances de taille moyenne à optimalité ou quasi-optimalité.

Au cours de la dernière décennie, les algorithmes évolutionnaires ont été intensivement utilisés sur le problème de VRP. A titre d'exemple, l'algorithme génétique est l'un des algorithmes évolutionnaires qui ont été mis en oeuvre avec succès et testés sur les problèmes VRP ([Prins 2004, Baker 2003]). Du fait du temps de calcul important requis par ces algorithmes, ils sont généralement associés à des mouvements locaux (algorithmes évolutionnaires hybrides) afin d'accélérer la convergence vers les optima locaux.

2.3.3 Classification des problèmes VRP

Desrochers et al. [Desrochers 1990] proposent une classification des problèmes de tournées de véhicules. L'objectif de la classification est de distinguer la multitude de caractéristiques dans les problèmes pratiques de routage et d'ordonnancement des véhicules, et d'apporter une certaine uniformité dans la littérature sur le sujet. Ces caractéristiques définissent généralement quatre facettes du problème étudié : les caractéristiques du problème, les clients, la flotte, et l'objectif optimisé.

Les caractéristiques du problème Il peut s'agir du graphe modélisant le problème qui peut être orienté, non-orienté ou mixte. Il peut s'agir aussi des paramètres du problème tels que la matrice des coûts de trajet qui peut respecter l'inégalité triangulaire ou non, ou bien des demandes qui peuvent être livrées complètes ou en partie. Enfin, les demandes et le coût de trajet sont soit déterministes ou aléatoires.

Les clients Dans une représentation graphique, les clients sont représentés par les nœuds, les arcs ou les deux en même temps. Il existe deux types de clients : des clients particuliers nommés dépôts et les clients simples. Certains problèmes utilisent plusieurs dépôts, qui peuvent être intermédiaires. Les clients sont parfois caractérisés par des poids à livrer. Ces poids sont de nature déterministe ou aléatoire, et ils sont livrés et parfois aussi ramassés par des véhicules. Des intervalles de temps (souples ou dures) peuvent être imposés pour les opérations de livraisons et de ramassage des poids. En plus des poids, les clients sont parfois aussi affectés par des priorités ou des fréquences de visites pour une période de temps donnée. Dans d'autres cas, certains clients ne peuvent pas être visités à cause de quelques contraintes de la flotte.

La flotte Les restrictions sur la flotte utilisée considèrent le nombre de véhicules comme étant fixe, pour les utiliser tous, ou comme étant une variable à déterminer. Une contrainte de capacité, qui limite les charges, attribue des valeurs similaires pour tous les véhicules (homogène) ou des valeurs différentes (hétérogène). La capacité peut être représentée sous forme d'un seul volume ou d'un ensemble de compartiments. Des intervalles de disponibilité sont parfois imposés aux véhicules, qui nécessitent une synchronisation par la suite. De même, des quotas maximaux peuvent être imposés sur les coûts de trajet effectués par les véhicules. Comme pour les capacités, les quotas peuvent être identiques ou différents.

L'objectif du problème Les variantes du problème de VRP diffèrent aussi en fonction des critères optimisés. Les critères possibles sont : le coût de trajet, la taille de la flotte, le niveau de service attendu, et une ou des pénalités dues à la violation des contraintes de capacité, de quota, ou des intervalles de disponibilité des véhicules. Quand plusieurs objectifs sont considérés, ils peuvent être optimisés simultanément.

Dans les travaux réalisés dans cette thèse, nous nous sommes intéressés principalement à deux variantes du VRP à savoir le problème de tournées de véhicules capacitaire (CVRP) et le problème de tournées de véhicules sélectif (TOP) que nous présentons dans la prochaine section.

2.4 Variantes du VRP étudiées

2.4.1 Problème de tournées de véhicules capacitaire (CVRP)

Le probème de tournées de véhicules capacitaire (CVRP) est l'une des variantes les plus importantes dans la classe des problèmes VRP. CVRP a été proposé pour la première fois par Dantzig and Ramser [Dantzig 1959] sous le nom de "track dis-

patching problem". Soit un ensemble de clients (ou de villes) chacun ayant une demande connue, CVRP essaie de trouver l'ensemble de routes à parcourir par les véhicules réalisant au moindre coût le trajet total en commençant par un dépôt de départ et finissant par un dépôt d'arrivée. Les solutions trouvées doivent respecter des contraintes de capacité fixe pour chaque véhicule.

2.4.2 Problème de tournées de véhicules sélectives (TOP)

Le problème de tournées de véhicules sélectives (TOP "*Team Orenteering Problem*") a d'abord été étudié par Chao *et al.* [Chao 1996] et il appartient à la famille des problèmes de tournées de véhicules avec profit. TOP a été inspiré par un jeu joué en équipe dans les zones montagneuses. Ce jeu se compose d'un ensemble de joueurs et d'un ensemble de points où chaque point est associé à un profit. L'objectif est de trouver les parcours disjoints pour les joueurs qui maximisent le profit total récolté. Dans ces itinéraires, tous les points ne peuvent pas être visités en raison du budget limité dans le temps (quota).

Lorsqu'un seul véhicule est considéré, le problème est connu sous le nom de «Orienteering Problem» (OP) (voir Golden *et al.* [Golden 1987]) qui a été introduit en 1987 juste avant TOP. OP a été utilisé dans différentes applications pratiques telles que le problème de conception de voyage touristique et le problème de "crowd sourcing mobile".

Lorsque plus d'un véhicule est considéré, le problème est connu sous le nom de problème de tournées de véhicules sélectives TOP ("Team Orienteering Problem"). Ce problème consiste en une combinaison (multi-niveau) du problème de sac à dos (pour choisir un sous-ensemble de clients à visiter) et du problème du voyageur de commerce (pour trouver un itinéraire pour les clients sélectionnés). Il a de nombreuses applications dans la vie réelle. OP et ses variantes (TOP, l'OP dépendant du temps (TDOP), TOPTW) ont reçu une grande attention ces dernières années. Quelques études sur ces problèmes, y compris les applications théoriques et pratiques peuvent être trouvées dans [Vansteenwegen 2011]. Pour une revue plus récente sur TOP et OP, le lecteur peut se référer à Gunawan *et al.* [Gunawan 2016].

Parmi les méthodes exactes qui ont été appliquées au TOP nous pouvons citer les travaux de [Dang 2013a], [Keshtkaran 2016] et [Poggi 2010], où seules les instances moyennes ont été résolues. Le problème TOP, comme toutes les variantes OP, est un problème NP-difficile. Par conséquent, la plupart des méthodes proposées sont principalement basées sur des méta-heuristiques qui avaient pour objectif de maximiser le profit (approches mono-objectifs) comme les colonies de fourmies [Ke 2008], la recherche locale guidée [Vansteenwegen 2009], la recherche par voisinage [Kim 2013], l'algorithme mémétique [Bouly 2010], l'optimisation par essaims particulaires *et al.* [Muthuswamy 2011], l'algorithme mémétique basé sur l'optimisation par essaims particulaires *et al.* [Dang 2011], Recuit simulé *et al.* [Lin 2013]. Notons également que l'optimisation par essaims particulaires proposée dans Dang *et al.* [Dang 2013b] reste la méthode la plus efficace (en termes de qualité des solu-

tions obtenues) sur les instances de Chao *et al.* [Chao 1996], même si cela nécessite un temps d'exécution moyen élevé pour fournir les bornes trouvées.

2.5 Optimisation sous incertitudes

2.5.1 Optimisation sous incertitude

L'optimisation sous incertitude est apparue pour la première fois en 1950, où un problème d'optimisation linéaire a été étudié par Dantzig et Beale. Depuis le début du 21ème siècle, Dantzig considère toujours l'optimisation sous incertitude comme l'un des problèmes les plus importants en matière d'optimisation, grâce à leur difficulté qui accroit avec l'élargissement de l'espace d'incertitude. De tels problèmes apparaissent dans de nombreux domaines d'application tels que : la planification de la production, la localisation et le transport, l'allocation des ressources dans les systèmes financiers, etc.

Dans les problèmes d'optimisation, l'incertitude peut être due à un manque de connaissances, ou à une erreur de mesures/échantillonnage, ou une variabilité des caractéristiques du système dans le temps.
Dans la littérature, trois niveaux d'incertitudes sont distingués [Ha-Duong 2002] : incomplétude, incertitude, risque et floue.

- **Incomplétude** (surprise) : illustre les situations dans lesquelles on ne connaît pas limitativement les conséquences possibles.
- **Incertitude** (incertitude non-spécifique, incertitude dure ou ambiguité) : illustre les situations dans lesquelles les conséquences possibles sont connues mais on ne dispose pas de probabilités.
- **Risque** (stochastique) : les situations de risque sont vues comme une classe particulière de situations d'incertitude. Dans ce cas, non seulement les différentes conséquences possibles sont connues, mais en plus on dispose de probabilités bien fondées.
- **Floues** (imprécises, vagues ou encore mal définies) : une catégorie d'incertitude totalement distincte du risque. C'est une incertitude attachée aux sens des mots en langage naturel, comme "grand" ou "petit".

Puisque l'incertitude n'est pas occasionnelle, la meilleure façon de la gérer est de la comprendre pour pouvoir la structurer et l'intégrer dans le processus de prise de décision.

La modélisation mathématique complète, ainsi que l'optimisation du problème sous incertitude étudié diffèrent d'une approche à une autre.

2.5.2 Critères d'incertitude

L'hétérogénéité des préférences est un élément très important pour expliquer les échanges entre individus. Par conséquent, un critère de rationalité doit être choisi pour juger la qualité d'une solution. Ce dernier représente une fonction qui traduit

les préférences du décideur. Le choix d'un critère dépend de l'approche adoptée, en plus de la nature d'incertitude. Différents critères d'incertitudes existent dans la littérature.

Dans le cas du risque, les critères les plus utilisés dans la littérature sont : max-max, max-min, espérance de gain et l'efficacité moyenne-variance. En économie, ces critères peuvent être unifiés dans le cadre de la maximisation de l'utilité espérée pour exprimer la rationalité économique.

En ce qui concerne l'incertitude en général, le choix s'avère plus difficile. Pour cela, les préférences personnelles du décideur sont parfois ajoutées comme des probabilités subjectives, afin de ramener l'incertitude au risque, ou de ramener l'incertitude à la certitude en attribuant des valeurs estimées aux paramètres.

2.5.3 Approches d'optimisation sous incertitudes

Différentes approches ont été proposées dans la littérature pour gérer l'incertitude. Elles peuvent être regroupées en quatre groupes principaux (figure 2.1). La première est l'optimisation stochastique qui comprend l'optimisation stochastique classique fondée sur les recours, l'optimisation stochastique robuste et l'optimisation stochastique probabiliste (contrainte aléatoire). La deuxième est la programmation floue qui comprends la programmation floue flexible et la programmation floue possibiliste. La troisième est l'analyse de sensibilité et la quatrième est l'optimisation robuste.

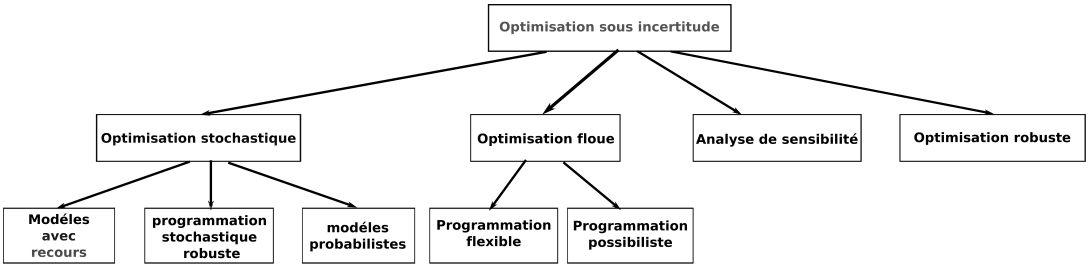


FIGURE 2.1 – Les approches d'optimisation sous incertitude

2.5.3.1 Optimisation stochastique

L'optimisation stochastique n'est utilisée que quand le comportement des paramètres incertains peut être décrit par des probabilités de distribution. Cette méthode est puissante, cependant quelques inconvénients sont à prendre en compte : (1) les distributions de probabilité sous-jacentes doivent être connues, (2) les solutions peuvent devenir infaisables pour certaines réalisations d'événements aléatoires, par exemple lorsque le recours est impossible.

Optimisation stochastique avec recours Le modèle de l'optimisation stochastique avec recours (ou à deux étapes) est le plus utilisé dans l'optimisation sto-

chastique. Il est utilisé généralement quand les données d'entrée sont récupérées en temps réel. Pour cette raison, un premier modèle est établi à partir des données actuelles avant la réalisation des données incertaines, puis un deuxième modèle est formé en vue de corriger la faisabilité du premier modèle en prenant en compte les nouvelles informations.

Le modèle classique de la première étape est formulé comme suit :

$$\begin{cases} \min_{x \in R^n} g(x) = c^T x + E_\xi[Q(x,\xi)] & (1) \\ s.c. & \\ Ax = b & (2) \\ x \geq 0 & (3) \end{cases}$$

Où $x \in R^n$ sont les variables de décision. $Q(x,\xi)$ est la valeur objectif du second problème et ξ représente les données incertaines. Dans la première étape, la fonction objectif optimise le coût c^T plus le coût prévu pour la deuxième étape, c'est à dire en considérant l'hypothèse que le décideur est neutre en terme de risque.

Cette formulation a été utilisée dans certains domaines d'applications telles que : la conception de réseau de télécommunication [Laguna 1998] et la planification financière [Kouwenberg 2001].

Optimisation stochastique robuste Dans le cas de l'optimisation stochastique robuste, la notion de risque est prise en compte dans le modèle. Pour cela, une modification sur la fonction objectif, proposée par [Mulvey 1995], peut être considérée.

$$\min \; c^T x + E_\xi[Q(x,\xi)] + \lambda f(w,x)$$

où f est une mesure de variabilité telle que la variance du coût de la deuxième étape. λ est un scalaire non négatif qui représente la tolérance au risque du décideur. Des grandes valeurs de λ aboutissent à des solutions qui réduisent la variance tandis que de faibles valeurs de λ réduisent les coûts attendus.

Optimisation stochastique probabiliste L'approche fondée sur les recours de la programmation stochastique exige que le décideur affecte un coût aux activités de corrections qui sont prises pour assurer la faisabilité du problème de la deuxième étape. Dans l'optimisation stochastique probabiliste les infaisabilités dans la deuxième étape sont autorisées moyennant une certaine pénalité qui doit être minimisée. Cette fiabilité du système est donc exprimée comme une exigence minimale sur la probabilité de satisfaire les contraintes.

2.5.3.2 Programmation mathématique floue

Comme la programmation stochastique, la programmation floue traite également les problèmes d'optimisation sous incertitude. L'une des principales différences entre les approches d'optimisation stochastique et floue réside dans la manière dont l'incertitude est modélisée. Dans le cas de la programmation stochastique, l'incertitude

est modélisée par des fonctions de probabilité discrètes ou continues. D'autre part, la programmation floue considère les paramètres aléatoires comme des nombres flous et les contraintes sont traitées comme des ensembles flous.

Il est à noter que pas mal des développements dans le domaine de la programmation mathématique floue sont basés sur l'article fondateur de Bellman et Zadeh (1970) [Bellman 1970]. Le domaine a été récemment popularisé par le travail de Zimmermann (1991) [urgen Zimmermann 1991]. Deux types de programmation floue existent : la programmation flexible et la programmation possibiliste. Dans les deux types de programmation floue, une fonction est utilisée pour représenter le degré de satisfaction des contraintes, les attentes du décideur concernant le niveau de fonction objectif et la plage d'incertitude des coefficients.

Programmation flexible La programmation flexible traite des incertitudes sur les contraintes. Les contraintes peuvent être représentées par des ensembles flous (contrainte douce), pour lesquelles des violations sont autorisées, plutôt que par des inégalités dures. Encore, les fonctions objectifs peuvent être représentées par un objectif flou. [Okuda 1975, urgen Zimmermann 1991]. Certaines violations de contraintes sont autorisées et le degré de satisfaction d'une contrainte est défini comme la fonction d'appartenance de la contrainte. Les fonctions objectif sont traitées comme des contraintes, où les bornes inférieures et supérieures de ces contraintes définissant les attentes du décideur.

Programmation possibliste la programmation possibiliste [Tanaka 1984] reconnaît les incertitudes dans les coefficients de la fonction objectifs ainsi que dans les coefficients des contraintes.

2.5.3.3 Analyse de sensibilité

Dans les applications réelles, il est généralement difficile de trouver une probabilité de distribution adaptée aux valeurs aléatoires (incertaines). L'analyse de sensibilité est une alternative à l'optimisation stochastique. Elle consiste à calculer une solution optimale en attribuant une valeur unique à chaque paramètre incertain, puis à déterminer la plage de variation pour ce paramètre de façon à ce que la valeur optimale reste stable. Pour l'analyse de sensibilité, peu de papiers se sont intéressés aux problèmes combinatoires dans la littérature à savoir : le problème d'ordonnancement [Jiang 2008], le problèmes d'affectation, du voyageur de commerce [Sotskov 1995], le problèmes de routage avec temps de trajet incertains [JANSSENS 2005], le problème de sac à dos [Al-Maliky 2018].

L'objectif de l'analyse de sensibilité est d'analyser un modèle mathématique en étudiant l'impact de la variabilité des facteurs en entrée sur la variable de sortie. Elle consiste à calculer des indices de sensibilité, en faisant varier les facteurs incertains sur leurs domaines, pour déterminer les entrées responsables de cette variabilité. Par la suite, l'analyse de sensibilité permet de prendre les mesures nécessaires pour

diminuer la variance de la sortie ou encore d'alléger le modèle en fixant les entrées dont la variabilité n'influe pas la variable de sortie.

Plusieurs indices de sensibilité existent dans la littérature, parmi lesquels, on peut citer : "Standardized Regression Coeficient" (SRC) et les indices de Sobol. Le choix de l'indice de sensibilité, appelé aussi mesure d'importance, dépend du nombre et de la corrélation (dépendance ou indépendance) entre les variables d'entrée étudiées, de la forme de la fonction de sortie, et du type de l'analyse de sensibilité choisie. Il existe de nombreuses méthodes pour calculer les indices de sensibilité par exemple ANOVA, Monte Carlo etc.

Les méthodes d'analyse de sensibilité peuvent être regroupées en trois classes :

Les méthodes de screening présentées [Campolongo 2007], qui consistent en une analyse qualitative de la sensibilité (variabilité) de la variable de sortie aux variables d'entrée. Elles permettent d'ordonner les variables d'entrée en fonction de leurs influences sur la réponse du système.

Les méthodes d'analyse de sensibilité locale évaluent quantitativement l'impact d'une petite variation autour d'une valeur nominale donnée des entrées sur la valeur de la sortie à travers le calcul d'un indice de sensibilité. La méthode locale la plus classique est OAT (*"One factor At Time"*).

Les méthodes d'analyse de sensibilité globale consistent à quantifier la variabilité de la variable de sortie, dans l'intégralité de son domaine de variation, due à la perturbation d'une ou des variables d'entrée. Elles s'intéressent à la variabilité de la sortie du modèle dans l'intégralité de son domaine de variation. Ceci, en déterminant quelle variance est due à la variabilité de quelles entrées. Si l'analyse de sensibilité locale s'intéresse plus à la valeur de la variable réponse, l'analyse de sensibilité globale s'intéresse quant à elle à sa variabilité.

L'une des limites de l'analyse de sensibilité est que les intervalles de perturbation, des variables d'entrée, construits avec cette pré-optimisation, peuvent ne pas correspondre aux vraies valeurs de la vie réelle causant une infaisabilité.

2.5.4 Optimisation robuste

Dans cette thèse, nous allons nous intéresser à l'optimisation robuste (RO). Elle provient, originalement, du travail de Soyster (1973) [Soyster 1973]. C'est une autre alternative à l'optimisation stochastique où le décideur doit prendre une décision en présence de l'incertitude et aucune fonction de recours n'est effectuée quand l'incertitude est réalisée. Le terme "optimisation robuste" est introduit pour indiquer les approches d'optimisation qui protègent le décideur de l'incertitude. Dans une optimisation robuste, l'incertitude des données est exprimée en définissant l'ensemble d'incertitudes au lieu de la distribution de probabilités. Si nous disposons d'informations limitées sur les distributions de probabilités des données, il peut être plus

facile de définir l'incertitude que d'estimer la distribution de probabilité. Une solution robuste est une solution de compromis entre la performance et la protection contre l'incertitude. Cette approche a donc pour but de trouver une solution raisonnable pour tous les scénarios considérés. Par exemple, l'incertitude établie à partir des premier et deuxième moments de données aléatoires peut être définie sans spécifier une distribution de probabilité particulière. Elle était largement utilisée dans plusieurs applications : la logistique [Sungur 2008], la planification [Mittal 2011], l'emplacement de l'installation [Alumur 2012]) , etc. Pour une description détaillée sur l'optimisation robuste traditionnelle, voir les travaux de [Bertsimas 2011] et [Ben-Tal 2009].

Il est à noter que plusieurs autres concepts de robustesse ont également été proposés, parmi lesquels le modèle de robustesse ajustable [Ben-Tal 2004], le modèle de robustesse léger (light robustness model) [Fischetti 2009] et le modèle de robustesse de récupération (recovery robustness model) [Liebchen 2009]. Des contributions plus récentes ont considéré une optimisation robuste sur le plan de la distribution, qui suppose que des distributions de probabilités des données sont incertaines et appartiennent à un ensemble donné [Ben-Tal 2010, Goh 2010].

De façon générale, l'application d'une approche d'optimisation robuste peut être résumée en trois étapes.
La première consiste en un choix d'un modèle robuste pour les valeurs incertaines déterminant si les paramètres incertains sont représentés comme un nombre fini ou infini de scénarios, ou comme un ensemble convexe tels qu'un polyèdre ou un ellipsoïde. Deuxièmement, un critère de robustesse doit être choisi par le décideur pour exprimer ses préférences dans la sélection des solutions. Ces deux première étapes servent à construire la formulation mathématique du problème complet. Enfin, une méthode de résolution est utilisée pour la résolution du problème construit. Ces trois étapes sont détaillées dans les trois paragraphes suivants.

2.5.4.1 Modélisation de l'incertitude

Les valeurs incertaines peuvent être modélisés en attribuant un ensemble borné de valeurs, par exemple ensembles ellipsoïdaux ou polyédriques, ou par une affectation de valeurs plausibles à chaque paramètre du modèle, appelé scénario. En pratique, les scénarios réalistes sont formés à partir des archives ou des données historiques. Deux types de scénarios existent :

— Intervalle de scénarios : représente un ensemble fini de valeurs.
— Ensemble de scénarios discret : représente un ensemble infini de valeurs.

Chaque valeur dans l'ensemble d'incertitude représente un scénario possible pour le paramètre incertain. De même, un scénario est défini en choisissant une valeur de chaque ensemble d'incertitude relatif aux paramètres du problème étudié.

2.5.4.2 Critères de de robustesse

Le critère de robustesse définit l'attitude des décideurs envers l'incertitude dans le but de juger la qualité d'une solution. Les critères de robustesse (appelés aussi

mesures de rationalité en économie) font la supposition suivante : une solution robuste est une solution qui est faisable et raisonnable pour toutes les configurations des scénarios incertains. Nous parcourons brièvement dans ce qui suit les principaux critères de robustesse de l'état de l'art.

Critère du pire cas ou "min-max" : Il s'agit du critère le plus utilisé dans le domaine de l'optimisation robuste.

Le critère *min-max* (pour un problème de minimisation de coût) ou son dual *max-min* (pour un problème de maximisation de profit) : cherche une solution minimisant (maximisant) le pire coût (profit) sur tous les scénarios. Il est introduit pour la première fois par Von Neumann (1928) [Neumann 1928], considéré comme le point initiant la théorie de jeux. La décision résultante de ce critère est considérée conservatrice comme c'est une anticipation sur la production du pire cas dans la meilleure façon possible.

min-max regret (ou min-max déviation) : Le critère *min-max regret (ou min-max déviation)* [Savage 1951] vise à minimiser l'écart (la différence) maximal (regret) aux valeurs optimales de la solution. Pour cela, la déviation de la solution actuelle par rapport à la solution optimale est calculée pour chaque scénario. Puis le critère min-max est appliqué sur les écarts calculés pour choisir une solution. A noter qu'il existe une autre variante de ce critère appelée min-max regret relatif, qui se différencie dans le calcul du regret qui représente le ratio entre la valeur de la solution et la valeur optimale pour chaque scénario. Par la suite, le min-max est appliqué pour choisir la solution qui minimise le ratio maximal.

min-max lexicographique : Le critère *min-max lexicographique* introduit par [Dresher 1961] dans la théorie de jeux ne considère pas que le pire scénario. Il commence par ordonner les scénarios, en fonction de leurs valeurs objectif, du pire au meilleur. Puis, pour choisir entre deux solutions, il compare entre leurs scénarios suivant l'ordre établi : s'ils ont la même valeur pour leurs pires scénarios, il passe aux second pire et ainsi de suite. Pour réaliser cela on peut utiliser par exemple la moyenne ordonnée OWA ("*ordered weighted average*").

***bw*-robustesse :** Le critère *bw-robustesse* a été récemment proposé par [Roy 2010]. Ce dernier est inspiré par les critères min-max. Il utilise deux paramètres déterminés par le décideur b et w. Une solution robuste est une solution garantissant un coût maximum inférieur à w pour tous les scénarios, et maximisant le nombre de scénarios où le coût (regret) est inférieur ou égale à b. Le critère *bw*-robustesse a été implémenté pour le problème du plus court chemin robuste dans [Gabrel 2013b] et pour un problème de planification des missions militaires robuste [Tang 2011].
Une extension de ce critère a été proposée par [Gabrel 2013a] dans le critère *pw-robustesse*. Dans cette variante, une solution est dite robuste si elle garantit une

valeur w pour tous les scénarios, et si elle atteint une valeur b avec un pourcentage de p scénarios.

α-robustesse lexicographique : Le critère *α-robustesse lexicographique* repose sur une critique des critères min-max qui ne prennent pas en compte la dimension subjective de la robustesse. Il s'agit d'un mélange des deux approches de robustesse les plus utilisées dans la littérature, à savoir les critères min-max et la p-robustesse [Snyder 2006]. En effet, il prend en compte le pire scénario ainsi que le concept de quasi-optimalité en considérant tous les scénarios et en introduisant un seuil de tolérance α. Cette approche considère non seulement le pire coût mais aussi le deuxième pire coût, le troisième et ainsi de suite. Plus précisément, chaque vecteur de performance est réorganisé du pire au meilleur et une solution robuste est une solution dont le vecteur de performance réordonné est proche, dans un seuil donné, d'un vecteur de référence fixe.

2.5.4.3 Méthode de résolution

Beaucoup de travaux dans l'état de l'art se sont intéressés à la résolution des problèmes d'optimisation combinatoire incertains avec des méthodes exactes tel que le branch-and-cut. Cependant, partant du constat que le problème des tournées de véhicules est un problème NP-difficile, et rajoutant à cela que la prise en compte des incertitudes fait augmenter sa complexité, plusieurs travaux se sont intéressés à l'application des méthodes méta-heuristiques. Ces méthodes permettent de trouver des solutions proches-optimales en un temps raisonnable. Ces méthodes sont bien adaptées dans le cas des problèmes du monde réel qui sont généralement de grande taille.

2.6 Problème de tournées de véhicules robuste (RVRP)

Le problème de tournées de véhicules robuste (RVRP) est un VRP auquel on associe une prise en compte de l'incertitude dans le modèle afin de garantir la faisabilité de la solution. Cette formulation répond à un besoin pratique qui consiste à prendre en compte l'incertitude des demandes et des coûts de trajet.
Beaucoup de travaux de recherche se sont intéressés aux problèmes de tournées de véhicules robustes. Ces travaux peuvent être regroupés selon le type d'incertitudes pris en compte. Nous distinguons trois catégories principales à savoir : les RVRP avec demandes incertaines, les RVRP avec des coûts de trajets incertains et les RVRP avec à la fois des demandes et des coûts de trajets incertains.

2.6.1 RVRP avec demandes incertaines

Dans la majorité des travaux considérant le RVRP, l'incertitude affecte les demandes des clients. Le premier problème de RVRP avec demandes incertaines a été proposé par [Sungur 2008]. Les demandes incertaines induisent généralement à

une insatisfaction des clients. Pour cela, le nombre de demandes insatisfaites est considéré parfois comme un objectif à minimiser. D'autres travaux optent pour l'application des stratégies de recours (retour au dépot [Cao 2014], redistriburion de la capacité libre [Sungur 2008]) pour palier à ce problème.

Dans la littérature du RVRP avec demandes incertaines, plusieurs variantes ont été étudiées. Nous pouvons citer par exemple : le VRP avec des contraintes de fenêtres de temps (VRPTW), le VRP capacitaires (CVRP), le VRP avec flotte hétérogène (HVRP).
Pour la résolution de ces problèmes, certains travaux ont utilisé des méthodes exactes, principalement l'algorithme "branch-and-cut" renforcé par des coupes : inégalités valides et des techniques de "lifting" [Noorizadegan 2012], "Rounded Capacity Inequalities"(RCI) [Gounaris 2013].

D'autres travaux se sont concentrés plutôt sur l'utilisation des méta-heuristiques pour résoudre des instances à grande échelle applicable dans la pratique. [Erera 2010] a proposé une heuristique de recherche tabou pour résoudre un problème de VRP avec des demandes stochastiques, où des contraintes de durée maximale sont introduites pour chaque route. Cette méthode requiert la résolution du problème adversaire qui détermine la durée maximale d'une route *a priori*. [Moghaddam 2012] a adapté un algorithme d'optimisation par essaims particulaires (PSO) pour résoudre le problème de VRP avec des demandes incertaines, couplé avec une approche locale de type recherche à voisinage variable (VNS), où le coût de trajet et l'équilibre de chargement des véhicules sont optimisés.
Dans[Cao 2014], des stratégies robustes ont été proposées pour gérer l'incertitude dans un VRP en appliquant un algorithme d'évolution différentielle pour minimiser le coût de trajet et les demandes non-satisfaites.

2.6.2 RVRP avec coûts de trajet incertains

Il n'existe pas une grande variété de travaux qui se sont intéressés au VRP avec des coûts de trajets incertains comparativement au VRP avec des demandes incertaines. Parmi les méthodes exactes qui se sont penchées sur ce problème, nous pouvons citer [Han 2013] qui a proposé un algorithme de "branch and cut" en introduisant des coupes pour résoudre une formulation stochastique avec recours d'un problème de VRP avec contraintes de durée maximale de trajets, où le temps de trajet peut prendre sa valeur dans un intervalle avec une probabilité connue. [Agra 2013] étudie le VRPTW où les temps de trajets appartiennent à un polytope. Ils ont proposé une extension aux deux modèles "resource inequalities" et "path inequalities" en introduisant des techniques de décomposition pour la gestion de l'incertitude. Les résultats obtenus permettent de trouver des résultats similaires que ceux de l'état de l'art concernant les modèles avec recours, mais avec des temps de résolutions plus rapides.

[Toklu 2013a, Toklu 2013b] considèrent le problème de VRP classique avec des temps de trajets dans un intervalle. Une méta-heuristique de colonie de fourmis

multiples est proposée. Cette méthode repose sur la techniques de [Bertsimas 2003] qui est utilisée pour générer des degrés de conservatisme différents à chaque itération. En plus, un mécanisme de "sharing" est ajouté pour permettre aux populations d'échanger leurs meilleures solutions périodiquement.
[Solano-Charris 2015] considère le VRP classiques où le temps de trajet appartient à un ensemble discret de scénarios. Plusieurs heuristiques de recherche locale sont adaptées au problème en vue de minimiser le pire coût de trajet à l'aide du critère de robustesse min-max lexicographique.

2.6.3 RVRP avec coûts de trajets et demandes incertains

Dans [Lee 2012], un VRP avec durée limitée pour les véhicules et des incertitudes sur les demandes et les coûts de trajet est considéré. L'algorithme de "branch-and-price" a été proposé pour la résolution de la formulation de décomposition du modèle "Dantzig-Wolfe" pour permettre d'encapsuler les incertitudes dans le sous-problème. Le sous problème a été modélisé sous forme d'un problème du plus court chemin avec des contraintes de ressource et un algorithme dynamique fut appliqué.

[Ordonez 2010] à proposé une reformulation du modèle avec demandes incertaintes de [Sungur 2008] afin de gérer les temps et les frais de déplacement, ainsi que les clients.

Le tableau 2.2 propose de synthétiser quelques travaux traitant les variantes du problème VRP robustes, en citant dans l'ordre : la variante du VRP étudiée, le modèle d'incertitude considéré, le critère de robustesse utilisé et enfin la méthode de résolution utilisée.

2.7 Optimisation multi-objectif

La plupart des problèmes d'optimisation du monde réel impliquent naturellement plusieurs critères. L'optimisation multi-objectif regroupe l'ensemble des méthodes qui cherchent à optimiser plusieurs objectifs simultanément. Ces objectifs sont généralement antagonistes : l'amélioration d'un objectif s'accompagne par la détérioration d'un autre objectif. Par conséquent, le résultat de l'optimisation n'est plus représenté par une solution unique comme c'est le cas dans l'optimisation mono-objectif, mais par un ensemble de solutions qui traduisent un compromis entre les critères à optimiser.

2.7.1 Le principe de dominance

Une solution x_1 domine une autre solution x_2 si et seulement si les deux conditions suivantes sont vérifiées :

— x_1 n'est pas pire que x_2 sur tous les objectifs,
— x_1 est strictement meilleur que x_2 pour au moins un objectif.

TABLE 2.2 – Synthèse des travaux du VRP robuste.

Auteurs	Variante VRP	Modèle d'incertitude	Critère de robustesse	Méthode de résolution
Sungur et al. [Sungur 2008]	CVRP	convexe,ellipsoïde,boite	pire cas	"Branch-and-cut"
Moghaddam et al. [Moghaddam 2012]	CVRP	Intervalle	pire cas	optimisation par essaims particulaires
Noorizadegan et al. [Noorizadegan 2012]	HVRP	polyèdre	pire cas	"Branch-and-cut"
Gounaris et al. [Gounaris 2013]	VRP	polyèdre	pire cas	"Branch-and-cut"
Cao et al. [Cao 2014]	OVRP	convexe	pire cas	algorithme d'évolution différentielle
Toklu et al. [Toklu 2013a, Toklu 2013b]	CVRP	intervalle	Degré de conservatisme	colonie de fourmis multiples
Han et al. [Han 2013]	VRP avec contraintes de durée maximale	intervalle	Degré de conservatisme	"Branch-and-cut"
Agra et al. [Agra 2013]	VRPTW incapacitaire	polytope	Degré de conservatisme	technique de "Cutting plane"
Solano-Charris et al. [Solano-Charris 2015]	CVRP	scénarios discrets	min-max lexicographique	méta-heuristiques basé recherche locale
Lee et al. [Lee 2012]	VRP avec contraintes de durée maximale	intervalle	Degré de conservatisme	"branch-and- price"
Ordonez [Ordonez 2010]	CVRP	convexe	pire cas	Analytique

2.7.2 L'optimalité de Pareto

L'optimum de Pareto sert à sélectionner un ensemble fini de solution pour le présenter au décideur. Cet ensemble est le résultat d'un tri effectué sur tous l'ensemble de la population de solution, où toutes les paires de solutions sont comparées à l'aide du principe de dominance. Cela permet de classer les solutions en rangs, où les solutions appartenant à chaque rang sont incomparables entre elles. Le premier rang est dominé par tous les autres rangs, le deuxième rang est dominé par tous les autres rangs sauf le premier, et ainsi de suite. À la fin, un dernier rang est obtenu qui n'est dominé par aucun autre rang. Ce dernier représente l'optimum de Pareto.

Solutions incomparables : une comparaison entre chaque paire de solutions montrent que l'une des solutions est meilleure pour un objectif mais mauvaise pour l'autre.

2.7.3 Méthodes classiques

Les méthodes classiques existantes essaient de réduire la complexité d'un problème d'optimisation multi-objectif en transformant plusieurs objectifs en une seule fonction objectif. A noter que l'agrégation n'optimise généralement pas tous les objectifs existants en même temps, donc un objectif peut être moins considéré par l'agrégation qu'un autre. L'optimisation multi-objectif semble être plus efficace, car tous les objectifs sont simultanément optimisés, donc aucun objectif n'est favorisé par rapport à un autre. Avant de passer en revue les principales méthodes multi-objectifs, nous parcourons rapidement quelques méthodes classiques d'agrégation.

2.7.3.1 Méthode d'agrégation

La méthode d'agrégation est la plus populaire. Elle consiste à réduire les M objectifs du problème en une équation linéaire qui les additionne en les associant chacun avec un poids p_o qui traduit l'importance relative à cet objectif o auprès du décideur.

$$f(x) = \sum_{1}^{M} p_o f_o(x)$$

Cependant, cette méthode permet de résoudre uniquement des problèmes simples avec des fronts de Pareto convexes, et ne permet pas de s'attaquer à des problèmes non-linéaires ou des problèmes dont la surface de Pareto contient des région non-convexes.

2.7.3.2 Méthode de ε-contraintes

La méthode de ε-contraintes introduit une subjectivité dans le problème étudié en permettant de garder l'un des objectifs du problème dans la fonction objectif et de transformer les autres en contraintes bornés par une valeur ε_o pour chaque objectif o fixés par le décideur.

$$\begin{cases} \min_x f_\eta(x) \\ st. \\ \qquad\qquad f_o(x) \leq \varepsilon_o \ \forall o = \{1, ..., M\} \ et \ o \neq \eta \end{cases}$$

Contrairement à la méthode d'agrégation pondérée, la méthode de ε-contraintes est capable de trouver des solutions de Pareto appartenant à des régions non-convexes. Cependant, le choix du vecteur ε est très important et lié à certaines difficultés relatives à sa position par rapport au front de Pareto.

2.7.3.3 Mesure de Chebyshev

La mesure de Chebyshev repose aussi sur une pondération de la fonction objectif à optimiser qui est de la forme :

$$Argmin\{\max_m(p_m|f_m(x) - y_m|)\}$$

Où les p_m représentent les poids de pondération des m objectifs à optimiser, et y est le vecteur idéal qui représente les bornes inférieures de chaque objectif. Comme la méthode ε-contraintes, cette méthode permet de retrouver des fronts de Pareto non-convexes.

2.7.4 Algorithmes évolutionnaires

Les algorithmes évolutionnaires sont des algorithmes d'optimisation appartenant à la famille des métaheuristiques. Ces algorithmes bio-inspirés reposent sur le paradigme darwinien de l'évolution naturelle des espèces. Selon cette théorie, les individus les plus aptes survivent à la sélection naturelle et ont plus de chance de se reproduire au cours des générations. Les algorithmes évolutionnaires font évoluer une population d'individus de façon itérative dans le but de rechercher les solutions optimum. Il existe plusieurs familles d'algorithmes évolutionnaires parmi lesquelles nous pouvons citer : la Programmation Evolutionnaire (EP) [Fogel 1966], les algorithmes génétiques (GA) proposés par [Holland 1975], et popularisés par [Goldberg 1989], les stratégies d'évolutions (ES)[Rechenberg 1973], et enfin la programmation génétique [Koza 1992, Koza 1994].

2.7.4.1 Vocabulaire et principe de fonctionnement

Nous présentons dans cette section le vocabulaire associé aux algorithmes évolutionnaires ([Bibai 2010, Yagoubi 2012]) :

- Les points de l'espace de recherche sont appelés des **individus**
- Un ensemble fini d'individus est appelé **population**
- La fonction objectif à optimiser est appelée **performance**
- Le calcul de la performance d'un individu est appelé **évaluation**
- La **génération** correspond à une population en une certaine itération

— Pour générer de nouveaux individus (appelés enfants) à partir d'autre individus (appelés parents), des **opérateurs de variations** sont utilisés, et sont généralement au nombre de deux : le **croisement** qui consiste à échanger des gènes entre deux ou plusieurs individus, et la **mutation** qui consiste en la modification d'un ou plusieurs gènes d'un individu
— La **sélection** est le processus de choix des individus en se basant sur leur performance
— Le **remplacement** est le processus de formation d'une nouvelle population à partir des parents et des enfants.

Nous décrivons à présent le principe de fonctionnement général d'un algorithme évolutionnaire (figure 2.2). La première étape est l'initialisation où une population initiale d'individus P_0 est générée aléatoirement. A chaque itération t, la fonction objectif des individus de la population P_t est évaluée, et les individus les plus performants sont sélectionnés. Par la suite, les opérateurs de croisement et de mutation sont appliqués afin de générer la population des enfants. Après évaluation des enfants nouvellement générés, ces derniers sont combinés avec leur parents et les meilleurs vont remplacer certains parents (moins performants) ce qui permet de construire la génération suivante P_{t+1}. Cette procédure est répétée jusqu'à ce qu'un critère d'arrêt soit rencontré.

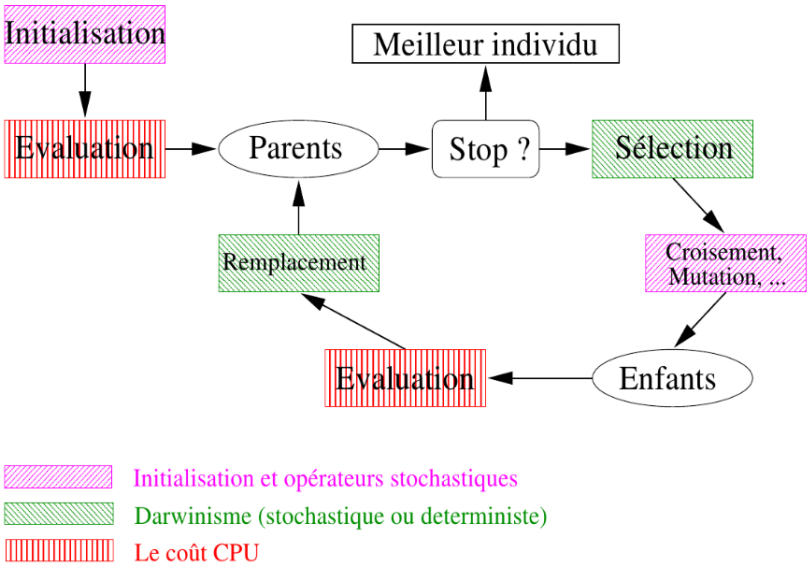


FIGURE 2.2 – Principe de fonctionnement de l'algorithme évolutionnaire [Schoenauer 2003]

2.7.4.2 Le compromis exploration/exploitation

Tous les algorithmes évolutionnaires ont pour objectif de réaliser un bon compromis entre l'exploitation des meilleures solutions et l'exploration de l'espace de recherche. L'exploitation des meilleures solutions permet d'améliorer les solutions en effectuant des mouvements locaux. Cependant, l'utilisation excessive de l'exploitation peut mener à une convergence prématurée. Pour cela, des techniques d'exploration de l'espace de recherche sont utilisées afin d'explorer d'autres régions de l'espace de recherche et éviter ainsi une convergence vers une optimum local. Ces techniques favorisent donc la diversité, mais leur utilisation excessive peut engendrer la non-convergence de l'algorithme. Par conséquent, un bon algorithme évolutionnaire est celui qui arrive à établir un bon compromis entre l'exploration et l'exploitation. Malheureusement, il n'existe pas de règles universelles pour ce type de réglage et le dosage "exploitation/exploration" est souvent défini empiriquement à travers l'expérience et dépend ainsi du problème en question.

2.7.4.3 Algorithmes évolutionnaires multi-objectif (MOEA)

IL existe plusieurs approches évolutionnaires qui ont cherché à adapter les algorithmes évolutionnaires au cas de l'optimisation multi-objectif. L'objectif de ces approches est le même à savoir converger vers des solutions Pareto-optimale tout en assurant une certaine diversité entre elles. Cependant, ils diffèrent dans la façon de procéder notamment les opérateurs de sélection et de diversification des solutions. Parmi ces algorithmes nous pouvons citer NSGA-II [Deb 2002], SPEA2 [Zitzler 2001], IBEA [Zitzler 2004], ε-MOEA [Deb 2003]. Plusieurs études ont cherché à comparer ces algorithmes à la fois sur des fonctions test et des problématiques réelles, et le classement change souvent selon le problème étudié (voir [Yagoubi 2012]). Cependant, l'algorithme NSGA-II semble être l'algorithme le plus populaire dans ce domaine grâce à sa robustesse dans la résolution d'une grande diversité de problèmes du monde réel.

2.7.4.4 NSGA-II

L'algorithme NSGA-II ("Non-dominated Sorting Genetic Algorithm") est considéré comme l'un des algorithmes évolutionnaires mulit-objectifs les plus importants. Le schéma globale de l'algorithme reprend les grandes étapes d'un algorithme évolutionnaire décrites précédemment. La spécification de NSGA-II réside dans la procédure de comparaison des solutions. A partir de la population des enfants Q_t de taille N créée à partir de celle des parents P_t (de taille N aussi), les deux populations sont ensuite réunies pour former une population mixte R_t de taille $2N$. Cette population est triée selon le principe de dominance au sens de Pareto pour former les fronts successifs : le premier front $\mathcal{F}_1$ correspond à l'ensemble des solutions non-dominées de R_t. Le deuxième front $\mathcal{F}_2$ est obtenu en réalisant un nouveau tri selon la dominance sur R_t après avoir enlevé les points de $\mathcal{F}_1$. Cette procédure est répétée jusqu'à ce que tous les points de R_t soient attribués à un front. Par la suite, la nouvelle population

P_{t+1} est créée et est remplie au fur et à mesure avec les différents fronts successifs. Quand la taille de P_{t+1} dépasse N, le reste des fronts seront éliminés. Cependant, le nombre de points du dernier front considéré peut être supérieur aux nombres de places restantes dans la nouvelle population. Dans ce cas, un deuxième critère de diversification est utilisé pour départager les solutions du dernier front. Ce critère, appelé le critère de surpeuplement, cherche à améliorer la diversité de la population en favorisant les individus dans les régions les "moins peuplées". Il est calculé de la façon suivante : La distance de surpeuplement d_i d'une solution i est une mesure de l'espace de recherche autour de i qui n'est pas occupé par aucune autre solution dans la population.
L'algorithme 1 résume la procédure de l'algorithme NSGA-II.

Algorithme 1 NSGA-II algorithm

Input. Une instance RVRP.

Output. Une population de solutions.

Set $t = 0$ and $P_0 \leftarrow random()$, $|P_0| = N$
tant que (critère d'arrêt non atteint) **faire**
  Q_t = variation (Select(P_t)) // construit une population d'enfants
  $R_t \leftarrow P_t \cup Q_t$ // population mixte : parents + enfants
  $\mathcal{F} \leftarrow$ Pareto-Rank(R_t), where $\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, ...)$ //
  $P_{t+1} \leftarrow \emptyset$, $i \leftarrow 1$ // générer la nouvelle population
  tant que $(|P_{t+1}| + |\mathcal{F}_i| < N)$ **faire**
    $P_{t+1} \leftarrow P_{t+1} \cup \mathcal{F}_i$;
    $i \leftarrow i + 1$;
  $P_{t+1} \leftarrow$ Cronwding-Distance-Rank(P_{t+1}) ;
  $P_{t+1} \leftarrow P_{t+1}[0 : N]$ // choisi les N premiers individus $t \leftarrow t + 1$;

2.8 Problème de tournées de véhicules multi-objectif

Nous présentons dans cette section l'état de l'art des problèmes VRP pour lesquels plusieurs objectifs sont à optimiser simultanément, parmi lesquels nous pouvons citer :

- **La minimisation du nombre de véhicules (NV)** : qui affecte les coûts fixes d'envoi des véhicules (FC) et le coût de rémunération des conducteurs (RC).
- **La minimisation du coût de trajet (TD)** : qui affecte le temps de trajet (TDT), le coût de carburant calculé par la consommation d'énergie pour la livraison (CPL) et pour le ramassage (CPR). Ça peut affecter aussi la pollution représentée par l'émission du CO_2 par la flotte (EC). Certains considèrent un autre objectif équivalent qui est la minimisation de la longueur maximale entre les routes (LM).
- **L'optimisation de l'équilibre des routes** : cet objectif affecte l'équité de la charge de travail entre les véhicules (ER), qui peut dépendre de la distance parcourue, des poids chargés, etc. Cette optimisation permet de minimiser les coûts de retour au dépôt pour le réapprovisionnement.

— **La minimisation des demandes non remplies** : qui affecte le niveau de satisfaction des clients (SC). A noter que parfois le niveau de satisfaction des clients pour livraison (SCL) et pour le ramassage (SCR) sont considérés séparément. La satisfaction des clients dépend aussi de la ponctualité des conducteur (TR).

Les premiers travaux qui se sont intéressés au VRP à plusieurs objectifs, proposaient une optimisation à plusieurs étapes ("*multi-stages*"). Pour cela, chaque objectif est optimisé séparément dans un ordre de priorité pré-établi par le décideur. On peut citer à titre d'exemple le travail dans [Homberger 2005] qui a introduit une recherche hybride en deux étapes, où la première étape minimise le nombre de véhicules utilisant une stratégie d'évolution, et la deuxième étape minimise ensuite la distance totale en utilisant un algorithme de recherche tabou.
L'inconvénient principal de cette méthode reste cependant le choix de priorité entre les objectifs, car dans certains cas, ces objectifs ont une importance égale, ce qui conduit à des résultats biaisés par l'ordre pré-établi entre eux. Nous présentons dans ce qui suit les travaux plus récents portant sur le VRP multi-objectif. Ces travaux peuvent être organisés en deux grandes catégories à savoir : la transformation en un problème mono-objectif et l'optimalité de Pareto.

2.8.1 Transformation mono-objectif

La majorité des travaux dans cette catégorie ont considéré la méthode d'*agrégation*. [Tuyttens 2004] étudie un problème qui combine quatre variantes du problème VRP. Un recuit simulé a été utilisé pour optimiser l'ensemble des objectifs potentiels, dans une fonction d'agrégation avec des poids différents. [Chand 2010] propose un algorithme évolutionnaire pour la résolution du problème de VRP classique. Le nombre de véhicules et le coût de trajet total sont optimisés dans une fonction agrégée selon un vecteur de poids. [Jiang 2014] étudie le VRPTW avec demandes stochastiques, où le coût de trajet et la rémunération des conducteurs sont minimisés. Une approche évolutionnaire hybride est utilisée pour l'ensemble de sous-populations évoluant simultanément. La relation entre les voisinages est définit par la différence de la somme pondérée de leurs objectifs.

D'autres travaux se sont intéressés à l'application de la méthode ε-*contrainte* parmi lesquels nous pouvons citer le travail de [Muller 2010] qui traite le problème de tournées de véhicules VRPTW avec trois objectifs conflictuels à minimiser à savoir : le coût du trajet, le nombre des véhicules et les pénalités issues de la violation des intervalles de temps. La méthode $\varepsilon - contrainte$ a été utilisée à travers la transformation de la minimisation de pénalité en une contrainte pour borner sa valeur à l'aide d'une limite fixée par le décideur.

[Molina 2014] propose un modèle multi-objectif, utilisant la mesure de *Chebyshev*, pour le problème de VRP avec une flotte hétérogène. L'algorithme proposé combine une heuristique avec une méthode de recherche locale pour optimiser trois objectifs à savoir : le coût total, les émissions en CO_2 et les émissions polluantes.

2.8.2 Optimalité de Pareto

Plusieurs travaux de l'état de l'art se sont intéressés à l'optimisation multi-objectif des problèmes VRP. Tenant compte du fait que le problème VRP est NP-difficile, la plupart de ces travaux se sont concentrés sur les méta-heuristiques.

Quelques travaux ont opté pour une généralisation des heuristiques de recherche locale, afin de les adapter au contexte multi-objectif. [Geiger 2010] a généralisé une méthode constructive combinée avec une recherche locale itérative (VNS) pour améliorer le coût du trajet total et la ponctualité des services. [Assis 2013] propose de traiter le VRP avec livraison, et ramassage optionnel (VRPPD). Une version multi-objectif de l'algorithme de recherche locale itérative (MOILS) est proposée pour optimiser de façon simultanée le coût de trajet et les demandes de ramassage satisfaites.

Concernant les méthodes globales, plusieurs méta-heuristiques ont été utilisés pour résoudre le VRP multi-objectif. Dans [Yang 2008], un algorithme d'optimisation par essaims particulaires (PSO) est utilisé pour optimiser le nombre de véhicules ainsi que la longueur maximale des routes.

D'autre part plusieurs autres travaux se sont focalisés sur les algorithmes évolutionnaires multi-objectifs comme dans [BanOs 2013] où un algorithme évolutionnaire multi-objectif hybride est proposé pour la résolution du problème de VRPTW multi-objectif. Dans ce travail, un algorithme évolutionnaire est couplé avec un algorithme recuit simulé (SA-EA) et une stratégie de recherche locale de type "multi-start". En plus de la distance parcourue, un deuxième objectif a été considéré qui consiste à équilibrer la charge de travail selon deux points de vue : le coût de trajet et les poids chargés. [Murata 2007] utilise l'algorithme NSGA-II pour le problème VRP multi-objectif, où le coût de trajet maximal des véhicules et leur nombre sont minimisés. [Sulieman 2010] propose d'étudier le problème de VRP avec demandes stochastiques (VRPSD). Deux modèles bi-objectifs ont été proposés : le modèle d'entropie et le modèle écart-type, et trois algorithmes évolutionnaires multi-objectif ont été utilisés à savoir : IBEA, MOGA et NSGA-II. Le premier modèle consiste a minimiser la distance parcourue par le véhicule par rapport aux demandes incertaines et l'entropie sur la distance, tandis que le deuxième utilise comme second objectif l'écart type des distances. [Psychas 2015] traite une variante multi-objectif du VRP, appelée "*Energy Reduction Vehicle Routing Problem*". Il considère trois fonctions objectifs : le temps de déplacement entre deux clients ou un client et un dépôt, la distance parcourue pour la livraison et celle pour le ramassage. Une version modifiée de NSGA-II, couplée avec la méthode VNS, a été proposée, où plusieurs populations évoluant en parallèle sont utilisées.

D'autres travaux utilisant les algorithmes évolutionnaires multi-objectif pour le problème VRP se sont concentrés sur les opérateurs de diversification, souvent utilisés en complément des critères de convergence (basés sur la dominance de Pareto). Parmi ces critères de diversification, nous pouvons citer : l'indice de surpeuplement, la grille adaptative [Zhang 2012], le plus proche voisin [Chand 2013a], la méthode histogramme [Chand 2013b], la mesure de similarité [Garcia-Najera 2011], la densité

[Yang 2008], la diversification élitiste [Jozefowiez 2009], etc.

Contrairement aux algorithmes évolutionnaires multi-objectifs classiquement conçus pour des problèmes paramétrés, ceux proposés pour le VRP multi-objectif sont présentés avec des opérateurs génétiques spéciales. Par conséquent, une adaptation de ces opérateurs au contexte VRP devient nécessaire. Ces opérateurs sont choisis en fonction des contraintes de la variante du VRP étudié (pour éviter les infaisabilités) et des objectifs optimisés.

Concernant les opérateurs de croisement, l'opérateur "Best-cost-best-route crossover" (BC-BR-C) est le plus utilisé, surtout, quand il s'agit de minimiser le coût de trajet et le nombre de véhicule en même temps. Plus récemment, [Pierre 2014] a proposé une version améliorée de cet opérateur appelée "Partially Optimized Cyclic Shift Crossover" (POCSX) qui permet de réduire le coût de calcul. [Chand 2013a] a utilisé deux opérateurs de croisement adaptés au problème VRP à savoir le "Best Route Crossover" (BRC), et le "Sub Route Mapped Crossover" (SRMC).

Parmi les opérateurs de mutation utilisés dans le contexte du VRP nous pouvons citer : "sequenced based mutation" (SBM) [Ghannadpour 2014], patial swap (PS) et "Sub Route Exchange Mutation" (SREM) [Chand 2013b].

Le tableau 2.3 propose de synthétiser quelques travaux traitant les variantes du problème VRP multi-objectif, en citant dans l'ordre : la variante du VRP étudiée, les objectifs considérés, les techniques de convergence et de diversification, le recours ou non à la parallélisation et enfin la méthode de résolution utilisée.

2.9 Conclusion

Dans ce chapitre nous avons parcouru l'état de l'art relatif à la prise en compte d'incertitudes dans le VRP, ainsi que l'optimisation multi-objectif. Pour la modélisation d'incertitudes, le choix du critère de robustesse s'avère être une étape importante du modèle de robustesse. Les principaux critères proposés dans la littérature sont soit très conservateurs ou bien très coûteux à mettre en œuvre. Ce constat a motivé les travaux sur la proposition d'un nouveau critère qui a pour objectif d'établir un compromis entre conservatisme et facilité de mise en œuvre. Ce travail fera l'objet du prochain chapitre. D'autre part, pour la résolution des variantes VRP robustes multi-objectifs proposés, il existe peu de travaux qui se sont intéressés à la résolution du problème en utilisant une approche multi-objectif basée sur l'optimalité de Pareto. La plupart des travaux utilisent les techniques classiques d'agrégations, qui sont limitées à cause de la subjectivité qu'elles incorporent dans la fonction objectif. Par ailleurs, il n'existe pas, à notre connaissance de méthodes multi-objectif qui s'intéressent à d'autres objectifs que le profit, considéré dans la totalité des méthodes mono-objectif de l'état de l'art sur TOP.

TABLE 2.3 – Synthèse des travaux du VRP multi-objectif. TDB : temps de trajet entre deux clients. P : le profit. TAP : temps d'accessibilité prévu. E : entropie. D : écart type. CT : compacité des route. L : demandes incertaines. TW : fenêtres de temps incertaines.

Auteurs	Variante VRP	Objectifs	Convergence	Diversification	Parallélisation	Méthode de résolution
Tan et al. [Tan 2006]	VRPTW	NV, TD	Optimum de Pareto	/	/	HMOEA (GA)
Ombuki et al. [Ombuki 2006]	VRPTW	NV, TD	Optimum de Pareto	/	/	GA + BC-BR-C
Cheong et al. [Cheong 2006, Tan 2007]	VRPTWSD	NV, TD, RC	Optimum de Preto	/	/	HMOEA
Murata et al. [Murata 2007]	VRP	NV, LM	Optimum de Pareto	Indice de surpeuplement	/	NSGA-II+mesures de similarité
Ombuki-Berman et al. [Ombuki-Berman 2007]	VRPTW	NV, TD, ER, CT	Optimum de Pareto	/	/	MOEA (GA)
Yang [Yang 2008]	VRPFDT	NV, LM	Optimum de Pareto	Densité	oui	PSO
Jozefowiez et al. [Jozefowiez 2009]	VRPRB	TD, ER	Optimum de Pareto	diversification élitiste+sharing	oui	HMOEA
Ghoseiri et al. [Ghoseiri 2010]	VRPTW	TD, NV	Optimum de Pareto	/	/	GA+"goal programming"
Geiger et al. [Geiger 2010]	VRP	TD, TR	Optimum de Pareto	/	/	Savings+LS
Sulieman et al. [Sulieman 2010]	VRPSD	TD, E/ TD, D	Optimum de Pareto	Selon l'algorithme	/	IBEA,NSGAII,MOGA
Garcia-Najera et al. [Garcia-Najera 2011]	VRPTW	TD, TDT, NV	Optimum de Pareto	Mesure de similarité	/	MOEA
Assis et al. [Assis 2013]	VRPPD	TD, SC	Optimum de Pareto	Indice de surpeuplement	/	MOILS
Zhang et al. [Zhang 2012]	VRPFTW	(TD+FC,TR), SCR	Challenge Cup rules	Grille adaptative	/	MOQEA
BanOs et al. [BanOs 2013]	VRPTW	TD, ER	Optimum de Pareto	/	/	Multi-start SA-EA
Zhou et al. [Zhou 2013]	VRPTW	TD, ER	Optimum de Pareto	/	/	GA
Chand et al. [Chand 2013a]	VRP	TD, NV	Optimum de Pareto	Plus proche voisin	/	GA+SBRC+PS
Chand et al. [Chand 2013b]	VRP	TD, NV	Optimum de Pareto	Méthode hystogramme	/	GA+SRMC+SEMM
Ghannadpour et al. [Ghannadpour 2014]	DVRPFTW	TD, TR, NV, SC	Optimum de Pareto	/	/	GA + BC-BR-C + SBM
Pierre et al. [Pierre 2014]	VRPTW	TD, NV	Optimum de Pareto	/	/	GA+POCSX
Psychas et al. [Psychas 2015]	VRP	TDB, CPR, CPL	Optimum de Pareto	Indice de surpeuplement	oui	NSGA-II+VNS

Chapitre 3

Critère de robustesse MNSQW

Sommaire

Les travaux présentés dans ce chapitre ont fait l'objet des publications [Wu 2015] et [Wu 2017].

3.1 Introduction

Durant les dernières années, l'évolution des technologies de l'information a permis d'améliorer de façon efficace la qualité du réseau social. Cependant, dans le cas des applications du monde réel, plusieurs informations sont malheureusement souvent inconnues ou incertaines, telles que le rendement des actifs à risque après un an, le temps de trajet d'une route le lendemain matin, etc. Par conséquent, la prise de décision sous incertitudes peut être rencontrée dans de nombreux domaines, tels que le transport, la logistique, les télécommunications, la fiabilité et la gestion de la production, etc. Malgré les progrès technologiques des dernières années, résoudre un problème d'optimisation combinatoire, même avec des paramètres certains reste un sujet difficile.

La plupart des modèles mathématiques tentent de faire face à l'incertitude en remplaçant une donnée incertaine par une série de paramètres déterministes tels que l'intervalle de variation, une valeur d'expérience, la variance, etc. Par exemple,

dans la programmation mathématique, certaines approches ont déjà été proposées pour représenter approximativement des événements incertains. Parmi les approches de l'état de l'art qui se sont distinguées on peut citer : l'analyse de sensibilité et l'optimisation stochastique.

L'analyse de sensibilité peut être considérée comme une procédure d'évaluation post-calcul qui vise à trouver un intervalle de variation pour chaque paramètre incertain. Un tel intervalle est utilisé pour garantir la stabilité d'une solution optimale en considérant que certains paramètres du problème d'origine peuvent être perturbés. En optimisation stochastique, les événements incertains sont caractérisés par leurs distributions de probabilité. Cependant, les informations relatives à chaque événement incertain sont généralement incomplètes ou inconnues, ce qui complique la détermination de sa distribution de probabilité. Une alternative possible pour palier à ce problème, serait d'essayer d'associer à chaque paramètre incertain, un ensemble de valeurs qui peuvent également être qualifiées de scénarios. Chaque scénario correspond à une valeur potentielle pouvant être atteinte par un paramètre incertain. Dans le cas idéal, on recherche une solution qui fournit les meilleures performances sur tous les scénarios possibles. Une telle solution est souvent difficile à trouver et, parfois, n'existe pas. Dans ce contexte, l'un des objectifs importants de l'optimisation robuste est de développer des critères de décision utilisés afin de caractériser la robustesse d'une solution durant l'optimisation.

Il est à noter que l'optimisation robuste a été proposée pour une variété de problèmes d'optimisation combinatoire (voir Kouvelis et Yu (1997) [Kouvelis 1997] et Gabrel, Murat et Thièle (2014) [Gabrel 2014]). Dans ce chapitre, nous proposons un nouveau critère robuste pour le problème de tournées de véhicules avec incertitude sur le temps de déplacement (RVRP) (cf. Bertsimas et Simchi-Levi, 1996 [Bertsimas 1996]). L'approche proposée est basée sur le concept théorique développé par Roy (2010) [Roy 2010] selon lequel une solution qualifiée sur la majorité des scénarios n'est jamais trop mauvaise.

Le problème VRP, décrit dans le chapitre précédent, est connu pour être un problème NP-difficile. L'objectif du problème de VRP est de déterminer la solution approvisionnant tous les clients avec un coût de transport minimal. Dans la version standard du VRP, le coût de transport est habituellement évalué par la distance parcourue. Cependant, dans les applications du monde réel, le coût du transport dépend fortement du temps de déplacement. Or contrairement à la distance de déplacement, le temps de déplacement est souvent incertain, particulièrement lors de son évaluation dans le futur. Afin de développer des modèles informatiques permettant la gestion de l'incertitude liée aux paramètres du problème, il est devenu plus intéressant de concevoir des modèles robustes basés sur des scénarios discrets (voir Adida et Perakis, 2010 [Adida 2010] ; Han, Lee et Park, 2013 [Han 2013]).

Dans ce travail, le temps de déplacement est représenté par un ensemble de scénarios, où chaque scénario représente une valeur potentielle du temps de déplacement requis pour le parcours d'un itinéraire. Par ailleurs, une solution est considérée comme robuste, si elle est qualifiée selon un critère robuste préfixé. Notons que dans les modèles d'optimisation robustes, un critère robuste est souvent considéré

comme une fonction objective. Par conséquent, trouver la meilleure solution pour un critère robuste équivaut à déterminer la meilleure solution optimisant la fonction objective liée au critère robuste. Les critères robustes élaborés dans la littérature sont généralement basés sur l'attitude du décideur vis à vis du risque (cf. Zymler, Kuhn, et Rustem, 2013 [Zymler 2013] ; Zhu et Fukushima, 2009 [Zhu 2009]). Parmi les critères robustes proposés dans la littérature, nous retrouvons : le critère du meilleur cas, le critère du pire cas (cf. Solano-Charris, Prins et Santos, 2015 [Solano-Charris 2015]), le critère min-max déviation (voir Aissi, Bazgan et Vanderpooten, 2009 [Aissi 2009]) et le critère bw-robustesse (meilleur-pire) (cf. Gabrel, Murat et Wu, 2013 [Gabrel 2013b]). (voir le chapitre précédent pour plus de détails).

En plus de la définition des modèles et le développement des algorithmes,une autre difficulté qui peut être rencontrée par l'optimisation robuste réside dans l'évaluation des solutions (c'est-à-dire soit optimale ou approximative) obtenues en utilisant différents critères robustes. [Roy 2010] a proposé trois mesures de robustesse pour un critère considéré, qui peuvent être résumées comme suit :

- **Conclusions parfaitement robustes :** l'optimalité de la solution a été prouvée en utilisant un algorithme exact ;
- **Conclusions approximativement robustes :** la solution a été approximativement déterminée en fournissant le rapport d'approximation ;
- **Conclusions pseudo-robustes :** la solution a été calculée en utilisant une méta-heuristique sans fournir d'informations sur son optimalité.

En raison de la complexité importante du RVRP, fournir une conclusion parfaitement robuste pour un critère robuste devient impossible, surtout, lorsque la taille du problème augmente.A cet effet, il existe peu de travaux qui se sont intéressés à l'élaboration de critères robustes pour la famille du problème de tournées des véhicules. Le travail présenté dans ce chapitre a pour objectif de proposer un mécanisme qui fournit, soit la conclusion parfaitement robuste, soit la conclusion pseudo-robuste pour un critère robuste considéré. Le critère robuste que nous proposons est évalué sur deux ensembles d'instances. Le premier ensemble contient les instances à petite échelle et le second comprend les instances à moyenne et à grande échelle. Sur le premier ensemble, nous appliquons un solveur exact (Soveur Cplex) pour résoudre le problème à son optimalité et fournir les conclusions parfaitement robustes. Les solutions optimales obtenues sont comparées à celles fournies en utilisant des critères robustes classiques. Sur le deuxième benchmark, nous adoptons une méta-heuristique basée sur la recherche à voisinage large pour résoudre approximativement le RVRP. Afin d'évaluer la performance du nouveau critère robuste et de l'heuristique proposée, nous comparons les solutions approximatives obtenues avec celles fournies par le solveur Cplex dans un temps d'exécution limité.

Le reste de ce chapitre est organisé comme suit. La section 3.2 introduit quelques critères classiques basés sur une approche de programmation linéaire entière pour le RVRP, et décrit le nouveau critère que nous proposons pour le RVRP en donnant un exemple numérique pour illustrer ses propriétés. La section 3.3 résume le principe de la méta-heuristique proposée. Dans la section 3.4, le nouveau critère est évalué en utilisant à la fois l'algorithme exact et la méta-heuristique afin de fournir des

conclusions parfaitement robustes et des conclusions pseudo-robustes. La section 3.5 conclut ce chapitre

3.2 Modèle d'incertitudes avec scénarios discrets

Dans cette section, nous décrivons quelques définitions et notations qui seront utilisées dans la suite. A noter que dans ce chapitre, nous considérons la variante du VRP capacitaire (CVRP). Étant donné un dépôt central (noté v_0), un ensemble de n clients $V = \{1, ..., n\}$ et un ensemble de véhicules identiques, chaque client i a une quantité de marchandises c_i à délivrer et chaque véhicule est caractérisé par une capacité C. L'objectif du CVRP est de déterminer une liste des itinéraires réalisables desservant tous les clients avec une distance de déplacement minimale et en utilisant un nombre de véhicules minimal. En effet, une instance du VRP peut être représentée sous la forme d'un graphe complet $G = (V, E)$, où V désigne l'ensemble des sommets et $E = \{(i, j)/i, j \in V\}$ désigne l'ensemble des arcs. Par conséquent, un programme linéaire standard pour le VRP (voir, Toth & Vigo, 2002 [Toth 2002]) peut être décrit comme suit :

$$
(ILP_{vrp}) \begin{cases}
Z = & \min \mu \times m + \sum_{i=1}^{n}(t_{0i}x_{0i} + t_{i0}x_{i0}) + \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij}x_{ij} & (1) \\
s.c. & & \\
& \sum_{i=1}^{n} x_{0i} \leq m \quad \text{and} \quad \sum_{i=1}^{n} x_{i0} \leq m & (2) \\
& \sum_{j=1}^{n} x_{ij} = 1, \ \forall i = \{1, ..., n\} & (3) \\
& \sum_{j=1}^{n} x_{ji} = 1, \ \forall i = \{1, ..., n\} & (4) \\
& \ell_j - \ell_i - C \times x_{ij} \leq c_j - C, \ \forall i \neq j = \{1, ..., n\} & (5) \\
& c_i \leq \ell_i \leq C, \ \forall i = \{1, ..., n\} & (6) \\
& x_{ij} \in \{0, 1\} \forall i, j = \{1, ..., n\}, \ \ell_i \in \mathbb{N}, \ \forall i = \{1, ..., n\}, \ m \in \mathbb{N} & (7)
\end{cases}
$$

Où, t_{ij} représente le temps de déplacement du client i au client j ; ℓ_i est la charge actuelle du véhicule lorsque le client i est desservi ; la variable de décision $x_{ij} = 1$ si l'un des véhicules passe par l'arc (i, j) (dans l'itinéraire), sinon elle est égale 0 ; m représente le nombre de véhicules utilisés. La fonction objectif (1) vise à minimiser le nombre de véhicules utilisés et le temps de déplacement total dans la même fonction. Le paramètre μ mesure l'impact de la minimisation du nombre de véhicules utilisés sur la fonction objective. L'inégalité (2) garantit que le nombre de véhicules qui sortent (ou retournent) du dépôt ne doit pas dépasser la valeur de la variable m. Les contraintes (3) et (4) sont utilisées pour assurer la propriété du circuit, d'où un cycle commence du dépôt et y retourne. Comme une route (un véhicule) est représentée par un seul circuit, les contraintes (5) sont utilisées pour éliminer tous les sous-circuits. La contrainte (6) est utilisée pour assurer la capacité du véhicule. Enfin, la contrainte (7) garantit l'intégralité de toutes les variables de décision.

Dans la littérature, les contraintes (5) sont référencées comme les contraintes d'élimination de sous-circuits de Miller-Tucker-Zemlin (MTZ), qui sont habituellement considérées comme faibles (cf. Miller, Tucker et Zemlin, 1960 [Miller 1960]). Contrairement aux contraintes de MTZ, les contraintes d'élimination de sous-circuits de Dantzig-Fulkerson-Johnson (DFJ) sont connues par la communauté comme de fortes contraintes d'élimination de sous-circuits (voir Dantzig, Fulkerson et Johnson, 1954). Ces dernières sont basées sur les contraintes de la clique et sont capables de calculer de meilleures bornes de relaxation linéaire que celles calculées par MTZ. Étant donné que les contraintes DFJ sont de complexité exponentielle en fonction du nombre de clients, l'application de DFJ pour éliminer les sous-circuits lorsque la taille de l'instance est importante, devient difficile. Comme le montrent Bekta et Gouveia (2014) [Bektaş 2014], la résolution d'un programme de relaxation linéaire basé sur les contraintes DFJ est coûteuse en terme de temps d'exécution. Pour cette raison, nous avons opté dans ce travail pour l'application des contraintes MTZ dans notre modèle pour résoudre exactement les petites instances scalaires considérées pour le RVRP.

Dans le reste de ce chapitre, $D(.)$ est utilisé pour désigner le coût de déplacement pour parcourir une route (R) ou le coût total de déplacement pour une liste de routes (LR). En effet, le coût de déplacement de LR (c'est-à-dire $D(LR)$) est équivalent à la fonction d'objectif (1) moins le nombre de véhicules utilisés. En bref, le VRP peut être décrit comme suit :

$$(ILP_{vrp}) \begin{cases} Z = & \min D(LR) = \alpha \times m + \sum_{\forall R \in LR} D(R) \\ s.c. & \\ & contraintes\ (2) - (7) \end{cases}$$

Ces dernières années, plusieurs travaux de recherche se sont intéressés au développement des approches à base de scénarios discrets pour fournir une solution robuste sous incertitude. Pour le RVRP, le coût de déplacement pour parcourir un itinéraire est approximativement évalué par un ensemble de q scénarios discrets, où chaque arc $(i, j) \in E$ est caractérisé par un ensemble fini de q valeurs $\{t_{ij}^1; ...; t_{ij}^q\}$. En d'autres termes, le coût de déplacement pour chaque itinéraire est évalué par q observations. Il s'ensuit que chaque plan d'acheminement faisable pour le RVRP (c'est-à-dire, LR) est également caractérisé par q valeurs : $\{D(LR)^1; ...; D(LR)^q\}$. Par conséquent, l'objectif du problème RVRP est de déterminer une solution robuste ayant de bonnes performances sur toutes les observations (ou scénarios). Dans ce qui suit, nous allons présenter certains critères classiques utilisés pour évaluer la robustesse de la solution. Ces critères sont généralement basés sur le comportement des humains lorsqu'ils prennent une décision sous incertitude. Le but du travail présenté est de concevoir un nouveau critère robuste afin de fournir des solutions qui semblent plus raisonnables que celles fournies par les critères classiques.

3.2.1 Le critère de meilleur cas : (meil)

Le critère de meilleur cas peut être formellement défini comme suit :

$$(ILP_{vrp}^{meil}) \begin{cases} Z = & \min_{i=\{1,...,q\}} \min D^i(LR) \\ s.c. & \\ & contraintes\ (2)-(7) \end{cases}$$

Un tel cas consiste à trouver la meilleure solution pour tous les scénarios considérés. Généralement, la solution fournie par P_{meil} a toujours été considérée comme la décision la plus risquée. Un tel décideur est connu être un décideur amoureux du risque. Le décideur essaie de trouver la meilleure possibilité qui permet de maximiser son gain. Partant de ce fait, la solution fournie pourrait se concentrer uniquement sur un scénario spécifique (le meilleur scénario), et ignore les autres scénarios. Une telle solution pourrait être extrêmement mauvaise lorsque les scénarios sont diversifiés.

3.2.2 Le critère du pire cas : (pire)

Contrairement au critère de meilleur cas, ce critère du pire cas tente de fournir une solution minimisant la pire évaluation sur tous les scénarios :

$$(ILP_{vrp}^{pire})\ \min_{i=\{1,...,q\}} \max\ D^i(LR)$$

Le programme linéaire en nombre entier correspondant peut être écrit comme suit :

$$(ILP_{vrp}^{pire}) \begin{cases} \min \sigma & & (8) \\ s.c. & & \\ & D^i(LR) \leq \sigma & \forall i = \{1,...,q\} \\ & contraintes\ (2)-(7) & \end{cases}$$

En choisissant ce type de solution, le décideur cherche à éviter tous les risques car il est réticent au risque. Nous pouvons constater que les modèles basés sur les critère meilleur et pire se réfèrent à deux stratégies de décision extrêmes.

3.2.3 Le critère de min-max Déviation : (Dev)

Dans la gestion des risques, la recherche de solutions optimales en tenant compte des différentes préférences face au risque est toujours encourageante. Dans la littérature, une alternative classique aux cas ci-dessus consiste à minimiser l'écart maximal, pour tous les scénarios considérés, entre la valeur objective courante et la valeur optimale (voir Aissi et al., 2009 [Aissi 2009]). Soit Opt^i la valeur objective optimale pour le scénario i ($\forall i = \{1,...,q\}$), le programme linéaire basé sur le critère de min-max déviation peut être écrit comme suit :

$$(ILP_{vrp}^{dev}) \begin{cases} \min Dev & & (9) \\ s.c. & & \\ & D^i(LR) - Opt^i \leq Dev \forall i = \{1,...,q\} & (10) \\ & contraintes\ (2)-(7) & \end{cases}$$

Nous rappelons également que le VRP est un problème d'optimisation combinatoire NP-difficile bien connu. Résoudre le VRP à l'optimum, pour chaque scénario i,

$\forall i = \{1, ..., q\}$, est intraitable et son temps d'exécution devient très coûteux, surtout lorsque le nombre de clients ou le nombre de scénarios augmente. Par conséquent, le critère de min-max déviation n'est pas convenable pour le RVRP, parce que même la construction de ILP^{dev}_{rvrp} a déjà une grande complexité.

3.2.4 Nouveau modèle robuste

Dans cette section, nous proposons un modèle modifié inspiré du critère du pire cas. Soit $Pire^*$ la solution optimale du problème ILP^{pire}_{vrp} et soit $D^i(Pire^*)$ la valeur objective liée à $Pire^*$ sur le scénario i ; $\forall i = \{1, ..., q\}$. Le modèle proposé vise à trouver une solution LR' maximisant le nombre de scénarios tels que $D^i(LR')$ fonctionne mieux que $D^i(Pire^*)$; $\forall i = \{1, ..., q\}$. Pour le RVRP, le modèle peut être formulé comme suit :

$$(ILP^{nouv}_{vrp}) \begin{cases} \max \sum_{i=1}^{q} s_i & (11) \\ s.c. & \\ \quad D^i(LR) \leq W \times (1 - s_i) + D^i(Pire^*) \times s_i \forall i = \{1, ..., q\} & (12) \\ \quad s_i \in \{0, 1\} \ \forall i = \{1, ..., q\} & \\ \quad contraintes\ (2) - (7) & \end{cases}$$

Où, la variable de décision $s_i = 1$ si la solution actuelle offre un temps de déplacement strictement inférieur à $D^i(Pire^*)$; $s_i = 0$ autrement ; W est une constante relativement grande. Dans notre cas, W est expérimentalement égal à 2 fois $D^i(Pire^*)$. La fonction objectif (11) maximise le nombre de scénarios qui se comportent mieux que ceux de $Pire^*$. Enfin, l'inégalité (12) garantit que $s_i = 0$ si la solution actuelle ne parvient pas à améliorer le temps de déplacement de $Pire^*$ pour le i-ème scénario.

Nous rappelons que ILP^{nouv}_{rvrp} est basé sur le critère du pire cas et son objectif (fonction objectif) consiste à maximiser le nombre de scénarios réalisant un meilleur temps de trajet que celui réalisé par la solution optimale de ILP^{Pire}_{rvrp} (c'est-à-dire $Pire^*$). Dans le reste du chapitre, le critère proposé est référencé par "maximisation du nombre de scénarios qualifiés par le pire" (Maximizing the Number of Scenarios Qualified by the Worst, MNSQW). Soit $MNSQW^*$ la solution optimale de ILP^{nouv}_{rvrp}. À partir du modèle ILP^{nouv}_{rvrp}, on peut observer qu'à chaque fois que le nombre de scénarios, pour lequel la solution optimale de ILP^{nouv}_{rvrp} (c'est-à-dire $MNSQW^*$) domine $Pire^*$ dépasse $q/2$, $MNSQW^*$ est considéré comme plus robuste que $Pire^*$. Cela veut dire que la solution actuelle satisfait la majorité des scénarios. De façon évidente, $MNSQW^*$ doit être égal à $Pire^*$ si la valeur objective de ILP^{nouv}_{rvrp} est inférieure à $q/2$. Formellement, nous faisons la proposition suivante qui énonce le critère robuste utilisé par le modèle ILP^{nouv}_{rvrp}.

Proposition 1. Soit V^* la valeur objective optimale de ILP^{nouv}_{rvrp} et q le nombre total de scénarios disponibles.

— Si $V^* > q/2$, soit $MNSQW^*$ égal à la solution optimale qualifiée par MNSQW ;
— Sinon, soit $MNSQW^*$ égal à la solution optimale de ILP^{pire}_{rvrp}.

L'exemple suivant illustre la robustesse de toutes les solutions robustes obtenues grâce à l'utilisation de différents critères robustes.

Exemple 1. Soit G un graphe indirect comme illustré à la Fig 3.1. Le temps de déplacement de chaque arc est représenté par 5 scénarios. Le tableau 3.1 détaille les solutions optimales obtenues en considérant chaque scénario séparément et celles obtenues selon différents critères robustes. Par exemple, Opt^1 correspond à la solution optimale en considérant uniquement le temps de déplacement lié au scénario 1, et $Meilleur^*$ correspond à la solution optimale en utilisant le critère robuste de meilleur cas, etc. Le tableau 3.2 montre le nombre de scénarios sur lesquels la solution considérée fonctionne mieux que la solution optimale qualifiée par le critère de pire cas ($Pire^*$).

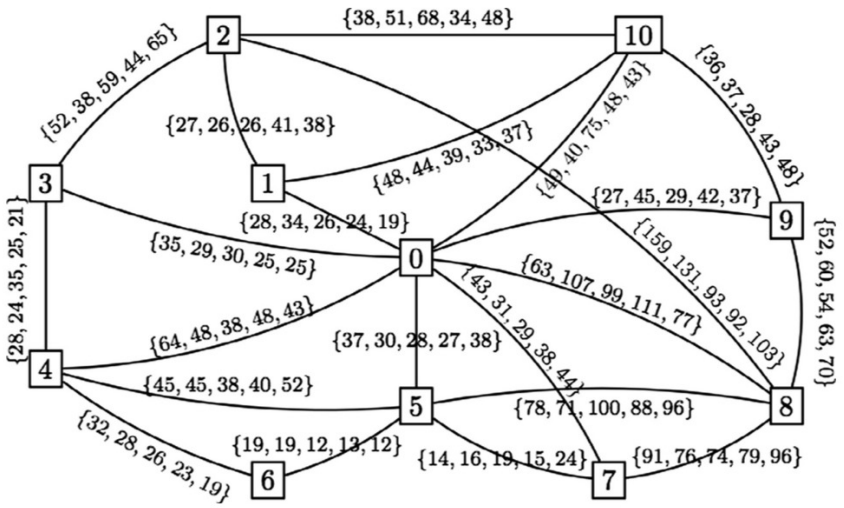


FIGURE 3.1 – Une description graphique d'une instance de RVRP avec 5 scénarios

Critères	LR	$D^1(LR)$	$D^2(LR)$	$D^3(LR)$	$D^4(LR)$	$D^5(LR)$
Opt^1	r1 :{0 ;3 ;4 ;6 ;5 ;7 ;0} ; r2 :{0 ;8 ;9 ;10 ;2 ;1 ;0}	415	462	452	455	445
Opt^2	r1 :{0 ;6 ;4 ;3 ;2 ;1 ;0} ; r2 :{0 ;7 ;5 ;8 ;9 ;10 ;0}	476	436	515	495	510
Opt^3	r1 :{0 ;2 ;1 ;10 ;8 ;9 ;0} ; r2 :{0 ;7 ;5 ;6 ;4 ;3 ;0}	496	494	415	510	498
Opt^4	r1 :{0 ;1 ;10 ;2 ;8 ;9 ;0} ; r2 :{0 ;3 ;4 ;6 ;5 ;7 ;0}	523	512	460	427	459
Opt^5	r1 :{0 ;3 ;4 ;6 ;5 ;7 ;0} ; r2 :{0 ;8 ;9 ;10 ;2 ;1 ;0}	415	462	452	455	445
$Meilleur^*$	r1 :{0 ;2 ;1 ;10 ;8 ;9 ;0} ; r2 :{0 ;7 ;5 ;6 ;4 ;3 ;0}	496	494	415	510	498
$Pire^*$	r1 :{0 ;3 ;4 ;6 ;5 ;7 ;0} ; r2 :{0 ;8 ;9 ;10 ;2 ;1 ;0}	415	462	452	455	445
Dev^*	r1 :{0 ;3 ;4 ;6 ;5 ;7 ;0} ; r2 :{0 ;9 ;8 ;10 ;2 ;1 ;0}	451	460	423	458	467
$MNSQW^*$	r1 :{0 ;3 ;4 ;0} ; r2 :{0 ;6 ;5 ;7 ;8 ;9 ;10 ;2 ;1 ;0}	469	451	448	453	467

TABLE 3.1 – Les solutions pour chaque chemin de 1 à 5.

D'après les tableaux 3.1 et 3.2, on peut observer ce qui suit :

Opt^1	Opt^2	Opt^3	Opt^4	Opt^5	$Meilleur^*$	Dev^*	$MNSQW^*$
0	1	1	1	0	1	2	3

TABLE 3.2 – Les scénarios qualifiés par rapport au critère du pire cas.

— $Pire^*$ coïncide avec Opt^1 (resp. Opt^5), qui représente une solution optimale en considérant le premier (resp. le cinquième) scénario ;
— $Pire^*$ domine les autres solutions optimales sur la plupart des scénarios : $Opt^2, Opt^3, Opt^4, Meilleur^*$ et Dev^* ;
— $MNSQW^*$ atteint un meilleur temps de déplacement que $Pire^*$ sur 3 scénarios : $D^2(LR)$; $D^3(LR)$ et $D^4(LR)$.

D'après le tableau 3.1, nous remarquons également que $MNSQW^*$ est équivalent à Dev^*. En effet, $MNSQW^*$ domine Dev^* sur deux scénarios ($D^2(LR)$; $D^4(LR)$), est dominés par Dev^* sur deux autres ($D^1(LR)$; $D^3(LR)$) et donne le même temps de déplacement pour le cinquième scénario ($D^5(LR)$).

La raison principale pour laquelle nous adoptons $Pire^*$ comme référence pour évaluer l'autre solution robuste est que $Pire^*$ est considéré comme la stratégie avec le moindre risque. Soit LR une solution potentielle du RVRP, soit B l'ensemble des scénarios où LR est meilleure que $Pire^*$; soit W l'ensemble des scénarios où LR effectue un chemin de coût plus mauvais que $Pire^*$. Par conséquent, les valeurs liées aux scénarios appartenant à B peuvent être considérées comme le bonus obtenu en augmentant son risque sur les scénarios appartenant à W. Par exemple, en remplaçant $Pire^*$ par $MNSQW^*$, nous pourrions améliorer notre gain sur les scénarios 2 ; 3 et 4, mais augmenterions notre perte sur les scénarios 1 et 5. Si le décideur est prêt à supporter le risque sur les scénarios 1 et 5 en améliorant le gain sur les scénarios 2 ; 3 et 4, $MNSQW^*$ est considéré comme plus robuste que $Worst^*$ pour le décideur.

3.3 Heuristique pour la validation du critère MNSQW

Nous présentons dans cette section l'heuristique que nous avons utilisée pour valider le critère proposé MNSQW. La recherche à voisinage large (Large Neighborhood Search, LNS), proposée par Shaw (1998) [Shaw 1998], est une technique courante utilisée pour améliorer une solution locale. LNS consiste à construire et à explorer alternativement le voisinage issu d'une solution locale donnée. Dans les travaux de recherches les plus récents, les heuristiques basées sur le LNS ont suscité un grand intérêt pour l'optimisation combinatoire, en particulier pour résoudre les problèmes d'optimisation à grande échelle (voir par exemple Pisinger & Ropke, 2010 [Pisinger 2010] ; Hansen, Mladenovic, Brimberg et Pérez, 2010 [Hansen 2010] ; Hifi, Saleh, & Wu, 2014 [Hifi 2014] et Wei, Zhang, Zhang & Lim, 2015 [Wei 2015]). Dans cette section, nous introduisons une heursitique hybride, qui applique une procédure basée sur LNS pour faire varier l'espace de solutions, puis effectue une série de mouvements locaux pour améliorer la solution actuelle.

Algorithme 2 La procédure de recherche à voisinage large

Précondition : Une instance de RVRP et un critère de robustesse c.
Postcondition : Meilleure solution possible LR_{best} pour G.
Définir une fonction objectif, notée par OF_c, selon le critère c ;
Appliquer une procédure gloutonne, en se basant sur OF_c, afin de trouver une solution faisable LR pour G ;
Soit $LR_{best} = LR_s = LR$;
tant que max_{iter} n'est pas atteint
Construire de manière aléatoire des voisins $N(LR_s)$ de LR_s en supprimant $\alpha \times n$ visites de LR_s ;
Réappliquez la procédure glouton pour explorer l'espace de voisinage $N(LR_s)$ et garder LR comme solution actuelle ;
Appliquer des stratégies d'amélioration locales pour améliorer la solution LR ;
si $OF_c(LR)$ est meilleure que $OF_c(LR_{best})$ **alors**
  remplacer LR_{best} par LR ;
fin si
si $OF_c(LR)$ est pire que $OF_c(LR_{best}) \times \beta$ **alors**
  $LR_s = LR_{best}$;
sinon
  $LR_s = LR$;
fin si
fin tant que
retourner LR_{best} comme étant la meilleure solution possible pour G.

L'algorithme 2 décrit les étapes principales de l'heuristique LNS pour résoudre une instance du problème du RVRP. L'algorithme proposé est alors référencé sous le nom de LNSR. Cet algorithme requiert une instance G de RVRP en entrée, les étapes 1-3 visent à fournir une solution initiale faisable pour le RVRP donné en utilisant une procédure gloutonne (voir les sections 4.1 et 4.2). La boucle principale de LNSR (les étapes 4-10) s'arrête quand le nombre d'itérations maximal (noté max_iter) est atteint. L'étape 5 applique une procédure aléatoire pour supprimer $\alpha \times n$ clients de la liste actuelle des routes (c.-à-d., LR), où n est le nombre total de clients visités. L'espace de solutions résultant est alors noté $N(LR_s)$. $N(LR_s)$ peut également être considéré comme un voisinage de LR_s. L'étape 6 ré-applique la procédure gloutonne sur $N(LR_s)$ pour fournir une nouvelle solution du RVRP. À l'étape 7, une série de mouvements locaux est appliquée pour améliorer la solution actuelle LR. À l'étape 8, LR_{best} est mis à jour par la meilleure solution obtenue lors de toutes les itérations précédentes. L'étape 9 ajuste la solution initiale pour la prochaine itération de LNSR. En effet, si la fonction objectif de LR est bien pire que celle de LR_{best}, nous adoptons LR_{best} comme solution de départ pour la prochaine itération ; Sinon, $LR_s = LR$. Le paramètre β est utilisé pour déterminer le seuil pour utiliser soit la solution actuelle (c'est-à-dire LR) ou la meilleure solution (LR_{best}) comme point de départ pour la prochaine itération. LNSR finit par trouver la meilleure solution optimale locale jusqu'à ce que le nombre d'itérations maximal fixe soit dépassé. Rappelons que le critère MNSQW est basé sur celui du pire cas. Par conséquent, il faut d'abord résoudre ILP^{pire}_{rvrp} afin de trouver soit une solution optimale, soit une solution réalisable Pour ILP^{nouv}_{rvrp}. Dans ce qui suit, nous allons

décrire comment l'algorithme 2 peut être appliqué pour résoudre les deux modèles, c'est-à-dire le pire cas et le MNSQW.

3.3.1 La recherche à voisinage large selon le critère de pire cas

L'idée fondamentale de la procédure gloutonne utilisée pour trouver une solution réalisable pour ILP^{pire}_{rvrp} consiste à ajouter itérativement les clients non visités dans une liste de routes. Au début, nous générons la liste de clients selon un ordre aléatoire. À chaque étape, nous essayons de localiser le premier client disponible dans la liste à une position (o_i, o_j), de sorte que l'emplacement du client actuel à (o_i, o_j) induit l'augmentation la plus basse pour OF^{pire} (voir l'équation (8)). Comme mentionné précédemment, l'algorithme basé sur le LNS est composé d'une procédure de suppression de clients et une autre procédure d'exploration. Dans notre cas, la procédure d'exploration peut être considérée comme l'application de la procédure gloutonne sur la solution réduite après l'application de la procédure de suppression.

Pour la procédure de suppression, nous appliquons un concept similaire à celui utilisé par Shaw (1998) [Shaw 1998]. Compte tenu d'une solution réalisable, un client est d'abord supprimé de la solution actuelle (c'est-à-dire une liste de routes). Pour les prochaines itérations, le rapprochement entre les clients qui sont réellement présents dans la solution et le client qui a été supprimé à la dernière itération (noté v) est mesuré. Deuxièmement, un client est supprimé au hasard selon la formule suivante :

$$Rapprochement_{iv} = \frac{1}{1+WorstTime_{iv}} \quad \forall i \in V \quad et \quad \tfrac{1}{2} i \notin LR \quad (13)$$

Où, $WorstTime_{iv} = max\{t^1_{iv}; ...; t^q_{iv}\}$. Soit N^r le nombre total de clients qui sont effectivement présents dans LR (c'est-à-dire, sans tenir compte des clients supprimés). Selon la formule (13), nous ordonnons d'abord tous les clients disponibles dans LR par ordre décroissant de leurs valeurs de rapprochement. Ensuite, nous supprimons le i_r-th client, de la liste ordonnée, de la LR. La valeur de i_r peut être calculée comme suit :

$$i_r = N^r \times rn^{\lambda},$$

Où rn est un nombre réel généré aléatoirement dans $[0, 1[$

3.3.2 La recherche par voisinage selon le critère MNSQW

Pour MNSQW, LNSR est généralement le même que celui pour Pire. Le changement le plus important est la fonction objectif utilisée pour exécuter la procédure gloutonne. Soit $Pire^0$ la meilleure solution trouvée par LNSR pour *pire* et soit $(o_i, o_j)^s$ la position où le client sera situé. La fonction objectif OF_{MNSQW} appliquée par l'algorithme glouton pour trouver une solution faisable à ILP^{nouv}_{rvrp} est basée sur la formule suivante :

$$(o_i, o_j)^s = argmin_{\forall o_i \in LR \quad \forall o_j \in LR} \sum_{i=1}^{q} \{max\{(D^i(pire^0) - D^i LR^0)), 0\}\},$$

Où LR^0 est la liste des routes quand le client est positionné dans la position (o_i, o_j).

3.3.3 Recherche locale

La recherche locale est une méthode largement utilisée pour affiner la qualité de l'optima local. Dans ce travail, nous avons essayé d'appliquer plusieurs mouvements locaux efficaces (voir Groër, Golden, & Wasil, 2010 [Groër 2010]) afin d'améliorer les performances de LNSR. Quatre mouvements locaux sont introduits dans LNSR :

- Un point : délocalise un client assigné dans une nouvelle position ;
- Deux points : échange les positions de deux clients assignés ;
- Deux-opt : supprime deux arêtes existantes de la solution actuelle et produit une nouvelle solution en ajoutant deux nouvelles arêtes ;
- Trois-opt : supprime trois arêtes qui existent dans l'itinéraire et donne un nouveau chemin en ajoutant trois nouvelles arêtes ;

En effet, pour tous ces mouvements, les échanges ne sont effectués que si la valeur de la fonction objectif considérée est améliorée (c'est-à-dire l'équation (8) pour le pire et l'équation (11) pour MNSQW). En outre, lors de l'exécution de LNSR, les quatre mouvements locaux sont exécutés suivant un ordre aléatoire.

3.4 Étude expérimentale

Afin d'analyser la performance de l'approche proposée, le modèle robuste est comparé en utilisant les trois critères robustes suivants : meilleur cas, pire cas et min max déviation. Tous les programmes linéaires entiers ont été résolus par le logiciel Cplex 12.6 et les tests ont été implémentés en C++ et exécutés sur un Intel Pentium Core i7-4790 avec 3.6 GHz.

3.4.1 Structure des instances testées

Les modèles robustes considérés ont été testés sur un ensemble d'instances aléatoires du benchmark de Solomon (1987) [Solomon 1987]. Étant donné que le VRP et le RVRP sont NP-Difficile, et afin de fournir des conclusions parfaitement robustes, nous n'avons considéré que les instances à petite échelle. Par conséquent, le nombre de clients est fixé successivement à 10 et 15 et le nombre de scénarios varie dans l'ensemble $\{5; 10; 15; 20\}$. Pour chaque paramètre, nous avons généré au hasard 10 échantillons. Comme l'incertitude du RVRP est simulée par des scénarios discrets, la distance de déplacement de chaque arc est évaluée par q scénarios. Chaque scénario est généré au hasard dans $[d; 2d]$, où d est la distance euclidienne entre deux clients. Les instances utilisées sont référencées comme suit : DSGr_q_n_i, où, $Gr = \{C1; C2; R1; R2; RC1; RC2\}$ correspond au nom de la classe du benchmark de Salomon ; $Q = \{5; 10; 15; 20\}$ (resp.$N = \{10; 15; 20\}$) représente le nombre de scénarios (resp. le nombre de clients) ; $I = \{1; ...; 10\}$ est l'indice de l'échantillon. Enfin, pour définir la capacité du véhicule, on utilise la formule suivante :

$(max_{i \in V}\{d_i\} + \sum_{i \in V} d_i) \times coef$, où, $coef$ est pris à 0.3 pour les instances des groupes $C1, R1, RC1$ et 0.6 pour celles appartenant aux groupes $C2, R2, RC2$.
Le deuxième ensemble contient les instances générées à partir du benchmark de Salomon avec 25, 50 et 100 clients. Ces instances sont référencées comme suit : DSGr_q_n, où, $Gr = \{C1, C2, R1, R2, RC1, RC2\}$; $Q = \{10, 30, 50\}$ (resp. $N = \{25, 50, 100\}$) représente le nombre de scénarios (resp. Clients). Chaque scénario est généré au hasard dans $[d, 2d]$, où d est la distance euclidienne entre deux clients. Le deuxième ensemble est généralement utilisé pour fournir des conclusions pseudo-robustes pour le Pire et MNSQW.

3.4.2 Conclusions parfaitement robustes sur le nouveau modèle

Le tableau 3.3 montre les résultats obtenus avec différents critères de robustesse sur des instances du RVRP avec 10 et 15 clients. La colonne "Inst" Affiche les labels des instances. Les colonnes de 2 à 5, montrent le nombre moyen de scénarios (c'est-à-dire la valeur moyenne sur 10 instances) qui présentent un meilleur comportement en comparant "Meilleur" (B) avec "Pire" (W), "min max déviation" (D) avec "pire", "MNSQW" (M) avec "Pire" et "MNSQW" avec "min max déviation", respectivement. A titre d'exemple, les chiffres indiqués dans la ligne DSC1_5_10 indiquent, qu'en moyenne, 40% des scénarios liés à "Meilleur" font mieux que ceux liés à "Pire" et dans 78% des scénarios "min-max déviation" domine "Pire", etc. Les 3 dernières lignes du tableau 3.3 affichent le comportement des critères robustes en fonction du nombre d'instances. La ligne "égale" présente le nombre d'instances, où les deux critères comparés trouvent les mêmes solutions. La ligne "supérieure" présente le nombre d'instances, où un des critère domine strictement l'autre. La ligne "aussi bon" donne la somme des deux lignes précédentes.

En examinant le tableau 3.3, nous pouvons observer ce qui suit. Le critère du Meilleur cas (c'est-à-dire B) est moins bon que le critère du Pire cas (c'est-à-dire W) dans la plupart des cas. De plus, B montre un meilleur comportement que W uniquement pour 97 (resp. 54) sur 240 instances pour le cas de 10 (resp. 15) clients. En général, le critère de min-max déviation (c'est-à-dire D) est relativement meilleur que le critère du pire cas. Min max déviation montre un meilleur comportement que le Pire pour 176 instances (resp. 151) sur 240 instances pour le cas de 10 (resp. 15) clients. En tant qu'une extension de Pire, MNSQW domine faiblement Pire sur toutes les instances, avec une dominance forte pour 164 (resp. 212) sur 240 instances pour le cas de 10 clients (respectivement 15). MNSQW montre aussi un meilleur comportement que min-max déviation pour 215 instances (respectivement 204) sur 240 instances avec une forte dominance pour 138 instances (resp. 158) sur 240 instances pour le cas de 10 clients (respectivement 15).

3.4.3 Conclusions pseudo-robustes pour le nouveau modèle

En se basant sur les résultats obtenus après quelques essais, nous avons choisi l'ensemble des valeurs qui assure la meilleure performance pour LNSR. Les para-

Inst.	B vs W	D vs W	M vs W	M vs D	Inst.	B vs W	D vs W	M vs W	M vs D
DSC1_5_10	40%	78%	78%	80%	DSC1_5_15	32%	60%	68%	58%
DSC1_10_10	49%	50%	88%	63%	DSC1_10_15	27%	55%	72%	64%
DSC1_15_10	35%	57%	80%	74%	DSC1_15_15	25%	44%	73%	72%
DSC1_20_10	50%	61%	82%	72%	DSC1_20_15	34%	56%	69%	54%
DSC2_5_10	40%	72%	74%	66%	DSC2_5_15	30%	56%	70%	66%
DSC2_10_10	54%	66%	86%	79%	DSC2_10_15	30%	64%	73%	58%
DSC2_15_10	54%	58%	77%	79%	DSC2_15_15	43%	70%	73%	75%
DSC2_20_10	54%	81%	76%	73%	DSC2_20_15	32%	54%	74%	61%
DSR1_5_10	52%	68%	74%	66%	DSR1_5_15	44%	60%	72%	66%
DSR1_10_10	39%	80%	78%	67%	DSR1_10_15	27%	64%	79%	70%
DSR1_15_10	45%	69%	71%	65%	DSR1_15_15	37%	57%	77%	71%
DSR1_20_10	46%	65%	80%	80%	DSR1_20_15	40%	48%	70%	76%
DSR2_5_10	32%	78%	78%	74%	DSR2_5_15	46%	72%	80%	56%
DSR2_10_10	35%	59%	81%	75%	DSR2_10_15	48%	58%	82%	73%
DSR2_15_10	37%	62%	77%	81%	DSR2_15_15	31%	65%	75%	77%
DSR2_20_10	51%	71%	79%	73%	DSR2_20_15	23%	66%	80%	65%
DSRC1_5_10	48%	62%	72%	66%	DSRC1_5_15	28%	78%	72%	60%
DSRC1_10_10	30%	82%	89%	86%	DSRC1_10_15	37%	56%	68%	78%
DSRC1_15_10	53%	59%	64%	76%	DSRC1_15_15	35%	55%	73%	65%
DSRC1_20_10	50%	65%	80%	73%	DSRC1_20_15	42%	61%	72%	71%
DSRC2_5_10	42%	74%	70%	66%	DSRC2_5_15	32%	68%	72%	72%
DSRC2_10_10	49%	68%	86%	72%	DSRC2_10_15	35%	59%	69%	63%
DSRC2_15_10	42%	81%	69%	64%	DSRC2_15_15	24%	45%	67%	74%
DSRC2_20_10	58%	71%	78%	72%	DSRC2_20_15	38%	51%	67%	66%
égale	25	79	76	77	égale	2	32	28	46
supérieure	72	97	164	138	supérieure	52	119	212	158
aussi bon	97	176	240	215	aussi bon	54	151	240	204

TABLE 3.3 – Analyse de robustesse sur des instances aléatoires de 10 et 15 clients, où chaque ligne de DSgr_q_n rapporte la moyennes des valeurs sur 10 instances.

		Pire			MNSQW		
Inst.	Cplex	$Best$	$P_{moyenne}$	cpu_m	P_{best}	$P_{moyenne}$	cpu_m
DSC1	839.67	777.44	778.14	2.71	85	82	3.66
DSC2	666.89	648.44	648.44	3.04	95	80	3.56
DSR1	1104.67	942.78	943.49	16.21	85	83	18.65
DSR2	815.11	808.00	808.12	55.00	81	72	63.03
DSRC1	1157.11	1002.11	1003.62	16.27	81	77	17.73
DSRC2	761.22	722.56	722.57	9.69	87	83	10.90

TABLE 3.4 – Performance de LNSR versus Cplex selon les critères de Pire et MNSQW (pour plus de détails, voir les tableaux 5 et 6).

mètres utilisés dans LNSR sont définis comme suit : α est généré de manière aléatoire dans $[0.1,0.5]$; λ est fixé à 15 ; β est réglé sur 1.03 (resp. 0.9) lors de l'application de LNSR à Pire (resp. MNSQW) ; Max_{iter} est fixé à 10000. De plus, et en raison de l'aspect aléatoire de LNSR, dix essais indépendants ont été effectués pour chaque instance. Afin d'analyser les performances de LNSR pour MNSQW, les résultats obtenus sont comparés à ceux fournis par Cplex. Pour les instances avec 25 et 50 clients, nous avons limité le temps d'exécution de Cplex à 3600s pour la résolution de ILP^{pire}_{rvrp} ou de ILP^{nouv}_{rvrp}. Toutefois, pour les instances avec 100 clients, nous avons augmenté le temps limite d'exécution de Cplex à 7200s.

Le tableau 3.4 montre les résultats obtenus avec Cplex et LNSR en utilisant le critère MNSQW. Étant donné que la solution liée à MNSQW dépend de Pire, nous comparons également les solutions fournies par LNSR selon Pire avec celles calculées par Cplex. La colonne «Inst.» affiche les labels des groupes, où chaque

groupe contient 9 instances. Les colonnes de 2 à 5 (respectivement les colonnes de 6 à 8) montrent les informations de la solution lors de l'application de Cplex et LNSR avec le critère Pire (resp. MNSQW). La colonne "Cplex" affiche la moyenne des valeurs objectifs calculées par Cplex. La colonne "Best" indique les meilleures valeurs trouvées par LNSR avec 10 tests tandis que la colonne "mean" affiche les valeurs moyennes trouvées par LNSR sur les 10 essais. Les colonnes "cpu_m" (c'est-à-dire les colonnes 5 et 8) correspondent à la moyenne des temps de CPU utilisés par LNSR pour trouver la meilleure solution pour chaque essai. Les colonnes P_{best} affichent le pourcentage (en moyenne) du nombre de scénarios, où les meilleures solutions trouvées par LNSR surpassent celles trouvées par Cplex. Les colonnes P_{moy} affichent les pourcentages des scénarios qualifiés lors de la comparaison de Cplex avec LNSR sur les moyennes. Les résultats complets sont détaillés dans les tableaux 3.5 et 5.13. À partir du tableau 3.4, nous pouvons faire les observations suivantes : Pour Worst, LNSR est capable de fournir de meilleures solutions que Cplex avec des temps de calcul raisonnables. En comparant les solutions obtenues pour le critère MNSQW, nous observons clairement que LNSR peut toujours trouver de meilleures solutions que Cplex. En effet, les solutions renvoyées par LNSR sont meilleures que celles renvoyées par Cplex sur la majorité des scénarios (voir tableau 5.13) : $81 - 95\%$ pour le cas des meilleures solutions et $72 - 83\%$ pour le cas des moyennes.

Dans le tableau 3.5, nous affichons la borne inférieure (voir, la colonne "BI_v") calculée par Cplex et le temps d'exécution cpu_{time} requis (voir la colonne "BI_t") pour la résolution de chaque instance. A noter que la valeur marquée avec le symbole * (voir, la colonne "Cplex") signifie que Cplex réussit à prouver l'optimalité de la solution. Dans tableau 3.5, on peut constater que la qualité de la borne inférieure calculée par Cplex est insatisfaisante pour résoudre de façon optimale les instances à grande échelle du RVRP. Ceci est dû à la grande complexité du RVRP. Nous pouvons également remarquer que LNSR réussit à fournir des solutions de bonne qualité pour ces instances complexes en moins de temps en comparaison avec Cplex.

3.5 Conclusion

Dans ce chapitre, notre objectif consistait à fournir des solutions robustes pour le problème de tournées de véhicules avec des coûts de trajet incertains. La contribution principale de ce chapitre est de proposer un nouveau critère robuste et d'essayer d'analyser la robustesse de la solution trouvée avec le critère proposé. Le critère robuste proposé MNSQW est généralisé à partir du critère du pire cas. Son objectif est de trouver une solution qui fonctionne mieux que la solution fournie par le critère du pire cas sur la majorité des scénarios. Le modèle proposé est représenté sous la forme d'un programme linéaire entier et évalué sur un ensemble d'instances aléatoires. Il est à noter que, RVRP est encore plus complexe que le VRP classique qui est déjà un problème NP-difficile. Pour cela, nous avons essayé d'évaluer MNSQW selon deux axes : parfaitement robustes et pseudo-robustes. La méthode exacte n'est

appliquée que pour résoudre des petites instances afin de fournir des conclusions parfaitement robustes. D'un point de vue pratique, nous avons proposé également une méta-heuristique efficace pour résoudre des instances à grande échelle afin de fournir des conclusions pseudo-robustes. En comparaison avec les autres critères classiques robustes, les résultats fournis ont montré que MNSQW est capable de produire des solutions robustes dans la majorité des cas.

Inst	BI_v	BI_t	Cplex best		moy	cpu_m	gen1	gen2	gen3	gen4	gen5	gen6	gen7	gen8	gen9	gen10
DSC1_10_25	148	0.02	320	316	316	0.03	316	316	316	316	316	316	316	316	316	316
DSC1_10_50	292	0.08	642	614	614	1.95	614	614	614	614	614	614	614	614	614	614
DSC1_10_100	604	0.43	1456	1344	1345.1	7.41	1345	1346	1344	1346	1350	1344	1344	1344	1344	1344
DSC1_30_25	150	0.03	343	337	337	0.15	337	337	337	337	337	337	337	337	337	337
DSC1_30_50	300	0.18	650	627	627	0.31	627	627	627	627	627	627	627	627	627	627
DSC1_30_100	618	1.13	1552	1389	1392.1	4.99	1398	1392	1392	1389	1389	1393	1392	1392	1395	1389
DSC1_50_25	152	0.05	352	339	339	0.1	339	339	339	339	339	339	339	339	339	339
DSC1_50_50	308	0.29	646	631	631	0.24	631	631	631	631	631	631	631	631	631	631
DSC1_50_100	625	1.35	1596	1400	1402.1	9.24	1403	1402	1400	1404	1402	1400	1404	1406	1400	1400
DSC2_10_25	244	0.02	333	333	333	0.08	333	333	333	333	333	333	333	333	333	333
DSC2_10_50	454	0.08	605	584	584	0.61	584	584	584	584	584	584	584	584	584	584
DSC2_10_100	844	0.42	1015	998	998	1.21	998	998	998	998	998	998	998	998	998	998
DSC2_30_25	259	0.03	341	341	341	0.17	341	341	341	341	341	341	341	341	341	341
DSC2_30_50	450	0.21	608	587	587	1.04	587	587	587	587	587	587	587	587	587	587
DSC2_30_100	852	1.05	1062	1019	1019	4.41	1019	1019	1019	1019	1019	1019	1019	1019	1019	1019
DSC2_50_25	260	0.05	344	344	344	0.33	344	344	344	344	344	344	344	344	344	344
DSC2_50_50	460	0.27	607	606	606	2.86	606	606	606	606	606	606	606	606	606	606
DSC2_50_100	863	1.53	1087	1024	1024	16.65	1024	1024	1024	1024	1024	1024	1024	1024	1024	1024
DSR1_10_25	481	0.02	568	568	568	0.08	568	568	568	568	568	568	568	568	568	568
DSR1_10_50	690	0.08	979	873	873	0.38	873	873	873	873	873	873	873	873	873	873
DSR1_10_100	994	0.47	1987	1359	1361.6	17.74	1365	1362	1363	1362	1363	1359	1361	1360	1362	1359
DSR1_30_25	498	0.03	581	580	580	0.08	580	580	580	580	580	580	580	580	580	580
DSR1_30_50	709	0.18	899	886	886	1.6	886	886	886	886	886	886	886	886	886	886
DSR1_30_100	999	1.31	1778	1380	1381.9	45.81	1386	1386	1381	1381	1381	1381	1380	1382	1380	1381
DSR1_50_25	480	0.04	573	573	573	0.28	573	573	573	573	573	573	573	573	573	573
DSR1_50_50	708	0.26	909	888	888	0.49	888	888	888	888	888	888	888	888	888	888
DSR1_50_100	1008	1.64	1668	1378	1379.9	79.46	1380	1381	1380	1380	1381	1378	1382	1379	1379	1379
DSR2_10_25	480	0.02	533	533	533	0.11	533	533	533	533	533	533	533	533	533	533
DSR2_10_50	686	0.08	771	771	771	4.17	771	771	771	771	771	771	771	771	771	771
DSR2_10_100	990	0.43	1108	1092	1092.2	33.48	1092	1092	1092	1092	1092	1092	1094	1092	1092	1092
DSR2_30_25	498	0.03	546	546	546	0.35	546	546	546	546	546	546	546	546	546	546
DSR2_30_50	707	0.18	782	782	782	3.41	782	782	782	782	782	782	782	782	782	782
DSR2_30_100	994	1.33	1117	1106	1106.4	169.25	1106	1106	1107	1106	1107	1106	1108	1106	1106	1106
DSR2_30_100	478	0.04	544	544	544	0.3	544	544	544	544	544	544	544	544	544	544
DSR2_50_50	704	0.26	786	786	786	9.84	786	786	786	786	786	786	786	786	786	786
DSR2_50_100	1003	1.60	1149	1112	1112.5	274.07	1112	1112	1112	1113	1112	1112	1112	1112	1112	1116
DSRC1_10_25	185	0.02	494	494	494	0.02	494	494	494	494	494	494	494	494	494	494
DSRC1_10_50	353	0.08	870	856	856	0.1	856	856	856	856	856	856	856	856	856	856
DSRC1_10_100	930	0.44	1901	1595	1602.8	17.75	1603	1610	1600	1600	1602	1613	1595	1615	1595	1595
DSRC1_30_25	192	0.04	513	508	508	0.08	508	508	508	508	508	508	508	508	508	508
DSRC1_30_50	361	0.22	909	890	890	0.19	890	890	890	890	890	890	890	890	890	890
DSRC1_30_100	952	1.22	2043	1635	1636.2	51.31	1636	1635	1635	1639	1635	1635	1638	1635	1639	1635
DSRC1_50_25	189	0.05	512	511	511	0.06	511	511	511	511	511	511	511	511	511	511
DSRC1_50_50	365	0.32	903	892	892	0.1	892	892	892	892	892	892	892	892	892	892
DSRC1_50_100	958	1.37	2269	1638	1642.6	76.83	1641	1641	1639	1638	1649	1644	1643	1649	1641	1641
DSRC2_10_25	185	0.02	391	380	380	0.09	380	380	380	380	380	380	380	380	380	380
DSRC2_10_50	350	0.08	677	621	621	0.55	621	621	621	621	621	621	621	621	621	621
DSRC2_10_100	919	0.45	1207	1115	1115	23.21	1115	1115	1115	1115	1115	1115	1115	1115	1115	1115
DSRC2_30_25	192	0.03	399	399	399	0.13	399	399	399	399	399	399	399	399	399	399
DSRC2_30_50	357	0.21	695	655	655	0.82	655	655	655	655	655	655	655	655	655	655
DSRC2_30_100	945	1.12	1196	1134	1134.1	38.72	1134	1134	1134	1134	1134	1134	1135	1134	1134	1134
DSRC2_50_25	189	0.05	412	397	397	0.26	397	397	397	397	397	397	397	397	397	397
DSRC2_50_50	363	0.30	679	661	661	0.87	661	661	661	661	661	661	661	661	661	661
DSRC2_50_100	948	1.42	1195	1141	1141	22.61	1141	1141	1141	1141	1141	1141	1141	1141	1141	1141

TABLE 3.5 – Performance de LNSR versus Cplex selon le critère de pire cas.

Inst.	Best	moy	cpu_m	gen1	gen2	gen3	gen4	gen5	gen6	gen7	gen8	gen9	gen10
DSC1_10_25	5	4.2	0.04	3	3	4	5	5	4	5	4	4	5
DSC1_10_50	10	9.5	3.08	9	8	10	9	10	10	9	10	10	10
DSC1_10_100	10	9.7	8.15	10	10	10	10	10	9	10	9	10	9
DSC1_30_25	22	20.7	0.22	22	19	22	22	22	20	22	22	20	16
DSC1_30_50	28	28	2.38	28	28	28	28	28	28	28	28	28	28
DSC1_30_100	30	30	5.56	30	30	30	30	30	30	30	30	30	30
DSC1_50_25	34	34	0.18	34	34	34	34	34	34	34	34	34	34
DSC1_50_50	39	36.3	0.79	33	39	36	39	36	33	39	39	33	36
DSC1_50_100	50	50	12.52	50	50	50	50	50	50	50	50	50	50
DSC2_10_25	10	6.3	0.1	7	6	7	6	7	7	1	6	10	6
DSC2_10_50	10	10	0.55	10	10	10	10	10	10	10	10	10	10
DSC2_10_100	10	9.4	1.31	10	8	10	8	10	10	8	10	10	10
DSC2_30_25	30	20.5	0.22	30	30	11	11	11	30	11	30	11	30
DSC2_30_50	26	24.2	1.15	24	24	24	26	24	24	24	24	24	24
DSC2_30_100	30	30	4.42	30	30	30	30	30	30	30	30	30	30
DSC2_50_25	50	39.3	0.42	24	23	50	23	50	50	23	50	50	50
DSC2_50_50	32	19.7	3.28	32	17	17	17	29	17	17	17	17	17
DSC2_50_100	50	50	20.54	50	50	50	50	50	50	50	50	50	50
DSR1_10_25	6	5.7	0.32	6	6	6	6	6	3	6	6	6	6
DSR1_10_50	10	10	0.4	10	10	10	10	10	10	10	10	10	10
DSR1_10_100	10	10	19.61	10	10	10	10	10	10	10	10	10	10
DSR1_30_25	30	28.6	0.05	30	30	30	30	30	30	16	30	30	30
DSR1_30_50	23	22.4	1.89	21	23	23	23	23	23	23	23	21	21
DSR1_30_100	30	30	52.48	30	30	30	30	30	30	30	30	30	30
DSR1_50_25	27	27	0.36	27	27	27	27	27	27	27	27	27	27
DSR1_50_50	37	33.4	0.51	33	33	33	33	33	33	33	37	33	33
DSR1_50_100	50	50	92.23	50	50	50	50	50	50	50	50	50	50
DSR2_10_25	10	6	0.13	10	10	2	10	2	10	10	2	2	2
DSR2_10_50	5	3.5	5.16	5	3	5	3	3	3	4	3	3	3
DSR2_10_100	10	9.9	36.73	10	10	10	10	9	10	10	10	10	10
DSR2_30_25	30	30	1,00	30	30	30	30	30	30	30	30	30	30
DSR2_30_50	14	14	4.01	14	14	14	14	14	14	14	14	14	14
DSR2_30_100	25	23.4	210.87	23	23	24	23	24	23	25	23	23	23
DSR2_30_100	50	42.2	1.47	50	50	27	50	25	50	50	50	20	50
DSR2_50_50	27	25.6	18.75	27	25	27	25	27	25	25	25	25	25
DSR2_50_100	46	45.7	289.11	46	46	46	43	46	46	46	46	46	46
DSRC1_10_25	7	5.9	0.01	6	6	6	5	6	6	6	5	6	7
DSRC1_10_50	9	8.6	0.22	9	9	9	9	8	9	8	8	9	8
DSRC1_10_100	10	10	29.64	10	10	10	10	10	10	10	10	10	10
DSRC1_30_25	17	17	0.15	17	17	17	17	17	17	17	17	17	17
DSRC1_30_50	25	22.1	0.24	21	21	25	25	21	21	22	21	19	25
DSRC1_30_100	30	30	58.56	30	30	30	30	30	30	30	30	30	30
DSRC1_50_25	30	30	0.07	30	30	30	30	30	30	30	30	30	30
DSRC1_50_50	33	30.5	0.72	31	29	31	32	29	32	32	28	28	33
DSRC1_50_100	50	50	69.99	50	50	50	50	50	50	50	50	50	50
DSRC2_10_25	8	5.5	0.11	5	5	6	5	6	5	6	8	4	5
DSRC2_10_50	10	10	10.15	10	10	10	10	10	10	10	10	10	10
DSRC2_10_100	10	10	20.47	10	10	10	10	10	10	10	10	10	10
DSRC2_30_25	21	20.8	0.15	21	19	21	21	21	21	21	21	21	21
DSRC2_30_50	28	26.6	1.06	28	26	26	26	26	28	28	26	26	26
DSRC2_30_100	30	30	39.54	30	30	30	30	30	30	30	30	30	30
DSRC2_50_25	31	30.1	0.28	28	31	31	28	31	31	31	31	28	31
DSRC2_50_50	40	38.8	1.39	40	40	31	40	40	40	37	40	40	40
DSRC2_50_100	50	50	24.91	50	50	50	50	50	50	50	50	50	50

TABLE 3.6 – Performance de LNSR versus Cplex selon le critère de MNSQW. Dans les colonnes de gen1 à gen10, nous affichons le nombre de scénarios où les solutions fournies par LNSR sont meilleures que celles fournies par Cplex.

Chapitre 4

Problème de tournées de véhicules avec des coûts de trajet incertains

Sommaire

Les travaux présentés dans ce chapitre ont fait l'objet des publications [Bederina 2016], [Bederina 2017c] et [Bederina 2018].

4.1 Introduction

La résolution des problèmes d'optimisation de type VRP liés aux applications du monde réel, se heurte à plusieurs défis.

D'une part, certains paramètres du VRP peuvent être incertains. Dans notre cas, nous considérons l'incertitude sur les coûts de déplacement entre les clients qui sont considérés comme aléatoires, c'est à dire chaque coût de déplacement peut être caractérisé par un ensemble fini de valeurs discrètes (voir chapitre précédent pour plus de détails sur le modèle d'incertitude). Le but étant d'essayer de déterminer une "bonne" solution finale qui satisfait tous les scénarios ou la plupart des scénarios considérés. Le problème résultant est connu sous le nom de VRP robuste (RVRP) [Solano-Charris 2014].

D'autre part, les applications réelles sont souvent caractérisées par plusieurs objectifs à optimiser simultanément. En raison de la présence de plusieurs objectifs, le résultat attendu pour de tels problèmes d'optimisation est souvent un ensemble de solutions optimales connues comme l'ensemble de solutions optimales de Pareto. De nombreuses méthodes de résolution sont disponibles dans la littérature pour traiter les problèmes multi-objectifs. Parmi ces méthodes de résolution, on distingue deux catégories principales : les méthodes de Pareto et les méthodes de transformation mono-objectif qui consistent princialement en des méthodes d'agrégation (voir section 2.7.3 pour plus de détails sur ces méthodes).

Une méthode de Pareto compare deux solutions selon le concept de non-dominance (section 2.7), alors qu'une méthode de transformation mono-objectif est basée sur le principe d'agrégation, comme l'agrégation linéaire pondérée. Autrement dit, la première méthode évalue les solutions et compare leurs qualités selon le principe de non-dominance tandis que la deuxième méthode tente de construire une fonction objectif unique et résout le problème résultant en un problème d'optimisation mono-objectif. L'algorithme génétique basé sur le tri de non-dominance (Non-dominated Sorting Genetic Algorithm NSGA-II) (cf. Deb *et al.* [Deb 2002]) essaye d'imiter la méthode de Pareto, en appliquant la classification au sens de la dominance et en assignant les solutions non dominées à différents fronts (voir 2.7.4.4 pour une description de NSGA-II). Dans ce cas, le but des fronts non dominés est de classer l'ensemble des solutions en fonction de leurs rangs et donc de créer la diversité dans les solutions de Pareto.

Dans ce chapitre, nous proposons une approche évolutionnaire basée sur l'optimisation multi-objectif pour résoudre le problème de tournées de véhicules avec coûts de trajet incertains (RVRP, Robust Vehicle Routing Problem). Nous considérons deux objectifs à minimiser de manière simultanée à savoir : un critère de robustesse sur le coût de trajet total de tous les véhicules (car ce dernier est incertain) et le nombre de véhicules à utiliser. Concernant les critères de robustesse, nous considérons deux critères à savoir : le critère du pire cas et le critère MNSQW que nous avons proposé dans le chapitre précédent. Le problème est résolu en utilisant l'algorithme NSGA-II, hybridé avec plusieurs stratégies de recherche locale, et une heuristique de destruction/construction.

La suite du chapitre est organisé comme suit : les formulations mathématiques du problème, selon les deux critères de robustesse, sont décrites dans la section 4.2. La section 4.3 présente l'approche HMOEA que nous proposons en décrivant les opérateurs de l'algorithme NSGA-II adapté ainsi que les opérateurs de diversification (heuristique de destruction/construction DCH) et les opérateurs d'intensification (recherche locale). Les expérimentations réalisées en deux temps : avec et sans DCH sont présentées dans la section 4.4. La section 4.5 conclut ce chapitre.

4.2 Modélisation du problème de tournées de véhicules robuste multi-objectif (MO-RVRP)

4.2.1 Formulation mathématique du probème

Soit $G = (V, E)$ un digraphe connexe, où V représente un ensemble de n nœuds, le nœud 0 représente le dépôt et les autres nœuds représentent les clients, chaque client $i \in N = \{1, ..., n\}$ a une quantité de biens ℓ_i à livrer. E désigne un ensemble d'arcs avec des poids non négatifs d_{ij} qui représentent les coûts de trajet.

La variable de décision binaire x_{ij} représente un chemin du nœud i au nœud j ; c'est-à-dire égal à 1 s'il y a un véhicule qui passe de i à j et 0 sinon. Le RVRP est également caractérisé par un ensemble de m véhicules identiques caractérisés par une capacité C.
Le problème linéaire standard du MO-RVRP doit respecter les contraintes suivantes :

$$\begin{cases} \sum_{i=1}^{n} x_{0i} \leq m \ \text{ and } \ \sum_{i=1}^{n} x_{i0} \leq m & (1) \\ \sum_{j=1}^{n} x_{ij} = 1, \ \forall i = \{1, ..., n\} & (2) \\ \sum_{j=1}^{n} x_{ji} = 1, \ \forall i = \{1, ..., n\} & (3) \\ \ell_j - \ell_i - C \times x_{ij} \leq c_j - C, \ \forall i \neq j = \{1, ..., n\} & (4) \\ c_i \leq \ell_i \leq C, \ \forall i = \{1, ..., n\} & (5) \\ x_{ij} \in \{0, 1\} \forall i, j = \{1, ..., n\}, \ \ell_i \in \mathbb{N}, \ \forall i = \{1, ..., n\}, \ m \in \mathbb{N} & (6) \end{cases}$$

Chaque contrainte de type (1) représente la contrainte de conservation du débit du dépôt et assure que le nombre de véhicules quittant le dépôt est le même que celui des véhicules qui y retournent, et doit être inférieur ou égal à m. Les contraintes de type (2) et (3) sont aussi les contraintes de conservation des nœuds (clients) en dehors du dépôt, elles assurent que le nombre de véhicules qui entrent dans un nœuds (visite un client) est le même que celui des véhicules qui le quittent et est égal obligatoirement à 1. Chaque contrainte de type (4) évite les sous-cycles dans le graphe construit. Chaque contrainte de type (5) assure que la somme des charges (demandes ou poids) dans un véhicule en arrivant à un nœuds doit être inférieure ou égale à la capacité du véhicule et au moins égale à la demande de ce client. Enfin, les équations (6) définissent les domaines de variation des variables de décision x_{ij} et l_i et m.

L'objectif principal du MO-RVRP proposé est de minimiser le coût de trajet total selon un critère de robustesse et le nombre de véhicules simultanément. Nous décrivons dans ce qui suit, le modèle d'incertitude utilisé pour le calcul du premier objectif à savoir le coût du trajet.

$D(.)$ indique la fonction qui calcule le coût de trajet pour une route (à savoir R) ou une liste de routes (itinéraires) (à savoir LR). Ensuite, le coût de déplacement

d'une route R est noté $D(R)$ et celui de LR est noté $D(LR)$ et est calculé comme suit :

$$D(LR) = \sum_{\forall R \in LR} D(R), \tag{4.1}$$

Tel que $D(R)$ est défini comme :

$$D(R) = \sum_{\forall (i,j) \in R} d_{ij}. \tag{4.2}$$

Le modèle RVRP est caractérisé par un ensemble de scénarios discrets car chaque arête $(i, j) \in E$ est caractérisée par un ensemble fini de q valeurs $\{d_{ij}^1, ..., d_{ij}^q\}$. Il s'ensuit que la solution réalisable LR de chaque RVRP est également caractérisée par q valeurs $\{D(LR)^1, ..., D(LR)^q\}$ (le vecteur représentant le coût de trajet total pour tous les scénarios considérés). Par conséquent, l'objectif du RVRP est de déterminer une solution robuste qui fonctionne relativement bien pour tous les scénarios envisagés. Par conséquent, certains critères classiques ont été utilisés pour évaluer la robustesse de la solution. Ces critères sont généralement basés sur le comportement des humains alors qu'ils prennent une décision dans l'incertitude.

4.2.2 Formulation mathématique selon le critère du pire des cas

Dans cette étude, nous utilisons le critère robuste de *min-max*, pour le premier objectif à optimiser qui est le coût de trajet. En choisissant ce type de solution, le décideur cherche à éviter tous les risques et on dit qu'il est averse au risque. Il consiste à chercher, entre les scénarios dans Q des solutions réalisables, ceux réalisant le coût maximal LR^ω (pire scénarios) :

$$LR^\omega = \max\Big\{D(LR)^s \mid s \in Q\Big\}.$$

L'objectif du critère robuste de *min-max* est de déterminer un indice de solution ω réalisant le plus petit pire scénario ; c'est la solution telle que :

$$\omega = \text{argmin}\Big\{LR^\omega \mid LR \in \mathcal{S}\Big\},$$

Où $\mathcal{S}$ désigne l'ensemble des solutions réalisables.

Le problème représente un problème d'optimisation multi-objectif. La fonction objectif (7) essaie de minimiser simultanément le plus mauvais (pire) coût de trajet total effectué par tous les véhicules et aussi le nombre de véhicules. Chaque contrainte de type (8) garantit que δ est une *borne supérieure* des coûts de trajet total de tous les scénarios.

$$(ILP_{vrp}^{worst}) \begin{cases} Z = & \min(\sigma, m) & (7) \\ s.c. & & \\ & D^i(LR) \leq \sigma \quad \forall i = \{1, ..., q\} & (8) \\ & s_i \in \{0, 1\} \ \forall i = \{1, ..., q\} & \\ & \textit{contraintes } (1) - (6) & \end{cases}$$

4.2.3 Formulation mathématique selon le critère MNSQW

La fonction objectif (7) essaie de minimiser le nombre de véhicules et de maximiser le nombre de scénarios où la solution courante LR fonctionne mieux que celle du pire cas $Pire^*$, simultanément. La contrainte (10) assure que $s_i = 0$ si la solution actuelle n'améliore pas le coût de trajet de $Pire^*$ pour le i-ème scénario, sinon $s_i = 1$.

$$(ILP_{vrp}^{mnsqw}) \begin{cases} Z = & \min(-\sum_{i=1}^{q} s_i, m) & (9) \\ s.c. & & \\ & D^i(LR) \leq W \times (1 - s_i) + D^i(Worst^*) \times s_i \forall i = \{1, ..., q\} & (10) \\ & s_i \in \{0, 1\} \ \forall i = \{1, ..., q\} & \\ & contraintes \ (1) - (6) & \end{cases}$$

4.3 Optimisation évolutionnaire multi-objectif hybride (HMOEA)

Beaucoup de problèmes du monde réel sont multi-objectif par nature, impliquant l'optimisation de plusieurs objectifs contradictoires. Il convient de noter que la plupart des approches de résolution proposées pour le RVRP ne tiennent souvent compte que d'un objectif unique qui est de réduire le coût de déplacement de tous les véhicules. Dans la description formelle de la section 4.2, au moins deux objectifs comprenant le nombre de véhicules et le coût de trajet total doivent être minimisés simultanément. Selon Tan *et al.* [Tan 2006], ces objectifs peuvent être contradictoires.

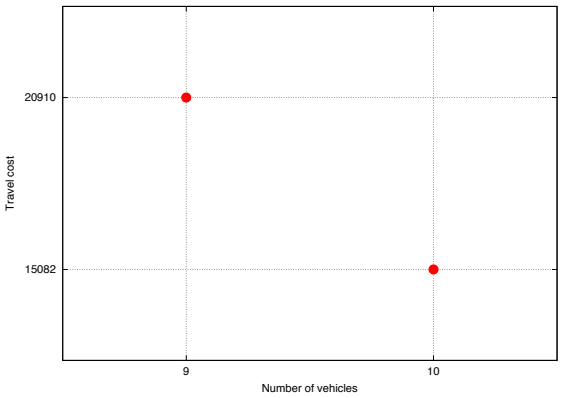


FIGURE 4.1 – Valeurs objectives conflictuelle pour l'instance n100-m10-p10-d50

En effet, la figure 4.1 illustre le comportement conflictuel des deux fonctions objectifs, en particulier pour une instance de l'état de l'art à grande échelle du RVRP . La solution notée A réalise un coût de trajet de 15082 en utilisant 10 véhicules alors que la solution B a besoin de seulement 9 véhicules, mais elle réalise un coût de trajet plus mauvais de 20910.

On peut observer que l'augmentation du coût de trajet peut permettre de réduire le nombre de véhicules à utiliser (et vice versa). Ainsi, les algorithmes évolutionnaires multi-objectifs s'avèrent être de bons candidats pour résoudre de tels problèmes, grâce notamment à leur robustesse et leur flexibilité. Dans ce travail, nous proposons une approche hybride basée sur un algorithme évolutionnaire multi-objectif à savoir NSGA-II et des opérateurs de recherche locale. Le choix de NSGA-II est justifié par le fait qu'il est l'un des algorithmes les plus populaires dans la famille des algorithmes évolutionnaires multi-objectifs. A noter que la version originale de l'algortihme NSGA-II a été porposée pour résoudre des problèmes avec des variables paramétriques. Comme dans le MO-RVRP une solution est représentée par une structure complexe, et pas avec un ensemble de variables. L'application de NSGA-II sur le problème MO-RVRP que nous proposons de résoudre, nécessite l'adaptation de certains opérateurs . Nous présentons dans la prochaine section les principales adaptations effectuées sur NSGA-II pour l'adapter au problème MO-RVRP

4.3.1 L'algorithme NSGA-II pour la résolution du MO-RVRP

Nous présentons dans cette section, les principaux opérateurs de l'algorithme NSGA-II incluant les adaptations réalisées pour résoudre le MO-RVRP. Dans tout ce qui suit, les notations suivantes seront utilisées : une population d'individus est noté P et chaque individu de la population est noté ω.

4.3.1.1 La représentation du chromosome

Une solution du problème VRP consiste en un certain nombre de véhicules, et un itinéraire, permettant de visiter certains clients commençant et se terminant par le dépôt, pour chaque véhicule. Le chromosome permet de représenter une solution complète comme le montre la figure 4.2. Le nombre de gènes est égal au nombre de véhicules. Chaque gène est une structure complexe. Il représente la séquence des clients visités et desservis par le véhicule correspondant. Une telle représentation, de longueur variable, est efficace et permet de manipuler et d'améliorer directement le nombre de véhicules et les séquences de clients pour une optimisation multi-objectif dans le RVRP.

4.3.1.2 Population Initiale

Une procédure gloutonne est utilisée pour générer la population initiale. Les routes sont construites de manière parallèle. Ainsi, les clients non desservis sont affectés à chaque fois au véhicule suivant qui a encore une capacité suffisante pour le prendre en charge. Donc, chaque fois qu'un client est affecté à la première route, le deuxième client entre dans la suivante. La procédure s'arrête lorsque tous les clients sont pris en charge par un véhicule.

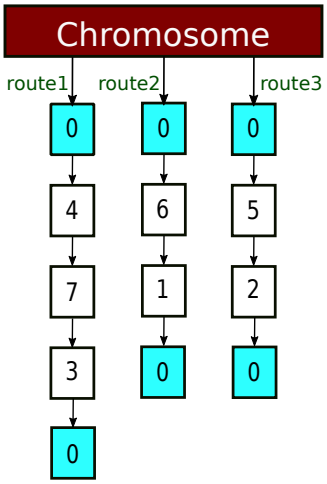


FIGURE 4.2 – La représentation du chromosome d'une solution VRP

4.3.1.3 Sélection par tournoi

L'opérateur de sélection de tournoi sélectionne des paires d'individus de la population. Ces individus (parents) représentent des candidats potentiels au croisement. Afin de sélectionner chaque parent, deux individus sont sélectionnés de la population, puis le meilleur parmi les deux individus est choisi comme étant un des parents. Un des paramètres importants de la sélection par tournoi est la taille du tournoi (T). Plus T augmente, plus les individus les moins performants voient leur chance de gagner le tournoi diminuer.

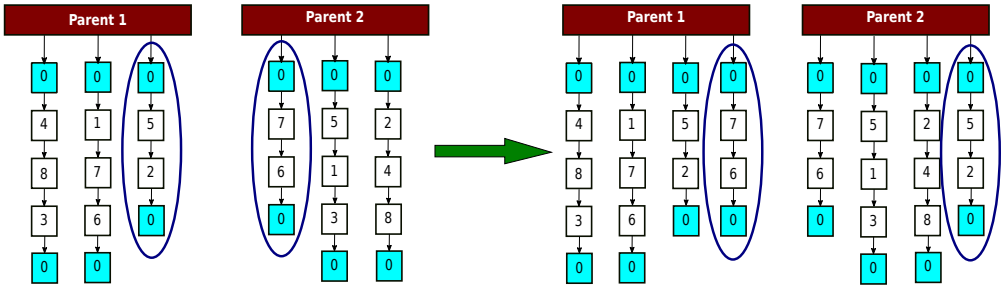
4.3.1.4 Opérateur de Croisement

L'opération de croisement permet de partager les meilleurs itinéraires entre les chromosomes. Son algorithme implique les deux chromosomes résultants de l'opération de sélection de tournoi. L'opérateur de croisement utilisé dans ce travail est celui proposé par Cheong *et al.* [Cheong 2006]). Ce dernier peut être résumé dans les points suivants :

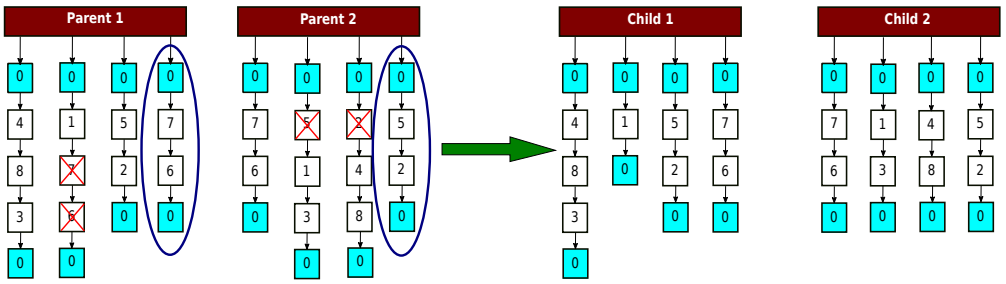
— L'opération de croisement échange les meilleures routes des deux parents sélectionnés, sans supprimer aucune des routes d'origine, comme indiqué par le premier schéma étiqueté 4.3a dans la figure 4.3.

— Afin d'assurer la faisabilité des routes fournies, les clients répétés sont ensuite supprimés des routes d'origine, comme indiqué par le second schéma étiqueté 4.3b dans la figure 4.3, en laissant les routes ajoutées intactes.

Cette opération ne viole pas la contrainte de capacité. En revanche, elle peut violer la contrainte du nombre de véhicules maximal autorisé. Pour cette raison nous avons relaxé cette contrainte dans notre modèle.



(a) Copie des meilleures routes



(b) Suppression des clients répétés

FIGURE 4.3 – Illustration de l'opération de croisement

4.3.1.5 Opérateurs de mutation

Plusieurs opérateurs de mutation pour le VRP ont été proposés dans l'état de l'art. Dans ce travail, nous nous sommes inspirés des opérateurs proposés par Cheong *et al.* [Cheong 2006]. L'algorithme proposé consiste à implémenter une opération de mutation basée sur quatre procédures. Ces procédures sont calibrées avec des paramètres permettant de fixer leurs probabilités d'application. Le taux élastique représente la probabilité avec laquelle on procède pour effectuer un échange partiel – "Partial Swap" (noté PS) entre deux itinéraires (routes). Le taux de compression est le taux d'exécution d'une fusion entre les routes les plus courtes – "Merge Shortest Routes" (noté MSR). Si cette dernière opération n'est pas appliquée, un fractionnement de la route la plus longue – "Split Longest Route" (noté SLR) est exécuté. À la fin de l'opérateur de mutation, un quatrième taux correspondant à un désordre aléatoire qui permet de brouiller l'ordre des clients dans chaque itinéraire de l'individu actuel est considéré. Ce dernier est appelé "Random Shuffling" (noté RS). Nous décrivons dans ce qui suit succinctement le principe de fonctionnement des opérateurs de mutation.

— Échange partiel (PS) : l'opération consiste en un certain nombre de mouvements d'échange, où deux itinéraires dans l'individu sont choisis au hasard. Ensuite, un segment est choisi aléatoirement de chaque route (pas nécessairement de la même taille). Ces segments sont basculés de telle sorte qu'un segment remplace l'autre sans supprimer aucun client n'appartenant pas au

segment.
- Fusion des routes les plus courtes (MSR) : Cette opération recherche deux itinéraires ayant les meilleurs coûts et les combine en une seule route.
- Fractionnement le la route la plus longue (SLR) : l'opération recherche l'itinéraire avec le coût le plus mauvais et le divise à un point aléatoire pour former deux routes séparées.
- Désordre aléatoire (RS) : À la fin de l'opérateur de mutation, une telle opération change et brouille l'ordre des clients de chaque itinéraire de l'individu actuel.

Nous décrivons à présent l'algorithme de mutation basé sur les opérateurs cités ci-haut.

Soit r^{op} un taux d'exécution d'un opérateur *op*. *op* correspondant à l'un des quatre opérateurs : PS, MSR, SLR, RS. La procédure de mutation multi-modale nécessite comme paramètres d'entrée, la population, la probabilité de PS, la probabilité de MSR, la probabilité de SLR et la probabilité de RS. L'algorithme retourne la population avec des individus mutés suivant les taux fixés au préalable.

La mutation *multi-modale* est appliquée à chaque solution (individu) de la population. Tout d'abord, le *swap partiel* est effectué avec une probabilité r^{PS}. Si le *swap partiel* n'est pas effectué alors l'opérateur de *fusion des routes les plus courtes* est appliqué avec une probabilité r^{MSR}, sinon le *fractionnement de la route la plus longue* est exécuté sur l'individu en cours. A la fin, l'opérateur de *désordre aléatoire* est appliqué avec une probabilité inférieur ou égal à r^{S}.

Afin d'améliorer la qualité locale des solutions au cours de l'optimisation, une heuristique de diversification (destruction/construction) ainsi que des opérateurs de recherche locale sont appliqués à chaque solution générée par l'algorithme de recherche globale NSGA-II. Ces procédures sont appliquées dans le but d'améliorer le premier objectif qui est le coût du trajet.

4.3.2 La procédure de diversification (heuristique de destruction/construction DCH)

Dans cette partie, nous avons étendu le travail effectué en incorporant l'heuristique de destruction et de construction (DCH). Cette heuristique contribue à améliorer la diversification et à fournir de meilleures solutions réalisables. Elle est injectée sur les solutions issues de l'application du croisement et de la mutation. Des techniques de diversification sont utilisées pour explorer différentes zones de l'espace de recherche afin d'échapper aux optima locaux. La diversification opérée dans ce travail est basée sur la dégradation de $\alpha\%$ de l'ensemble de la solution. α peut être considéré comme le degré de diversification. Cette procédure se compose de deux phases, la première libère certains clients de la solution d'entrée. Ensuite, une phase de reconstruction est appliquée en utilisant un algorithme glouton pour compléter la solution avec les clients supprimés en vue de trouver une solution différente qui conduit à un meilleur optima local lors de l'utilisation de l'intensification.

Une adaptation de DCH est utilisée pour étendre l'algorithme NSGA-II. Le schéma global de cette adaptation DCH est décrit dans l'algorithme 3. Cet algorithme est appliqué à une solution potentielle assignée. L'algorithme commence par fixer V_{best} comme étant le coût du trajet de la solution courante $V(\omega)$. Par la suite, à chaque itération, les $\alpha\%$ clients sélectionnés sont supprimés de la solution actuelle, et la solution résultante est placée dans une nouvelle copie nommée ω'. Après avoir exécuté la procédure destruction, la procédure construction reformule ω' en ajoutant les clients supprimés de manière gloutonne. Si le coût de trajet de la solution obtenue est meilleur que V_{best}, V_{best} ainsi que la meilleure solution ω_{best} sont actualisés. Cette procédure est répétée jusqu'à atteindre le nombre d'itérations maximal.

Algorithme 3 La procédure de Destruction et Construction (DCH)

Précondition : $\omega, V(\omega), \alpha$
$Iter = 1$;
$\omega_{best} = \omega; V_{best} = V(\omega)$;
tant que $Iter \leq MaxIter$
 $\omega' = Destruction(\omega, \alpha)$;
 $\omega' = Construction(\omega')$;
 si $V(\omega') \leq V_{best}$ **alors**
  $\omega_{best} = \omega'$;
  $V_{best} = V(\omega')$;
 fin si
 $Iter = Iter + 1$;
fin tant que
$\omega = \omega_{best}; V(\omega) = V_{best}$;

Une fois la procédure de diversification appliquée, une procédure d'intensification peut être appliquée juste après afin exploiter le voisinage local de l'espace de recherche résultant.

4.3.3 La procédure d'intensification (opérateurs de recherche locale)

La recherche locale (LS) est une technique utilisée pour affiner la qualité d'une solution afin de converger vers un optimum local. Cette technique est connue pour son efficacité lorsqu'elle est associée à des méta-heuristiques. En effet, elle peut contribuer à l'intensification en optimisant les résultats. Elle est considérée comme une procédure complémentaire pour les opérateurs évolutionnaires qui se concentrent principalement sur l'exploration globale.

Pour les problèmes du voyageur de commerce et de VRP, cinq stratégies de recherche locale bien connues de l'état de l'art sont généralement utilisées, à savoir : les déplacements locaux à un-point (Un-point), à deux-points (Deux-points), 2-opt, 2-opt inversé et la meilleure insertion (BI). Ces opérateurs visent à améliorer la qualité de chaque route séparément. Les déplacements locaux à un point (également connus avec relocalisation) et à deux-points sont également utilisés entre deux routes pour améliorer la solution globale. Nous décrivons à présent ces opérateurs de façon succincte.

— ***2-opt :*** cet opérateur supprime deux arêtes et les remplace par deux nouvelles arêtes en inversant la route qui se situe entre les deux. La figure 4.4 montre le mouvement 2-opt où les arêtes (A,B) et (E,D) sont remplacées par les arêtes (A,D) et (B,E) en inversant le segment [B,C,D] à [D,C,B]. Ce remplacement est par la suite évalué pour voir s'il induit une amélioration de la solution.

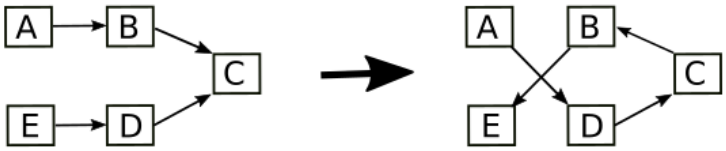


FIGURE 4.4 – L'opérateur 2-opt

— ***2-opt inversé :*** cet opérateur cherche deux nouvelles arêtes pour remplacer deux arêtes existantes chaque fois qu'une amélioration des objectifs est trouvée. Contrairement à 2-opt, tous les sous-tours sont inversés sauf celui entre les deux arêtes supprimées. La figure 4.5 montre un exemple de ce mouvement, où les arêtes [B,C] et [D,E] sont supprimées.

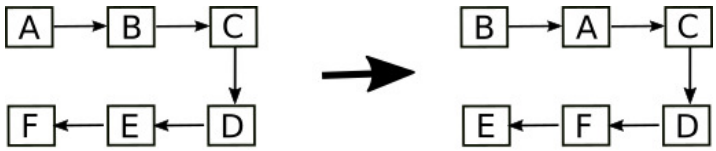
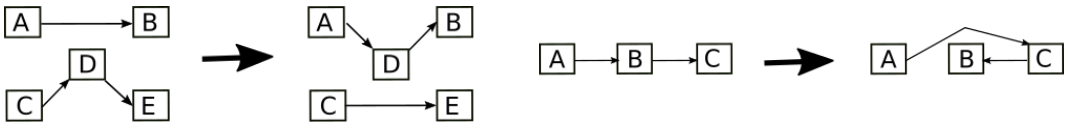


FIGURE 4.5 – L'opérateur 2-opt inversé

— ***Meilleure insertion :*** cet opérateur insère successivement les clients un par un dans les meilleures positions qui améliorent le coût de déplacement de chaque itinéraire séparément.
— ***Un-point :*** cet opérateur déplace un client assigné dans une nouvelle position. Ce client peut être déplacé soit sur la route où se trouve sa position actuelle, soit sur une autre route comme indiqué dans la figure 4.6. Par exemple, la figure 4.6a décrit le mouvement de délocalisation qui traite deux routes et cherche à améliorer la solution entière, où le client D est déplacé de la route [C,D,E] à l'autre route [A,B] pour avoir à la fin les nouvelles routes [C,E] et [A,D,B]. La figure 4.6b quant à elle décrit la délocalisation d'un client dans la même route, où le client C est délocalisé avant le client B. La délocalisation n'est effectuée que si la nouvelle position améliore la solution globale.
— ***Deux-points :*** échange les positions de deux clients assignés comme indiqué dans la figure 4.7. Dans la figure 4.7a, le mouvement deux-points est utilisé pour échanger les positions de B et E qui appartiennent à deux routes dif-



(a) Un-point pour deux routes (b) Un-point pour une seul route

FIGURE 4.6 – L'opérateur un-point

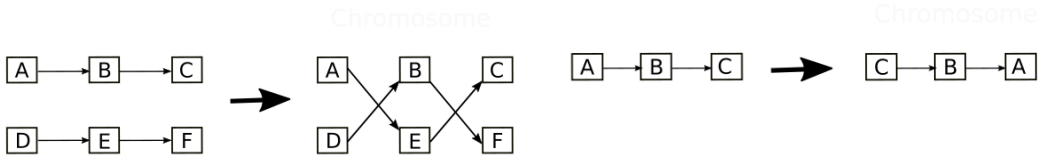
férentes. Cependant, dans la figure 4.7b les positions des clients A et C qui appartiennent à la même route sont échangées. Par conséquent, ces clients peuvent appartenir au même itinéraire ou à deux itinéraires différents.

Algorithme 4 Schéma global de la procédure de recherche locale

Précondition : P, α, $MaxIter$

pour tout ω de P **faire**
 Pour j = 1 à it_{max} **faire**
  $V(\omega') \leftarrow V(\omega)$;
  $\omega' \leftarrow \omega$;
  Pour i = 1 à 5 **faire**
   chosen[i] = False ;
  fin pour
  $i \leftarrow 1$;
  $op \leftarrow rand[1, 5]$;
  Tant que $(i \leq 5)$ et (chosen[op] $\neq$ Vrai) **faire**
   Si $op = 1$ **alors** $Un\_point(\omega', V(\omega'))$;
    chosen[1] = True ;
   fin si
   Si $op = 2$ **alors**
    $Deux\_point(\omega', V(\omega'))$;
    chosen[2] = True ;
   fin si
   Si $op = 3$ **alors**
    $2\_opt(\omega', V(\omega'))$;
    chosen[3] = True ;
   fin si
   Si $op = 4$ **alors**
    $2\_opt\_invers(\omega', V(\omega'))$;
    chosen[4] = True ;
   fin si
   Si $op = 5$ **alors**
    $Meilleure\_insertion(\omega', V(\omega'))$;
    chosen[5] = True ;
   fin si
   $i \leftarrow i + 1$;
   $op \leftarrow rand[1, 5]$;
  fin tant que
  Si $V(\omega') < V(\omega)$ **faire**
   $V(\omega) \leftarrow V(\omega')$;
   $\omega \leftarrow \omega'$;
  fin si
  $DCH(\omega, V(\omega), \alpha)$ // (voir algorithme 3) ;
 fin pour
fin pour

Suite à l'application de l'un de ces mouvements locaux, les nouveaux individus atteints ne seront conservés que s'ils sont meilleurs.



(a) Deux-points pour deux routes (b) Deux-points pour une route

FIGURE 4.7 – L'opérateur deux-points

L'algorithme 4 décrit la procédure de recherche locale. Sa boucle principale lance la recherche locale sur tous les individus ω de la population P. La boucle recherche la meilleure solution améliorante et s'arrête quand le nombre d'itérations maximal est atteint. Tout d'abord, $V(\omega')$ et ω' sont initialisés avec $V(\omega)$ et ω respectivement. Par la suite, les opérateurs de recherche locale sont appliqués dans un ordre aléatoire d'un individu ω à un autre, en s'assurant que tous les opérateurs sont appliqués une et une seule fois. A la fin, si la solution obtenue ω' permet d'améliorer la solution ω, ω' remplace ω.

Algorithme 5 Procédure des opérateur de recherche locale

Précondition : ω', $V_{\omega'}$, *op*
répéter
$V_{best} \leftarrow V(\omega')$;
$\omega_{best} \leftarrow \omega'$
$b \leftarrow faux$;
pour tout Mouvement l de L **faire** // L : liste des mouvements de type *op*
 Calculer ω^l et $V(\omega^l)$;
 si $V(\omega^l) < V_{best}$ **alors**
  $V_{best} \leftarrow V(\omega^l)$;
  $\omega_{best} \leftarrow \omega^l$
  $b \leftarrow vrai$;
 fin si
fin pour
si $b = vrai$ **alors**
 $\omega' \leftarrow \omega_{best}$
 $V(\omega') \leftarrow V_{best}$;
fin si
jusqu'à aucun mouvement améliorant trouvé

L'algorithme 5 décrit l'utilisation de chacun des opérateurs de recherche locale. Soit *op* le type d'opérateur et L l'ensemble de tous les mouvements possibles de type *op* qui peuvent être effectués. L'algorithme a comme entrées l'individu ω', son coût de trajet $V(\omega')$, et l'opérateur de recherche locale *op*. La boucle *répéter* assure de

continuer à améliorer l'individu en utilisant le même opérateur pendant qu'il y a des mouvements possibles qui peuvent améliorer sa qualité. Initialement, le meilleur coût V_{best} est initialisé pour chaque itération de la boucle (*répéter*) par le coût associé à l'individu actuel $V(\omega')$. Le paramètre b représente un booléen qui est égal à "$vrai$" quand l'individu a été amélioré suite au processus de déplacement, et "$faux$" sinon. Cette variable est initialisée à "faux" avant d'essayer les déplacements dans L. La boucle *pour* permet de tester tous les déplacements dans L. Puis, le coût $V(\omega^l)$ résultant de l'application du mouvement est calculé. Si la qualité de la solution obtenue est meilleure que toutes celles trouvées jusqu'ici, alors le meilleur coût V_{best} et le meilleur individu ω_{best} sont tous les deux mis à jour, et b reçoit la valeur "$vrai$". Après avoir essayé tous les mouvements (fin boucle *pour*), si b est égal à "vrai", alors le meilleur mouvement l_{best} est appliqué et le coût du trajet l'individu ω' et son coût de trajet $V(\omega')$ sont mis à jour.

4.4 Expérimentation

4.4.1 Instances utilisées et paramètres

La performance de l'algorithme d'optimisation évolutionnaire multiobjectif hybride (HMOGA) proposé est évaluée sur un ensemble d'instances de référence tirées de Solano *et al.* [Solano-Charris 2014]. Ces instances sont composées de 18 petites instances et de 20 grandes instances.

- Les petites instances contiennent 10, 15 et 20 clients, le nombre de véhicules varie de 2 à 3, alors que le nombre de scénarios varie de 10 à 30.
- Les grandes instances contiennent 50 et 100 clients, le nombre de véhicules peut prendre les valeurs 5 et 10, alors que le nombre de scénarios est fixé à 10 ou 20.

Les étiquettes des instances sont de la forme `n-m-p` pour les instances de petite taille et `n-m-p-d` pour les instances de grande taille, où n indique le nombre de clients, m est le nombre maximal de véhicules autorisé, p est le nombre de scénarios et d est la déviation en pourcentage des différents scénarios par rapport au coût de trajet de base. La méthode proposée a été implémentée en Langage C et exécutée sur un processeur Intel Pentium Core i5 avec 2.4 Ghz et 4 Go de RAM. Afin d'évaluer le comportement de la méthode proposée par rapport aux solutions optimales (quand cela est possible), nous avons également considéré les solutions de GLPK pour les instances de petite taille (comme envisagé dans Solano *et al.* [Solano-Charris 2014]) en fixant la limite d'exécution à quatre heures pour chaque instance résolue.

Généralement, lorsqu'on utilise des algorithmes approchés pour résoudre des problèmes d'optimisation combinatoire, il est bien connu que différents réglages de paramètres entraînent une variabilité de la qualité des résultats. Pour notre HMOEA, plusieurs paramètres doivent être définis tels que la taille de la population, le nombre de générations, les probabilités de mutation et de croisement. Pour étudier l'impact de ces paramètres sur les résultats, nous avons effectué un plan d'expérience détaillé dans le tableau 4.1 en utilisant les valeurs suivantes pour les probabilités des opéra-

Inst	Croisement		Mutation		PS		MST		RS	
	0.8	**1**	0.4	**0.6**	0.5	**0.7**	0.5	**0.7**	0.3	**0.5**
10-2-10	**217**	**217**	**217**	**217**	**217**	**217**	**217**	**217**	**217**	**217**
10-2-20	284.41	**284**	284.48	**284**	**284**	284.44	284.48	**284**	284.48	**284**
10-2-30	**302.86**	**302.86**	303.62	**302.86**	303.62	**302.86**	**302.86**	**302.86**	**302.86**	**302.86**
15-2-10	352.20	**349.13**	351.51	**349.13**	**349.13**	350.10	351.51	**349.13**	**349.13**	350.82
15-2-20	374.79	**374.48**	375.10	**374.48**	375.10	**374.48**	**374.48**	375.58	**374.48**	375.27
15-2-30	409	**407.31**	409.75	**407.31**	409.75	**407.31**	**407.31**	409.34	**407.31**	409.44
20-2-10	438.31	**440.48**	**440.48**	439.79	**440.48**	440.44	442.68	**440.48**	442.68	**440.48**
20-2-20	**486.96**	491.82	491.82	**486.96**	491.82	**486.96**	491.82	**486.96**	**486.96**	491.93
20-2-30	**510.31**	514.68	514.68	**510.31**	514.68	**510.31**	**510.31**	513.24	**510.31**	514.82
10-3-10	**255**	255.24	**255**	**255**	**255**	**255**	**255**	**255**	**255**	**255**
10-3-20	**316.65**	316.93	**316.65**	316.89	316.93	**316.65**	**316.65**	316.89	316.93	**316.65**
10-3-30	**337**	**337**	**337**	**337**	**337**	**337**	**337**	**337**	**337**	**337**
15-3-10	401.20	**395.75**	398.10	**395.75**	**395.75**	398.51	398.10	**395.75**	398.10	**395.75**
15-3-20	406.79	**403.44**	405.17	**403.44**	405.17	**403.44**	**403.44**	406.06	405.17	**403.44**
15-3-30	**437.86**	439.44	439.44	**437.86**	439.44	**437.86**	439.44	**437.86**	439.44	**437.86**
20-3-10	**471.10**	475.51	**471.10**	472.86	**471.10**	474.27	475.51	**471.10**	475.51	**471.10**
20-3-20	517.58	**516.86**	518.82	**516.86**	518.82	**516.86**	518.82	**516.86**	518.82	**516.86**
20-3-30	545.44	**539.72**	545.44	**539.72**	545.44	**539.72**	**539.72**	543.51	**539.72**	544.13

TABLE 4.1 – Meilleurs coûts de trajet moyens atteints par différentes paramétrisations sur les instances de petite taille

teurs : le taux de croisement est fixé à 0.8 ou 1, le taux de mutation prend les valeurs 0.4 et 0.6, le taux de l'opérateur de PS prend les valeurs 0.5 et 0.7, l'opérateur de MSR est réglé à 0.5 ou 0.7 et l'opérateur de RS a été testé avec les valeurs 0.3 et 0.5. La taille de la population ainsi que le nombre de générations sont fixés à 100. Enfin, 30 essais indépendants sont considérés pour chaque configuration.

Premièrement, les résultats obtenus pour ces paramètres sont présentés dans le tableau 4.1. Dans ce tableau, les solutions présentant les meilleurs coûts de trajet moyens, pour chaque taux assigné à un paramètre donné avec toutes les configurations possibles des autres paramètres, sont signalées. Les meilleures solutions obtenues, en moyenne, pour chaque instance sont marquées en gras.

A partir du tableau 4.1, on peut constater qu'aucune amélioration significative ne peut être associé à un choix spécifique de valeurs pour les paramètres, même si certaines valeurs restent plus intéressantes que d'autres. En effet, en analysant les résultats affichés dans le tableau, on observe ce qui suit :

1. Le nombre de meilleures solutions (moyennes) obtenues avec un taux de croisement de 1 est égal à 12 et celui des solutions obtenues avec un taux fixé à 0.8 sont égales à 9.
2. Le nombre de meilleures bornes augmente avec un taux de mutation égal à 0.6, où 15 meilleures solutions sont atteintes, tandis que seulement 6 meilleures solutions sont trouvées avec la valeur 0.4.
3. Pour le taux de PS, 13 meilleures bornes sont réalisées avec la valeur 0.7 et 8 meilleures bornes sont atteintes avec la valeur 0.5.

4. Le taux de MSR avec la valeur 0.7 atteint 12 meilleures solutions, tandis que les 10 meilleures solutions sont réalisées avec la valeur 0.5.
5. Enfin, le taux de RS avec la valeur 0.5 atteint 12 meilleures solutions et seulement 9 meilleures solutions sont atteintes avec un taux fixé à 0.3.

Sur la base des observations ci-dessus, nous considérons tout de même les valeurs qui ont permis d'obtenir le nombre maximal de meilleures bornes dans les expériences suivantes.

#Inst.	PopSize=100	PopSize=1000
10-2-10	217	*217*
10-2-20	284	*284*
10-2-30	301	*301*
10-3-10	255	*255*
10-3-20	316	*316*
10-3-30	337	*337*
15-2-10	346	*346*
15-2-20	373	*373*
15-2-30	406	**404**
15-3-10	381	*381*
15-3-20	401	**399**
15-3-30	435	**433**
20-2-10	431	**426**
20-2-20	484	**470**
20-2-30	506	**501**
20-3-10	460	**448**
20-3-20	505	**497**
20-3-30	536	**523**

TABLE 4.2 – Résultats pour une taille de population fixée à 100 ou à 1000 sur des petites instances.

Deuxièmement, nous avons étudié l'impact de la taille de la population (notée PopSize) sur les résultats obtenus par HMOEA. Dans ce cas, deux valeurs représentant la taille de la population ont été considérées : 100 et 1000. Les tableaux 4.2 et 4.3 montrent les résultats obtenus sur les instances à petite et à grande échelle respectivement. Notons que les solutions améliorées avec une taille de population 1000, par rapport à celles réalisées avec une taille de population 100, sont marquées en **gras**, alors qu'elles sont exprimées en *italique* lorsqu'elles sont égales. Des deux tableaux, nous observons ce qui suit :

1. Avec la taille de la population fixée à 100, HMOEA est capable d'atteindre 50% des valeurs optimales trouvées par le solveur GLPK. Le pourcentage devient plus important avec la taille de la population 1000 (sauf pour un seul cas avec 20 clients).

2. Pour les instances à grande échelle, les résultats atteints avec 1000 individus surpassent ceux obtenus avec 100 individus. Pour 25% des instances de 50 clients (marqués en italique), la méthode a atteint les mêmes solutions.

#Inst.	PopSize=100	PopSize=1000
50-5-10-10	8637	**8633**
50-5-10-50	*9936*	*9936*
50-5-10-100	*11534*	*11534*
50-5-20-10	8088	**7966**
50-5-20-50	*10014*	*10014*
50-5-20-100	10984	**10913**
50-10-10-10	*10882*	*10882*
50-10-10-50	13052	**12849**
50-10-10-100	16409	**16261**
50-10-20-10	*11581*	*11576*
50-10-20-50	14367	**14337**
50-10-20-100	15133	**15097**
100-10-10-10	12920	**12767**
100-10-10-50	15402	**15151**
100-10-20-10	13866	**13627**
100-10-20-50	16509	**16200**
100-20-10-10	22472	**22271**
100-20-10-50	25374	**25054**
100-20-20-10	20260	**20131**
100-20-20-50	25632	**24967**

TABLE 4.3 – Rrésultats pour une taille de population fixée à 100 et à 1000 sur des grandes instances

Par conséquent, nous pouvons conclure qu'une taille de population plus grande permet d'obtenir de meilleures solutions grâce à une meilleure exploration de l'espace de recherche.

Troisièmement, sur la base des résultats ci-dessus, HMOEA est ensuite appliqué avec le paramétrage suivant : la taille de la population et le nombre de générations sont fixés à 1000, la recherche locale est appliquée toutes les 50 générations, l'opérateur de croisement est appliqué avec une probabilité équivalente à 1, la probabilité d'appliquer l'opérateur de mutation est égale à 0.4. Chaque opération liée à la mutation multimodale est exécutée en utilisant les taux 0.5, 0.5 et 0.3 pour PS, MST et RS respectivement. En raison de l'aspect aléatoire de l'algorithme évolutionnaire, nous avons envisagé 30 essais indépendants pour chaque instance.

4.4.2 Analyse des résultats avec HMOEA

4.4.2.1 HMOEA sans l'heuristique DCH

Les résultats obtenus par la méthode HMOEA sans l'utilisation de l'heuristique de diversification DCH sont comparés, en terme du pire coût de trajet, avec les meilleures bornes trouvées par GLPK, ainsi que les résultats atteints par deux méthodes de la littérature. La première est une méthode globale basée sur un algorithme génétique, publiée dans [Solano-Charris 2014] – (notée GA). La deuxième est une

recherche locale itérative diversifiée avec la méthode "multi-start", où la génération initiale est faite avec la technique du tour géant, publiée dans [Solano-Charris 2015] – ("Multi-Start Iterated Local Search Giant Tour", notée MS-ILS-GT).

Le tableau 4.4 rapporte les résultats obtenus pour les petites instances. La colonne 1 du tableau 4.4 affiche l'étiquette de l'instance, la colonne 2 rapporte les solutions trouvées par GLPK. Les colonnes 3 et 4 montrent les solutions réalisées dans la littérature par MS-ILS-GT et GA respectivement. Les résultats fournis par HMOEA, en terme de pire coût de trajet et de nombre de véhicules, sont affichés dans les colonnes 5 et 6 respectivement. La colonne 7 montre le coût moyen trouvé par l'algorithme HMOEA.

Instance	GLPK (meilleur coût)	MS-ILS-GT (meilleur coût)	GA (meilleur coût)	HMOEA (meilleur coût)	HMOEA (meilleur Nv)	HMOEA (coût mouen)
10-2-10	217	**217**	**217**	**217**	2	217
10-2-20	284	**284**	**284**	**284**	2	284
10-2-30	301	**301**	**301**	**301**	2	301.29
10-3-10	255	**255**	**255**	**255**	3	255
10-3-20	316	**316**	**316**	**316**	3	316.07
10-3-30	337	**337**	**337**	**337**	3	337
15-2-10	346	**346**	**346**	**346**	2	347.95
15-2-20	373	**373**	374	**373**	2	374.512
15-2-30	404	**404**	409	**404**	2	406.4146
15-3-10	381	**381**	405	**381**	3	388.9756
15-3-20	399	**399**	411	**399**	3	403.707
15-3-30	433	**433**	436	**433**	3	435.707
20-2-10	423	**422***	447	*426*	2	435.29
20-2-20	470	**470**	493	**470**	2	486.146
20-2-30	501	**497***	520	**501**	2	508.9
20-3-10	448	**448**	484	**448**	3	467.756
20-3-20	497	**497**	520	**497**	3	512.19
20-3-30	528	**523***	551	**523***	3	536.56

TABLE 4.4 – Comparaison de HMOEA avec GLPK, GA et MS-ILS-GT sur l'ensemble des instances à petite échelle

Du tableau 4.4, on peut observer ce qui suit :

1. HMOEA est capable d'atteindre 16 valeurs optimales et d'améliorer une borne, et ne parvient pas à résoudre à optimalité une seule instance de 20 clients, avec un petit écart de 4 unités par rapport à la solution de GLPK.
2. GA atteint 7 valeurs optimales sur 18.
3. MS-ILS-GT est capable d'atteindre 15 valeurs optimales et d'améliorer trois bornes.
4. En termes de pourcentage, GA atteint 38.89% des solutions optimales, alors que HMOEA et MS-ILS-GT atteignent 94.44% et 100% des valeurs optimales respectivement.
5. En terme de nombre de véhicules utilisés, toutes les méthodes réalisent le même résultat, qui est équivalent aux valeurs maximales autorisées.

La figure 4.8 illustre le meilleur coût de trajet trouvé sur tous les essais et leurs moyennes pour chaque instance. L'axe horizontal se réfère aux instances et l'axe vertical représente le coût de déplacement trouvé lors de l'application de HMOEA. Les

meilleures bornes sont représentées par les barres rouges et les moyennes des essais sont affichées dans les barres vertes. Pour 10 clients, la meilleure borne est égale ou très proche de la moyenne, car HMOEA obtient la meilleure valeur de solution pour la majorité des essais, quel que soit le nombre de véhicules ou le nombre de scénarios pour les valeurs incertaines. HMOEA atteint les meilleures bornes sur beaucoup d'essais, pour les instances avec 15 clients, mais avec un gap entre la meilleure valeur et la moyenne un peu plus grand par rapport aux instances avec 10 clients. En plus, le nombre de véhicules peut aussi empêcher HMOEA d'atteindre la meilleure borne pour certains essais quand il est grand. Encore, moins d'essais atteignent la meilleure borne pour les instances avec 20 clients, où l'écart entre la meilleure valeur et la moyenne devient plus important. Par conséquent, le comportement de HMOEA sur les différents essais considérés, peut être affecté négativement par la croissance du nombre de clients ainsi que du nombre de véhicules.

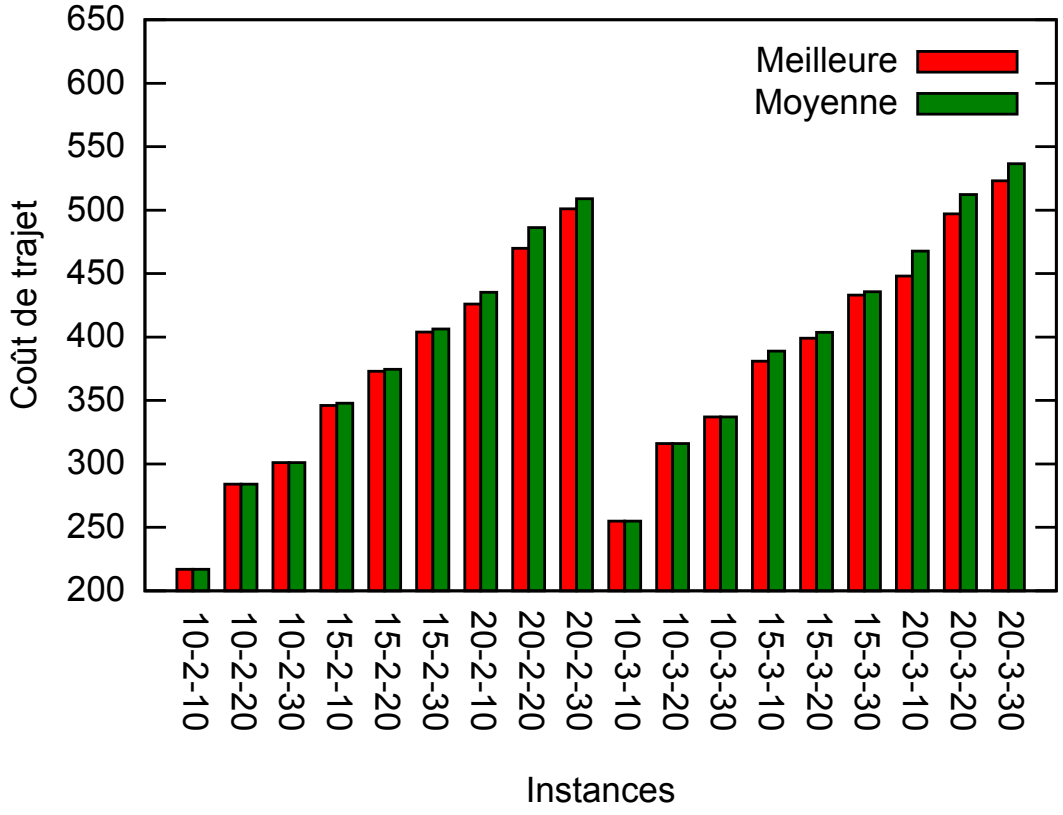


FIGURE 4.8 – La variation de l'écart entre le meilleur coût de trajet et la moyenne des coûts de trajet sur les différents essais pour les petites instances

D'autre part, le tableau 4.5 montre les résultats obtenus par HMOEA, MS-ILS-GT et GA sur les instances de grande taille. Dans ce cas, le solveur GLPK n'est pas capable de résoudre ces instances à optimalité à cause de la taille importante du problème. La première colonne présente les labels des instances. Les solutions obtenues par MS-ILS-GT et GA sont affichées dans la deuxième et la troisième colonnes

respectivement. La quatrième et la cinquième colonnes représentent les solutions (coût de trajet et nombre de véhicules) atteintes par le HMOEA proposé sans l'utilisation de l'heuristique de destruction/construction dans un premier temps, alors que la dernière colonne affiche le coût moyen pour HMOEA sur les différents essais considérés.

A partir du tableau 4.5, on peut observer que MS-ILS-GT et HMOEA dominent GA. En effet, pour les instances à grande échelle, HMOEA est capable de fournir de meilleures bornes pour toutes les 20 instances par rapport à GA. Cependant, en comparant le comportement de HMOEA avec MS-ILS-GT, nous observons une dégradation des solutions HMOEA principalement avec un nombre de client égal à 100. Globalement, HMOEA atteint les même résultats que MS-ILS-GT sur 50% des instances. Cependant, parallèlement à cela, nous pouvons constater que le nombre de véhicules trouvé par notre approche a été amélioré pour la plupart des instances pour lesquelles le coût se voit dégradé. Ceci illustre aussi la notion de compromis que nous avons introduit en considérant plusieurs objectifs antagonistes à optimiser dans notre approche multi-objectif.

Instance	MS-ILS-GT (meilleur coût)	GA (meilleur coût)	HMOEA (meilleur coût)	HMOEA (Nv)	HMOEA (coût moyen)
50-5-10-10	8633	9797	8633	5	8637.707
50-5-10-50	9936	12002	9936	5	9938.39
50-5-10-100	11525	13891	11534	5	11645.171
50-5-20-10	7966	9302	7966	5	8043.268
50-5-20-50	10014	11040	10014	5	10063.488
50-5-20-100	10913	13358	10913	5	10970.268
50-10-10-10	10882	12394	10882	10	10947.756
50-10-10-50	12849	15469	12849	10	13007.219
50-10-10-100	16212	18413	16261	10	16465.122
50-10-20-10	11576	13060	11576	10	11577.09
50-10-20-50	14337	15893	14337	10	14427.146
50-10-20-100	15042	16802	15097	10	15179.66
100-10-10-10	12767	14813	12767	9	13035.88
100-10-10-50	15082	19812	15151	10	15517.29
100-10-20-10	13614	16980	13627	10	13792.146
100-10-20-50	15890	19878	16200	10	16579.66
100-20-10-10	22252	26596	22271	19	22559.707
100-20-10-50	24690	30954	25054	19	25394.49
100-20-20-10	19942	23553	20131	19	20307.658
100-20-20-50	24715	28594	24967	19	25349.854

TABLE 4.5 – Comparaison de HMOEA avec GLPK, GA et MS-ILS-GT sur l'ensemble des instances à grande échelle

La Figure 4.9 contient les variations des meilleures coût de trajet obtenus par HMOEA, ainsi que les moyennes correspondantes de tous les essais pour les instances à moyenne échelle. À partir de cette figure, on peut observer que la différence entre les deux valeurs représentées devient globalement plus importante lorsque le nombre de clients, le nombre de véhicules et la valeur de déviation augmentent.

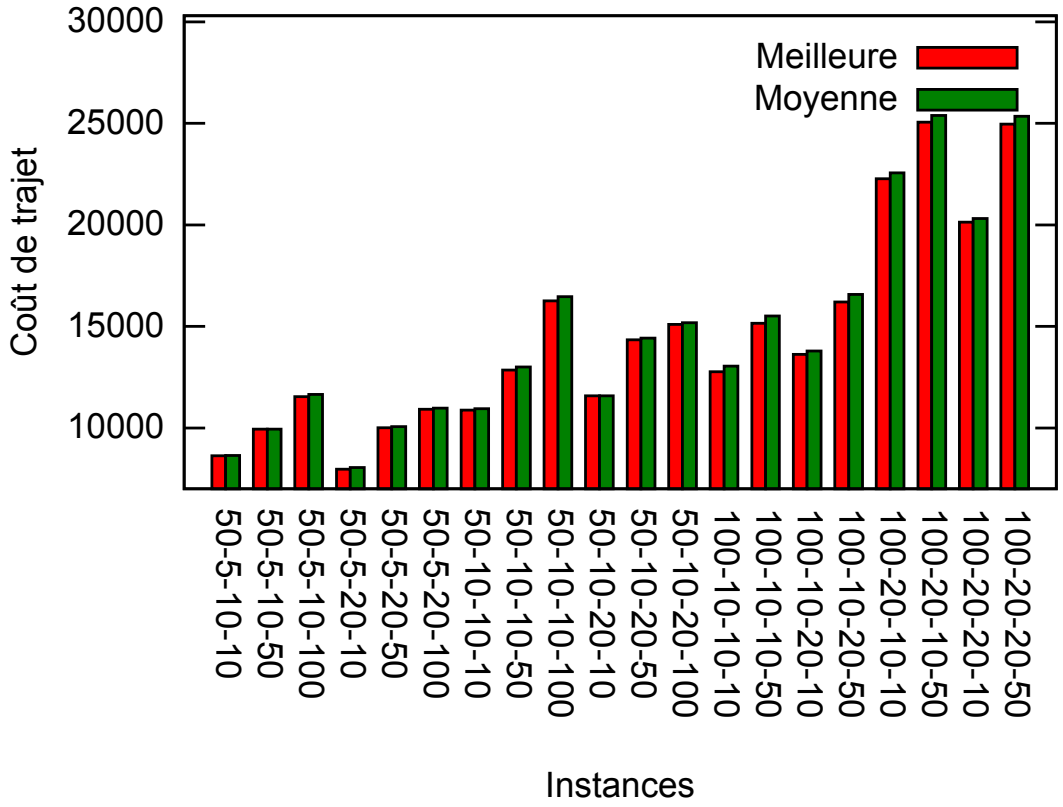


FIGURE 4.9 – La variation de l'écart entre le meilleur coût de trajet et la moyenne des coûts de trajet sur les différents essais pour les grandes instances

4.4.2.2 HMOEA avec l'heuristique DCH

Une version modifiée de la méta-heuristique est ensuite adoptée dans les prochaines expérimentations dans le but d'améliorer son comportement. Dans cette version, des stratégies de destruction et de construction sont incorporées dans l'algorithme HMOEA, afin de diversifier l'espace de recherche et d'éviter la convergence prématurée. Pour analyser l'impact de l'heuristique DCH sur le comportement de HMOEA, six valeurs différentes ont été testées pour le degré de diversification à savoir : $\alpha = \{10\%, 20\%, 40\%, 50\%, 60\%, 80\%\}$. Les résultats obtenus sont reportés dans le tableau 4.6. La colonne 1 représente l'étiquette de l'instance. La colonne 2 montre le meilleur coût obtenu pour les différentes valeurs de α. Dans les colonnes suivantes, les solutions correspondant à chaque α sont illustrées dans un ordre croissant de α.

À partir du tableau 4.6, nous remarquons que : pour les instances de grande taille avec 50 clients, HMOEA atteint toutes les meilleures bornes lorsque α est égal à 80%. La meilleure borne de l'instance 50-10-20-100 est ratée lorsque α est égal à 20% et 60%. Pour des degrés de diversification de 50% et 60%, HMOEA n'a pas réussi à atteindre les meilleures bornes pour deux instances : 50-5-10-100

Instance	Meilleur	$\alpha = 0.1$		$\alpha = 0.2$		$\alpha = 0.4$		$\alpha = 0.5$		$\alpha = 0.6$		$\alpha = 0.8$	
	C	C	Nv	C	Nv	C	Nv	C	Nv	C	Nv	C	Nv
50-5-10-10	8633	8633	5	8633	5	*8637*	5	8633	5	8633	5	8633	5
50-5-10-50	9936	9936	5	9936	5	9936	5	9936	5	9936	5	9936	5
50-5-10-100	11525	11525	5	11525	5	11525	5	*11534*	5	11525	5	11525	5
50-5-20-10	7966	7966	5	7966	5	7966	5	7966	5	7966	5	7966	5
50-5-20-50	10014	10014	5	10014	5	10014	5	10014	5	10014	5	10014	5
50-5-20-100	10913	10913	5	10913	5	10913	5	10913	5	10913	5	10913	5
50-10-10-10	10882	*10886*	10	10882	10	*10886*	10	*10886*	10	10882	10	10882	10
50-10-10-50	12849	12849	10	12849	10	12849	10	12849	10	12849	10	12849	10
50-10-10-100	16212	16212	10	16212	10	*16238*	10	16212	10	*16238*	10	16212	10
50-10-20-10	11576	*11621*	10	11576	10	11576	10	11576	10	11576	10	11576	10
50-10-20-50	14337	*14342*	10	14337	10	14337	10	14337	10	14337	10	14337	10
50-10-20-100	15042	15042	10	*15097*	10	15042	10	15042	10	*15097*	10	15042	10
100-10-10-10	12767	*12802*	9	*12770*	9	12767	9	*12818*	9	12767	9	12767	9
100-10-10-50	15082	*15422*	10	15082	10	*15185*	10	*15183*	10	*15309*	10	15082	10
100-10-20-10	**13589***	*13712*	10	*13681*	10	*13598*	10	13589	10	*13662*	10	*13634*	10
						/14070	/9	/13758	/9	*/13965*	/9	*/13844*	/9
100-10-20-50	16051	*16212*	10	*16301*	10	*16404*	10	*16453*	10	16051	10	*16246*	10
100-20-10-10	**22247***	22247	19	*22326*	19	*22371*	19	22260	19	*22290*	19	22260	19
100-20-10-50	24851	*24945*	19	*25007*	19	*24926*	19	*24985*	19	*25093*	19	24851	19
100-20-20-10	19942	*20133*	20	19942	19	*20098*	19	*20042*	19	*20082*	20	*20050*	20
		/20135	/19							*/20226*	/19	*/20161*	/19
100-20-20-50	24747	*24972*	19	*24827*	19	*24811*	19	*25143*	19	*24898*	19	24747	19

TABLE 4.6 – Comparaison de HMOEA-DCH avec plusieurs valeurs de α sur les grandes instances.

α	0.1	0.2	0.4	0.5	0.6	0.8
Performance (%)	50	70	50	60	60	85

TABLE 4.7 – Impact de α sur le comportement de HMOEA sur l'ensemble des grandes instances.

et 50-10-10-10 ; 50-10-10-100 et 50-10-20-100 respectivement. La pire performance est trouvée avec les degrés de diversification de 40% et 10% où trois bornes sont manquées pour les instances : 50-5-10-10, 50-10-10-10 et 50-10-10-100 ; 50-10-10-10, 50-10-20-10 et 50-10-20-50 respectivement. En comparant les meilleures solutions, globalement trouvées avec un α égal à 80%, avec les anciennes solutions (sans DCH), on observe que :

- La stratégie DCH a permis d'améliorer les coûts de trajet dans trois instances avec 50 clients : 50-5-10-100, 50-10-10-100 et 50-10-20-100 sans dégrader le nombre de véhicules.
- Toutes les instances de 100 clients sont améliorées avec l'incorporation de la stratégie DCH.
- Nous remarquons également que HMOEA couplé avec la stratégie de destruction/construction ramène deux nouvelles bornes de valeurs 13589 et 22247 pour les instances 100-10-20-10 et 100-20-10-10 respectivement (marquée en gras dans le tableau).

Le tableau 4.7 et la figure 4.10 rapportent les pourcentages des meilleures bornes, affichées dans la deuxième colonne (Meilleur C) du tableau 4.6, obtenues pour chaque α testé dans HMOEA-DCH. Les colonnes de 2 à 7 représentent les stratégies utilisées, c.à.d. les valeurs de α (10 %, 20 %, 40%, 50%, 60% et 80%), et la ligne affiche la performance correspondante. Ce dernier indique que lorsque le degré de dégradation

Instance	MS-ILS-GT (meilleur C)	HMOEA (meilleur C)	HMOEA (Nv)	HMOEA-DCH (meilleur C)	HMOEA-DCH (Nv)
50-5-10-10	8633	8633	5	8633	5
50-5-10-50	9936	9936	5	9936	5
50-5-10-100	11525	11534	5	11525	5
50-5-20-10	7966	7966	5	7966	5
50-5-20-50	10014	10014	5	10014	5
50-5-20-100	10913	10913	5	10913	5
50-10-10-10	10882	10882	10	10882	10
50-10-10-50	12849	12849	10	12849	10
50-10-10-100	16212	16261	10	16212	10
50-10-20-10	11576	11576	10	11576	10
50-10-20-50	14337	14337	10	14337	10
50-10-20-100	15042	15097	10	15042	10
100-10-10-10	12767	12767	9	12767	9
100-10-10-50	15082	15151	10	15082	10
100-10-20-10	13614	13627	10	**13589***	10
100-10-20-50	15890	16200	10	*16051*	10
100-20-10-10	22252	22271	19	**22247***	19
100-20-10-50	24690	25054	19	*24851*	19
100-20-20-10	19942	20131	19	19942	19
100-20-20-50	24715	24967	19	*24747*	19

TABLE 4.8 – Comparaison de HMOEA-DCH avec HMOEA et MS-ILS-GT sur l'ensemble des instances à grande échelle.

(destruction) de la solution est plus important, la méta-heuristique arrive à améliorer plus de solutions en termes de coût de trajet sans dégrader le nombre de véhicules.

Le tableau 4.8 permet de comparer les meilleures valeurs de coût de trajet obtenus par HMOEA-DCH avec les approches : HMOEA, MS-ILS-GT. Nous n'avons pas rapporté les résultats de GA car ils sont déjà dominés par les résultat de HMOEA (sans DCH). En analysant ces résultats, nous pouvons constater une nette amélioration avec HMOEA-DCH avec lequel deux nouvelles bornes ont été atteintes pour les instances 100-10-20-10 et 100-20-10-10, tandis que MS-ILS-GT reste plus performant sur 3 instances avec 100 clients.

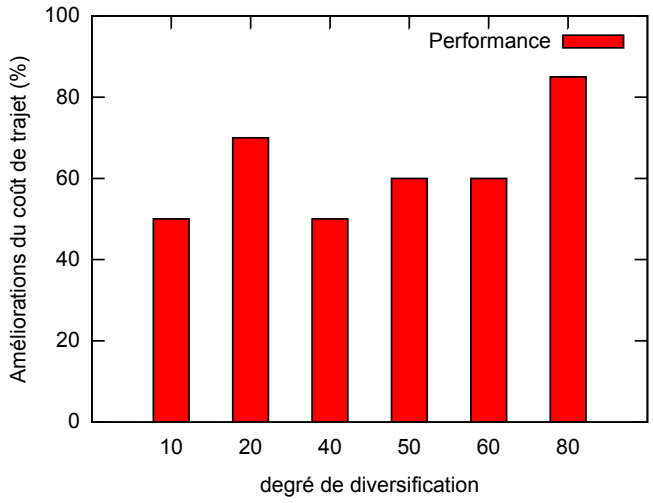


FIGURE 4.10 – Le taux d'amélioration des coûts de trajet pour chaque valeur de α

4.4.2.3 Évaluation de l'impact de l'heuristique destruction/construction

Afin d'évaluer l'impact de l'incorporation de l'heuristique destruction/construction sur le comportent de l'algorithme HMOEA, une comparaison basée sur un indicateur de performance multi-objectif a été réalisée. Plusieurs indicateurs de performance ont été proposés dans la littérature de l'optimisation multiobjectif, parmis lesquels nous pouvons citer : l'indicateur de l'hypervolume [Zitzler 1999], l'indicateur epsilon [Zhao 2003]. Dans ce qui suit, nous utilisons l'indicateur de l'hypervolume pour comparer les deux variantes de l'algorithme HMOEA à savoir : avec et sans destruction/construction . Cet indicateur permet de mesurer la qualité du front de Pareto obtenu en termes de convergence et de diversité.

L'indicateur de performance de l'hypervolume est calculé pour chaque instance du benchmark en utilisant les fronts de Pareto successifs trouvés par l'algorithme. Le calcul de l'hypervolume nécessite la désignation d'un point de référence. Nous avons fixé ce dernier à ($f_1 = 1000, f_2 = 30$) pour les instances moyennes (50 clients) et ($f_1 = 90000, f_2 = 70$) pour les grandes instance (100 clients).

La figure 4.11 montre l'évolution de l'hypervolume durant les 1000 générations accomplies pour une grande instance "100-10-10-10 pour les deux variantes : HMOEA et HMOEA-DCH respectivement. A noter que plus l'hypervolume est grand, plus le front de Pareto est meilleur. Nous pouvons remarquer que l'algorithme HMOEA-DCH est capable d'obtenir une meilleur front que celui obtenu par HMOEA après 1000 générations.

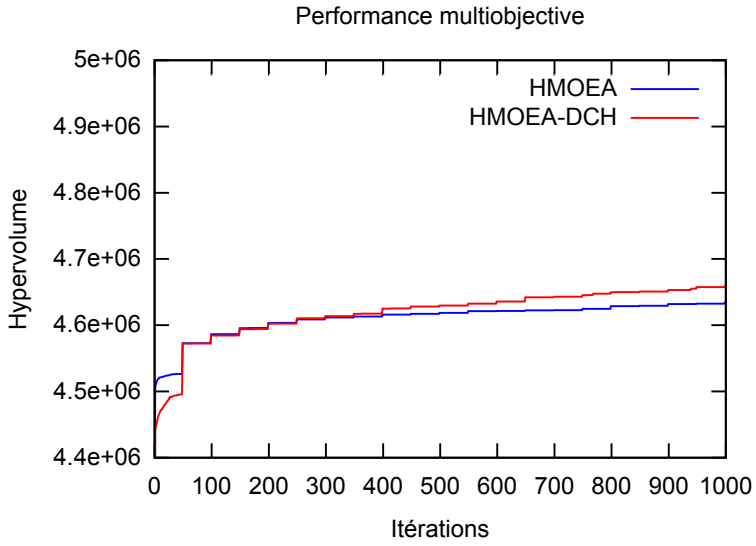


FIGURE 4.11 – Evolution de l'hypervolume de HMOEA et HMOEA-DCH pour l'instance 100-10-10-10

Dans le tableau 4.9 sont rapportés les résultats des deux variantes pour toutes les instances de grande taille. La colonne 1 présente les étiquettes des instances. les valeurs de l'hypervolume pour HMOEA et HMOEA-DCH sont rapportées dans

les colonne 2 et 3 respectivemet. Les valeurs correspondant à HMOEA-DCH sont marquées d'une * lorsqu'elles sont plus grandes (meilleures) que les valeurs correspondantes à HMOEA et elles sont marquées avec un - sinon. De cette table, nous observons :

— Pour les cas avec 50 clients, HMOEA-DCH surpasse HMOEA sur 7 instances parmi 12 (un pourcentage de 58,33%) et échoue sur 5 instances.
— Pour les instances avec 100 clients, le pourcentage passe à 75% où HMOEA-DCH améliore 6 parmi 8 instances.

Nous pouvons conclure de cette comparaison que l'heuristique destruction/construction permet globalement d'améliorer la qualité des résultats trouvés.

Instance	HMOEA	HMOEA-DCH
50-5-10-10	5.28878e+06	5.28887e+06*
50-5-10-50	5.20393e+06	5.20316e+06-
50-5-10-100	5.09254e+06	5.09471e+06*
50-5-20-10	5.3272e+06	5.32815e+06*
50-5-20-50	5.1961e+06	5.19756e+06*
50-5-20-100	5.1369e+06	5.1362e+06-
50-10-10-10	4.74314e+06	4.74547e+06*
50-10-10-50	4.61957e+06	4.6275e+06*
50-10-10-100	4.4121e+06	4.41138e+06-
50-10-20-10	4.70538e+06	4.70036e+06-
50-10-20-50	4.53438e+06	4.53309e+06-
50-10-20-100	4.48922e+06	4.4914e+06*
100-10-10-10	4.63489e+06	4.66015e+06*
100-10-10-50	4.46947e+06	4.46586e+06-
100-10-20-10	4.58022e+06	4.58846e+06*
100-10-20-50	4.40556e+06	4.40693e+06*
100-20-10-10	3.43949e+06	3.44295e+06*
100-20-10-50	3.29495e+06	3.29363e+06-
100-20-20-10	3.55241e+06	3.55322e+06*
100-20-20-50	3.29576e+06	3.29705e+06*

TABLE 4.9 – Indicateur de l'hypervolume pour HMOEA et HMOEA-DCH sur l'ensemble des grandes instances.

4.4.2.4 Évaluation du critère MNSQW

Dans la dernière partie des expérimentation, nous avons essayé d'étendre le travail pour évaluer le critère de robustesse MNSQW proposé dans le chapitre précédent. Ce critère a déjà fait l'objet d'une évaluation avec une méthode exacte et une heuristique basée sur la recherche à voisinage large (LNS). Pour HMOEA, nous avons évalué le critère MNSQW sur les grandes instances. Les résultats sont rapportés dans le tableau 4.10. On peut constater de ce tableau que le critère MNSQW domine faiblement le critère du pire cas avec une amélioration (dominance stricte) des résultats des instances 50-5-10-100, 50-10-10-100 et 100-10-10-50.

Instance	$HMOEA^{worst}$	$HMOEA^{MNSQW}$
50-5-10-10	8633	8633
50-5-10-50	9936	9936
50-5-10-100	11534	**11525**
50-5-20-10	7966	7966
50-5-20-50	10014	10014
50-5-20-100	10913	10913
50-10-10-10	10882	10882
50-10-10-50	12849	12849
50-10-10-100	16261	**16212**
50-10-20-10	11576	11576
50-10-20-50	14337	14337
50-10-20-100	15097	15097
100-10-10-10	12767	12767
100-10-10-50	15151	**15126**
100-10-20-10	13627	13627
100-10-20-50	16200	16200
100-20-10-10	22271	22271
100-20-10-50	25054	25054
100-20-20-10	20131	20131
100-20-20-50	24967	24967

TABLE 4.10 – Comportement des critères MNSQW versus pire cas, sur un premier ensemble d'instances.

4.5 Conclusion

Dans ce chapitre, nous avons proposé une approche évolutionnaire hybride multi-objectif pour résoudre le problème de tournées de véhicules robuste avec un coût de trajet incertain. L'approche proposée tente d'optimiser simultanément le nombre de véhicules utilisés et le critère min-max sur le coût de trajet. Afin de pouvoir appliquer NSGA-II sur le problème MO-RVRP, ses opérateurs ont été adaptés pour bien représenter le problème étudié. L'algorithme proposé HMOEA est une hybridation de l'algorithme NSGA-II adapté et de plusieurs opérateurs de recherche locale. De plus, une heuristique de diversification basée sur l'algorithme destruction/construction (DCH) a été incorporée dans notre approche (HMOEA-DCH). Afin d'évaluer l'approche proposée, cette dernière a été implémentée et testée sur une série d'instances tirée de l'état de l'art. Les résultats obtenus ont été comparés avec ceux d'une méthode exacte (GLPK) et de deux méthodes méta-heuristiques de la littérature. L'analyse des résultats a montré la capacité de notre approche à atteindre la quasi-totalité des bornes trouvées par GLPK pour les petites instances (sauf pour une instance). La comparaison des résultats obtenus avec HMOEA-DCH par rapport aux deux autres méta-heuristiques de l'état de l'art a permis de montrer que notre

approche domine totalement la première (algorithme génétique mono-objectif), et est compétitive avec la deuxième (recherche locale itérative "multi-start").

Chapitre 5

Problème de tournées de véhicules sélectives

Sommaire

Les travaux présentés dans ce chapitre on fait l'objet des publications [Bederina 2017a] et [Bederina 2017b].

5.1 Introduction

Le problème de tournées de véhicules sélectives (TOP, Team Orienteering Problem) est un problème d'optimisation combinatoire considéré comme l'une des variantes les plus étudiées des problèmes de tournées de véhicules. L'objectif du TOP est de sélectionner un sous-ensemble de clients dans l'ensemble complet pour chaque véhicule de telle sorte que la somme des scores obtenus en visitant ces clients soit maximisée sous réserve de la contrainte de coût de trajet maximal.

Le TOP traditionnel n'optimise généralement que le profit total (voir section 2.4.2), tandis le coût de trajet total est souvent optimisé en utilisant des opérateurs de recherche locale seulement. Cependant, les coûts de déplacement (coût de trajet)

et les bénéfices (profit) sont deux objectifs antagonistes (tel que mentionné dans [Bouly 2010]), car lorsque le coût du trajet diminue, il est possible d'ajouter davantage de clients aux itinéraires ce qui conduit à l'augmentation du bénéfice. Il est à noter également que la seule méthode multi-objectif consacrée à l'étude de TOP que nous avons pu trouver dans la littérature est celle proposée par [Schilde 2009], où plusieurs aspects de profits étaient considérés lors de la visite de clients tels que : la valeur culturelle, le nombre de magasins, etc. L'optimisation multi-objectif semble être un bon candidat pour résoudre des problèmes avec des objectifs conflictuels plutôt que de les agréger en une seule fonction. Elle permet d'optimiser simultanément tous les objectifs considérés en utilisant le concept de dominance de Pareto décrit dans le chapitre 2. Dans ce chapitre, nous proposons de résoudre une formulation multi-objectif du TOP en utilisant un algorithme évolutionnaire multi-objectif hybridé par une procédure de recherche locale. Notre choix s'est porté sur l'algorithme NSGA-II (cf. Deb *et al.* [Deb 2000]) qui est l'un des algorithmes les plus populaires dans ce domaine, comme nous l'avons mentionné dans le chapitre de l'état de l'art.

Dans un premier temps, nous nous intéressons à la résolution d'une variante déterministe du problème TOP multi-objectif proposé, en optimisant le profit total et le coût du trajet total.

Dans un deuxième temps, nous nous focalisons sur le TOP multi-objectif avec un temps de trajet incertain. Le temps de trajet entre deux emplacements peut prendre sa valeur dans un ensemble de scénarios. Cet ensemble de scénarios peut être continu ou discret. Au meilleur de nos connaissances, il n'y a qu'un seul travail dans la littérature [Ke 2013] qui a adopté une approche d'optimisation robuste sur un problème de tournées de véhicules sélectives avec des données incertaines dans un intervalle (TOPID), dans lequel tous les paramètres du modèle, i.e les score, les temps de services et les temps de déplacements prennent leurs valeurs dans des intervalles, où un algorithme de "branch and price algorithm" est développé pour résoudre la contrepartie ("counterpart") robuste. En plus, il y a des travaux qui ont abordé le problème de TOP avec une incertitude de nature stochastique. Le problème de l'orientation par équipe stochastique (TOPSO) a été étudié dans la littérature en considérant l'incertitude soit sur les coûts de trajet [Verbeeck 2014, Verbeeck 2016] entre deux emplacements ou sur le profit à chaque emplacement [Ilhan 2008]. Dans ce chapitre, l'approche de l'optimisation robuste (RO) est choisie pour modéliser et résoudre la variante TOP multi-objectif avec incertitudes. Le coût de déplacement incertain est modélisé à l'aide de scénarios discrets.

Le reste de ce chapitre est organisé comme suit. La section 5.2 décrit formellement le TOP multi-objectif avec ses deux variantes déterministe et robuste. La section 5.3 est dédiée à la description de l'algorithme hybride multi-objectif proposé. La section 5.4 décrit les opérateurs de recherche locale utilisés pour hybrider la méthode globale. Dans la section 5.5 nous présentons les expérimentations réalisées et discutons les résultats obtenus. Enfin, la section 5.6 conclut ce chapitre et donne quelques perspectives.

5.2 Formulation du problème TOP Multi-objectif

5.2.1 Variante multi-objectif du TOP déterministe (MO-TOP)

Soit $G = (V, E)$ un graphe connexe où V représente l'ensemble des nœuds (ou lieux ou clients à visiter) $V = \{1, ..., n\}$. Chaque nœud i a un score s_i à attribuer au touriste (véhicule) lors de sa visite. E indique l'ensemble des arcs entre chaque deux nœuds. Chaque arc (i, j) entre les nœuds i et j est décrit par un coût de déplacement non négatif C_{ij}. La variable de décision binaire x_{ijp} est le chemin direct reliant le nœud i au nœud j. x_{ijp} est égal à 1 quand l'arc (i, j) est croisé dans une route p (touriste ou véhicule), et 0 sinon. Une variable de décision binaire y_{ip} est égale à 1 si le client i existe dans la route p et 0 sinon. La variable de décision entière u_{ip} donne la position du client i dans le chemin p. Si le client i n'est pas visité dans le chemin p, alors u_{ip} est égal à 0. Le TOP est également décrit par m véhicules identiques. Le coût du trajet pour chaque route R est noté $D(R)$. $D(R)$ est limité par un coût maximum $T_{\max}$ (quota).

Formellement, le problème du TOP multi-objectif peut être écrit comme suit :

$$
\text{TOP}\begin{cases}
Z = & \min\big(-P(LR), D(LR)\big) & (1)\\
s.c. & \sum_{p=1}^{m}\sum_{i=2}^{n-1} x_{1ip} \le m & (2)\\
& \sum_{p=1}^{m}\sum_{i=2}^{n-1} x_{inp} \le m & (3)\\
& \sum_{p=1}^{m} y_{kp} \le 1,\ \forall k = 2, ..., n-1 & (4)\\
& \sum_{j=1}^{n-1} x_{ijp} = y_{ip}, & \\
& i = \{2, \ldots, n-1\}\ p = \{1, .., m\} & (5)\\
& \sum_{j=2}^{n} x_{jip} = y_{ip}, & \\
& i = \{2, \ldots, n-1\}\ p = \{1, .., m\} & (6)\\
& \sum_{i=1}^{n-1}\sum_{j=2}^{n-1} C_{ij}x_{ijp} \le T_{\max}, & \\
& p = \{1, ..., m\} & (7)\\
& u_{ip} - u_{jp} + 1 \le (n-1)(1 - x_{ijp}) & \\
& i, j = \{2, ..., n\}, p = \{1, ..., m\} & (8)\\
& x_{ijp}, y_{ip} \in \{0, 1\}, & \\
& \forall i, j = \{2, ..., n\}, p = \{1, ..., m\} & (9)\\
& 2 \le u_{ip} \le n, & \\
& \forall i = \{2, ..., n\}, p = \{1, ..., m\}, & (10)
\end{cases}
$$

Où la fonction objectif consiste en deux objectifs : le premier essaie de minimiser le coût du trajet total $D(LR)$ tandis que le second maximise le profit (bénéfice) total collecté $P(LR)$. Le coût de trajet total $D(LR)$ est donné par la somme des coûts de trajets pour chaque véhicule (à savoir $LR = \{R_1, R_2, ..., R_m\}$) et peut être calculé

comme suit :

$$D(LR) = \sum_{R \in LR} \sum_{(i,j) \in R} C_{ij}. \tag{5.1}$$

Le bénéfice total collecté $P(LR)$ de tous les véhicules peut être calculé comme suit :

$$P(LR) = \sum_{R \in LR} \sum_{i \in R} s_i. \tag{5.2}$$

Les contraintes (2) et (3) garantissent que le nombre de véhicules qui quittent le dépôt de départ 1 est le même que le nombre de véhicules qui reviennent au dépôt d'arrivée n et est égal à m au plus. La contrainte (4) garantit que chaque client de 2 à $n-1$ est visité au plus une fois. Les contraintes (5) et (6) représentent les contraintes de conservation de flux des clients et assurent que le nombre de véhicules qui entrent dans un nœud est le même que le nombre de véhicules qui le quittent. La contrainte (7) garantit que le coût de déplacement de chaque route n'excède pas la valeur T_{max}. La contrainte (8) évite les sous-cycles dans les routes. La contrainte (9) représente les variables binaires x_{ijp} et y_{ip}, et la contrainte (10) dénote les bornes des variables u_{ip}.

5.2.2 Variante multi-objectif du TOP robuste (MO-RTOP)

Dans la variante robuste MO-RTOP que nous proposons, les coûts de trajet entre les villes sont incertains. Les arcs (i, j) dans le graphe sont représentés par un ensemble de q scénarios où chaque scénario est associé avec un coût de déplacement possible C_{ij}^s où s indique l'indice du scénario, $s = \{1, ..., q\}$. Par conséquent, le coût de déplacement d'un tour R est calculé à chaque fois pour un scénario particulier s, c.a.d. q valeurs de coût de trajet pour chaque tour, et est noté $D^s(R)$. Chacun des $D^s(R)$ calculés est limité par le coût maximum T_{max}. De plus, les frais de voyage de LR sont indiqués par $D^s(LR)$. La formulation considérée est basée sur le critère de robustesse du Pire cas et peut être définie comme suit :

$$\text{RTOP} \begin{cases} Z = & \min\big(-P(LR), \sigma\big) & (11) \\ s.c. & D^s(LR) \leq \sigma \quad \forall s = \{1, ..., q\} & (12) \\ & \sum_{i=1}^{n-1} \sum_{j=2}^{n-1} C_{ij}^s x_{ijp} \leq T_{\max}, & \\ & p = \{1, ..., m\} \; s = \{1, ..., q\} & (13) \\ & (2) - (6) \; et \; (8) - (10) & \end{cases}$$

Où les deux objectifs à optimiser sont : la minimisation du pire coût total de déplacement $D^s(LR)$ parmi tous les scénarios s et la maximisation du bénéfice total collecté $P(LR)$.

La contrainte (12) assure que σ est le coût de déplacement du pire scénario. La contrainte (13) garantit que les différents scénarios de coûts de déplacement de chaque route n'excède pas la valeur T_{max}.

5.3 Optimisation évolutionnaire multi-objectif hybride (HMOEA) pour le MO-TOP

Afin de résoudre le problème du TOP en utilisant la description formelle ci-dessus, l'algorithme évolutionnaire multi-objectif NSGA-II (NSGA-II, Deb *et al.* [Deb 2000]) est considéré. Cet algorithme est reconnu comme l'une des meta-heuristiques les plus populaires lorsque des problèmes multi-objectifs sont abordés. La méthode hybride proposée NSGA-II avec certains opérateurs de recherche locale, où les opérateurs servent à mettre en évidence la qualité des solutions atteintes à chaque étape de NSGA-II.

5.3.1 L'algorithme multi-objectif NSGA-II

Nous présentons dans cette section l'adaptation des opérateurs de l'algorithme NSGA-II (décrit dans le chapitre 2) pour résoudre le problème TOP. Cette adaptation ressemble à celle réalisée sur le problème VRP (cf chapitre 4) avec quelques spécificités liées au problème TOP que nous décrivons dans ce qui suit.

5.3.1.1 La représentation du chromosome

Le chromosome représente une solution complète. Le nombre de gènes est égal au nombre de véhicules. Chaque gène représente une route (véhicule). Les routes commencent à partir d'un dépôt de départ noté 1 puis visitent un sous-ensemble de clients et se terminent en visitant le dépôt d'arrivée noté n. Le coût de trajet de chaque route (gène) ne peut pas dépasser la valeur fixe T_{max}. Cela signifie que certains clients peuvent être insatisfaits (non desservis) en raison de la restriction T_{max}.

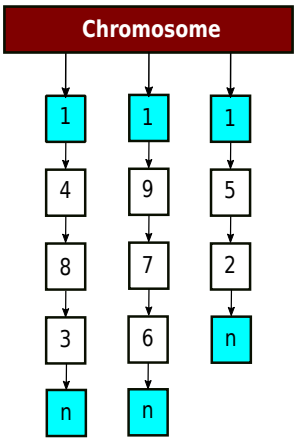


FIGURE 5.1 – La représentation du chromosome pour la solution au problème du TOP

5.3.1.2 Population initiale

Un simple algorithme glouton est utilisé pour construire les solutions initiales qui sont utilisées comme première génération de l'algorithme. Les routes sont construites séparément, où pour chaque route, un client aléatoire est ajouté à la dernière position en vérifiant la constante T_{max}. Ensuite, les routes sont séquentiellement créées en suivant le processus susmentionné jusqu'à ce que les m véhicules sont remplis.

5.3.1.3 La procédure de croisement

Il existe différentes manières de gérer l'opérateur de croisement. Dans ce travail, nous avons opté pour un opérateur simple proposé par Cheong *et al.* [Cheong 2006]. Il s'agit de sélectionner deux parents en utilisant *la sélection de tournoi* et d'appliquer le *croisement* entre les parents afin de générer deux enfants. *La sélection de tournoi* sélectionne chaque parent en choisissant aléatoirement une paire de chromosomes de la population parente. Une comparaison de dominance est ensuite effectuée et le meilleur parent est conservé.

Le *croisement* permet de partager les meilleures routes entre les deux parents sélectionnés, où la meilleure route liée à *parent1* est copiée vers *parent2* et *vice versa* (comme illustré dans la figure 5.2).

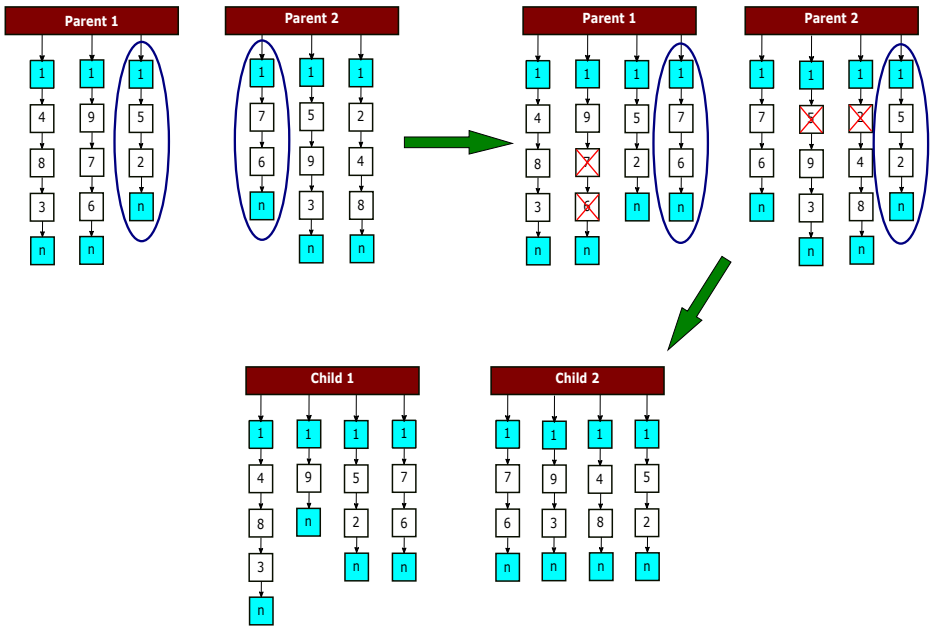


FIGURE 5.2 – Illustration de l'opération de croisement

Toutefois, l'opérateur de croisement augmente le nombre de véhicules d'une unité chaque fois qu'il crée des enfants auprès de leurs parents. Par conséquent, les nouvelles configurations ne sont pas nécessairement réalisables car elles peuvent dépasser le nombre maximum de véhicules m. Ainsi, une fonction de réparation est propo-

sée afin de corriger la faisabilité des solutions construites lors de l'utilisation de l'opérateur de croisement. Cette procédure sera décrite dans la section 5.3.2.

5.3.1.4 La procédure de mutation multimodale

La procédure multimodale de mutation consiste à introduire des opérateurs simples et efficaces, tels que *le swap partiel*, *fusion des routes les plus courtes*, *fractionnement de la plus longue route* et *ordre aléatoire*. Dans ce qui suit, nous décrivons brièvement les opérateurs de mutation :

- *Swap partiel :* un segment aléatoire est déplacé d'une route aléatoire vers une autre route aléatoire dans une position aléatoire. Ce déplacement est effectué uniquement si le coût de déplacement des routes résultantes ne viole pas le $Tmax$.
- *Fusion des routes les plus courtes :* les deux routes ayant les meilleurs coûts de transport sont sélectionnées pour être fusionnées en une seule route. La solution résultante est faisable uniquement si elle ne viole pas la contrainte de $Tmax$.
- *Fractionnement de la plus longue route :* la route qui a le pire coût de déplacement se divise en un point aléatoire.
- *Ordre aléatoire :* change l'ordre des clients dans chaque itinéraire.

Chacun de ces opérateurs est appliqué avec une certaine probabilité telle qu'utilisé dans Cheong *et al.* [Cheong 2006]. Le *swap partiel* est appliqué avec une certaine probabilité (appelée le taux élastique). Si le swap partiel n'est pas appliqué, la *fusion des routes les plus courtes* est effectuée avec une certaine probabilité (appelée le taux de compression). Le *fractionnement de la plus long route* est considéré chaque fois qu'aucun des deux premiers opérateurs n'est exécuté. Enfin, l'opérateur *ordre aléatoire* est exécuté avec une certaine probabilité. Notons que chaque déplacement de ces opérateurs est effectué uniquement si la contrainte de T_{max} n'est pas violée.

5.3.2 Fonction de correction de faisabilité

L'individu obtenu après l'application des opérateurs de croisement et de mutation n'est pas forcément faisable. Ceci est dû particulièrement à la violation des contraintes du modèle. Dans le cas où la contrainte T_{max} est violée, l'individu est rejeté. Par ailleurs, la contrainte sur le nombre de véhicules maximal m peut être violée par l'application des opérateurs de croisement et de mutation. Pour cette raison, nous proposons une fonction de réparation de faisabilité. Cette dernière consiste à calculer le profit collecté pour chaque route et de trier par la suite les routes avec un ordre décroissant en fonction de leurs profits. Enfin, les m premières routes sont gardées et toutes les routes restantes sont supprimées comme montré sur la figure 5.3.

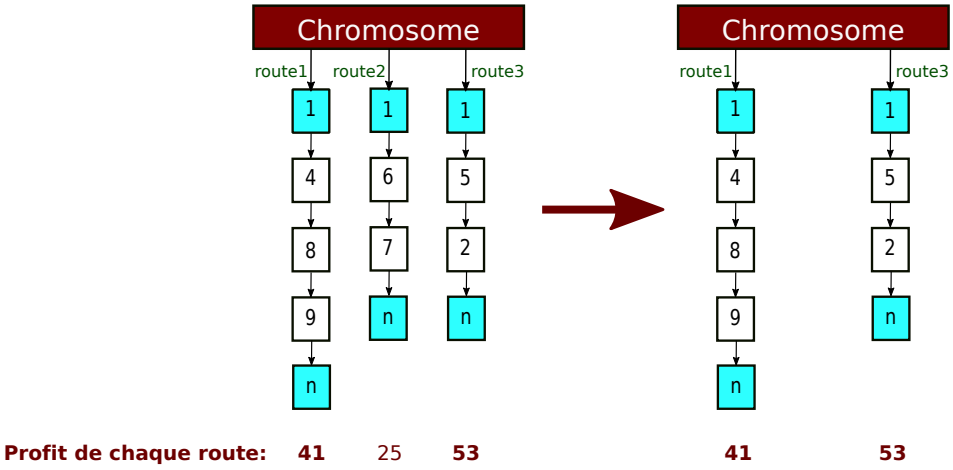


FIGURE 5.3 – Réparation de l'individu, avec un nombre de véhicules maximale de 2

5.4 Heuristiques d'amélioration

5.4.1 Opérateurs de recherche locale

La recherche locale est une technique utilisée pour améliorer la qualité des solutions en effectuant des changements locaux. Dans ce chapitre, nous considérons 4 opérateurs de recherche locale. Ces derniers ont déjà été décrits dans le chapitre précédent (voir section 4.3.3). Nous nous contentons donc de les citer : Un-point, Deux-point, 2-opt, meilleure insertion. La façon d'appliquer chacun des opérateurs de recherche locale est aussi décrite dans la section 4.3.3.

La solution résultante de l'utilisation de l'un de ces mouvements n'est conservée que si elle améliore la qualité de la solution en place et respecte la contrainte de temps $Tmax$.

5.4.2 Stratégie d'amélioration du profit

Comme les opérateurs de recherche locale se concentrent principalement sur l'amélioration de l'objectif temps de trajet, nous avons adopté une nouvelle stratégie afin d'améliorer le second objectif qui est le profit. Elle est introduite après la recherche locale et elle consiste à ajouter les clients non visités. Les principales étapes de la stratégie d'amélioration du profit peuvent être résumées dans les points suivants :

1. Déterminer les villes non visitées et les mettre dans une liste ;
2. Trier la liste selon le score de chaque ville ;
3. Ajouter la ville qui se trouve en tête de liste dans la première position trouvée ;
4. La troisième étape est répétée tant que la contrainte sur T_{max} n'est pas violée.

5.5 Résultats expérimentaux

5.5.1 Instances et paramétrage utilisés

La performance de l'algorithme HMOEA proposé est évaluée sur les instances extraites de Chao *et al.* [Chao 1996]. La série contient sept groupes et composée de 387 petites et grandes instances, où le nombre de véhicules varie de 2 à 4. L'algorithme proposé a été implémenté en Langage C et exécuté sur un Intel (R) Xeon (R) avec 2,60 Ghz et 4 Go de RAM.

Concernant le problème du MO-RTOP, comme ce problème n'a pas encore été traité dans la littérature, il n'existe pas d'instances prenant en considération les inceritudes. Par conséquent, en s'inspirant des instances de Chao *et al.*'s [Chao 1996] pour le TOP, nous avons créé de nouvelles instances avec des incertitudes. Les valeurs incertaines du coût de trajet, entre deux emplacements (villes) i et j, coûtent C_{ij}^s pour chaque scénario $s \in S$. Les valeurs de C_{ij}^s ont été générées en utilisant la distance euclidienne d_{ij}, calculée en utilisant la formule : $C_{ij}^s = d_{ij} - (1-\beta) \times (d_{ij} \times \delta)$ où β est un nombre aléatoire dans $[0, 1]$ et l'écart type δ est fixé à 40% pour l'ensemble des expériences.

En raison de l'aspect aléatoire de la méthode proposée, nous avons effectué 30 essais indépendants pour chaque instance considérée. Pour toutes les instances traitées, la taille de la population était fixée à 1000 et tous les essais sont arrêtés après 1000 générations. L'opérateur de croisement a été appliqué avec un taux égal à 100% et le taux d'opérateur de mutation est égal à 40%. En outre, les probabilités d'application des opérateurs : swaps partiels, fusion des routes les plus courtes et l'opérateur ordre aléatoire, étaient de 50% , 50% et 30% respectivement. Enfin, la recherche locale incorporée à la méthode est appliquée toutes les 50 générations.

Le graphique de la figure 5.4 représente le front de Pareto obtenu par la méthode proposée, qui illustre les solutions optimales pour l'instance $p4.2.m$ (extraite de Chao *et al.* [Chao 1996]). À partir de ce front de Pareto, on peut observer que l'augmentation du profit collecté induit la réduction du coût total du trajet. Un tel phénomène confirme le comportement conflictuel des deux objectifs considérés, en particulier pour le problème de TOP. De façon plus générale, ces solutions Pareto-optimales offrent au décideurs un panel de choix caractérisé par un compromis entre le profit et le coût.

5.5.2 Comparaison du MO-TOP avec d'autres approches de l'état de l'art

Afin d'évaluer la performance de la méthode proposée (notée HMOEA), nous comparons les résultats obtenus (sur les instances de TOP) avec ceux obtenus par les meilleures méthodes disponibles dans la littérature (Dang *et al.* [Dang 2013b]) : un algorithme mémétique (MA)[Bouly 2010], un algorithme d'optimisation par essaims particulaires (PSOMA) [Dang 2011] et un algorithme inspiré par l'optimisation par essaims particulaires (PSOIA) [Dang 2013b].

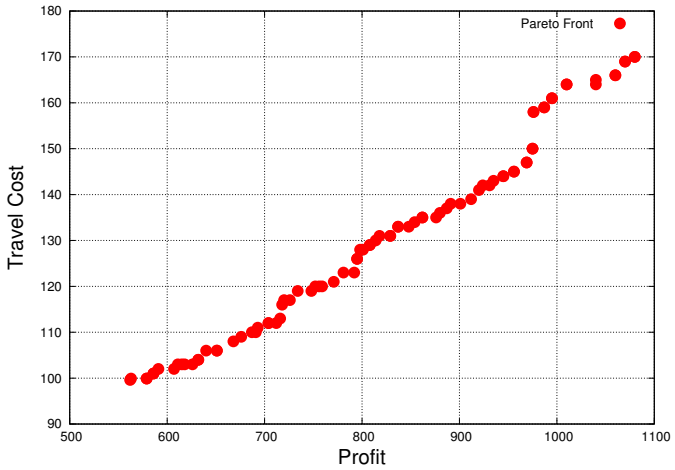


FIGURE 5.4 – Front de Pareto pour l'instance $p.4.2.m$

Les mêmes mesures que celles utilisées dans la majorité des travaux testant les instances de Chao sont utilisées pour tester les performances de notre méthode par rapport aux méthodes de la littérature comparées. Les meilleurs profits collectés entre toutes ces méthodes sont utilisés pour évaluer le pourcentage d'erreurs relatives (RPE). Ce profit est noté P_{best}. Le meilleur profit collecté, pour chaque instance (parmi les différents testes effectués) en utilisant la méthode proposée est noté P_{max}. RPE est calculé comme suit :

$$RPE = ((P_{best} - P_{max})/P_{best}) * 100$$

Plus la valeur RPE est faible pour une certaine instance, plus la capacité de la méthode d'atteindre le meilleur profit est grande. Quand HMOEA et la méthode de la littérature trouvent les mêmes solutions, RPE est égale à 0. Quand HMOEA est meilleure, RPE est strictement négative. Elle est strictement positive dans le cas contraire.

Le bénéfice moyen collecté dénoté par P_{avg} est également utilisé pour calculer l'erreur relative moyenne en pourcentage (APRE). Cela évalue plus tard la stabilité de la méthode par rapport à P_{best}. ARPE est calculé en utilisant la formule suivante :

$$ARPE = ((P_{best} - P_{avg})/P_{best}) * 100$$

Plus la valeur ARPE est petite pour une certaine instance, plus la méthode est stable sur les différents essais indépendants.

Après avoir calculé RPE et ARPE pour chaque instance, la moyenne de ces deux mesures est calculée pour chaque groupe parmi les sept groupes d'instances en fonction des RPE/ARPE de ses instances. Par la suite, nous avons comparé notre méthode avec les méthodes de la littérature à savoir : MA, PSOMA et PSOIA.

Le Tableau 5.1 montre la RPE pour chaque groupe de 1 à 7. La RPE est égale à 0 pour l'ensemble 2, ce qui signifie que HMOEA effectue la même performance en

Méthode	Année	RPE moyen pour chaque groupe d'instances						
		1	2	3	4	5	6	7
MA	2011	-0.067204	0.0	-0.083333	0.026158	-0.037193	-0.168919	-0.01607
PSOMA	2011	-0.067204	0.0	-0.083333	0.034441	-0.009178	-0.168919	-0.022303
PSOIA	2012	-0.067204	0.0	-0.083333	0.060184	0.0	-0.168919	-0.006623
Meilleur	2017	-0.067204	0.0	-0.083333	0.06295	0.0	-0.168919	-0.006623

TABLE 5.1 – Comparaison des performances en fonction de RPE pour chaque groupe des instances pertinentes.

Method	Year	ARPE moyen pour chaque groupe d'instances						
		1	2	3	4	5	6	7
MA	2011	-0.067204	0.0	-0.086158	0.004910	-0.046283	-0.162774	-0.060427
PSOMA	2011	-0.067204	0.0	-0.083333	-0.085807	-0.043305	-0.152226	-0.114805
PSOIA	2012	-0.067204	0.0	-0.083333	0.100539	-0.008759	-0.152226	-0.004728
Meilleur	2017	-0.067204	0.0	-0.083333	0.229180	0.011689	-0.152226	0.017871

TABLE 5.2 – Comparaison de robustesse en fonction de ARPE pour chaque groupe des instances pertinentes.

termes de profit en comparaison avec les trois méthodes de la littérature. Cependant, l'approche proposée surpasse les méthodes comparées pour les groupes 1, 3, 6 et 7, car le RPE est de valeur négative. Ainsi, HMOEA atteint de meilleurs profits pour certaines instances testées. A partir du RPE calculé pour le groupe 5, on remarque que l'approche multi-objectif proposée surpasse les approches mono-objectifs MA et PSOMA, mais elle est de performance égale avec celle de PSOIA. Sur le groupe 4 l'approche HMOEA a été surpassée par les autres approches.

L'ARPE est également calculée pour tous les groupes de données par rapport aux méthodes considérées et est indiquée dans le Tableau 5.2. Ce dernier se comporte généralement de la même manière que RPE. Ceci peut être expliqué par la faible différence entre les meilleurs profits atteints et les profits moyens (entre les essais). La performance de HMOEA est plus efficace sur les ensembles 1, 3, 5, 6 et 7 par rapport à toutes les méthodes de la littérature et se comporte également bien sur l'ensemble 2. Cependant, les méthodes MA et PSOIA sont plus efficaces, en moyenne, sur le groupe 4.

Nous avons comparé les méthodes en fonction des temps de calcul pour chaque groupe. Le tableau 5.3 illustre le temps maximal effectué entre les instances de chaque groupe et le tableau 5.4 illustre le temps de calcul moyen de toutes les instances d'un groupe. D'après le tableau 5.3, HMOEA est plus rapide dans quatre groupes sur sept à savoir : 1,3,6,7. PSOMA effectue les meilleurs temps CPU pour les groupes 2 et 5, juste avant HMOEA. Par conséquent, MA et PSOIA sont les méthodes qui prennent plus de temps. Concernant le temps de calcul moyen, l'ordre des méthodes de la plus rapide à la plus lente est PSOMA, HMOEA, MA, PSOIA pour les groupes de 1 à 4. Par contre, HMOEA arrive en tête du classement pour

Method	Year	Le temps CPU maximal pour chaque groupe						
		1	2	3	4	5	6	7
MA	2011	5.36	0.83	4.73	374.27	96.04	72.54	280.06
PSOMA	2011	3.09	0.03	3.61	201.95	59.14	34.75	157.24
PSOIA	2012	8.33	1.65	8.16	588.22	124.68	87.02	277.18
HMOEA	2017	1.179	0.2248	3.02	369.316	63.645	28.869	124.903

TABLE 5.3 – Comparaison de robustesse en fonction du temps CPU maximal pour chaque ensemble d'instances pertinentes.

Method	Year	Le temps CPU moyen pour chaque groupe						
		1	2	3	4	5	6	7
MA	2011	1.954	0.2348	2.06	193.06	35.334	39.0708	112.7546
PSOMA	2011	0.177	0.01	0.4938	87.331	14.719	7.59	49.088793
PSOIA	2012	2.149	0.409	3.181	240.4056	49.4958	47.08	97.47
HMOEA	2017	0.318	0.10479	0.72	118.3028	10.28	6.19	19.28

TABLE 5.4 – Comparaison de robustesse en fonction du temps CPU moyen pour chaque ensemble d'instances pertinentes.

les groupes de 5 à 7. En général, les temps de calcul les plus élevés sont notés pour les groupes 4 et 7.

Comme l'approche proposée est basée sur une optimisation multi-objectif, en plus de l'évaluation du profit, le coût du trajet total doit, également, être analysé et comparé à celui trouvé avec les approches basées sur l'optimisation mono-objectif de la littérature. Pour cela, le PSOIA est comparée car elle est la meilleure méta-heuristique par rapport aux deux autres méthodes (GA et PSOMA).

La comparaison du deuxième objectif considéré entre PSOIA et HMOEA est montrée dans le tableau 5.5. Cette comparaison est faite en fonction de l'écart moyen entre les coûts de trajet D_{HMOEA} et D_{PSOIA} atteints par les deux méthodes. Ce dernier est calculé pour chaque instance de chaque ensemble suivant cette formule : $D^i_{PSOIA} - D^i_{HMOEA}$. Lorsque le coût de trajet trouvé par PSOIA est plus élevé (moins bon) que celui trouvé par HMOEA, l'écart est positif, sinon, quand PSOIA surpasse HMOEA, cet écart est négatif. Après avoir calculé l'écart pour chaque instance, l'écart moyen est calculé pour chaque groupe. Ces moyennes sont représentées dans le graphique 5.5. Le taux d'amélioration en nombre d'instances est, aussi, calculé et illustré dans la deuxième ligne du même tableau ainsi que sous forme d'histogramme dans la figure 5.5. Nous remarquons que l'écart moyen est positif pour les groupes de 1 à 6 et négatif pour le groupe 7. Le meilleur écart est atteint pour l'ensemble 2 malgré que les deux méthodes fournissent la même performance en terme de profit. Les plus faibles améliorations sont obtenues pour les ensembles 5 et 6. Donc, les amélioration peuvent être lié à la complexité du problème. Généralement, le deuxième objectif n'est pas moins considéré par la méthode proposée, car le coût du trajet est amélioré pour la majorité des groupes. Ainsi, HMOEA garantit

	Les groupes						
	1	2	3	4	5	6	7
Gap moyen	0.11895	0.5986	0.2416	0.0674	0.058	0.108	-0.0193
Améliorations (%)	33.33	63.636	30	29.7297	24.324	12.5	6.8966

TABLE 5.5 – Comparaison de robustesse en fonction du coût de trajet pour chaque groupe d'instances pertinentes.

un meilleur compromis entre le profit récolté et le coût du trajet en leur accordant la même importance.

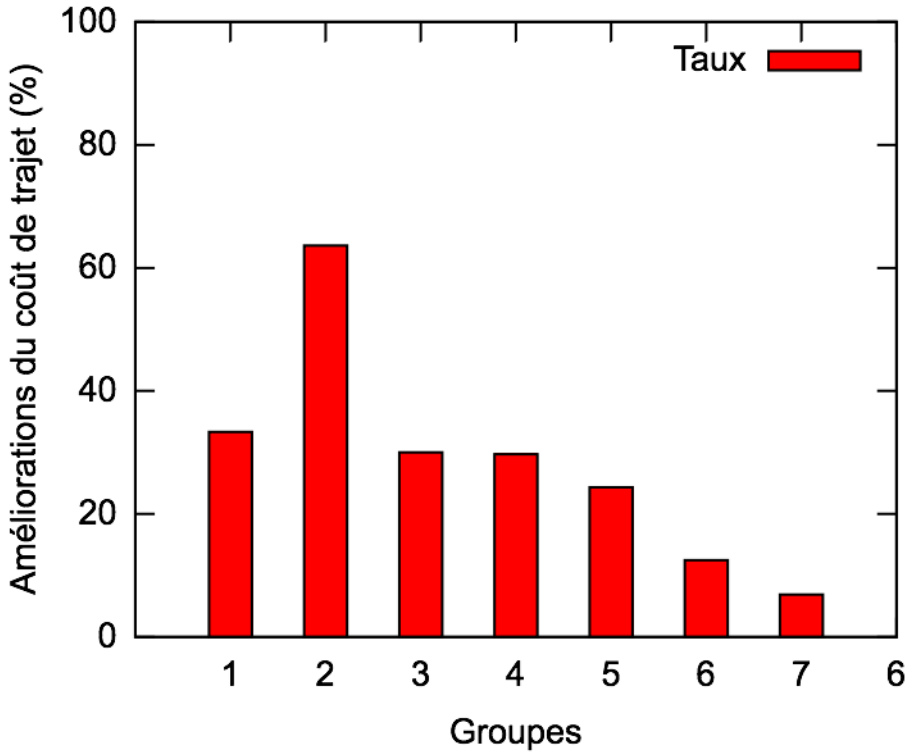


FIGURE 5.5 – Le taux d'amélioration des coûts de trajet par NSGAII pour chaque groupe

Les Tableaux 5.6 à 5.14 contiennent les résultats détaillés obtenus avec les trente essais et ceux extraits de Dang *et al.* [Dang 2013b]. La colonne 1 représente l'étiquette de chaque instance testée. La colonne 2 affiche le meilleur profit trouvé sur tous les essais et la colonne 3 montre le coût de trajet correspondant. Les colonnes 4 et 5 montrent le temps de calcul effectué pour atteindre la solution finale et le temps de calcul moyen entre tous les essais respectivement. La colonne 6 montre le meilleur profit fourni par l'heuristique mono-objectif la plus efficace pour le TOP (PSOIA) avec le coût de trajet correspondant affiché dans la colonne 7. Enfin, la colonne 8 indique le temps de calcul moyen pris pour résoudre l'instance.

Selon ces résultats, quatre cas ont été améliorés en termes de profit. De plus, on peut remarquer que HMOEA est capable d'améliorer le coût du trajet sur plusieurs instances.

5.5.3 Expérimentations réalisées avec MO-RTOP

Pour le RTOP, nous avons également considéré les solutions de CPLEX pour les instances testées en fixant la limite d'exécution à chaque fois supérieure au temps d'exécution consommé par HMOEA. Le Tableau 5.15 rapporte les résultats obtenus par CPLEX et HMOEA. La première colonne représente l'étiquette de l'instance. Les deuxième et troisième colonnes indiquent respectivement le nombre d'emplacements et le nombre de scénarios considéré. La colonne (4) illustre ensuite le bénéfice P et le coût de trajet D trouvés par CPLEX dans le temps limite indiqué dans l'autre colonne. De plus, la colonne (5) illustre le meilleur bénéfice P et le meilleur coût de trajet D correspondant à une solution atteinte par HMOEA. La dernière colonne présente le temps de calcul effectué par HMOEA pour trouver la solution.

A partir du Tableau 5.15, on peut observer que HMOEA a donné de meilleurs résultats que CPLEX en termes de profit ou de temps de déplacement dans la majorité des cas. En outre, comme HMOEA est une approche multi-objectif, elle trouve plusieurs solutions (avec les essais considérés) qui offrent différentes options au décideur. Par exemple, HMOEA atteint souvent un meilleur profit, et atteint même parfois le même bénéfice avec un meilleur temps de déplacement. HMOEA atteint un meilleur temps de déplacement pour les instances p3.2.g, p3.2.g, p3.2.m avec 20 scénarios, p3.3.e, p3.4.i, p3.4.i, et atteint un meilleur temps de déplacement avec un meilleur bénéfice pour l'instance p3.2.m avec 10 scénarios. Le seul cas où CPLEX surpasse HMOEA est celui de l'instance p5 .2.q. On observe aussi l'augmentation du temps de calcul avec l'augmentation du nombre de scénarios.

5.6 Conclusion

Dans ce chapitre, nous avons proposé un algorithme évolutionnaire hybride multi-objectif pour résoudre approximativement les problèmes d'orientation (ou problème de tournées de véhicules sélectives multi-objectif MO-TOP). Deux vairantes ont été considérées à savoir : le TOP déterministe (MO-TOP) et le TOP robuste avec un temps de trajet incertains (MO-RTOP). L'objectif de la méthode proposée est d'optimiser simultanément le profit total et le coût du trajet. Pour le cas robuste, le pire cas du coût de trajet incertain est considéré en plus du profit. Certaines stratégies de recherche locale ont été intégrées dans l'algorithme afin d'améliorer ses performances, en particulier pour fournir des solutions de meilleure qualité en terme de coût de trajet. Une deuxième stratégie d'insertion de clients non visités a été intégrée pour améliorer le profit et compenser la recherche locale. La première partie des expérimentations sur le TOP déterministe a démontré le comportement conflictuel des deux objectifs et a montré que la méthode est capable d'offrir au décideur une large gamme de solutions Pareto-optimales. Les résultats obtenus sur le TOP montrent également que la méthode proposée est compétitive en atteignant la majorité des meilleures bornes disponibles dans la littérature et en produisant de nouvelles bornes pour plusieurs instances. De plus, la comparaison avec des méthodes de l'état de l'art, a permis de montrer que notre méthode surpasse les autres

méthodes comparées pour la majorité des groupes. La deuxième partie des expérimentations s'est focalisée sur le RTOP. Après avoir généré une série d'instances pour ce problème, la méthode HMOEA a été comparée avec CPLEX. La comparaison a permis d'établir que HMOEA était compétitive par rapport à CPLEX sur plusieurs instances en terme de profit total et de temps de trajet en même temps.

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p1.2.b	15	8.81527	0.023	0.0238	15	8.81527	0.0
p1.2.c	20	13.64219	0.03	0.0304	20	13.64219	0.0
p1.2.d	30	16.62751	0.04	0.045	30	16.62751	0.01
p1.2.e	45	22.4968	0.05	0.0529	45	22.4968	0.43
p1.2.f	80	29.078	0.086	0.0869	80	29.078	1.1
p1.2.g	90	33.57499	0.118	0.118	90	33.7308	1.47
p1.2.h	110	38.03516	0.16	0.169	110	38.0943	3.29
p1.2.i	135	45.3866	0.21	0.216	135	45.3866	1.77
p1.2.j	155	48.79547	0.277	0.28	155	49.1482	2.87
p1.2.k	175	54.18672	0.327	0.335	175	54.1868	4.33
p1.2.l	195	59.3987	0.419	0.442	195	59.7238	2.96
p1.2.m	215	63.6638	0.536	0.54	215	63.6638	6.9
p1.2.n	235	69.24734	0.637	0.64	235	69.2474	5.95
p1.2.o	240	70.4509	0.727	0.73	240	70.4509	5.21
p1.2.p	250	74.04173	0.779	0.78	250	74.0418	6.51
p1.2.q	265	78.4768	0.89	0.903	265	78.4768	7.87
p1.2.r	280	84.649300	1.047	1.053	280	84.6493	8.33
p1.3.c	15	8.81527	0.023	0.024	15	8.81527	0.0
p1.3.d	15	8.81527	0.023	0.024	15	8.81527	0.0
p1.3.e	30	15.00843	0.047	0.0477	30	15.00843	0.0
p1.3.f	40	26.52455	0.068	0.0708	40	26.52455	0.02
p1.3.g	50	31.5707	0.075	0.076	50	32.8137	0.29
p1.3.h	70	36.83009	0.098	0.099	70	36.8809	0.74
p1.3.i	105	44.26063	0.151	0.147	105	44.2607	1.82
p1.3.j	115	48.88913	0.178	0.182	115	48.8892	1.79
p1.3.k	135	53.42448	0.238	0.24	135	54.0137	1.51
p1.3.l	155	58.4112	0.303	0.468	155	58.4112	2.09
p1.3.m	175	62.9288	0.34	0.354	175	62.9288	2.74
p1.3.n	190	66.23986	0.45	0.447	190	66.8366	2.64
p1.3.o	205	69.3588	0.516	0.521	205	69.3588	2.53
p1.3.p	220	73.128	0.558	0.56	220	73.128	4.54
p1.3.q	230	76.96987	0.626	0.66	230	76.96987	3.93
p1.3.r	250	82.23907	0.786	0.797	250	82.3358	4.03
p1.4.d	15	8.81527	0.0247	0.025	15	8.81527	0.0
p1.4.e	15	8.81527	0.0239	0.024	15	8.81527	0.0
p1.4.f	25	20.87147	0.046	0.047	25	20.871470	0.0
p1.4.g	35	29.01338	0.0596	0.06	35	30.606300	0.0
p1.4.h	45	34.27967	0.089	0.09	45	34.27967	0.02
p1.4.i	60	42.75491	0.1035	0.104	60	42.75494	0.22
p1.4.j	75	46.9333	0.125	0.126	75	47.0163	0.39
p1.4.k	100	51.845	0.1659	0.166	100	51.8552	0.98
p1.4.l	120	56.1558	0.222	0.227	120	56.564	1.07
p1.4.m	130	59.68608	0.245	0.25	130	60.0113	2.01
p1.4.n	160	67.27509	0.319	0.326	155	66.3585	2.37
p1.4.o	165	70.26392	0.38	0.38	165	70.6996	1.67
p1.4.p	175	71.91069	0.419	0.428	175	72.212200	1.34
p1.4.q	190	76.0082	1.133	1.179	190	76.0082	2.65
p1.4.r	210	81.6914	0.537	0.64	210	81.6914	2.79

TABLE 5.6 – Résultats pour le groupe 1 du benchmark Chao

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p2.2.a	90	13.88921	0.04848	0.048992	90	13.88921	0.07
p2.2.b	120	17.12468	0.080145	0.080797	120	17.72381	0.24
p2.2.c	140	20.77461	0.082621	0.083118	140	21.4497	0.48
p2.2.d	160	23.67856	0.105484	0.106142	160	23.6867	0.78
p2.2.e	190	26.459	0.121474	0.122134	190	26.459	0.65
p2.2.f	200	26.54121	0.161570	0.162431	200	27.9984	0.81
p2.2.g	200	26.54121	0.165211	0.166211	200	27.9984	0.96
p2.2.h	230	32.30188	0.190224	0.194181	230	33.2712	0.99
p2.2.i	230	30.79003	0.206107	0.207540	230	32.7948	0.95
p2.2.j	260	38.6921	0.211219	0.212248	260	38.6921	1.09
p2.2.k	275	43.9554	0.223135	0.224851	275	43.9554	1.65
p2.3.a	70	13.31976	0.040886	0.041129	70	13.31976	0.0
p2.3.b	70	10.11585	0.043199	0.043542	70	10.11585	0.01
p2.3.c	105	19.16391	0.06688	0.067397	105	19.99991	0.06
p2.3.d	105	16.20039	0.068823	0.06931	105	16.2829	0.09
p2.3.e	120	19.3271	0.081995	0.082909	120	20.77509	0.15
p2.3.f	120	17.12468	0.084494	0.085394	120	17.72381	0.16
p2.3.g	145	30.75488	0.108277	0.107827	145	31.22777	0.32
p2.3.h	165	33.93007	0.119992	0.120796	165	33.9307	0.46
p2.3.i	200	36.12769	0.14722	0.148971	200	36.181	0.81
p2.3.j	200	30.61974	0.161162	0.161922	200	36.2229	0.81
p2.3.k	200	26.54121	0.181852	0.185516	200	27.0876	0.92
p2.4.a	10	2.67915	0.013065	0.013199	10	2.67915	0.0
p2.4.b	70	13.31976	0.044268	0.044618	70	13.31976	0.0
p2.4.c	70	10.96165	0.044290	0.044661	70	10.96165	0.0
p2.4.d	70	10.11585	0.043526	0.043999	70	10.11585	0.0
p2.4.e	70	10.11585	0.043686	0.044018	70	10.11585	0.01
p2.4.f	105	19.22241	0.072018	0.072525	105	20.35371	0.05
p2.4.g	105	19.16391	0.072273	0.072933	105	19.22634	0.05
p2.4.h	120	19.3271	0.086199	0.086904	120	20.77509	0.17
p2.4.i	120	17.36086	0.086631	0.087951	120	17.36086	0.17
p2.4.j	120	17.12468	0.086223	0.086938	120	17.42343	0.19
p2.4.k	180	43.02006	0.136089	0.137141	180	43.0208	0.42

TABLE 5.7 – Résultats pour le groupe 2 du benchmark Chao

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p3.2.a	90	13.11367	0.053578	0.054105	90	14.42356	0.02
p3.2.b	150	18.37674	0.088201	0.08882	150	18.37674	0.51
p3.2.c	180	20.989	0.117265	0.117874	180	20.989	1.44
p3.2.d	220	28.687	0.171088	0.172057	220	28.687	1.03
p3.2.e	260	34.1088	0.203524	0.207892	260	34.1088	1.84
p3.2.f	300	39.4855	0.228442	0.232791	300	39.4855	3.66
p3.2.g	360	44.3297	0.268965	0.273292	360	44.3297	4.5
p3.2.h	410	49.6889	0.334657	0.342909	410	49.688900	2.97
p3.2.i	460	53.85861	0.430647	0.844621	460	54.599	6.9
p3.2.j	510	59.224740	1.136561	1.166545	510	59.2248	6.16
p3.2.k	550	62.94801	0.584801	0.585594	550	63.0544	6.02
p3.2.l	590	69.1343	1.532791	1.580153	590	69.1343	7.51
p3.2.m	620	72.4763	0.740157	0.865969	620	72.4763	6.89
p3.2.n	660	78.133	2.058948	2.131352	660	78.133	7.9
p3.2.o	690	83.5681	0.899454	0.920091	690	83.5681	6.32
p3.2.p	720	88.0713	2.249619	2.354967	720	88.0713	5.94
p3.2.q	760	94.28964	1.183619	1.196015	760	94.2897	7.14
p3.2.r	790	98.9919	3.007743	1.54582	790	98.9919	8.16
p3.2.s	800	100.8873	1.437263	1.470566	800	100.8873	7.22
p3.2.t	800	100.328	1.563057	1.598637	800	101.1356	4.54
p3.3.a	30	4.71525	0.034136	0.034538	30	4.71525	0.0
p3.3.b	90	15.77221	0.06161	0.061718	90	15.77221	0.0
p3.3.c	120	19.55063	0.075512	0.075693	120	19.55063	0.07
p3.3.d	170	27.45916	0.123019	0.123304	170	27.47993	0.43
p3.3.e	210	32.69571	0.166043	0.167263	200	32.1021	0.75
p3.3.f	230	36.87557	0.187702	0.189233	230	36.87557	0.9
p3.3.g	270	43.19026	0.252231	0.256159	270	43.19027	1.07
p3.3.h	300	45.16222	0.300861	0.303058	300	45.70092	1.32
p3.3.i	330	48.44124	0.32074	0.321935	330	48.44124	1.89
p3.3.j	380	56.19053	0.384533	0.39296	380	56.19053	2.62
p3.3.k	440	62.7226	0.44932	0.460391	440	62.7226	3.4
p3.3.l	480	66.74023	0.520743	0.528670	480	67.0864	4.15
p3.3.m	520	73.0405	1.392297	1.598258	520	73.0405	4.13
p3.3.n	570	77.43	0.685431	0.686292	570	77.43	6.48
p3.3.o	590	81.5639	1.720281	0.805575	590	81.7913	5.42
p3.3.p	640	87.06405	1.901319	1.925194	640	87.7364	4.58
p3.3.q	680	93.25534	2.178666	2.778275	680	93.3276	3.46
p3.3.r	710	94.67892	1.181329	1.190045	710	97.3364	4.83
p3.3.s	720	100.9528	1.209510	1.217444	720	100.9528	4.79
p3.3.t	760	107.9744	2.780575	3.020999	760	107.9744	6.91

TABLE 5.8 – Résultats pour le groupe 3 du benchmark Chao

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p3.4.a	20	3.49653	0.024427	0.025201	20	3.49653	0.0
p3.4.b	30	4.71525	0.035909	0.036181	30	4.71525	0.0
p3.4.c	90	15.77221	0.063646	0.064507	90	15.77221	0.0
p3.4.d	100	20.39403	0.075485	0.076641	100	20.39403	0.01
p3.4.e	140	28.24269	0.101008	0.102745	140	28.24269	0.13
p3.4.f	190	36.56235	0.156675	0.157221	190	36.56235	0.52
p3.4.g	220	40.91423	0.205095	0.205651	220	40.91432	0.83
p3.4.h	240	45.06544	0.2301	0.230651	240	45.0655	0.74
p3.4.i	270	52.86067	0.268447	0.271236	270	52.8607	1.69
p3.4.j	310	57.8503	0.346761	0.353937	310	57.8503	0.49
p3.4.k	350	59.98566	0.404623	0.410884	350	61.4219	1.94
p3.4.l	380	64.56744	0.453319	0.454999	380	64.5675	2.07
p3.4.m	390	68.19041	0.468364	0.472559	390	68.6207	1.15
p3.4.n	440	73.96912	0.568896	0.581802	440	75.9417	1.61
p3.4.o	500	82.35863	0.637749	0.656115	500	82.3587	3.7
p3.4.p	560	87.11066	0.734249	0.745504	560	87.7785	5.23
p3.4.q	560	85.97428	0.837911	0.860526	560	87.598	1.78
p3.4.r	600	93.1031	1.860524	1.385827	600	93.1031	4.39
p3.4.s	670	102.9797	1.052982	1.054038	670	102.9798	7.81
p3.4.t	670	99.29247	1.175323	1.177284	670	100.7541	2.91

TABLE 5.9 – Résultats pour le groupe 3 du benchmark Chao (suite)

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p4.2.a	206	49.6252	0.463864	0.478349	206	49.6252	5.88
p4.2.b	341	59.3783	1.11936	1.213705	341	59.3783	39.21
p4.2.c	452	69.7586	1.825759	2.302475	452	69.7586	67.12
p4.2.d	529	79.57429	21.486837	28.571930	531	79.3943	124.29
p4.2.e	618	89.3428	8.58298	10.03488	618	89.3428	197.94
p4.2.f	686	99.50263	50.783669	46.674264	687	99.7844	322.64
p4.2.g	755	109.4005	62.68808	69.001786	757	109.993	206.54
p4.2.h	828	119.7387	73.459785	85.438925	835	119.626	257.24
p4.2.m	1132	169.942500	45.53434	88.387565	1132	169.9425	321.08
p4.2.n	1174	178.5981	68.764954	136.857698	1174	178.5982	427.5
p4.2.o	1218	189.7543	152.581055	274.557427	1218	189.7544	415.43
p4.2.p	1242	199.359	237.172958	287.384582	1242	199.359	347.47
p4.2.q	1267	209.694	325.413086	296.938012	1268	209.799	588.22
p4.2.r	1292	219.659	319.8013	369.316745	1292	219.659	470.01
p4.2.s	1304	229.255	117.653419	332.471746	1304	229.255	486.19
p4.2.t	1306	230.517	223.28894	213.573276	1306	230.517	408.65
p4.3.e	468	89.07742	2.128216	5.468217	468	89.6263	36.41
p4.3.f	579	99.5188	3.526328	9.259082	579	99.5189	72.88
p4.3.g	653	109.3394	4.360299	5.587141	653	109.3395	70.44
p4.3.h	728	119.2186	47.028309	46.845958	729	119.6306	194.18
p4.3.i	809	129.6855	18.341478	49.510294	809	129.6855	247.26
p4.3.j	861	139.6934	26.407854	53.562586	861	139.6934	229.11
p4.3.k	919	149.1174	51.448936	38.211347	919	149.1174	275.31
p4.3.q	1253	209.6688	210.92337	239.196114	1253	209.6688	448.69
p4.3.r	1271	218.3586	290.02301	284.984468	1273	219.286000	288.72
p4.3.s	1295	228.1291	114.555885	314.613249	1295	228.1291	278.0
p4.3.t	1305	238.6624	307.724731	321.915184	1305	238.6624	305.85
p4.4.i	657	129.3443	13.746482	27.174536	657	129.3443	65.48
p4.4.j	732	138.0991	19.531019	33.937493	732	138.0991	81.35
p4.4.k	821	149.5589	26.813089	50.383256	821	149.5589	119.45
p4.4.l	879	159.1262	72.996002	79.215366	880	159.4477	101.6
p4.4.m	917	169.293	92.61895	92.017899	919	168.8906	223.19
p4.4.n	977	178.7289	131.081772	124.873923	977	178.7289	257.14
p4.4.o	1061	189.5323	37.893475	47.992948	1061	189.5323	208.36
p4.4.p	1124	199.0781	50.497623	85.485291	1124	199.0781	193.14
p4.4.q	1161	209.0136	109.00779	104.399884	1161	209.013600	252.98
p4.4.r	1216	218.7492	99.093781	119.368077	1216	218.7492	260.06

TABLE 5.10 – Résultats pour le groupe 4 du benchmark Chao

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p5.2.b	20	8.47214	0.112626	0.113839	20	8.47214	0.0
p5.2.c	50	14.12899	0.119944	0.120701	50	14.12899	0.07
p5.2.d	80	16.47214	0.185328	0.192354	80	16.47214	0.76
p5.2.e	180	24.47214	0.249480	0.251939	180	24.47214	4.9
p5.2.f	240	29.97569	0.319467	0.320777	240	29.97569	13.47
p5.2.g	320	32.47214	0.480224	0.493781	320	33.8416	36.47
p5.2.h	410	39.62123	0.599267	0.629474	410	39.62123	51.98
p5.2.i	480	44.9442	0.784915	0.824944	480	44.9442	71.12
p5.2.j	580	49.7502	0.998283	1.088651	580	49.7502	54.37
p5.2.k	670	54.129	1.30975	1.420333	670	54.129	60.78
p5.2.l	800	59.6212	1.607933	1.775201	800	59.6212	88.15
p5.2.m	860	64.47214	1.678488	2.037560	860	64.4722	90.5
p5.2.n	925	69.2239	2.093309	2.201936	925	69.2239	72.46
p5.2.o	1020	74.85263	2.714294	2.791368	1020	74.8527	65.93
p5.2.p	1150	79.6212	3.219487	3.280995	1150	79.6212	109.63
p5.2.q	1195	84.7082	3.318788	4.147733	1195	84.7082	116.62
p5.2.r	1260	88.47214	6.521168	13.872945	1260	88.4722	92.78
p5.2.s	1340	94.77032	17.575302	29.852249	1340	94.7704	85.78
p5.2.t	1400	99.6212	5.513312	13.127730	1400	99.6212	111.2
p5.2.u	1460	104.4721	22.285212	34.219752	1460	104.4722	110.39
p5.2.v	1505	108.7082	22.509615	41.477294	1505	108.7082	112.4
p5.2.w	1565	113.6212	25.967432	61.243182	1565	113.6213	124.13
p5.2.x	1610	119.6212	27.247011	33.407132	1610	119.6212	124.68
p5.2.y	1645	124.7082	28.999472	38.130964	1645	124.7082	112.34
p5.2.z	1680	128.4721	32.401222	51.713656	1680	128.4722	122.55
p5.3.b	15	8.76243	0.063836	0.068648	15	8.76243	0.0
p5.3.c	20	8.47214	0.101426	0.101707	20	8.47214	0.0
p5.3.d	60	19.41641	0.139984	0.144983	60	19.43964	0.03
p5.3.e	95	22.80477	0.211397	0.215829	95	22.80478	0.35
p5.3.f	110	24.80477	0.286435	0.287108	110	25.57559	0.53
p5.3.g	185	33.2901	0.312798	0.313960	185	33.2901	5.03
p5.3.h	260	36.80477	0.500964	0.515062	260	37.2269	5.11
p5.3.i	335	44.9431	0.567412	0.572134	335	44.9431	15.76
p5.3.j	470	48.80478	0.829840	0.850155	470	48.8909	24.37
p5.3.k	495	54.1401	0.981561	1.003135	495	54.1401	33.27
p5.3.l	595	58.78229	1.162903	1.817310	595	58.8684	53.91
p5.3.m	650	60.89084	1.993544	2.009626	650	61.2269	17.01
p5.3.n	755	69.2901	1.640455	1.809742	755	69.2901	48.68
p5.3.o	870	74.6253	2.153387	2.572295	870	74.6253	48.93
p5.3.p	990	79.4163	2.784479	3.168274	990	79.4163	106.58
p5.3.q	1070	84.32943	3.808869	4.204687	1070	84.3295	51.54
p5.3.r	1125	89.5141	3.618553	4.341126	1125	89.5141	46.7

TABLE 5.11 – Résultats pour le groupe 5 du benchmark Chao

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p5.3.s	1190	94.0162	11.124921	11.094935	1190	94.0163	59.48
p5.3.t	1260	96.8048	10.990310	12.835058	1260	96.8048	69.12
p5.3.u	1345	103.9862	13.449559	17.245195	1345	103.9862	57.97
p5.3.v	1425	109.2077	14.723844	21.360095	1425	109.2077	56.21
p5.3.w	1485	114.5655	46.145836	58.668375	1485	114.5656	76.2
p5.3.x	1555	118.7823	22.070265	44.018401	1555	118.7823	76.78
p5.3.y	1595	123.0064	27.952612	63.645418	1595	123.0065	70.54
p5.3.z	1635	128.0087	28.057005	39.627319	1635	128.0087	84.24
p5.4.c	20	11.68324	0.088520	0.092982	20	11.68324	0.0
p5.4.d	20	8.47214	0.111851	0.116873	20	8.47214	0.0
p5.4.e	20	8.47214	0.114011	0.118791	20	8.47214	0.0
p5.4.f	80	25.88854	0.222008	0.239544	80	25.91178	0.23
p5.4.g	140	33.13741	0.286752	0.306867	140	33.22349	0.99
p5.4.h	140	33.13741	0.37052	0.403018	140	33.13742	0.48
p5.4.i	240	44.45112	0.444176	0.453216	240	44.4512	2.32
p5.4.j	340	49.1374	0.613248	0.638772	340	49.1374	5.13
p5.4.k	340	49.1374	0.836008	0.871104	340	49.1374	1.82
p5.4.l	430	59.9106	0.876359	0.881097	430	59.9106	11.22
p5.4.m	555	64.2679	1.128479	1.207649	555	64.2679	20.76
p5.4.n	620	65.1374	1.489265	1.750096	620	65.1374	12.44
p5.4.o	690	73.50978	1.710827	1.780459	690	73.5098	42.91
p5.4.p	765	78.74616	4.927735	11.793891	765	78.8323	48.35
p5.4.q	860	81.1374	2.386095	2.723501	860	81.1374	61.58
p5.4.r	960	89.8884	2.710395	3.041734	960	89.8884	76.4
p5.4.s	1030	95.05977	9.164079	13.008218	1030	95.0598	30.82
p5.4.t	1160	99.5004	3.832092	4.691872	1160	99.5004	47.63
p5.4.u	1300	104.1867	4.552927	5.121086	1300	104.1868	67.26
p5.4.v	1320	105.8885	5.116637	6.210471	1320	106.556	97.97
p5.4.w	1390	114.064	15.065608	16.351960	1390	114.064	40.5
p5.4.x	1450	118.8634	18.865892	29.755306	1450	118.8634	63.41
p5.4.y	1520	121.8885	14.685776	15.710628	1520	122.2224	102.12
p5.4.z	1620	129.1374	8.523828	12.355489	1620	129.1374	86.55

TABLE 5.12 – Résultats pour le groupe 5 du benchmark Chao (suite)

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p6.2.d	192	29.6568	0.263648	0.274138	192	29.6568	5.76
p6.2.e	360	34.62741	0.657654	0.680527	360	34.7433	41.91
p6.2.f	588	39.598	1.227758	1.300669	588	39.598	64.83
p6.2.g	660	43.59799	1.503719	1.950784	660	43.598	66.09
p6.2.h	780	49.2548	2.066651	2.228414	780	49.2548	82.46
p6.2.i	888	54.9116	2.320512	2.477249	888	54.9116	87.02
p6.2.j	948	59.856	2.909728	3.00044	948	59.856	54.1
p6.2.k	1032	62.2254	3.842620	4.310187	1032	62.2254	49.41
p6.2.l	1116	67.8822	14.526771	16.991614	1116	67.8822	66.31
p6.2.m	1188	73.53914	15.23856	17.627547	1188	73.5392	58.34
p6.2.n	1260	79.196	17.501884	20.03487	1260	79.196	62.26
p6.3.g	282	44.5578	0.500248	0.515931	282	44.5578	5.85
p6.3.h	462	49.89625	0.999974	1.127345	444	49.4559	12.34
p6.3.i	642	54.4263	1.817372	1.984370	642	54.4263	31.16
p6.3.j	828	59.4882	2.452394	2.658249	828	59.4882	75.2
p6.3.k	894	64.06469	19.310684	6.308957	894	64.6413	61.6
p6.3.l	1002	68.5499	12.656151	10.480966	1002	68.5499	50.91
p6.3.m	1080	74.2254	4.250898	8.857152	1080	74.2254	56.64
p6.3.n	1170	77.53915	18.73402	28.869215	1170	79.8822	52.52
p6.4.j	366	59.5021	0.834165	0.88232	366	59.5021	5.23
p6.4.k	528	64.2208	1.379988	1.518798	528	64.2208	8.56
p6.4.l	696	68.751	2.642709	6.062391	696	68.751	27.37
p6.4.m	912	74.8744	3.304759	3.630714	912	74.8744	38.97
p6.4.n	1068	79.3784	4.302679	4.869004	1068	79.3784	65.1

TABLE 5.13 – Résultats pour le groupe 6 du benchmark Chao

Inst	P_{best}^{HMOEA}	D^{HMOEA}	CPU^{HMOEA}	CPU_{avg}^{HMOEA}	PSOIA	D^{PSOIA}	CPU_{avg}^{PSOIA}
p7.2.a	30	18.94427	0.067921	0.071768	30	18.94427	0.0
p7.2.b	64	36.6593	0.143163	0.147458	64	36.6593	0.0
p7.2.c	101	57.6852	0.223166	0.234515	101	57.6852	0.48
p7.2.d	190	78.6737	0.402485	0.434141	190	78.6737	4.36
p7.2.e	290	98.3631	1.962095	2.065331	290	98.3631	18.56
p7.2.f	387	119.4096	1.209718	1.212726	387	119.4096	40.48
p7.2.g	459	138.6004	1.824302	2.044384	459	138.6004	126.9
p7.2.h	521	158.2426	1.988263	2.033893	521	158.2426	188.66
p7.2.i	580	179.7669	7.865879	15.719058	580	179.7669	181.96
p7.2.j	646	199.9453	8.79159	9.797227	646	199.9453	134.9
p7.2.k	705	218.861	21.789476	26.769872	705	218.861	170.92
p7.2.l	767	237.613	11.805183	14.334448	767	237.613	210.89
p7.2.m	827	258.5923	14.570998	15.705872	827	258.593	177.68
p7.2.n	888	278.298	18.829735	41.435757	888	278.298	186.65
p7.2.o	945	297.669	19.531033	23.360537	945	297.669	208.93
p7.2.p	1002	318.314	22.068663	26.174366	1002	318.314	241.83
p7.2.q	1044	339.732	31.964184	81.160834	1044	339.732	181.26
p7.2.r	1094	358.424	54.688782	73.660024	1094	358.424	182.32
p7.2.s	1136	379.288	59.170967	101.423029	1136	379.288	228.1
p7.2.t	1179	398.308	71.725883	124.903681	1179	398.308	277.18
p7.3.b	46	31.59337	0.102304	0.102698	46	31.59338	0.0
p7.3.c	79	54.6593	0.217171	0.21801	79	54.6593	0.0
p7.3.d	117	76.0941	0.336565	0.337302	117	76.0941	0.18
p7.3.e	175	95.7573	0.430687	0.452283	175	95.7573	0.97
p7.3.f	247	117.6235	0.708656	0.721138	247	117.6235	3.84
p7.3.g	344	137.6793	1.081733	1.083179	344	137.6794	21.75
p7.3.h	425	155.3399	4.014586	4.184722	425	155.3399	27.88
p7.3.i	487	177.8798	5.316876	6.780594	487	179.0485	45.65
p7.3.j	564	198.0369	2.871576	5.365472	564	198.0369	54.98
p7.3.k	633	217.9935	3.120921	3.165544	633	217.9935	88.79
p7.3.l	684	235.4293	4.023306	8.347942	684	235.4294	100.72
p7.3.m	762	258.9108	11.01853	12.0035	762	258.9109	127.04
p7.3.n	820	277.6986	5.682399	11.423556	820	277.6986	175.64
p7.3.o	874	296.0697	14.981923	16.546106	874	296.0697	196.91
p7.3.p	929	319.191	36.409851	57.242781	929	319.191	162.61
p7.3.q	987	336.56	16.539906	19.677966	987	336.56	168.7
p7.3.r	1026	359.248	61.592854	65.397322	1026	359.248	203.02
p7.3.s	1081	376.339	25.095951	27.047493	1081	376.339	242.0
p7.3.t	1120	396.56	31.876381	81.068143	1120	396.56	151.73
p7.4.b	30	18.94427	0.076155	0.077821	30	18.94427	0.0
p7.4.c	46	31.59337	0.102328	0.105021	46	31.59337	0.0
p7.4.d	79	54.6593	0.234626	0.235499	79	54.659300	0.0
p7.4.e	123	88.8355	0.338977	0.339896	123	88.8355	0.1
p7.4.f	164	114.4281	0.477227	0.478597	164	114.4281	0.54
p7.4.g	217	131.8378	0.725311	0.727282	217	131.837800	1.72
p7.4.h	285	152.1288	0.966910	1.000444	285	152.1288	4.44
p7.4.i	366	176.4856	1.352264	1.402686	366	176.4857	12.68
p7.4.j	462	196.6349	2.086532	2.124165	462	196.6349	21.25
p7.4.k	520	216.0388	2.797775	8.305839	520	216.0388	39.94
p7.4.l	590	238.5599	7.835466	21.891327	590	238.5599	53.8
p7.4.m	646	252.5119	4.053904	4.166477	646	252.5119	100.05
p7.4.n	730	277.9552	11.220297	11.975724	730	277.9552	110.63
p7.4.o	784	297.804	13.718845	19.498943	781	295.5152	93.27
p7.4.p	846	314.0214	16.409977	26.486556	846	314.0214	124.32
p7.4.q	909	335.2712	34.900154	47.568832	909	335.2712	134.35
p7.4.r	970	355.8278	8.445887	10.045565	970	355.8278	143.04
p7.4.s	1022	376.4271	20.829704	26.744507	1022	376.4271	150.72
p7.4.t	1077	397.9319	23.992	51.351520	1077	397.9319	128.13

TABLE 5.14 – Résultats pour le groupe 7 du benchmark Chao

			CPLEX		CPU	$HMOEA^{worst}$		CPU
Instance	Nc	Snr	P	D		P	D	
p3.2.g	33	10	410	43.72	214	420/410	43.78/42.48	0.229731
p3.2.g	33	20	390	40.70	214	430/390	43.406/39.33	43.360916
p3.2.m	33	10	650	70.69	300	700/660	73.02/66.84	228.67
p3.2.m	33	20	670	71.057	214	690/670	71.94/68.80	0.106059
p3.3.e	33	10	210	28.52	214	220/210	32.96/28.49	0.044919
p3.3.e	33	20	220	30.48	214	220	30.48	0.053809
p3.4.i	33	10	300	49.38	214	320/300	49.21/38.98	0.0524
p3.4.i	33	20	300	47.169	214	320/300	49.94/46.36	0.109534
p5.2.q	66	10	1280	83.39	300	1405	83.85	291.31
p5.2.q	66	20	1420	83.8	2840	1405	84.50	2833.005
p5.3.w	66	10	840	108.59	400	1680	112.56	376.714020
p5.3.w	66	20	750	101.84		1675	110.30	8648.87
p5.4.m	66	10	470	53.07	214	705	62.64	1.092805
p5.4.m	66	20	/	/	/	690	59.026	1183.98
p6.2.i	64	10	612	53.53	214	1074	53.66	0.347492
p6.2.i	64	20	438/936	48.815/52.7	5010	1062	54.68	5008.36
p6.4.k	64	10	102	29.45	214	882	62.74	0.389556
p6.4.k	64	20	/	/	/	912	61.65	327.08
p7.2.m	102	10	765	252.74	700	954	256.34	699.904541
p7.2.g	102	10	331	121.748	214	539/373	138.56/97.26	0.090698
p7.2.g	102	20	/	/	/	542	138.05	1918.01

TABLE 5.15 – Comportement de $HMOEA^{worst}$ versus CPLEX sur les instances avec 10 scénarios et une déviation de valeur 40.

Chapitre 6

Conclusion générale

Les travaux décrits dans ce manuscrit de thèse sont dédiés à l'étude du problème de tournées de véhicules. L'objectif principal étant de généraliser le problème VRP afin de le rendre suffisamment flexible pour résoudre les problématiques du monde réel en se focalisant sur deux axes majeurs. Le premier concerne la prise en compte des incertitudes dans la modélisation du problème VRP, tandis que le deuxième concerne la considération de plusieurs objectifs à optimiser simultanément. Le modèle d'incertitude que nous avons considéré dans les travaux de ce manuscrit concerne les coûts de trajet, un domaine qui a été faiblement exploré dans la littérature de l'optimisation robuste. Dans ce modèle, le coût de trajet prend sa valeur dans un ensemble discret de scénarios.
Nous nous sommes focalisés dans un premier temps sur l'étude des critères de robustesse permettant de prendre en compte l'incertitude engendrée par les différents scénarios. Après avoir étudié les critères de robustesse classiques de l'état de l'art comme : le meilleur cas, le pire cas et le pire regret, nous avons proposé un nouveau critère appelé MNSQW ("Maximising the Number of Scénarios Qualified by the Worst"). Le but de ce critère étant d'être moins conservateur que le critère du pire cas et d'améliorer la qualité des résultats tout en étant plus rapide que le critère du min-max regret (ou pire regret). Il consiste à maximiser le nombre de scénarios où la solution actuelle est plus performante que celle du pire cas. Ce critère a été évalué en utilisant deux approches algorithmiques. La première est une méthode exacte pour laquelle un ensemble de 480 instances à petite échelle, générées à partir des instances de référence de Solomon, a été utilisé. La deuxième approche est une heuristique qui a été appliquée sur un ensemble d'instances de moyenne et grande échelles. Les résultats obtenus ont montré que le critère que nous proposons permet de produire les solutions robustes dans la majorité des cas par rapport aux autres critères de robustesse.

Une perspective naturelle de ce travail serait de considérer d'autres types d'incertitudes comme l'incertitude sur les demandes ou bien sur les clients. Une autre perspective qui va dans le sens de la réduction de la complexité du problème (amplifiée par la prise en compte des incertitudes) serait d'appliquer une méthode de décomposition exacte telle que la décomposition de Benders par exemple qui consiste à décomposer le problème en sous problèmes afin de réduire sa complexité.

La deuxième partie de cette thèse a porté sur la résolution des problèmes VRP capacitaires incertains en utilisant une méta-heuristique. Un algorithme évolutionnaire multi-objectif hybridé avec des opérateurs de recherche locale a été proposé

pour optimiser deux objectifs de façon simultanée à savoir : le critère du pire cas du coût de trajet total et la taille de la flotte. L'application de cet algorithme a nécessité la réalisation de quelques adaptations notamment sur la représentation du chromosome et les opérateurs de variation (croisement et mutation). Différentes stratégies d'amélioration locale ont été utilisées afin d'améliorer localement les solutions trouvées par l'approche globale basée sur NSGA-II. De plus, une heuristique de destruction/construction (DCH) a été incorporée dans le but de diversifier davantage les solutions. Un certain nombre d'expérimentations ont été menées pour évaluer la performance de l'approche proposée sur le problème étudié (RVRP). L'algorithme proposé a été appliqué sur des instances de référence tirées de la littérature et regroupées en deux catégories : petites et grandes instances. Les résultats obtenus ont été comparés à ceux obtenus avec une méthode exacte utilisant le solveur GLPK et avec deux méthodes méta-heuristiques parmi les plus récentes de l'état de l'art. A noter que toutes ces méthodes que nous avons utilisées pour la comparaison sont des méthodes mono-objectif. Ceci est expliqué par l'inexistence d'une approche multi-objectif traitant du problème de tournées de véhicules robuste. Pour cela nous avons opté pour une comparaison d'un des deux objectifs que nous avons considéré à savoir le pire coût de trajet avec les approches mono-objectif considérées. L'analyse des résultats a permis de constater que toutes les solutions trouvées par le solveur GLPK sur les petites instances ont été atteintes sauf pour une instance, tandis qu'une nouvelle borne a été établie pour une autre instance. La comparaison avec la première approche méta-heuristique qui utilise un algorithme génétique mono-objectif a permis d'établir une nette amélioration des résultats obtenus sur toutes les instances. Concernant l'autre approche méta-heuristique qui est basée sur une approche de recherche locale itérative, la comparaison a montré une similarité sur la majorité des instances avec deux nouvelles bornes trouvées par notre approche et trois instances où notre approche a été surpassée.

Une perspective intéressante de ce travail serait de considérer d'autres objectifs à optimiser comme l'équilibre de la charge de travail des conducteurs par exemple (voir chapitre 2 pour la liste des objectifs potentiels) . Cependant, il est à noter qu'à partir de quatre objectifs, les approches appartenant au domaine de l'optimisation multi-objectif risquent de rencontrer des difficultés. Les approches dite "Many Objective Optimization" semblent être plus adaptées pour ce cas de figure, et représentent une perspective intéressante pour l'extension des travaux .
Une autre perspective porterait sur l'affinement des opérateurs de croisement et de mutation pour les faire participer davantage dans le processus d'optimisation en minimisant le coût de trajet des solutions créées avec ces opérateurs.

Dans la troisième partie de cette thèse, nous nous sommes focalisés sur une autre variante du VRP à savoir le problème de tournées sélectives (TOP). Pour cette variante, les problèmes d'optimisation multi-objectif ont été faiblement explorés. La formulation multi-objectif que nous avons considérée propose d'optimiser simultanément le profit collecté (souvent considéré dans les optimisations mono-objectifs) qui est à maximiser et le coût de trajet qui est plutôt à minimiser. Pour la résolution de ce problème, nous avons considéré deux variantes : déterministe (MO-TOP) et

robuste (MO-RTOP), et nous avons opté, comme pour le RVRP, pour une approche évolutionnaire multi-objectif hybridée avec des opérateurs de recherche locale. Cette approche a été appliquée sur une série d'instances proposée dans la littérature. Les résultats obtenus ont permis tout d'abord de confirmer l'antagonisme entre les deux objectifs considérés. Cette approche a été par la suite comparée avec trois autres approches de la littérature. Tout comme le RVRP, ces approches reposent sur des algorithmes mono-objectifs. La comparaison avec ces approches a permis de trouver quatre nouvelles bornes avec notre approches pour le profit. De façon générale, notre approche surpasse les autres approches de l'état de l'art sur 4 groupes d'instances, a un comportement similaire sur 2 groupes et est dominée sur 1 groupe. Après avoir étudié le TOP multi-objectif de façon déterministe, nous nous sommes focalisés dans un deuxième temps sur la variante du TOP multi-objectif avec incertitude (MO-RTOP). L'incertitude considérée a porté sur le coût de trajet entre deux villes pour lequel plusieurs scénarios sont possibles. Comme il n'existe pas d'instances dédiées au TOP robuste dans la littérature, de nouvelles instances ont été générées en se basant sur celle proposées pour le TOP déterministe. Les résultats obtenus ont été comparés à ceux trouvés par une méthode exacte (CPLEX). Il ressort de la comparaison que notre approche dépasse le CPLEX sur la majorité des instances, tout en étant plus rapide en termes de temps d'exécution.

Une perspective pour le problème de TOP qui va dans le sens de l'amélioration des solutions de ce problème serait de considérer d'autres méthodes de génération pour la population initiale, comme la technique de "Giant tour" qui a été appliquée avec succès sur d'autres problèmes dans la littérature.

De façon plus générale, bien que les algorithmes évolutionnaires multi-objectifs offrent davantage de choix au décideur grâce à leur flexibilité et robustesse dans le traitement de plusieurs problèmes divers, l'application de ces derniers sur les problématiques réelles se heurte à la complexité grandissante avec la croissance du nombre de clients ou de scénarios liés à l'incertitude, et aux nombreux objectifs qui sont à optimiser. L'utilisation des méthodes approchées comme les méta-heuristiques a certes permis d'avoir des avancées considérables dans la résolution de ces problèmes. Néanmoins, l'évolution importante des capacités de calcul durant les dernières années a permis aux techniques de parallélisation de se positionner comme une perspective évidente afin de continuer à résoudre des problèmes d'une plus grande complexité.

Bibliographie

[Adida 2010] Elodie Adida et Georgia Perakis. *Dynamic pricing and inventory control : robust vs. stochastic uncertainty models a computational study.* Annals of Operations Research, vol. 181, no. 1, pages 125–157, 2010. 34

[Agra 2013] Agostinho Agra, Marielle Christiansen, Rosa Figueiredo, Lars Magnus Hvattum, Michael Poss et Cristina Requejo. *The robust vehicle routing problem with time windows.* Computers and Operations Research, vol. 40, no. 3, pages 856–866, 2013. 21, 23

[Aissi 2009] Hassene Aissi, Cristina Bazgan et Daniel Vanderpooten. *Min–max and min–max regret versions of combinatorial optimization problems : A survey.* European journal of operational research, vol. 197, no. 2, pages 427–438, 2009. 35, 38

[Al-Maliky 2018] Ferhan Al-Maliky, Mhand Hifi et Hedi Mhalla. *Sensitivity analysis of the setup knapsack problem to perturbation of arbitrary profits or weights.* International Transactions in Operational Research, vol. 25, no. 2, pages 637–666, 2018. 16

[Alumur 2012] Sibel A Alumur, Stefan Nickel et Francisco Saldanha-da Gama. *Hub location under uncertainty.* Transportation Research Part B : Methodological, vol. 46, no. 4, pages 529–543, 2012. 18

[Assis 2013] Luciana P Assis, Andre L Maravilha, Alessandro Vivas, Felipe Campelo et Jaime A Ramirez. *Multiobjective vehicle routing problem with fixed delivery and optional collections.* Optimization letters, vol. 7, no. 7, pages 1419–1431, 2013. 30, 32

[Baker 2003] Barrie M Baker et MA Ayechew. *A genetic algorithm for the vehicle routing problem.* Computers and Operations Research, vol. 30, no. 5, pages 787–800, 2003. 10

[Baldacci 2008] Roberto Baldacci, Nicos Christofides et Aristide Mingozzi. *An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts.* Mathematical Programming, vol. 115, no. 2, pages 351–385, 2008. 10

[Baldacci 2011] Roberto Baldacci, Aristide Mingozzi et Roberto Roberti. *New route relaxation and pricing strategies for the vehicle routing problem.* Operations research, vol. 59, no. 5, pages 1269–1283, 2011. 2, 10

[BanOs 2013] RauL BanOs, Julio Ortega, ConsolacioN Gil, Antonio L MaRquez et Francisco De Toro. *A hybrid meta-heuristic for multi-objective vehicle routing problems with time windows.* Computers and Industrial Engineering, vol. 65, no. 2, pages 286–296, 2013. 30, 32

[Bederina 2016] Hiba Bederina et Mhand Hifi. *Evolutionary multi-objective optimization approach for the vehicle routing problem with uncertain travel time.*

In Control Engineering & Information Technology (CEIT), 2016 4th International Conference on, pages 1–6. IEEE, 2016. 51

[Bederina 2017a] Hiba Bederina et Mhand Hifi. *A hybrid multi-objective evolutionary algorithm for the team orienteering problem.* In 4th International Conference on Control, Decision and Information Technologies (CoDIT), pages 898–903, April 2017. 79

[Bederina 2017b] Hiba Bederina et Mhand Hifi. *Solving the Multi-Objective Team Orienteering Problem with Uncertain Travel Time.* In International Conference on Multiple Objective Programming anf Goal Programming, 10 2017. 79

[Bederina 2017c] Hiba Bederina et Mhand Hifi. *Tackling the Robust Vehicle Routing Problem with Discrete Set of Scenarios.* In International Conference on Multiple Objective Programming anf Goal Programming, 10 2017. 51

[Bederina 2018] Hiba Bederina et Mhand Hifi. *A Hybrid Multiobjective Evolutionary Optimization Approach for the Robust Vehicle Routing Problem.* Applied Soft Computing, page .Submitted (In revision), 2018. 51

[Bektaş 2014] Tolga Bektaş et Luis Gouveia. *Requiem for the Miller–Tucker–Zemlin subtour elimination constraints ?* European Journal of Operational Research, vol. 236, no. 3, pages 820–832, 2014. 37

[Bellman 1970] Richard E Bellman et Lotfi Asker Zadeh. *Decision-making in a fuzzy environment.* Management science, vol. 17, no. 4, pages B–141, 1970. 16

[Ben-Tal 2004] Aharon Ben-Tal, Alexander Goryashko, Elana Guslitzer et Arkadi Nemirovski. *Adjustable robust solutions of uncertain linear programs.* Mathematical Programming, vol. 99, no. 2, pages 351–376, 2004. 18

[Ben-Tal 2009] Aharon Ben-Tal, Laurent El Ghaoui et Arkadi Nemirovski. Robust optimization. Princeton University Press, 2009. 18

[Ben-Tal 2010] Aharon Ben-Tal, Dimitris Bertsimas et David B Brown. *A soft robust model for optimization under ambiguity.* Operations research, vol. 58, no. 4-part-2, pages 1220–1234, 2010. 18

[Bertsimas 1996] Dimitris J Bertsimas et David Simchi-Levi. *A new generation of vehicle routing research : robust algorithms, addressing uncertainty.* Operations Research, vol. 44, no. 2, pages 286–304, 1996. 34

[Bertsimas 2003] Dimitris Bertsimas et Melvyn Sim. *Robust discrete optimization and network flows.* Mathematical programming, vol. 98, no. 1, pages 49–71, 2003. 22

[Bertsimas 2011] Dimitris Bertsimas, David B Brown et Constantine Caramanis. *Theory and applications of robust optimization.* SIAM review, vol. 53, no. 3, pages 464–501, 2011. 18

[Bibai 2010] Jacques Bibai. *Segmentation et evolution pour la planification : le systeme Divide And Evolve.* Theses, Universite Paris Sud Paris XI, Octobre 2010. 25

[Bouly 2010] Hermann Bouly, Duc-Cuong Dang et Aziz Moukrim. *A memetic algorithm for the team orienteering problem.* 4or, vol. 8, no. 1, pages 49–70, 2010. 12, 80, 87

[Campolongo 2007] Francesca Campolongo, Jessica Cariboni et Andrea Saltelli. *An effective screening design for sensitivity analysis of large models.* Environmental modelling & software, vol. 22, no. 10, pages 1509–1518, 2007. 17

[Cao 2014] Erbao Cao, Mingyong Lai et Hongming Yang. *Open vehicle routing problem with demand uncertainty and its robust strategies.* Expert Systems with Applications, vol. 41, no. 7, pages 3569–3575, 2014. 21, 23

[Chand 2010] Padmabati Chand, Bhabani Sankar Prasad Mishra et Satchidananda Dehuri. *A multi objective genetic algorithm for solving vehicle routing problem.* International Journal of Information Technology and Knowledge Management, vol. 2, no. 2, pages 503–506, 2010. 29

[Chand 2013a] Padmabati Chand et JR Mohanty. *Multi objective genetic approach for solving vehicle routing problem.* International Journal of Computer Theory and Engineering, vol. 5, no. 6, page 846, 2013. 30, 31, 32

[Chand 2013b] Padmabati Chand et JR Mohanty. *A multi-objective vehicle routing problem using dominant rank method.* International Journal of Computer Application, pages 29–34, 2013. 30, 31, 32

[Chao 1996] I-Ming Chao, Bruce L Golden et Edward A Wasil. *The team orienteering problem.* European journal of operational research, vol. 88, no. 3, pages 464–474, 1996. 12, 13, 87

[Cheong 2006] Chun Yew Cheong, Kay Chen Tan, DK Liu et Jian-Xin Xu. *A multiobjective evolutionary algorithm for solving vehicle routing problem with stochastic demand.* In Evolutionary Computation, 2006. CEC 2006. IEEE Congress on, pages 1370–1377. IEEE, 2006. 32, 57, 58, 84, 85

[Clarke 1964] G. Clarke et J.W. Wright. *Scheduling of vehicles from a central depot to a number of delivery points.* Operations research, vol. 12, no. 4, pages 568–581, 1964. 10

[Dang 2011] Duc-Cuong Dang, Rym Nesrine Guibadj et Aziz Moukrim. *A pso-based memetic algorithm for the team orienteering problem.* In European Conference on the Applications of Evolutionary Computation, pages 471–480. Springer, 2011. 12, 87

[Dang 2013a] Duc-Cuong Dang, Racha El-Hajj et Aziz Moukrim. *A branch-and-cut algorithm for solving the team orienteering problem.* In International Conference on AI and OR Techniques in Constriant Programming for Combinatorial Optimization Problems, pages 332–339. Springer, 2013. 12

[Dang 2013b] Duc-Cuong Dang, Rym Nesrine Guibadj et Aziz Moukrim. *An effective PSO-inspired algorithm for the team orienteering problem.* European Journal of Operational Research, vol. 229, no. 2, pages 332–344, 2013. 12, 87, 92

[Dantzig 1959] George B Dantzig et John H Ramser. *The truck dispatching problem.* Management science, vol. 6, no. 1, pages 80–91, 1959. 11

[Deb 2000] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap et Tanaka Meyarivan. *A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization : NSGA-II.* In International Conference on Parallel Problem Solving From Nature, pages 849–858. Springer, 2000. 80, 83

[Deb 2002] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal et TAMT Meyarivan. *A fast and elitist multiobjective genetic algorithm : NSGA-II.* Evolutionary Computation, IEEE Transactions on, vol. 6, no. 2, pages 182–197, 2002. 27, 52

[Deb 2003] Kalyanmoy Deb, Manikanth Mohan et Shikhar Mishra. *Towards a quick computation of well-spread pareto-optimal solutions.* In Evolutionary multi-criterion optimization, pages 68–68. Springer, 2003. 27

[Desrochers 1990] Martin Desrochers, Jan Karel Lenstra et Martin WP Savelsbergh. *A classification scheme for vehicle routing and scheduling problems.* European Journal of Operational Research, vol. 46, no. 3, pages 322–332, 1990. 10

[Dresher 1961] Melvin Dresher. *Games of strategy : theory and applications.* Rapport technique, RAND CORP SANTA MONICA CA, 1961. 19

[Erera 2010] Alan L Erera, Juan C Morales et Martin Savelsbergh. *The vehicle routing problem with stochastic demand and duration constraints.* Transportation Science, vol. 44, no. 4, pages 474–492, 2010. 21

[Faure 2014] Robert Faure, Bernard Lemaire et Christophe Picouleau. Precis de recherche operationnelle : methodes et exercices d'application. Dunod, 2014. 6

[Fischetti 2009] Matteo Fischetti et Michele Monaci. *Light robustness.* In Robust and online large-scale optimization, pages 61–84. Springer, 2009. 18

[Fogel 1966] Lawrence J Fogel, Alvin J Owens et Michael J Walsh. *Artificial intelligence through simulated evolution.* 1966. 25

[Fukasawa 2006] Ricardo Fukasawa, Humberto Longo, Jens Lysgaard, Marcus Poggi de Aragao, Marcelo Reis, Eduardo Uchoa et Renato F Werneck. *Robust branch-and-cut-and-price for the capacitated vehicle routing problem.* Mathematical programming, vol. 106, no. 3, pages 491–511, 2006. 10

[Gabrel 2013a] Virginie Gabrel, Cecile Murat et Aurelie Thiele. *La pw-robustesse : pourquoi un nouveau critere de robustesse et comment l'appliquer.* 14eme Congres de la Societe FranÃ§aise de Recherche Operationnelle et D'Aide a la Decision (ROADEF), page 2p, 2013. 19

[Gabrel 2013b] Virginie Gabrel, Cecile Murat et Lei Wu. *New models for the robust shortest path problem : complexity, resolution and generalization.* Annals of Operations Research, pages 1–24, 2013. 19, 35

[Gabrel 2014] Virginie Gabrel, Cécile Murat et Aurélie Thiele. *Recent advances in robust optimization : An overview.* European journal of operational research, vol. 235, no. 3, pages 471–483, 2014. 34

[Garcia-Najera 2011] Abel Garcia-Najera et John A Bullinaria. *An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows.* Computers and Operations Research, vol. 38, no. 1, pages 287–300, 2011. 30, 32

[Geiger 2010] Martin Josef Geiger. *Fast approximation heuristics for multi-objective vehicle routing problems.* In European Conference on the Applications of Evolutionary Computation, pages 441–450. Springer, 2010. 30, 32

[Gendreau 2005] Michel Gendreau et Jean-Yves Potvin. *Metaheuristics in combinatorial optimization.* Annals of Operations Research, vol. 140, no. 1, pages 189–213, 2005. 10

[Ghannadpour 2014] Seyed Farid Ghannadpour, Simak Noori, Reza Tavakkoli-Moghaddam et Keivan Ghoseiri. *A multi-objective dynamic vehicle routing problem with fuzzy time windows : Model, solution and application.* Applied Soft Computing, vol. 14, pages 504–527, 2014. 31, 32

[Ghoseiri 2010] Keivan Ghoseiri et Seyed Farid Ghannadpour. *Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm.* Applied Soft Computing, vol. 10, no. 4, pages 1096–1107, 2010. 32

[Gillett 1974] Billy E Gillett et Leland R Miller. *A heuristic algorithm for the vehicle-dispatch problem.* Operations research, vol. 22, no. 2, pages 340–349, 1974. 10

[Goh 2010] Joel Goh et Melvyn Sim. *Distributionally robust optimization and its tractable approximations.* Operations research, vol. 58, no. 4-part-1, pages 902–917, 2010. 18

[Goldberg 1989] David E Goldberg. *Genetic algorithms in search, optimization, and machine learning, 1989.* Reading : Addison-Wesley, 1989. 25

[Golden 1987] Bruce L Golden, Larry Levy et Rakesh Vohra. *The orienteering problem.* Naval research logistics, vol. 34, no. 3, pages 307–318, 1987. 12

[Golden 2008] Bruce L Golden, Subramanian Raghavan et Edward A Wasil. The vehicle routing problem : latest advances and new challenges, volume 43. Springer Science and Business Media, 2008. 10

[Gounaris 2013] Chrysanthos E Gounaris, Wolfram Wiesemann et Christodoulos A Floudas. *The robust capacitated vehicle routing problem under demand uncertainty.* Operations Research, vol. 61, no. 3, pages 677–693, 2013. 21, 23

[Groër 2010] Chris Groër, Bruce Golden et Edward Wasil. *A library of local search heuristics for the vehicle routing problem.* Mathematical Programming Computation, vol. 2, no. 2, pages 79–101, 2010. 44

[Gunawan 2016] Aldy Gunawan, Hoong Chuin Lau et Pieter Vansteenwegen. *Orienteering Problem : A survey of recent variants, solution approaches and applications*. European Journal of Operational Research, 2016. 12

[Ha-Duong 2002] Minh Ha-Duong. *Introduction aux approches économiques de l'incertitude*. 2002. 13

[Han 2013] Jinil Han, Chungmok Lee et Sungsoo Park. *A robust scenario approach for the vehicle routing problem with uncertain travel times*. Transportation science, vol. 48, no. 3, pages 373–390, 2013. 21, 23, 34

[Hansen 2010] Pierre Hansen, Nenad Mladenović et José A Moreno Pérez. *Variable neighbourhood search : methods and applications*. Annals of Operations Research, vol. 175, no. 1, pages 367–407, 2010. 41

[Hao 1999] Jin-Kao Hao, Philippe Galinier et Michel Habib. *Metaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes*. Revue d'intelligence artificielle, vol. 13, no. 2, pages 283–324, 1999. 10

[Hifi 2014] Mhand Hifi, Sagvan Saleh et Lei Wu. *A fast large neighborhood search for disjunctively constrained knapsack problems*. In International Symposium on Combinatorial Optimization, pages 396–407. Springer, 2014. 41

[Holland 1975] John H Holland. *Adaptation in natural and artificial systems. An introductory analysis with application to biology, control, and artificial intelligence*. Ann Arbor, MI : University of Michigan Press, 1975. 25

[Homberger 2005] Jörg Homberger et Hermann Gehring. *A two-phase hybrid metaheuristic for the vehicle routing problem with time windows*. European Journal of Operational Research, vol. 162, no. 1, pages 220–238, 2005. 29

[Ilhan 2008] Taylan Ilhan, Seyed MR Iravani et Mark S Daskin. *The orienteering problem with stochastic profits*. Iie Transactions, vol. 40, no. 4, pages 406–421, 2008. 80

[JANSSENS 2005] Gerrit JANSSENS, An Caris et Katrien Ramaekers. *Sensitivity analysis of vehicle routing solutions to uncertainty in travel times*. 2005. 16

[Jiang 2008] Zhen-duo Jiang, Shi-jie Sun et Zhi-gang Wu. *Sensitivity analysis for some scheduling problems*. Journal of Shanghai University (English Edition), vol. 12, no. 1, pages 20–25, 2008. 16

[Jiang 2014] Jing Jiang, Sen Bong Gee, Willson Amalraj Arokiasami et Kay Chen Tan. *Solving Vehicle Routing Problem with Stochastic Demand Using Multi-objective Evolutionary Algorithm*. In Soft Computing and Machine Intelligence (ISCMI), 2014 International Conference on, pages 121–125. IEEE, 2014. 29

[Jozefowiez 2009] Nicolas Jozefowiez, Frederic Semet et El-Ghazali Talbi. *An evolutionary algorithm for the vehicle routing problem with route balancing*. European Journal of Operational Research, vol. 195, no. 3, pages 761–769, 2009. 31, 32

[Ke 2008] Liangjun Ke, Claudia Archetti et Zuren Feng. *Ants can solve the team orienteering problem.* Computers and Industrial Engineering, vol. 54, no. 3, pages 648–665, 2008. 12

[Ke 2013] Liangjun Ke, Zongben Xu, Zuren Feng, Ke Shang et Xueming Qian. *Proportion-based robust optimization and team orienteering problem with interval data.* European Journal of Operational Research, vol. 226, no. 1, pages 19–31, 2013. 80

[Keshtkaran 2016] Morteza Keshtkaran, Koorush Ziarati, Andrea Bettinelli et Daniele Vigo. *Enhanced exact solution methods for the team orienteering problem.* International Journal of Production Research, vol. 54, no. 2, pages 591–601, 2016. 12

[Kim 2013] Byung-In Kim, Hong Li et Andrew L Johnson. *An augmented large neighborhood search method for solving the team orienteering problem.* Expert Systems with Applications, vol. 40, no. 8, pages 3065–3072, 2013. 12

[Kouvelis 1997] P. Kouvelis et G. Yu. *Robust discrete optimization and its applications.* Boston :Kluwer Academic., 1997. 34

[Kouwenberg 2001] Roy Kouwenberg. *Scenario generation and stochastic programming models for asset liability management.* European Journal of Operational Research, vol. 134, no. 2, pages 279–292, 2001. 15

[Koza 1992] John R Koza. Genetic programming : on the programming of computers by means of natural selection, volume 1. MIT press, 1992. 25

[Koza 1994] John R Koza. *Genetic programming as a means for programming computers by natural selection.* Statistics and computing, vol. 4, no. 2, pages 87–112, 1994. 25

[Laguna 1998] Manuel Laguna. *Applying robust optimization to capacity expansion of one location in telecommunications with demand uncertainty.* Management Science, vol. 44, no. 11-part-2, pages S101–S110, 1998. 15

[Laporte 2000] Gilbert Laporte, Michel Gendreau, J-Y Potvin et Frederic Semet. *Classical and modern heuristics for the vehicle routing problem.* International transactions in operational research, vol. 7, no. 4-5, pages 285–300, 2000. 10

[Laporte 2009] Gilbert Laporte. *Fifty years of vehicle routing.* Transportation Science, vol. 43, no. 4, pages 408–416, 2009. 10

[Lee 2012] Chungmok Lee, Kyungsik Lee et Sungsoo Park. *Robust vehicle routing problem with deadlines and travel time and demand uncertainty.* Journal of the Operational Research Society, vol. 63, no. 9, pages 1294–1306, 2012. 22, 23

[Lenstra 1981] Jan Karel Lenstra et AHG Kan. *Complexity of vehicle routing and scheduling problems.* Networks, vol. 11, no. 2, pages 221–227, 1981. 2

[Liebchen 2009] Christian Liebchen, Marco Lubbecke, Rolf Mohring et Sebastian Stiller. *The concept of recoverable robustness, linear programming recovery,*

and railway applications. In Robust and online large-scale optimization, pages 1–27. Springer, 2009. 18

[Lin 2013] Shih-Wei Lin. *Solving the team orienteering problem using effective multi-start simulated annealing.* Applied Soft Computing, vol. 13, no. 2, pages 1064–1073, 2013. 12

[Lysgaard 2004] Jens Lysgaard, Adam N Letchford et Richard W Eglese. *A new branch-and-cut algorithm for the capacitated vehicle routing problem.* Mathematical Programming, vol. 100, no. 2, pages 423–445, 2004. 10

[Miller 1960] Clair E Miller, Albert W Tucker et Richard A Zemlin. *Integer programming formulation of traveling salesman problems.* Journal of the ACM (JACM), vol. 7, no. 4, pages 326–329, 1960. 37

[Minoux 1983] M Minoux. *Programmation Mathematique Theorie et Algorithmes vol 1 (Paris : Dunod).* 1983. 7

[Mittal 2011] Shashi Mittal*et al.* *Algorithms for discrete, non-linear and robust optimization problems with applications in scheduling and service operations.* PhD thesis, Massachusetts Institute of Technology, 2011. 18

[Moghaddam 2012] Babak Farhang Moghaddam, Ruben Ruiz et Seyed Jafar Sadjadi. *Vehicle routing problem with uncertain demands : An advanced particle swarm algorithm.* Computers & Industrial Engineering, vol. 62, no. 1, pages 306–317, 2012. 21, 23

[Molina 2014] Jose Carlos Molina, Ignacio Eguia, Jesus Racero et Fernando Guerrero. *Multi-objective vehicle routing problem with cost and emission functions.* Procedia-Social and Behavioral Sciences, vol. 160, pages 254–263, 2014. 29

[Muller 2010] Juliane Muller. *Approximative solutions to the bicriterion vehicle routing problem with time windows.* European Journal of Operational Research, vol. 202, no. 1, pages 223–231, 2010. 29

[Mulvey 1995] John M Mulvey, Robert J Vanderbei et Stavros A Zenios. *Robust optimization of large-scale systems.* Operations research, vol. 43, no. 2, pages 264–281, 1995. 15

[Murata 2007] Tadahiko Murata et Ryota Itai. *Local search in two-fold EMO algorithm to enhance solution similarity for multi-objective vehicle routing problems.* In International Conference on Evolutionary Multi-Criterion Optimization, pages 201–215. Springer, 2007. 30, 32

[Muthuswamy 2011] Shanthi Muthuswamy et Sarah S Lam. *Discrete particle swarm optimization for the team orienteering problem.* Memetic Computing, vol. 3, no. 4, pages 287–303, 2011. 12

[Neumann 1928] J von Neumann. *Die Zerlegung eines Intervalles in abzahlbar viele kongruente Teilmengen.* Fundamenta Mathematicae, vol. 1, no. 11, pages 230–238, 1928. 19

[Noorizadegan 2012] Mahdi Noorizadegan, Laura Galli et Bo Chen. *On the heterogeneous vehicle routing problem under demand uncertainty.* 2012. 21, 23

[Okuda 1975] Tetsuji Okuda, Hideo Tanaka et Kiyoji Asai. *Decision-making and information in fuzzy events.* 1975. 16

[Ombuki-Berman 2007] Beatrice M Ombuki-Berman, Andrew Runka et FranklinT Hanshar. *Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms.* In Proceedings of the Third IASTED International Conference on Computational Intelligence, pages 91–97. ACTA Press, 2007. 32

[Ombuki 2006] Beatrice Ombuki, Brian J Ross et Franklin Hanshar. *Multi-objective genetic algorithms for vehicle routing problem with time windows.* Applied Intelligence, vol. 24, no. 1, pages 17–30, 2006. 32

[Ordonez 2010] Fernando Ordonez. *Robust vehicle routing.* In Risk and Optimization in an Uncertain World, pages 153–178. INFORMS, 2010. 22, 23

[Pecin 2017] Diego Pecin, Artur Pessoa, Marcus Poggi et Eduardo Uchoa. *Improved branch-cut-and-price for capacitated vehicle routing.* Mathematical Programming Computation, vol. 9, no. 1, pages 61–100, 2017. 10

[Pierre 2014] Djamalladine Mahamat Pierre et Nordin Zakaria. *Partially optimized cyclic shift crossover for multi-objective genetic algorithms for the multi-objective vehicle routing problem with time-windows.* In Computational Intelligence in Multi-Criteria Decision-Making (MCDM), 2014 IEEE Symposium on, pages 106–115. IEEE, 2014. 31, 32

[Pisinger 2010] David Pisinger et Stefan Ropke. *Large neighborhood search.* In Handbook of metaheuristics, pages 399–419. Springer, 2010. 41

[Poggi 2010] Marcus Poggi, Henrique Viana et Eduardo Uchoa. *The team orienteering problem : Formulations and branch-cut and price.* In OASIcs-OpenAccess Series in Informatics, volume 14. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2010. 12

[Prins 2004] Christian Prins. *A simple and effective evolutionary algorithm for the vehicle routing problem.* Computers and Operations Research, vol. 31, no. 12, pages 1985–2002, 2004. 10

[Prins 2014] Christian Prins, Philippe Lacomme et Caroline Prodhon. *Order-first split-second methods for vehicle routing problems : A review.* Transportation Research Part C : Emerging Technologies, vol. 40, pages 179–200, 2014. 10

[Psychas 2015] Iraklis-Dimitrios Psychas, Magdalene Marinaki et Yannis Marinakis. *A parallel multi-start NSGA II algorithm for multiobjective energy reduction vehicle routing problem.* In International Conference on Evolutionary Multi-Criterion Optimization, pages 336–350. Springer, 2015. 30, 32

[Rechenberg 1973] Ingo Rechenberg. *Evolution Strategy : Optimization of Technical systems by means of biological evolution.* Fromman-Holzboog, Stuttgart, vol. 104, 1973. 25

[Roy 2010] Bernard Roy. *Robustness in operational research and decision aiding : A multi-faceted issue.* European Journal of Operational Research, vol. 200, no. 3, pages 629–638, 2010. 19, 34, 35

[Savage 1951] Leonard J Savage. *The theory of statistical decision.* Journal of the American Statistical association, vol. 46, no. 253, pages 55–67, 1951. 19

[Schilde 2009] Michael Schilde, Karl F Doerner, Richard F Hartl et Guenter Kiechle. *Metaheuristics for the bi-objective orienteering problem.* Swarm Intelligence, vol. 3, no. 3, pages 179–201, 2009. 80

[Schoenauer 2003] Marc Schoenauer. *Les algorithmes evolutionnaires : etat de l art et enjeux.* In Algorithms Seminar, 2001-2002, page 113, 2003. vii, 26

[Shaw 1998] Paul Shaw. *Using constraint programming and local search methods to solve vehicle routing problems.* In International Conference on Principles and Practice of Constraint Programming, pages 417–431. Springer, 1998. 41, 43

[Snyder 2006] Lawrence V Snyder. *Facility location under uncertainty : a review.* IIE Transactions, vol. 38, no. 7, pages 547–564, 2006. 20

[Solano-Charris 2014] Elyn L Solano-Charris, Christian Prins et AndrÃ©a Cynthia Santos. *A robust optimization approach for the vehicle routing problem with uncertain travel cost.* In Control, Decision and Information Technologies (CoDIT), 2014 International Conference on, pages 098–103. IEEE, 2014. 51, 64, 67

[Solano-Charris 2015] Elyn Solano-Charris, Christian Prins et Andréa Cynthia Santos. *Local search based metaheuristics for the robust vehicle routing problem with discrete scenarios.* Applied Soft Computing, vol. 32, pages 518–531, 2015. 22, 23, 35, 68

[Solomon 1987] Marius M Solomon. *Algorithms for the vehicle routing and scheduling problems with time window constraints.* Operations research, vol. 35, no. 2, pages 254–265, 1987. 44

[Sotskov 1995] Yu N Sotskov, VK Leontev et Evgenii N Gordeev. *Some concepts of stability analysis in combinatorial optimization.* Discrete Applied Mathematics, vol. 58, no. 2, pages 169–190, 1995. 16

[Soyster 1973] Allen L Soyster. *Convex programming with set-inclusive constraints and applications to inexact linear programming.* Operations research, vol. 21, no. 5, pages 1154–1157, 1973. 17

[Sulieman 2010] Dalia Sulieman, Laetitia Jourdan et El-Ghazali Talbi. *Using multiobjective metaheuristics to solve VRP with uncertain demands.* In Evolutionary Computation (CEC), 2010 IEEE Congress on, pages 1–8. IEEE, 2010. 30, 32

[Sungur 2008] Ilgaz Sungur, Fernando Ordonez et Maged Dessouky. *A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty.* IIE Transactions, vol. 40, no. 5, pages 509–523, 2008. 18, 20, 21, 22, 23

[Tan 2006] Kay Chen Tan, Yoong Han Chew et LH Lee. *A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows.* Computational Optimization and Applications, vol. 34, no. 1, page 115, 2006. 32, 55

[Tan 2007] Kay Chen Tan, Chun Yew Cheong et Chi Keong Goh. *Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation.* European Journal of operational research, vol. 177, no. 2, pages 813–839, 2007. 32

[Tanaka 1984] H Tanaka et K Asai. *Fuzzy linear programming problems with fuzzy numbers.* Fuzzy sets and systems, vol. 13, no. 1, pages 1–10, 1984. 16

[Tang 2011] Luohao Tang, Cheng Zhu, Weiming Zhang et Zhong Liu. *Robust mission planning based on nested genetic algorithm.* In Advanced computational intelligence (IWACI), 2011 fourth international workshop on, pages 45–49. IEEE, 2011. 19

[Toklu 2013a] NE Toklu, Roberto Montemanni et Luca Maria Gambardella. *An ant colony system for the capacitated vehicle routing problem with uncertain travel costs.* In Swarm Intelligence (SIS), 2013 IEEE Symposium on, pages 32–39. IEEE, 2013. 21, 23

[Toklu 2013b] NE Toklu, Roberto Montemanni et Luca Maria Gambardella. *A robust multiple ant colony system for the capacitated vehicle routing problem.* In Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on, pages 1871–1876. IEEE, 2013. 21, 23

[Toth 2002] Paolo Toth et Daniele Vigo. The vehicle routing problem. SIAM, 2002. 36

[Toth 2014] Paolo Toth et Daniele Vigo. Vehicle routing : problems, methods, and applications. SIAM, 2014. 10

[Tuyttens 2004] Daniel Tuyttens, Jacques Teghem et Nasser El-Sherbeny. *A particular multiobjective vehicle routing problem solved by simulated annealing.* In Metaheuristics for multiobjective optimisation, pages 133–152. Springer, 2004. 29

[urgen Zimmermann 1991] Hans-J urgen Zimmermann. *Fuzzy Set Theory| and Its Applications*, 1991. 16

[Vansteenwegen 2009] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe et Dirk Van Oudheusden. *A guided local search metaheuristic for the team orienteering problem.* European Journal of Operational Research, vol. 196, no. 1, pages 118–127, 2009. 12

[Vansteenwegen 2011] Pieter Vansteenwegen, Wouter Souffriau et Dirk Van Oudheusden. *The orienteering problem : A survey.* European Journal of Operational Research, vol. 209, no. 1, pages 1–10, 2011. 12

[Verbeeck 2014] Cedric Verbeeck, El-Houssaine Aghezzaf et Pieter Vansteenwegen. *Solving the Stochastic Time-Dependent Orienteering Problem.* In MOSIM 2014, 10ème Conférence Francophone de Modélisation, Optimisation et Simulation, 2014. 80

[Verbeeck 2016] Cédric Verbeeck, Pieter Vansteenwegen et E-H Aghezzaf. *Solving the stochastic time-dependent orienteering problem with time windows.* Euro-

pean Journal of Operational Research, vol. 255, no. 3, pages 699–718, 2016. 80

[Wei 2015] Lijun Wei, Zhenzhen Zhang, Defu Zhang et Andrew Lim. *A variable neighborhood search for the capacitated vehicle routing problem with two-dimensional loading constraints.* European Journal of Operational Research, vol. 243, no. 3, pages 798–814, 2015. 41

[Wu 2015] Lei Wu, Hiba Bederina et Mhand Hifi. *New model for the robust vehicle routing problem.* In 45th International Conference on Computers and Industrial Engineering 2015 (CIE45), pages 1302–1309, 10 2015. 33

[Wu 2017] Lei Wu, Mhand Hifi et Hiba Bederina. *A new robust criterion for the vehicle routing problem with uncertain travel time.* Computers & Industrial Engineering, 2017. 33

[Yagoubi 2012] Mouadh Yagoubi. *Optimisation evolutionnaire multi-objectif parallele : application a la combustion Diesel.* PhD thesis, Universite Paris Sud-Paris XI, 2012. 25, 27

[Yang 2008] Ying-qing Yang et Jiu-ping Xu. *A class of multiobjective vehicle routing optimal model under fuzzy random environment and its application.* World Journal of Modelling and Simulation, vol. 4, no. 2, pages 112–119, 2008. 30, 31, 32

[Zhang 2012] Jingling Zhang, Wanliang Wang, Yanwei Zhao et Carlo Cattani. *Multiobjective quantum evolutionary algorithm for the vehicle routing problem with customer satisfaction.* Mathematical Problems in Engineering, vol. 2012, 2012. 30, 32

[Zhao 2003] Fuquan Zhao, Thomas N Asmus, Dennis N Assanis, John E Dec, James A Eng et Paul M Najt. *Homogeneous charge compression ignition (HCCI) engines.* Rapport technique, SAE Technical Paper, 2003. 74

[Zhou 2013] Wei Zhou, Tingxin Song, Fei He et Xi Liu. *Multiobjective vehicle routing problem with route balance based on genetic algorithm.* Discrete Dynamics in Nature and Society, vol. 2013, 2013. 32

[Zhu 2009] Shushang Zhu et Masao Fukushima. *Worst-case conditional value-at-risk with application to robust portfolio management.* Operations research, vol. 57, no. 5, pages 1155–1168, 2009. 35

[Zitzler 1999] Eckart Zitzler et Lothar Thiele. *Multiobjective evolutionary algorithms : a comparative case study and the strength Pareto approach.* IEEE transactions on Evolutionary Computation, vol. 3, no. 4, pages 257–271, 1999. 74

[Zitzler 2001] Eckart Zitzler, Marco Laumanns et Lothar Thiele. *SPEA2 : Improving the strength Pareto evolutionary algorithm.* 2001. 27

[Zitzler 2004] Eckart Zitzler et Simon Kunzli. *Indicator-based selection in multiobjective search.* In International Conference on Parallel Problem Solving from Nature, pages 832–842. Springer, 2004. 27

[Zymler 2013] Steve Zymler, Daniel Kuhn et Berç Rustem. *Worst-case value at risk of nonlinear portfolios.* Management Science, vol. 59, no. 1, pages 172–188, 2013. 35

Résumé : Les travaux de cette thèse s'articulent autour du domaine de l'optimisation robuste pour résoudre le problème de tournées de véhicules (VRP). Le VRP représente l'un des problèmes les plus importants dans le domaine de l'optimisation combinatoire, de part sa capacité à résoudre différents problèmes, offrant aux entreprises des leviers pour rendre leurs flottes de véhicules plus efficaces et réduire les coûts. Toutefois, dans les problèmes du monde réel, certains paramètres dépendent d'autres facteurs tels que l'environnement et le temps, et cela empêche de définir de façon précise les paramètres du modèle, ce qui les rendent incertains. Dans ce contexte, la prise en compte de ces incertitudes devient nécessaire afin d'obtenir des solutions robustes. D'autre part, ces mêmes problèmes réels cherchent généralement à optimiser plusieurs objectifs de façon simultanée, et de ce fait, l'optimisation multi-objectif semble être un bon candidat pour résoudre ces problèmes.

L'objectif principal de cette thèse est de contribuer à l'adaptation des problèmes VRP aux problématiques du monde réel en se focalisant sur deux axes principaux à savoir : la prise en compte des incertitudes à travers l'optimisation robuste et l'optimisation simultanée de plusieurs critères en utilisant l'optimisation multi-objectif. Dans un premier temps, nous nous sommes focalisés sur la modélisation du problème VRP sous incertitudes en proposant un nouveau critère de robustesse. Ce critère, appelé " *Maximizing the Number of scenarios Qualified by the Worst* (MNSQW)", a été évalué en utilisant deux approches : une méthode exacte et une méta-heuristique. Les résultats obtenus ont permis de démontrer l'efficacité du critère proposé à produire des solutions robustes en le comparant avec trois critères connus de l'état de l'art.

Dans un deuxième temps, nous avons étudié la résolution robuste multi-objectif de la variante VRP capacitaire (CVRP) avec incertitudes sur les coûts de trajets. Un algorithme évolutionnaire multi-objectif hybridé avec des opérateurs de recherche locale a été proposé pour optimiser le coût de trajet et la taille de la flotte simultanément. Les expérimentations réalisées sur des instances de l'état de l'art ont permis de comparer l'approche proposée avec une méthode exacte et deux autres approches méta-heuristiques de la littérature. Les résultats obtenus montrent que notre approche permet d'atteindre la quasi-totalité des solutions trouvées avec la méthode exacte (sauf une instance), et qu'une nouvelle borne a été établie sur une autre instance. La comparaison avec les deux méta-heuristiques, a permis d'observer une amélioration sur la globalité des résultats de la première (algorithme génétique mono-objectif), et d'obtenir des résultats compétitifs avec la deuxième (recherche local itérative) sur les petites instances, accompagnée par une petite dégradation sur les grandes instances.

La troisième partie de cette thèse a été consacrée à l'étude d'une autre variante du VRP à savoir : le problème de tournées de véhicules sélectives (TOP). Nous avons proposé dans un premier temps une approche évolutionnaire multi-objectif hybride pour résoudre une formulation multi-objectif de ce problème, permettant d'optimiser simultanément le profit collecté et le coût du trajet. Les résultats obtenus sur

une série d'instances de l'état de l'art, ont d'abord permis de confirmer le caractère antagoniste des objectifs optimisés. La comparaison avec trois approches de la littérature, a permis de montrer une amélioration des résultats obtenus, et d'établir quatre nouvelles bornes. Dans la deuxième partie de l'étude de TOP, nous avons proposé une variante robuste de ce dernier (RTOP), qui a été résolue en adaptant l'algorithme utilisé pour la variante déterministe. Afin d'évaluer l'approche proposée, de nouvelles instances pour le TOP robuste ont été générées à partir de celles de l'état de l'art.

Mots clés : Problème de tournées de véhicules, incertitude, optimisation robuste, critère de robustesse, optimisation multi-objectif, algorithmes évolutionnaires.