POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Programmation primale en nombres entiers pour la résolution efficace d'un problème de tournées de véhicules riche : théorie et pratique

MAYSSOUN MESSAOUDI
Département de mathématiques et de génie industriel

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*
Mathématiques

Décembre 2020

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Cette thèse intitulée :

Programmation primale en nombres entiers pour la résolution efficace d'un problème de tournées de véhicules riche : théorie et pratique

présentée par **Mayssoun MESSAOUDI**
en vue de l'obtention du diplôme de *Philosophiæ Doctor*
a été dûment acceptée par le jury d'examen constitué de :

Michel GENDREAU, président
Louis-Martin ROUSSEAU, membre et directeur de recherche
Issmaïl EL HALLAOUI, membre et codirecteur de recherche
Nadia LAHRICHI, membre
Jacques RENAUD, membre externe

DÉDICACE

À l'homme qui m'a appris à nuancer entre
la Chose et la Valeur,
le Mot et la Parole, Vivre et Exister,
À ta belle âme, toujours imposante et présente, protectrice et bienveillante,
À l'homme de toutes mes vies,
À papa...J'espère que tu es fier de moi, là OÙ tu es ...

REMERCIEMENTS

Dur a été ce long parcours, marqué par des moments extrêmes oscillants entre le chagrin, le deuil, la joie et la gratitude. Des moments, où l'on se sent dépassé, englouti dans un gouffre profond. Mais une lueur n'a jamais cessé de jaillir ... À toutes les personnes qui m'ont entourée et soutenue : c'est à mon tour de vous remercier.

De prime abord, mes plus sincères remerciements pour mes directeurs de recherche Pr. Issmaïl El Hallaoui et Pr. Louis-martin Rousseau. Je vous exprime mon immense gratitude pour votre patience, votre professionnalisme et les bonnes valeurs que vous m'avez inculqués. C'est grâce à votre sens prononcé de l'humain et votre indéfectible soutien que j'ai pu mener ce projet jusqu'à sa fin.

Je remercie Pr. Anouar Jamali et Pr. Nizar El hachemi. Vous avez posé les premières briques et établi les premiers ponts. C'est grâce à vos recommandations que je suis arrivée là.

Je remercie la SNTL, en le nom de tout son Directoire. Je vous remercie pour votre collaboration et votre transparence qui nous ont permis de concilier le monde de la recherche et celui de l'industrie.

Aux honorables membres du Jury : merci d'avoir accepté d'évaluer et de critiquer mon travail.

Après cinq longues années passées entre les murs du GERAD et du CIRRELT, on finit par se sentir enfant issu d'une longue tradition d'apprentissage et d'enseignement. Je remercie cette grande famille à laquelle je suis fière d'appartenir.

Mes frères et sœurs au GERAD, avec qui j'ai partagé les rouages de ce long parcours, nous nous sommes soutenus dans les moments les plus et les moins heureux.

Adil, j'espère que tu trouveras dans ces lignes l'expression de ma gratitude. Tu as été un frère qui m'a soutenu dans les moments les plus critiques. Merci pour toutes les fins des semaines consacrées et les longues nuits de travail.

Safaa, Fatiha, Kaoutar, Asmaa... votre présence a rendu ce parcours plus facile... vous étiez ma seconde famille et vous le resterez. Mille mercis.

Linda et Johanne, merci pour votre énergie, le thé du soir et les méditations reposantes. Merci d'avoir pensé à moi à chaque occasion.

Ma grande et petite famille, mes adorables neveux et nièces : vous êtes mon monde et mon équilibre. Merci pour votre soutien et votre amour inconditionnels.

RÉSUMÉ

Concilier enseignement théorique et réalités professionnelles ; telle est la mission de la recherche opérationnelle. Résoudre une problématique complexe avec toutes les contraintes du monde réel est d'une grande difficulté, qui requiert une méthodologie de recherche rigoureuse doublée d'une perspective d'intervention toujours présente. Concrètement, les problèmes industriels sont beaucoup plus complexes dans la réalité que ce qu'ils sont en théorie. Une bonne solution réalisable au sens mathématique n'a pas la même signification en pratique. Un bon algorithme pour l'industrie doit réunir des capabilités telles la rapidité, la fiabilité, la flexibilité et la simplicité.

Le problème de tournées de véhicules (VRP : *vehicle routing problem)* constitue un sujet impérieux de la recherche opérationnelle. Il représente en effet la traduction mathématique d'innombrables applications de la vie courante. Nous nous emploierons à résoudre une problématique de transport prépondérante chez la majorité des prestataires de services logistiques qui, rappelons-le, sont des facilitateurs intégrés dont la mission est de gérer les opérations logistiques de plusieurs donneurs d'ordre. Ils planifient notamment la circulation d'une légion de commodités le long de réseaux logistiques très complexes. C'est dans ce contexte que les VRPs sont particulièrement riches en contraintes, et très difficiles à résoudre. Un défi qui a progressivement édifié notre projet.

Après une analyse approfondie des flux et une cartographie logistique, le maillon faible de la chaîne a été identifié, en l'occurrence la livraison du dernier kilomètre. Celle-ci implique la distribution d'une marchandise variée depuis un centre de distribution urbain vers une panoplie de clients finaux (grande distribution, détaillant, particulier, etc.). Jusqu'à ce jour, les efforts se sont plutôt concentrés à modéliser ces difficultés comme un VRP riche, incluant un large spectre de contraintes endogènes (capacité, véhicules hétérogènes, fenêtres de temps, etc.) et exogènes (règles de la profession, options de livraison, compatibilité, etc.). Toutefois, une solution ne peut être réellement implémentable que si elle est en concordance avec les configurations opérationnelles et managériales.

Les méthodes de résolution des VRPs sont scindées en des méthodes heuristiques, et des méthodes exactes. Les méthodes heuristiques parviennent généralement à réaliser un bon compromis entre la qualité et le coût de la solution, mais tout en exigeant un effort substantiel pour raffiner les valeurs des paramètres. En contrepartie, les méthodes exactes garantissent l'optimalité, mais peinent à résoudre les variantes riches du VRP. L'examen de l'état de l'art montre que, à l'encontre des méthodes exactes de *Branch-and-Price*, les méthodes primales

exactes sont très peu abordées. La présente thèse serait la première implémentation des méthodes primales exactes dans un contexte de VRP riche et réel.

Nous avons utilisé et adapté un algorithme de génération de colonnes en nombres entiers dit ICG (ICG : *integral column generation*) qui combine un algorithme primal dans un schéma de génération de colonnes. Composé de quatre modules séquentiels, chacun coopère pour trouver une solution entière optimale ou presque optimale, sans avoir recours aux méthodes traditionnelles tels le branchement ou l'ajout des coupes. Cette capabilité est l'un des facteurs clés de réussite, qui a permis la résolution efficace d'un problème réel et difficile.

Dans la première contribution, nous avons réussi une première implémentation de ICG dans le cadre d'un VRP réel avec fenêtres de temps, flotte hétérogène interne et externe, contraintes de compatibilité de sites et un éventail de règles de métiers. Le problème maître (PM) a été formulé comme un problème de partitionnement d'ensembles SPP (SPP : *set partitioning problem*), et le sous-problème (SP) a été modélisé comme un problème de plus court chemin avec contraintes de ressources, modélisé sur un graphe orienté cyclique. Dans ICG, le problème maître restreint (PMR) est résolu à l'aide de l'algorithme primal ISUD (ISUD : *integral simplex using decomposition*), tandis que le SP est résolu à l'aide de la programmation dynamique. ISUD décompose le PMR en deux sous-problèmes. Le problème réduit (PR) contient les colonnes compatibles avec la solution entière courante S, et cherche une solution entière améliorante. D'autre part, le problème complémentaire (PC) contient les colonnes incompatibles avec S et vise à trouver des directions de descente entières. Littéralement, une colonne est dite compatible avec S si elle peut s'écrire comme combinaison linéaire des colonnes de la solution, sinon elle est dite incompatible. Les résultats de l'expérimentation, conduite sur sept journées opérationnelles, ont fait surgir aussi bien le potentiel d'amélioration de la méthode, que la nécessité de répondre à des questions relevant du fondement théorique.

Dans la deuxième contribution, la performance de ICG a été hissée en opérant deux stratégies d'amélioration : (i) au niveau du SP, nous avons proposé une modélisation intelligente et réaliste du réseau de distribution cyclique. Ceci a été performé grâce à une analyse affûtée des caractéristiques des clients et de l'historique des livraisons. Cette amélioration a permis d'économiser une partie considérable du temps investi pour résoudre les SPs. (ii) D'autre part, nous avons évité d'utiliser le branchement en implémentant une stratégie dite de multiphase, qui consiste à résoudre le PC uniquement pour les colonnes qui se trouvent à une certaine distance de la solution courante. Dans la majorité des itérations, le PC trouve directement des directions qui mènent vers des solutions entières. Pour le reste, le module de recherche de voisinage permet souvent d'aboutir rapidement à une solution entière dans le voisinage de la solution courante. Finalement, vu son éventuelle influence sur la performance de l'algorithme,

il était crucial d'analyser, aussi bien théoriquement qu'empiriquement, la qualité de la solution duale que nous utilisons. La performance confirmée de l'algorithme a été démontrée sur 30 instances réelles d'un VRP riche, incluant jusqu'à 199 clients, et ce en comparaison avec une méthode duale fractionnaire de type *Branch-and-Price*. Tous les résultats ont fait l'objet de tests rigoureux et dont la réalisabilité a été testée et validée en pratique.

À travers cette thèse, notre objectif ultime est de proposer non seulement un algorithme efficace, mais également la stratégie propice pour le mettre en place dans la pratique. La troisième contribution se veut d'apporter une brique manquante à la méthodologie de résolution des VRPs dans le milieu pratique. Nous avons capitalisé notre expérience dans une étude de cas réelle, qui étudie la problématique de VRP sous la perspective d'un prestataire logistique intégré. La stratégie expliquée et les outils exposés constituent un guide utile pour mener à bien la résolution d'un VRP en pratique tout en mettant à jour un savoir-faire bienvenu auprès de la communauté des chercheurs et des étudiants.

Dans la dernière contribution, nous avons présenté une étude initiale portant sur deux pistes d'amélioration susceptibles de renforcer la performance de la méthode primale ICG. L'objectif ultime serait d'implémenter une technique efficace de telle sorte que la recherche des solutions entières se fasse dans des voisinages potentiels, et ce, sans pénaliser le temps total de calcul. Dans un premier temps, nous avons implémenté une première approche dite *ZOOM*, où le processus de recherche est guidé par une direction de descente fractionnaire. Les résultats numériques préliminaires ont clairement montré que *ZOOM* a considérablement amélioré la performance de ICG. Ceci nous a conduits à proposer une seconde approche, qui consiste à remplacer *ZOOM* par une heuristique. Dans un premier temps, nous avons testé la viabilité d'une heuristique hybride adaptative, dite H-ALNS (H-ALNS : *Hybrid Adaptive Large Neighbourhood Search*) sur nos instances. Les résultats obtenus ont révélé que l'heuristique n'est relativement performante que sur les petites instances. Ce constat nous a incités à proposer des dispositifs adéquats afin de réussir l'incorporation de H-ALNS à ICG. Ces techniques exploitent les informations de nature duale et primale pour décider des arcs à éliminer. En effet, les arcs coûteux, longs ou rarement empruntés seraient dynamiquement enlevés. L'implémentation d'une telle stratégie d'hybridation est une piste de recherche qui est en cours d'exploration.

Les résultats des contributions réalisées jusqu'ici témoignent de la viabilité de la piste de recherche explorée dans le cadre de ce projet de recherche. Avec ces aboutissements, nous avons droit de penser que la programmation primale en nombres entiers est un paradigme facilement transférable en pratique, qui mérite d'être amplement employé afin de résoudre efficacement des VRPs riches.

ABSTRACT

The mission of operational research is to reconcile theoretical knowledge and practical business realities. Solving a complex problem with real-world constraints is very difficult, requiring a rigorous research methodology coupled with an unfailing intervention perspective. In practice, industrial problems are much more complex than they are in theory. A good feasible solution in the mathematical sense is not the same in practice. A practical algorithm for industry must combine capabilities such as speed, reliability, adaptability and simplicity. The vehicle routing problem (VRP) is a compelling topic for operational research. It represents the mathematical description of countless applications in everyday life. We aim to solve a major transport problem arising within most logistics service providers. The latter have the mission to manage the logistics operations of several contractors. In particular, they schedule the routing of a huge number of commodities across extremely complex logistics networks. In this context, VRPs are particularly rich in constraints, and very difficult to tackle. Such a challenge has gradually built up our project.

After an extensive flow analysis and logistics mapping, the weak leg of the value chain was identified, namely the last mile delivery. This involves the distribution of a variety of goods from urban distribution centers to a range of end consumers (supermarkets, retailers, individuals, etc.). To date, efforts have tended to focus on modeling this problem as a rich VRP, encompassing a wide range of constraints such as time windows, heterogeneous fleet, business rules, delivery options, and compatibility. Nevertheless, a proper solution can be implemented only if it is in line both with operational and managerial configurations.

The methods for solving VRPs are split into heuristic methods and exact methods. Heuristic methods generally manage to achieve a good compromise between the quality and the cost of the solution, but require a substantial effort to refine the parameters values. On the other hand, exact methods guarantee optimality, but are struggling to solve the rich variants, and are restricted to tractable problems. Examination of the state of the art showed that, in contrast to dual fractional methods, very little attention has been devoted to exact primal methods. The present thesis would be the first successful deployment of exact primal methods in a rich and real VRP context.

We have used and adapted the integral column generation (ICG) algorithm, which combines a primal algorithm in a column generation scheme. Composed of four modules, each one cooperates to find an optimal or near-optimal integer solution, without using traditional methods such as branching or adding cuts. This capability is one of the key success factors,

which enabled the effective resolution of a difficult real-world problem.

In the first contribution, we successfully performed a first implementation of ICG in the context of a real VRP problem with time windows, internal and external heterogeneous fleet, site compatibility constraints and a range of business rules. The master problem was formulated as a set partitioning problem (SPP), and the sub-problem (SP) was modeled as a shortest path problem with resource constraints, modeled on a cyclic oriented graph. ICG is a sequential algorithm where the restricted master problem (RMP) is solved using the integral simplex using decomposition (ISUD) algorithm, while the SP is solved using dynamic programming. ISUD decomposes the RMP into two sub-problems. The complementary problem (CP) contains the incompatible columns, and aims at finding integer descent directions. On the other hand, the reduced problem (RP) deals with the compatible columns and looks for an improving integer solution. Literally, a column is said to be compatible with a solution S if it can be written as a linear combination of its columns, otherwise it is said to be incompatible. The computational results, conducted over seven operational days, highlighted both the potential for improvement of the method and the need to address theoretical underlying issues.

In the second contribution, ICG's performance was enhanced by operating two improvement strategies: (i) at the SP level, we proposed an intelligent and realistic modeling of the cyclic distribution network. This was achieved through a careful analysis of the customers' characteristics and the historical delivery records. This improvement saved a considerable amount of time invested in solving the SPs. On the other hand, (ii) we avoided using branching by implementing a so-called multiphase strategy, which consists in solving the CP only for columns which are at a specific distance from the current solution. In the majority of iterations, the CP directly found directions leading to integer solutions. For the rest, the Neighborhood Search module generally leads quickly to an integer solution in the neighborhood of the current solution. Finally, given its possible impact on the performance of the algorithm, it was crucial to analyze, both theoretically and empirically, the quality of the dual solution we have used. The well-proven performance of the algorithm was demonstrated on 30 real instances of a rich VRP, with up to 199 clients, in comparison with a Branch-and-Price method. All results have been carefully tested and practically validated for feasibility.

Through this thesis, our main purpose is to propose not only an efficient algorithm, but also the appropriate strategy to put it into practice. The third contribution aims to provide a methodology of solving VRPs in the practical environment. We have capitalized on our experience through a real-life case study, which explores the problem of VRP from the perspective of an integrated logistics provider. The strategy outlined and the described tools

provide a useful guide to successfully resolve a VRP in practice, while at the same time updating a know-how that is welcomed by the research community and its students.

In the last contribution, we presented an introductory study for further improvements to enhance the performance of the primal ICG method. The idea is to implement an efficient technique so that the search for integer solutions is conducted in potential neighborhoods, yet without penalizing the total computing time. As a first step, we have deployed a first technique called *ZOOM*, where the search process is guided by a fractional descent direction. Preliminary numerical results clearly showed that *ZOOM* has significantly improved ICG performance in terms of computing time. Next, we applied a hybrid adaptive heuristic called H-ALNS (H-ALNS: Hybrid Adaptive Large Neighbourhood Search). To begin with, we tested the viability of the heuristic on our instances. The results achieved then prompted us to consider a suitable hybridization strategy, offering a good compromise between the heuristic's properties and those of an exact primal method. This subject is still an ongoing research topic.

The results of the contributions made so far testify to the viability of the research topic that was explored in this thesis. Thus, we believe that integer primal programming paradigm is easily transferable in practice and that merits to be widely employed in solving rich VRPs models.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

3PL	Third-Party Logistics
B&B	Branch-and-Bound
B&P	Branch-and-Price
CBM	Cubic Meter
CG	Column Generation
CP	Complementary Problem
CPP	Crew Pairing Problem
DCA	Dynamic Constraints Aggregation
DH	Diving Heuristic
DO	Delivery Order
DP	Dynamic Programming
DW	Dantzig–Wolfe
FTL	Full Truckload
H-ALNS	Hybrid Adaptive Large Neighbourhood Search
ICG	Integral Column Generation
IDRO	Integrated Delivery Routing Optimizer
IPS	Improved Primal Simplex
ISUD	Integral Simplex Using Decomposition
LSP	Logistics Service Provider
LTL	Less Than Truckload
MIP	Mixed-Integer Programming
MP	Master Problem
OVRP	Open Vehicle Routing Problem
PO	Purchase Order
RMH	Restricted Master Heuristic
RMP	Restricted Master Problem
RP	Reduced Problem
SDVRP	Site-Dependent Vehicle Routing Problem
SP	Sub-Problem
SPPRC	Shortest Path Problem with Resource Constraints
TDVRP	Time-Dependent Vehicle Routing Problem
TSPP	Traveling Salesman Problem with Profit
VCSP	Vehicle and Crew Scheduling Problem

VRP	Vehicle Routing Problem
VRPPC	Vehicle Routing Problem with Private fleet and common Carrier
VRPTW	Vehicle Routing Problem with Time Windows

CHAPITRE 1 INTRODUCTION

Étymologiquement, le mot logistique est apparenté au mot grec *logistikos*, relatif à l'art du calcul et au raisonnement. D'ailleurs, Platon (428-348 av. J.-C.) a utilisé cette expression pour désigner le calcul pratique. Ce n'est pas surprenant alors que le transport et la logistique constituent des terrains d'essais de choix pour la communauté de la recherche opérationnelle.

De nos jours, la chaîne logistique s'est transformée en un arrimage intégré de maillons, dont la mission est d'optimiser les opérations physiques liées au traitement des flux de produits depuis le fournisseur jusqu'à leur mise à disposition du consommateur final, tout en répondant au triptyque coût-qualité-délai. La mondialisation s'est accompagnée d'une tension accrue des flux, une multiplication des intervenants, une hausse du coût des opérations et tout un lot de taxations écologiques. La fonction logistique s'est imposée alors comme l'arme concurrentielle par excellence. Sa gestion efficiente n'est assurée que si l'entreprise contrôle les coûts de ses opérations, maintient un taux de service optimal tout en s'adaptant à des contraintes de plus en plus fortes. Dans cette perspective, la prestation logistique est devenue un secteur économique à forte valeur ajoutée. Le prestataire logistique agit comme un intégrateur de services au sein d'une chaîne de valeur étendue, comptant une multitude d'intervenants de différentes classes avec des processus, des contraintes et des objectifs distincts. Toute la difficulté réside donc dans l'intégration des solutions d'optimisation adaptées et résilientes.

Le réseau logistique global est généralement articulé en trois sous-systèmes :

- Logistique en amont (*inbound*) : orchestrée par les fournisseurs, elle englobe toutes les opérations qui assurent l'approvisionnement en matières nécessaires à la fabrication d'un produit ou d'un service.
- Logistique interne : fait référence aux opérations de transformation et de production prises en charge par les fabricants.
- Logistique en aval (*outbound*) : se réfère à l'ensemble des activités opérées au niveau du système de distribution afin d'assurer que le produit final soit disponible auprès du consommateur.

Dans ce sillage, le transport constitue une opération fondamentale assurant le transfert et l'acheminement efficace des flux entre chacun de ces maillons ; et plus particulièrement, au niveau de la logistique en aval, où le transport connait plus de complexité. En effet, il passe d'une fonction support vers une fonction génératrice de profit, opérant pour satisfaire la demande du client final et pour assurer un taux de service satisfaisant. Les problématiques de transport appartiennent à la famille populaire des problèmes de tournées de véhicules (VRP :

vehicle routing problem) ; l'un des problèmes les plus abordés de la littérature. Le VRP est un problème classique qui a évolué à travers les époques ; depuis l'import-export au large de la méditerranée avec les phéniciens, en arrivant à la livraison via un véhicule autonome. L'objectif est de satisfaire les demandes des clients avec le moindre coût, en leur acheminant le bon produit au bon moment tout en respectant les contraintes imposées et la capacité de la commodité utilisée. Cette formulation simpliste ne peut toutefois résumer toute la complexité de ce problème NP-difficile ; une complexité qui s'enrichit d'aspects plus saillants quand on se place dans un contexte pratique.

Cette théorie s'est concrétisée en conduisant une partie de ce projet de recherche au sein de l'entreprise partenaire ; *SNTL Supply chain*. C'est la filiale logistique du premier opérateur national de la logistique et de transport au Maroc. Depuis 1937, la compagnie gère une chaîne de valeur diversifiée qui s'articule autour de la gestion de flotte de l'État, l'immobilier logistique, les assurances et le ravitaillement militaire. Quant à notre partenaire, il exploite sa propre plateforme logistique multi flux, étalée sur une surface de 25 hectares d'entrepôts et de zones de services logistiques à valeurs ajoutées.

In fine, le déclencheur majeur de nos contributions est de répondre aux besoins de ce secteur vital et complexe en proposant de nouvelles méthodes et des techniques capables de produire des solutions optimisées, efficaces et fidèles à la réalité.

Après avoir introduit le contexte global de notre contribution, nous relatons dans ce qui suit les éléments de la problématique à laquelle nous sommes confrontés, ainsi que le déploiement de nos objectifs de recherche.

1.1 Définitions et concepts de base

Pour faciliter l'assimilation de ce document, nous employons la terminologie suivante :

- *3PL : Third-Party Logistics* est un prestataire de services qui a la responsabilité d'exécuter une partie ou la globalité de la logistique de ses clients comme l'entreposage, la gestion des stocks et la livraison ;
- *Taux de service :* c'est un indicateur qui mesure le pourcentage de commandes livrées dans le respect des exigences exprimées par le client ;
- *Dernier kilomètre :* dernier segment de la chaîne logistique qui consiste à acheminer un bien ou un service jusqu'au client final ;
- *Client final :* le dernier client d'un sous-système de la chaîne logistique ; c'est un particulier ou un point de vente ;

— *Sous-traitance :* ou externalisation, consiste à déléguer la gestion d'une partie ou la globalité des activités à une partie tierce ;

— *Messagerie :* métier logistique consistant à transporter des colis (dont le poids total est inférieur à trois tonnes) dans un délai rapide ; généralement inférieur à 48h ;

— *Business rules :* l'ensemble des règles de gestion régulant un métier ou une activité ;

— *Full Truckload (FTL) :* fait référence au chargement complet d'un véhicule, avec une expédition directe depuis l'origine à la destination, sans rupture de charge ;

— *Less than Truckload (LTL) :* fait référence à un véhicule partagé par plusieurs clients, qui effectue plusieurs arrêts afin de livrer les différentes destinations.

Un 3PL exploite une plateforme logistique centrale, où les flux multiples sont réceptionnés, consolidés, stockés et expédiés vers leurs destinations finales. Il a un large éventail de donneurs d'ordres (contractants), dont chacun a son propre réseau de clients. Simplement formulé, la demande est déclenchée par le client final, le contractant consolide ces demandes et les transfère au 3PL. Celui-ci doit opérer pour effectuer des livraisons qui respectent les exigences et les termes du contrat commercial qui le lie avec son contractant, faute de quoi, des pénalités strictes sont imposées. Les clients finaux sont répartis à l'échelle nationale, ils sont desservis par une large flotte hétérogène. Le prestataire réalise un gain s'il arrive à satisfaire les commandes journalières des clients tout en respectant la capacité et la compatibilité des véhicules, les fenêtres de temps et les *business rules* régulant les différents métiers.

Après avoir introduit les notions de base utilisées dans le domaine de la logistique et transport, et après avoir défini le contexte général d'un prestataire logistique, nous allons exposer les problématiques majeures inhérentes.

1.2 Éléments de la problématique

Si la complexité combinatoire des VRPs a été théoriquement démontrée dans la littérature de la recherche opérationnelle (RO), l'objectif ici est de mettre en exergue les facettes qui rendent leur résolution un véritable défi en pratique. Ceci constitue par ailleurs le déclencheur de notre projet de recherche.

1.2.1 Dimensions d'un VRP riche en pratique

La richesse d'un modèle de VRP est dictée par le nombre et la nature des contraintes qu'il doit satisfaire. Selon la classification générique de Toth et Vigo (2014), les attributs caractérisent les éléments du VRP qui affectent le réseau de transport, les clients, les demandes,

la flotte de véhicules et la fonction objectif. Plus un modèle inclut d'attributs, plus il a la chance de modéliser la configuration logistique réelle, et plus sa résolution devient difficile et problématique.

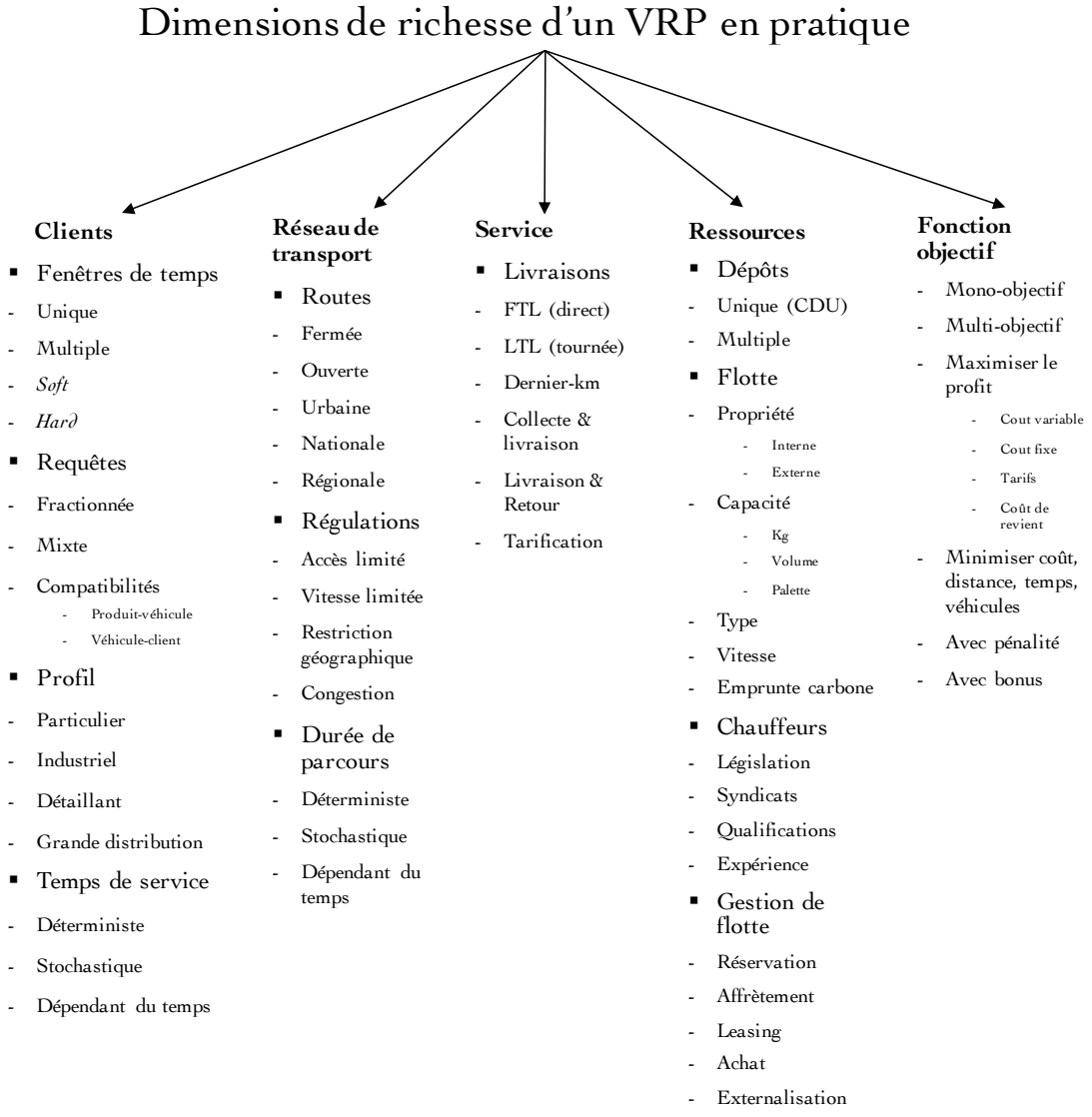


Figure 1.1 Dimensions de richesse d'un VRP en pratique

Dans la Figure 1.1, nous avons essayé de recenser les principales caractéristiques des VRPs relevant de plusieurs niveaux. Un modèle qui modélise le problème de transport chez un 3PL par exemple, est l'incarnation d'une combinaison horizontale et verticale des éléments énumérés ci-haut. La résolution d'un tel modèle, avec une méthode exacte ou même heuristique,

est non envisageable. Dans la plupart des configurations pratiques, il faut passer par une étape d'analyse et d'élicitation afin de prioriser les contraintes ayant le plus fort impact sur la performance opérationnelle et financière, et obtenir par la suite une formulation mathématique traitable.

1.2.2 Attributs du VRP chez un 3PL

Résoudre un VRP en pratique implique la considération des contraintes et des aspects problématiques réels. Ceux-ci apparaissent sous différentes facettes :

Fenêtres de temps. Le client doit être visité uniquement durant une plage horaire qu'il choisit. Dépendamment du secteur d'activité du client, plusieurs options se présentent. Les fenêtres de temps peuvent être : (i) multiples, signifiant que plusieurs plages horaires sont disponibles (e.g., [12h, 14h] ou [16h,18h]) ; (ii) strictes, signifiant que le respect de l'horaire choisi par le client est impératif ; et finalement (iii) *soft*, dans le sens où la contrainte peut être violée moyennant un coût léger.

Flotte de véhicules. Dans le souci de répondre aux besoins des clients, le service de transport s'appuie sur un parc roulant de différents types de véhicules. La flotte se compose de véhicules propres à l'entreprise et de véhicules externes affrétés au besoin. Ainsi, la flotte est hétérogène en termes du type (caractéristiques techniques), de capacité (charge nominale et poids), des coûts (fixes et variables) et de type de livraison (FTL, LTL, dernier kilomètre). Plus encore, pour satisfaire les demandes urgentes, les retours ou les reliquats, il est possible de sous-traiter certaines livraisons par un partenaire de messagerie.

Fonction du coût. Pour mesurer sa performance, le 3PL doit être en mesure d'évaluer son profit ; la différence entre le revenu de la vente du transport et le coût encouru pour son opération. Or en pratique, les logisticiens ne maitrisent pas leurs coûts et une modélisation structurée avec des composantes fixes et variables est souvent inexistante. Cette difficulté de modélisation est accentuée par la présence d'une flotte hétérogène et des unités de facturation distinctes.

Service. Le taux de service est un indicateur phare qui permet de mesurer le niveau de satisfaction de la demande. Cet indicateur de performance est suivi au quotidien et son infraction entraine des pénalités ainsi que des sanctions explicitées dans le contrat commercial.

Flux tirés. Les prestataires logistiques opèrent tous sous un système à flux tirés ce qui signifie que les livraisons ne sont déclenchées que suite à la demande des clients. Ces derniers exigent de plus en plus des livraisons fréquentes, fragmentées et personnalisées. La réactivité, la flexibilité et l'efficience sont alors les compétences primordiales d'un prestataire de qualité.

En plus des attributs que nous venons d'énumérer, la livraison du dernier kilomètre vient accentuer la complexité des opérations. Cette étape ultime de la chaîne de distribution représente le maillon le plus difficile, le plus couteux et le moins efficient de toute la chaîne. En effet, les clients sont situés dans des zones métropolitaines difficiles d'accès ; en plus, les livraisons sont très fragmentées et fréquentes. Une mauvaise planification et des données inexactes apparaissent comme les catalyseurs de plusieurs incidents, tels des accidents, des infractions et des retards.

1.2.3 Défis de résolution

Pour pouvoir accommoder les contraintes difficiles liées à l'organisation du transport, le VRP a recours à une formulation de partitionnement d'ensembles SPP (SPP : *set partitioning problem*), où le sous-problème apparaît comme un problème de plus court chemin avec contraintes de ressources, modélisé sur un graphe orienté cyclique et résolu moyennant un algorithme de programmation dynamique. Malgré les avantages qu'offre cette formulation compacte et flexible, deux problèmes majeurs surviennent : (i) les SPPs de grande taille souffrent de la dégénérescence sévère engendrée par des variables nulles dans la base. Ceci affecte largement la méthode de génération de colonnes classique ce qui se traduit par un grand nombre d'itérations. Plus encore, éviter la dégénérescence peut causer une expansion rapide de l'arbre de branchement. (ii) Le réseau de distribution est complexe et contient des cycles, en plus, gérer toutes les contraintes pratiques fait souvent appel à un nombre conséquent de ressources. Ainsi, la résolution d'un tel sous-problème nécessite-t-elle une méthode performante. Finalement, nous soulignons que le milieu industriel requiert un algorithme efficace, flexible, manipulant un nombre restreint de paramètres et avec la capacité de garantir des solutions réalisables à tout moment de la résolution.

À travers ce chapitre, nous avons exposé et analysé des problématiques fondamentales auxquelles fait face un prestataire logistique au quotidien. Notre contribution porte sur l'optimisation du transport chez un prestataire logistique intégré, avec la motivation de proposer des modèles de recherche opérationnelle capables de tenir compte de la facette réaliste et complexe des opérations.

1.3 Objectifs de recherche

L'objectif sommaire est de résoudre un VRP riche par le biais d'une approche générique, flexible et agile capable de traiter efficacement des instances difficiles et réelles. Pour ce faire, nous faisons appel à la programmation primale en nombres entiers (PPNE). À notre connaissance, cet ouvrage est le premier travail qui s'intéresse à étudier le potentiel des approches primales dans la résolution des VRPs riches.

Dans un premier lieu, nous avons résolu un VRP riche en utilisant, pour la première fois, un algorithme issu de la PPNE. Il s'agit de ICG ; un algorithme primal intégré dans un cadre de génération de colonnes. Le problème maître a été formulé comme un SPP, tandis que le sous-problème a apparu comme un problème de plus court chemin avec contraintes de ressources. Les résultats concluants de notre premier objectif nous ont poussé à intensifier la recherche dans les objectifs de : (i) propulser la performance de ICG, (ii) formaliser les évidences théoriques attestant son efficacité, (iii) affiner la modélisation du PCCCR et (iv) étendre le banc des expériences sur 30 instances réelles atteignant 199 clients. Finalement, nous avons matérialisé notre expérience sous la forme d'une étude de cas orientée afin de montrer les étapes cruciales dont passe l'implémentation d'un VRP en milieu pratique, en partant de la modélisation, la correction des données, jusqu'à la conception de l'architecture de l'outil. Finalement, l'objectif de notre dernier chapitre consistait à identifier l'intérêt et la viabilité de deux approches d'amélioration de ICG. Chacune de ces propositions a été décrite et expérimentée. Nous avons présenté les résultats préliminaires obtenus jusqu'à présent sur des instances réelles d'un VRP riche.

1.4 Plan de la thèse

Le plan de cette thèse est structuré comme suit. Le chapitre 2 est une revue de littérature où nous étudions les problèmes de tournées de véhicules, leurs classes ainsi que leur principales méthodes de résolution. Le chapitre 3 donne un aperçu sur l'organisation des axes principaux de cette thèse.

Le chapitre 4 présente le premier objectif de la thèse qui s'intéresse à la résolution d'un problème de VRP multi-attributs. Cet article de conférence a fait l'objet d'une publication dans *Combinatorial Optimization. ISCO 2020. Lecture Notes in Computer Science.*

Le chapitre 5 est consacré au deuxième objectif portant sur la résolution d'un problème de livraison du dernier kilomètre, où le transporteur utilise une flotte externe et sous-traite une partie de ses livraisons par un partenaire de messagerie. Ce deuxième objectif a fait l'objet d'un article soumis à la revue *OMEGA - The International Journal of Management Science.*

Le chapitre 6 est une étude de cas portant sur l'implémentation de l'algorithme en milieu pratique dans le contexte spécifique d'un 3PL opérant dans une économie émergente. Ce troisième objectif a fait l'objet d'un article soumis à la revue *INFORMS Journal on Applied Analytics*.

Le chapitre 7 examine le potentiel d'amélioration de la version de base de ICG, par le biais de deux approches distinctes. Nous présentons un aperçu de chacune de ces méthodes, ainsi que les résultats numériques préliminaires.

Le chapitre 8 entame une discussion globale synthétisant nos contributions. Finalement, les conclusions et les futures pistes de recherche clôturent cet ouvrage ; elles sont présentées dans le chapitre 9.

CHAPITRE 2 REVUE DE LITTÉRATURE

L'objectif de ce chapitre est de situer notre problématique et nos contributions en passant en revue les recherches adjacentes. En premier lieu, nous étudions le problème de tournées de véhicules et les approches primordiales proposées par la littérature pour résoudre ses différentes variantes. D'abord, nous abordons les méthodes de résolution exactes basées sur la génération de colonnes. Ensuite, nous donnons un aperçu sur les principales méthodes de résolution heuristiques ayant été employées dans le cas des VRPs. Puisque les méthodes primales constituent une pièce maîtresse de notre contribution, nous accordons une section pour en donner le principe et parcourir les algorithmes notables. Une synthèse achève cette étude bibliographique.

2.1 Problème de tournées de véhicules

Les problèmes de tournées de véhicules VRP (VRP : *vehicle routing problem*) tiennent leur popularité de leurs applications innombrables dans différents secteurs de transport, logistique, santé, urbanisme, télécommunications et plus encore. Fondamentalement décrit par Dantzig et Ramser (1959) et généralisé cinq ans plus tard par Clarke et Wright (1964), le VRP compte parmi les problèmes prépondérants de la recherche opérationnelle, et fait l'objet d'une littérature intensive. Parcourir toute la littérature est un travail onéreux, nous nous contenterons de relater les aspects relevant de notre périmètre de contribution. Nombreux sont les travaux de revue et de taxonomie sur les VRPs, nous conférons le lecteur à quelques contributions majeures relativement anciennes (Irnich et al., 2014; Braekers et al., 2015; Koç et al., 2016) et toutes récentes (Koç et al., 2020; Konstantakopoulos et al., 2020; Widuch, 2020).

2.1.1 Aperçu global

Le VRP classique est un problème combinatoire NP-difficile (Lenstra et Kan, 1981). Il est généralement défini sur un graphe orienté $G = (V, A)$. L'ensemble V contient $|N| + 2|D|$ sommets, un sommet pour chaque client $i \in N$ dont chacun a une demande d_i, et $|D|$ sommets pour représenter les dépôts. Une paire (s_i, t_i) pour chaque dépôt $i \in D$, où un ensemble K de véhicules homogènes de capacité Q sont disponibles. L'ensemble A contient des arcs de départ (s_i, j) $\forall i \in D$, $\forall j \in N$, des arcs d'arrivée (i, t_j) $\forall i \in N$, $\forall j \in D$ et des arcs d'inter-parcours (i, j) $\forall (i, j) \in N \times N$ de telle sorte que le client j puisse être visité immédiatement après le

client i dans une route réalisable. Dans sa version de base, le CVRP (CVRP : *Capacitated Vehicle Routing Problem*) consiste à trouver, pour chaque véhicule $k \in K$, une route sur G en s'assurant que : (i) chaque client soit visité une seule fois, (ii) le total des demandes transportées par le véhicule k ne dépasse pas la capacité Q de celui-ci et (iii) la route de chaque véhicule k commence et finisse au même dépôt.

Cette version de base est l'archétype de toutes les applications, mais ce n'est jamais assez pour les modéliser proprement, car des contraintes intrinsèques additionnelles s'y imposent. Les problème VRPs deviennent de plus en plus riches et complexes en intégrant des dimensions issues de la réalité (Coelho et al., 2015). Cette complexité apparente justifie l'engouement de la littérature envers les méthodes de résolution heuristiques au détriment des méthodes exactes. En pratique, ces dernières peinent à résoudre les problèmes de grande taille en un temps raisonnable, à l'exception de quelques approches généralement basées sur la génération de colonnes (GC) et la décomposition de Dantzig–Wolfe (DW).

2.1.2 Classes de VRP

Pour se rapprocher des problématiques réelles, la littérature a proposé un très grand nombre de classes de VRP, qui ont été largement étudiées, répertoriées et recensées dans plusieurs travaux (Irnich et al., 2014; Toth et Vigo, 2014). Dans ce qui suit, nous présentons les variantes les plus souvent rencontrées en milieu pratique, notamment dans les métiers de la *supply chain*.

VRP avec fenêtres de temps. Le VRPTW (VRPTW : *VRP with time windows*) a été introduit par Solomon (1987) dans le but de planifier des visites durant les fenêtres de temps imposées par les clients, celles-ci sont soit flexibles (*soft*) ou strictes (*hard*). Une description plus détaillée de cette classe populaire figure dans plusieurs travaux (Taillard et al., 1997; Vidal et al., 2013a). Les premières méthodes exactes avaient fondamentalement recours à l'algorithme *Branch-and-Bound*, puis se sont étendues vers des algorithmes plus performants, généralement basés sur la génération de colonnes ou la relaxation lagrangienne (Cordeau et al., 2001a). La contribution notable de Desrochers et al. (1992) a résolu le VRPTW comme suit : la relaxation linéaire de la formulation de partitionnement d'ensembles a été résolue par l'approche de génération de colonnes. Le sous-problème a été modélisé comme un problème du plus court chemin avec des fenêtres de temps et des contraintes de capacité. L'algorithme s'est avéré efficace sur une variété des instances de référence de taille pratique ; il a résolu de manière optimale des problèmes de 100 clients. Parmi les approches exactes notables, nous citons *Branch-and-Price-and-Cut* introduite par Kohl et al. (1999). Cette méthode est basée

sur la combinaison de *Branch-and-Bound* avec la génération de colonnes et les plans coupants. En effet, la relaxation linéaire du problème-maître est résolue par la méthode de génération de colonnes, alors que le sous-problème est un plus court chemin élémentaire avec contraintes de ressources (Feillet et al., 2004).
Dans certaines situations pratiques, il est possible de relâcher la contrainte des fenêtres de temps. Ceci permet d'effectuer une livraison hors la plage horaire imposée par le client. On parle alors du VRP avec fenêtres de temps souples (VRPSTW : *VRP with soft time windows*). Koskosidis et al. (1992) étaient les premiers à résoudre cette variante moyennant une approche heuristique. Leur procédure a été élaborée à partir de trois étapes fondamentales. Tout d'abord, les contraintes temporelles sont relâchées dans la fonction objectif. Ensuite, la complexité de la fonction objectif résultante est contournée à l'aide d'un modèle d'approximation approprié. Enfin, l'ensemble du processus est exécuté de manière itérative pour améliorer la solution. Par ailleurs, Taillard et al. (1997) on utilisé la recherche tabou pour résoudre le VRPSTW. Ils ont montré qu'en induisant des pénalités de violation au niveau de l'objectif, le problème est équivalent à celui avec fenêtres de temps strictes. Nous notons finalement l'existence de certaines méthodes unifiées qui ont réussi à résoudre plusieurs variantes du VRPTW comme l'algorithme hybride introduit par Vidal et al. (2013a) et l'algorithme de recherche tabou unifiée de Cordeau et al. (2001b).

VRP avec routes ouvertes. L'OVRP (OVRP : *open VRP*) est un problème de plus en plus fréquent en pratique, en particulier dans l'industrie de la prestation logistique où les compagnies font appel à des transporteurs tiers pour supporter leurs activités. C'est le cas également chez certains chargeurs industriels qui ont choisi d'externaliser une partie des opérations vers des parties tierces. Cette variante omet la contrainte imposée sur le retour de tous les véhicules au dépôt. En général, deux objectifs d'optimisation sont considérés : minimiser le nombre de véhicules, ensuite minimiser le coût (distance ou temps) de parcours (Zachariadis et Kiranoudis, 2010). Dans les articles classifiés par Braekers et al. (2015), les auteurs traitent généralement des problèmes avec flotte homogène, et emploient des approches métaheuristiques. Notons que tout OVRP peut également être formulé sous la forme d'un VRP sur un graphe orienté, en fixant à 0 le coût des arcs entrant au dépôt, toutefois, si le graphe est non orienté, une telle transformation n'est pas valide (Letchford et al., 2007).

VRP avec flotte interne et transporteur externe. Le VRPPC (VRPPC : *VRP with private fleet and common carrier*) a été anciennement introduit par Ball et al. (1983), mais il est peu abordé dans la littérature. Toujours est-il que c'est une classe qui modélise le mieux la réalité des opérations logistiques. En effet, considérant la fluctuation de la demande, les frais

d'investissement, la hausse du prix du baril, et le facteur du risque, le prestataire choisit d'être en possession d'une partie de sa flotte, et d'affréter, au besoin, d'autres véhicules provenant de transporteurs externes. Parfois, il s'agit également d'un affrètement du service de livraison, ce qui signifie qu'une partie des opérations soit complètement planifiée et opérée par une partie externe. Il s'agit d'un scénario très répandu dans l'industrie du fret, qui surgit plus particulièrement dans la distribution urbaine du dernier kilomètre ; le maillon difficile de toute la chaîne logistique. Cette configuration est à l'origine de plusieurs problématiques décisionnelles lorsqu'il est question de déterminer quand externaliser une livraison, et quand et combien affréter de véhicules. Les méthodes de résolution heuristiques qui ont été proposées dans la littérature reposent essentiellement sur des méthodes de construction et amélioration, recherche tabou, recherche locale itérative, etc. (Bolduc et al., 2008, 2007; Côté et Potvin, 2009; Potvin et Naud, 2011; Stenger et al., 2013; Euchi, 2017). Récemment, Dabia et al. (2019) ont introduit une formulation exacte basée sur *Branch-and-Price-and-Cut*, en prêtant un intérêt particulier à la modélisation de la fonction du coût qui inclut les prix de vente, les tarifs et les coûts d'affrètement. Son modèle a été testé sur des instances de la littérature avec 100 clients. Bien que cette variante soit une illustration appropriée du contexte de la *supply chain*, ce domaine est encore largement inexploré, et invite les chercheurs à proposer des solutions plus efficaces afin de traiter les contextes contraints et complexes.

VRP avec contraintes de site. Dans le SDVRP (SDVRP : *site-dependent VRP*), les contraintes de compatibilité font référence à la situation où chaque client a besoin d'un type de véhicule particulier pour être servi. La nécessité de modéliser de telles contraintes s'impose en pratique, dans le transport urbain en particulier, où les clients sont situés dans des zones résidentielles à accès limité, ou bien dont les quais de service ne peuvent recevoir qu'un type particulier de véhicules. Parfois, même les commandes peuvent nécessiter un équipement spécial pour les manutentionner. La plupart des problèmes de taille réelle ne peuvent être résolus dans un délai raisonnable par les approches exactes existantes. En revanche, les algorithmes heuristiques, principalement basés sur la recherche tabou, peuvent trouver des solutions de bonne qualité en un temps raisonnable (Cordeau et Laporte, 2001; Chao, 2002; Chao et Liou, 2005; Ropke et Pisinger, 2006). La stratégie répandue dans le cadre du SDVRP consiste à le résoudre en deux étapes. D'abord, les contraintes de compatibilité sont intégrées en résolvant un problème d'affectation afin de grouper les clients necessitant les mêmes véhicules. Ensuite, un CVRP est résolu pour construire des routes de coût minimal. Là encore, on a décrit une classe de VRP présente en pratique mais pas suffisamment étudiée par les travaux de recherche connexes. Enfin, nous rappelons que dans des applications comme le transport et la logistique, les contraintes de site se manifestent sous d'autres aspects : produit-véhicule

(sécurité), chauffeur-véhicule (compétence), chauffeur-client (règles du métier).

VRP avec profit. Ce problème propose de sélectionner les clients à visiter selon le profit total collecté par les visites. Il s'agit d'une généralisation du TSPP (TSPP : *traveling salesman problems with profit*). Deux objectifs sont considérés dans le TSPP : (i) maximiser le profit total collecté, et (ii) minimiser les coûts (Feillet et al., 2005). La combinaison de ces deux objectifs a donné naissance à trois problèmes génériques bien connus : (i) *Profitable Tour Problem* où l'objectif est de maximiser la différence entre le profit total collecté et le coût de la distance totale parcourue (Dell'Amico et al., 1995), (ii) *Orienteering Problem* où l'objectif est de sélectionner un ensemble de clients potentiels maximisant le profit avec une limitation sur le coût (Golden et al., 1987; Laporte et Martello, 1990) et (iii) *Prize Collecting TSP* où l'objectif est de sélectionner un ensemble de clients potentiels minimisant la distance totale du parcours et tout en s'assurant que le seuil du profit soit réalisé (Balas, 1989). Nombreuses sont les méthodes heuristiques recensées dans la littérature qui sont basées généralement sur recherche locale, recherche tabou, algorithmes génétiques et notamment des méthodes hybrides. Dans une application intéressante, Li et Lu (2014) ont développé un modèle d'aide à la décision au profit d'un chargeur industriel afin de décider des activités à assurer par flotte propre et celles à sous-traiter. Les auteurs font appel à un algorithme génétique hybride afin de pouvoir traiter des instances réelles. Boussier et al. (2007); Butt et Ryan (1999) quant à eux, ont proposé une méthode exacte de génération de colonnes qui a été expérimentée sur des instances incluant 100 clients.

VRP dépendant du temps. Si la durée du trajet est considérée comme étant constante dans les problèmes classiques du VRP, la réalité contredit parfois cette hypothèse, puisque ce paramètre dépend de plusieurs facteurs comme la congestion, la variation de la vitesse, les règles de gestion chez certains clients, etc. Les TDVRPs (TDVRP : *time-dependent VRP*) sont généralement caractérisés par un temps de parcours qui varie en fonction des périodes constituant un horizon défini, et par des objectifs qui peuvent être linéaires par morceaux ou convexes en fonction des temps de service aux clients (Ibaraki et al., 2005, 2008). Une panoplie de contraintes temporelles sur les routes ont été introduites. Elles sont intégrées au niveau d'un problème du *timing* qui consiste à déterminer les heures de départ et d'arrivée chez les clients. À notre connaissance, aucune méthode exacte n'a été développée jusqu'à présent pour résoudre les VRPs avec des durées de parcours ou bien des temps de services aux clients dépendants du temps. Excluons Soler et al. (2009), qui ont proposé une démarche particulière qui consiste à transformer un TDVRP avec fenêtres de temps à un VRP asymétrique résolu à l'optimalité, mais la taille des données traitées n'était pas réaliste. Les durées de parcours

dépendants du temps sont souvent modélisées selon la formulation de Ichoua et al. (2003), basée sur une décomposition d'un horizon opérationnel en des périodes, auxquelles on affecte un profil de vitesse constante. La discrétisation temporelle de l'horizon est importante, dans le sens où le modèle serait plus réaliste avec des intervalles serrés.
Tout récemment, Gmira et al. (2020) ont apporté une contribution notable aux versions connues du TDVRP dans le sens où les variations dans le temps sont basées sur le graphe routier original. En effet, leur approche consiste à travailler sur le réseau routier et à prendre en compte les variations de temps de parcours sur chaque segment de route. Le problème a été résolu moyennant une heuristique de recherche tabou qui prend en compte les plus courts chemins entre deux clients, et ce à différents moments de la journée. La performance de l'algorithme a été évaluée et démontrée sur un ensemble d'instances de référence.
Nous sommes en présence d'une variante qui est assez bien étudiée dans la théorie, profitant des modèles mathématiques rationnels et robustes, cependant, elle est très difficile à adopter en pratique. En fait, la précision et la disponibilité des données sont des conditions sine qua non de leur succès. Avec l'essor des systèmes intelligents de transport, connus sous le nom de la télématique embarquée, les outils de traçabilité et de calcul en temps réel devraient faciliter leur transfert en industrie.

2.1.3 VRPs commerciaux

Des challenges relevant de la mobilité urbaine, la congestion des routes, la hausse des prix, les règles législatives, etc. rendent la planification du transport une opération encore plus complexe et onéreuse, nécessitant un capital humain, des ressources technologiques adéquates et un logiciel de planification approprié (Crainic et al., 2009). Ainsi, automatiser la planification des livraisons est-il non pas un choix, mais une nécessité pour les compagnies de transport et de logistique. Ceux-ci investissent de plus en plus pour s'équiper en logiciels de planification de transport ou bien des VRPs commerciaux (VRPS) (VRPS : VRP *software*). Une étude conduite par *Gartner* (TEM Guide, 2020) a révélé que l'adoption de ces systèmes est entravée par une panoplie de facteurs. En effet, les industriels trouvent que ces systèmes sont très coûteux et difficiles à exploiter par les opérateurs. Plus encore, les solutions présentent un risque d'obsolescence suite à l'intégration de nouveaux clients ou secteurs d'activités. Dans leur étude sur les logiciels de VRP, Bräysy et Hasle (2014) rapportent que la majorité des systèmes assurent les fonctionnalités de base, comme la planification journalière ou hebdomadaire, la ré-optimisation suite à un incident, l'édition des rapports statistiques, etc. Néanmoins, des considérations telles la congestion des routes et les restrictions d'accès ne sont pas considérées altérant ainsi la fluidité et l'efficience des opérations. Les auteurs des principales revues sur les VRPS (Drexl, 2012; Rincon-Garcia et al., 2018) partagent le même constat : les méthodes

de résolution sont principalement basées sur un large ensemble d'heuristiques, ayant subi de longues périodes d'affinage et de calibrage, ce qui va à l'encontre des méthodes rationnelles. Les algorithmes implémentés sont issus des modèles publiés dans la littérature, ou bien des méthodes développées en interne. Les développeurs n'offrent aucune donnée sur la nature des benchmarks employés pour tester la fiabilité de leurs systèmes. Cependant, les mêmes études révèlent qu'il s'agit en fait des instances tirées de la littérature.

2.2 Méthodes duales

Les approches ayant été employées dans la résolution des VRPs sont essentiellement issues des méthodes exactes ou bien heuristiques. D'après Letchford et Lodi (2002), les méthodes de résolution exactes se répartissent en deux classes principales : (i) les méthodes duales fractionnaires, et (ii) les méthodes primales. Ces deux approches procèdent selon deux mécanismes différents. Les méthodes duales maintiennent itérativement l'optimalité de la fonction objectif et la faisabilité des contraintes linéaires jusqu'à ce que l'intégralité soit atteinte. Dans cette classe, l'algorithme de *Branch-and-Price* jouit d'une grande popularité, et dans la littérature du VRP en particulier. Il est basé sur la GC combinée avec *Branch-and-Bound*. Tout de même, il convient de mentionner que les méthodes traditionnelles de la GC ne sont pas des méthodes duales au sens strict du terme. En effet, elles procèdent selon une restriction au niveau des solutions examinées (approche primale), et aussi selon une relaxation des contraintes d'intégralité (approche duale). Les méthodes primales maintiennent à la fois la faisabilité et l'intégralité et s'arrêtent lorsque l'optimalité est atteinte. Cette classe n'est pas assez abordée dans la littérature, d'où le besoin de présenter les notions clés et les avancées majeures dans ce domaine.

2.2.1 Décomposition de Dantzig–Wolfe

Si la génération de colonnes figure parmi les méthodes exactes notables, ceci revient en premier lieu à la décomposition de Dantzig–Wolfe (DW). Cet algorithme permet de résoudre des problèmes de programmation linéaire de grande taille ayant une structure spéciale. Ainsi, est-il judicieux d'introduire d'abord les bases de cette décomposition afin de bien cerner les méthodes de résolution qui en découlent.

Bien que la méthode de DW puisse être sollicitée dans le cas d'un problème non linéaire, elle est généralement utilisée pour résoudre les problèmes linéaires en nombres entiers (PNE) de grande taille et dont les contraintes peuvent être scindées en deux sous-ensembles : (i) contraintes faciles et (ii) contraintes liantes, telles qu'en omettant ces dernières le problème

devient plus facile à résoudre. Les contraintes sont dites faciles dans le sens qu'elles sont séparables en sous-matrices indépendantes, pourtant ce sont des contraintes qui peuvent être difficiles produisant un grand gap d'intégralité.

Afin d'illustrer cette approche, considérons le PNE suivant :

$$
\begin{aligned}
(PNE): \quad \min_{x} \quad & \boldsymbol{c}^{\top}\boldsymbol{x} && (2.1)\\
\text{s.à}: \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} && (2.2)\\
& \boldsymbol{B}\boldsymbol{x} = \boldsymbol{e} && (2.3)\\
& \boldsymbol{x} \in \{0,1\}^{n} && (2.4)
\end{aligned}
$$

Avec $\boldsymbol{A} \in \mathbb{Z}^{m\times n}$, $\boldsymbol{B} \in \mathbb{Z}^{k\times n}$, $\boldsymbol{c} \in \mathbb{R}^{n}$, $\boldsymbol{b} \in \mathbb{Z}^{m}$, $\boldsymbol{e} \in \mathbb{Z}^{k}$, et $\boldsymbol{x}$ est le vecteur de solution. Soit $\mathcal{X}$ le domaine réalisable combinant les contraintes faciles et les contraintes d'intégralité sur les variables x tel que $\mathcal{X} = \{\boldsymbol{x} |\ \boldsymbol{B}\boldsymbol{x} = \boldsymbol{e};\ \boldsymbol{x} \in \{0,1\}^{n}\}$. Par souci de simplicité, nous supposons que $\mathcal{X}$ est borné et non vide, et notons par $conv(\mathcal{X})$ son enveloppe convexe.
Quand le problème possède cette structure particulière, il est alors propice à la décomposition de DW, dont l'idée est de reformuler le PNE original (2.1)-(2.4) en un problème équivalent ne contenant que les contraintes liantes (2.2) et les contraintes de convexité (2.7). Les contraintes faciles (2.3) forment un problème auxiliaire dit sous-problème (SP), dont l'objectif est de définir les nouvelles variables qui représentent les points extrêmes du domaine réalisable $\mathcal{X}$ (Wolsey et Nemhauser, 1999).

À titre d'illustration, considérons un problème de VRP classique où K véhicules hétérogènes doivent effectuer des visites chez $n \in N$ clients. Les contraintes faciles sont séparables par type de véhicule $k \in K$, le sous-problème est donc décomposé en k sous-domaines $\mathcal{X}^{k},\ k \in K$. Chacun d'entre aux se modélise comme un problème de plus court chemin avec des contraintes de ressources sur le graphe correspondant $G^{k}(V^{k}, A^{k})$. Chaque point extrême correspond à un chemin entre la source et le puits ; généralement obtenus moyennant un algorithme de programmation dynamique (Feillet et al., 2004; Irnich et Desaulniers, 2005).

Le principe fondateur de la méthode DW découle du théorème de Minkowski qui stipule que tous les polyèdres bornés (n'ayant aucun rayon extrême) sont des ensembles de combinaisons convexes de leurs points extrêmes. Ceci permet de représenter les points de $\mathcal{X}$ comme

combinaison convexe des points extrêmes de $conv(\mathcal{X})$ tel que :

$$\begin{aligned} \boldsymbol{x} &= \sum_{r\in\Omega} \theta_r \omega_r \\ &\sum_{r\in\Omega} \theta_r = 1 \\ &\theta_r \geq 0 \quad \forall r \in \Omega \end{aligned}$$

où Ω est l'ensemble des indices des points extrêmes, ω_r le r$^{\text{ème}}$ point extrême et θ_r le poids qui lui est associé. À ce stade, imposer les conditions d'intégralité contribue considérablement dans la réduction du gap d'intégralité, rendant ainsi le problème beaucoup plus facile à résoudre. En effet, cette décomposition a le grand avantage de couper une grande partie des points non entiers en réécrivant les polyèdres en termes de combinaisons convexes de points extrêmes (solutions entières).
Posons $c_r = \boldsymbol{c}^\top \omega_r$ et $a_r = \boldsymbol{A}\omega_r$, en effectuant une réorganisation des termes des coefficients, et en supposant qu'il y a un seul domaine $\mathcal{X}$, le problème ainsi reformulé est dit le problème maître (PM) :

$$(PM): \quad \min_{\theta} \quad \sum_{r\in\Omega} c_r\theta_r \tag{2.5}$$

$$\text{s.à.} \quad \sum_{r\in\Omega} a_r\theta_r = \boldsymbol{b} \tag{2.6}$$

$$\sum_{r\in\Omega} \theta_r = 1 \tag{2.7}$$

$$\theta_r \geq 0 \qquad \forall r \in \Omega \tag{2.8}$$

$$\boldsymbol{x} = \sum_{r\in\Omega} \theta_r \omega_r \qquad entiers. \tag{2.9}$$

Le PM impose que les contraintes liantes soient satisfaites, et demande des solutions supplémentaires au sous-problème de sorte que l'objectif global du programme linéaire initial soit amélioré. Un sous-problème adéquat s'assurera de ne générer que les points extrêmes prometteurs. Cette reformulation du PNE engendre moins de contraintes mais un très grand nombre de variables, soit une variable par point extrême de $\mathcal{X}$, en plus des contraintes d'intégralité, et c'est là où réside l'intérêt de la GC.

2.2.2 Génération de colonnes

Au cours des dernières décennies, la génération de colonnes a acquis une popularité considérable dans la résolution des problèmes de décision d'intérêt pratique. Les problèmes de tournées de véhicules ont représenté un véritable terrain d'essais pour cette méthode qui a prouvé son efficacité dans la résolution des classes d'instances larges et riches. Lübbecke et Desrosiers (2005) ont décrit les applications de la GC aux problèmes décisionnels comme les tournées de véhicules, l'affectation d'équipages en transport aérien et routier.

Dans la majorité des programmes linéaires résolus par l'algorithme simplex révisé, à chaque itération, la plupart des colonnes sont hors base, d'où l'idée maîtresse de la génération de colonnes (différée). En effet, au lieu de résoudre le PM complet pour toutes les variables, le résoudre plutôt pour un sous-ensemble intéressant de variables (actives). Le rôle du SP est de générer les nouvelles colonnes à entrer dans la base pour améliorer la fonction objectif. La GC est un processus itératif qui permet de résoudre la relaxation linéaire du problème maître (2.5)-(2.8). Soient π et σ les variables duales associées aux contraintes 2.6 et 2.7, respectivement. À chaque itération t, l'algorithme commence par résoudre le PM restreint (PMR^t) pour un sous-ensemble réduit de variables $\theta_r \in \Omega^t$. La résolution du PMR^t permet d'obtenir une solution primale $\boldsymbol{\theta}^t$ réalisable pour le PM, ainsi que la solution duale correspondante (π^t, σ^t). Les autres variables (colonnes) $\theta_r \in \Omega \setminus \Omega^t$ sont générées au besoin par le SP. En effet, en utilisant la solution duale π^t, SP^t permet de générer les colonnes de coût réduit négatif $\bar{c_r} = c_r - \pi^\top a_r - \sigma$. Ces colonnes sont ajoutées à Ω^t et forment un nouveau PMR. Le processus s'arrête lorsque le coût réduit de toutes les variables est positif. Ainsi, la solution courante $\boldsymbol{\theta}^t$ du PMR^t est optimale. Cette solution constitue aussi une solution optimale pour la relaxation linéaire du PM (RLPM). À ce stade, si la solution contient des variables à valeurs fractionnaires, un algorithme d'énumération implicite est invoqué.

Notons que la dualité constitue un aspect puissant. Elle a la capacité d'identifier et d'évaluer l'amélioration potentielle d'une variable non générée en regardant simplement son coût réduit. Nonobstant, en cas de dégénérescence, le coût réduit n'est pas un indicateur pertinent. Comme nous l'avons mentionné, dans les modèles de GC, le VRP apparaît souvent comme un problème de partitionnement d'ensembles, tandis que le SP est un problème de plus court chemin avec contraintes de ressources (PCCCR). Ce dernier consiste à trouver le chemin le moins coûteux de la source à la destination, tout en respectant les contraintes de consommation des ressources. Selon Desrosiers et Lübbecke (2005), la GC doit son efficacité au PCCCR, qui a la capacité de gérer les contraintes les plus difficiles qui introduisent un grand gap d'intégralité.

Agrégation dynamique des contraintes. Seule une partie très réduite des variables sont non nulles en cas de dégénérescence. De ce fait, un grand nombre de contraintes est redondant pour les variables non nulles. Basés sur ce constat majeur, Elhallaoui et al. (2005) ont introduit la méthode d'agrégation dynamique des contraintes DCA (DCA : *Dynamic Constraints Aggregation*) pour les problèmes de partitionnement d'ensembles. La DCA consiste à reformuler le problème original et à réduire le nombre des contraintes en regroupant certaines d'entre elles. L'efficacité de la méthode a été démontrée par le biais des résultats expérimentaux qui ont révélé que l'approche DCA surpasse clairement la méthode de GC standard, appliquée à un SPP relaxé.

2.2.3 Branch-and-Bound

La méthode de *Branch-and-Bound*, introduite par Land et Doig (2010), est une méthode de résolution reposant sur l'exploration récursive d'un arbre de branchement. L'algorithme est exact et garantit l'obtention de la solution optimale, mais en pratique cela peut entrainer un temps de calcul considérable. La racine de l'arbre représente la relaxation linéaire du problème original, et les différentes résolutions sont organisées dans un arbre de branchement. À chaque résolution, on procède à une étape d'évaluation qui permet de déduire les bornes inférieures en résolvant la relaxation linéaire du problème courant. Ensuite, l'étape de séparation consiste à séparer le problème en deux sous-problèmes en y imposant des décisions de branchement adéquates. Il existe plusieurs stratégies d'exploration de l'arbre de branchement ; en profondeur d'abord, meilleur d'abord, ou une combinaison de celles-ci. L'énumération est implicite dans le sens où le principe d'élagation est appliqué ce qui permet d'explorer juste la partie intéressante de l'arbre (Morrison et al., 2016).

2.2.4 Branch-and-Price

L'algorithme de *Branch-and-Price* est une méthode d'énumération implicite dans laquelle les bornes inférieures sont calculées par une méthode de GC. Il a été appliqué la première fois par Desrochers et al. (1992) dans le cadre de la planification d'équipages, où la GC a été utilisée pour résoudre chaque nœud de l'arbre de branchement. Dans la formulation du PM, les variables x sont assujetties à des contraintes d'intégralité, par conséquent, les règles de branchement sont appliquées sur ces variables ou sur leurs combinaisons au niveau du PM ou bien du SP.

L'inconvénient majeur de cette méthode réside dans la lenteur de la convergence. En particulier lorsqu'il s'agit de problèmes d'optimisation de grande échelle souffrant d'une sévère dégénérescence. Ces problèmes peuvent être contournés moyennant des stratégies adéquates

de stabilisation des variables duales, ou en utilisant la méthode de points intérieurs (*Cplex Barrier*) (Vanderbeck, 2005).

2.2.5 Heuristiques pour la génération de colonnes

Appliquer des règles exactes de branchement au niveau du PM est une méthode peu efficace, en outre, les appliquer au niveau du SP en accentue la complexité. Par conséquent, pour contourner les difficultés de calcul et de formulation, plusieurs heuristiques, basées sur la génération de colonnes, ont été proposées le long de la littérature. L'heuristique de plongeon DH (DH : *Diving Heuristic*) explore une seule branche de l'arbre de recherche. À chaque nœud, la variable ayant la plus grande valeur fractionnaire est fixée à un, suivi d'une réoptimisation du PMR. Le processus s'arrête lorsque la solution est entière. Dans le même contexte, RMH (RMH : *Restricted Master Heuristic*) est une heuristique de problème maître restreint qui consiste à résoudre par génération de colonnes le nœud racine. Les contraintes d'intégralité sont ensuite imposées aux variables au niveau du dernier PMR qui est ensuite résolu par un solveur MIP commercial. Plus de détails sur ces approches sont procurés par Joncour et al. (2010). Nous rappelons que ces méthodes sont heuristiques dans le sens où elles n'offrent aucune garantie sur l'optimalité de la fonction objectif. Pour un aperçu plus étendu et récent sur d'autres heuristiques primales, nous conférons le lecteur aux travaux suivants : Lübbecke et Puchert (2012); Sadykov et al. (2019); Zhao et al. (2020).

2.3 Méthodes heuristiques

Depuis des décennies, les méthodes de résolution heuristiques ont été largement sollicitées pour résoudre les problèmes combinatoires difficiles de grande taille. Certes, elles n'offrent aucune garantie d'optimalité dans la mesure où elles n'explorent qu'une partie de l'espace de recherche, mais elles exhibent une grande capacité à fournir des solutions satisfaisantes en un temps raisonnable. Dans la littérature des VRPs, l'enthousiasme envers ces approches est inhérent. Récemment, Blocho (2020) a realisé une revue extensive sur les heuristiques, métaheuristiques et hyperheuristiques adaptées aux VRPs. De même, Osaba et al. (2020) ont conduit une évaluation critique d'une certaine classe des heuristiques et leur viabilité pour les VRPs. Dans le travail notable de Vidal et al. (2013b), les auteurs ont passé en revue 48 heuristiques et 15 variantes du VRP, selon une démarche méthodologique structurée.

2.3.1 Méthodes de construction

Les heuristiques de construction sont très sollicitées pour produire des solutions initiales d'une grande gamme de méthodes, et ont été adaptées à de nombreuses variantes du VRP. Ces méthodes exécutent des décisions irréversibles telles l'insertion d'un client ou l'intersection de deux routes. Parmi les procédures notables, nous citons la méthode des économies (*saving*), qui construit une route en se basant sur le choix d'insertion séquentielle d'arcs permettant de réaliser une meilleure économie sur la distance (Clarke et Wright (1964)). Ordonner les clients, les balayer et les insérer à la fin de chaque route, tel est le principe de la méthode de balayage (*sweep*) (Gillett et Miller, 1974). Il existe également des approches à deux-phases, dont les plus notables sont : (i) *route-first cluster-second* qui génère une route géante visitant tous les clients, qui est ensuite segmentée selon les contraintes assignées ; (ii) *cluster- first route-second* consiste à affecter les clients en premier lieu, puis fixer les séquences.

2.3.2 Méthodes d'amélioration

L'objectif de ces méthodes est la recherche d'une meilleure solution dite optimum local, en se déplaçant au voisinage d'une solution de départ. Dans le cadre des VRPs, la méthode implique un échange de type suppression et réinsertion de λ segments de routes (arcs) afin d'améliorer la solution à chaque itération. Partant d'une solution primale $\boldsymbol{\theta}$, une heuristique de recherche locale explore le voisinage réduit $\mathcal{N}(\boldsymbol{\theta})$ en y effectuant des mouvements menant à une solution améliorante $\bar{\boldsymbol{\theta}}$. À travers la littérature, il existe une panoplie de voisinages conçus pour les VRPs. (i) La première famille recouvre les voisinages basés sur les échanges d'arcs, permettant l'optimisation séparée des routes. Un voisinage de type λ-*opt* (Lin, 1965) est engendré par les solutions obtenues suite à la suppression et la réinsertion de λ-paires d'arcs. Dans la littérature, les voisinages les plus courants sont ceux de *2-opt*, *3-opt*, *GENI*, etc. (ii) La seconde famille englobe les voisinages basés sur les échanges d'arcs et les changements de visites, permettant une amélioration concomitante de plusieurs routes. Parmi les voisinages fréquemment employés nous citons *insert*, *swap* et *crossover*. Finalement, (iii) les voisinages larges, basés sur la suppression (destruction) et la réinsertion (construction) des visites, ont prouvé leur efficacité dans le traitement des problèmes de grande taille.

2.3.3 Métaheuristiques

Les métaheuristiques relèvent de l'optimisation générale, et offrent une combinaison de techniques et de stratégies intelligentes qui permettent d'explorer et d'exploiter l'espace de

recherche tout en évitant les optimums locaux. Les dernières années ont été témoins de la prépondérance de ces approches. La littérature classifie ces méthodes en deux classes majeures :

Méthodes à trajectoire. Ces méthodes de recherche locale passent d'une solution à une autre en construisant une trajectoire dans l'espace des solutions candidates, jusqu'à ce qu'une solution considérée comme optimale soit trouvée ou que le temps imparti soit dépassé. Pour éviter de cycler dans un optimum local, elles opèrent des modifications et des détériorations successives des solutions. Le recuit simulé (Kirkpatrick et al., 1983), ou en anglais *simulated annealing* (SA), figure parmi les méthodes notables qui utilisent des arguments probabilistes dans le choix de solutions. La recherche tabou, introduite par Glover (1989), est référencée dans la littérature parmi les heuristiques les plus efficaces notamment pour le VRP et ses variantes ; offrant un bon compromis entre qualité et rapidité (Cordeau et Laporte, 2001). Cette méthode déterministe à mémoire consiste à chercher les solutions en évitant certains déplacements qui sont stockés dans une liste tabou. Partant du fait qu'un minimum local varie d'un voisinage à l'autre, la méthode de recherche à voisinage variable VNS (VNS : *variable neighbourhood search heuristic*) est un autre type de métaheuristiques qui est basé sur la variation du voisinage tout au long de la recherche (Hansen et Mladenović, 2001).

Méthodes à population. Elles font référence aux algorithmes génétiques dans le sens qu'elles sont inspirées des lois d'évolution naturelle des êtres vivants. Elles font usage de l'expérience acquise durant la recherche d'optimum ainsi que des concepts d'intelligence collective. La génération des solutions et leurs améliorations sont réalisées par le biais d'opérateurs de sélection, de croisement et de mutation (Barbosa, 2018). Employées seules, les solutions fournies sont peu intéressantes, néanmoins, elles sont plus efficaces quand elles sont couplées avec une heuristique de recherche locale par exemple. Les méthodes ayant réussi à résoudre des VRPs de grande taille incluent plusieurs mécanismes avancés et de plusieurs approches hybrides (Toth et Vigo, 2014).

Méthodes hybrides. Elles représentent un schéma de résolution coopératif où plusieurs techniques et méthodes heuristiques et exactes coexistent. Dans la littérature, on peut retrouver plusieurs algorithmes hybrides adaptés aux VRPs, ayant permis de réduire le temps de calcul et d'améliorer la qualité des solutions obtenues (Ibaraki et al., 2005; Kytöjoki et al., 2007). Cette classe prometteuse d'heuristiques représente un terrain d'essai propice pour évoluer vers des méthodes flexibles, résilientes et agiles.

2.4 Méthodes primales exactes

À travers cette section, nous introduisons le cadre général de l'approche primale, puis nous présentons les contributions notables, dans le contexte particulier des SPPs.

2.4.1 Généralités

Les méthodes primales opèrent directement sur le problème à traiter, et génèrent itérativement des solutions assurant une décroissance monotone de la fonction à minimiser. Elles consistent à trouver une direction où une meilleure solution entière réalisable puisse être atteinte. D'une manière simplifiée, un algorithme primal se base sur le schéma suivant : (i) à partir d'une solution entière non optimale, (ii) trouver une direction de descente, (iii) itérer le processus jusqu'à l'optimalité. Nous constatons que ce schéma, simplement décrit, est très similaire à celui des heuristiques décrites dans la section précédente. Une méthode primale exacte suit toutefois une démarche déterministe et rationnelle pour trouver, à chaque itération, une solution améliorante entière en suivant une direction de descente. L'approche primale présente un avantage indéniable ; la disponibilité d'une solution réalisable à tout moment est une capabilité très demandée en milieu industriel pour faire face aux incidents imprévus. Dans ce qui suit, nous relatons l'évolution de ces méthodes, les approches les plus notables ainsi que leurs avantages potentiels.

2.4.2 Evolution des méthodes primales

La théorie sous-jacente à l'approche primale a été initiée par Ben-Israel et Charnes (1963) avec l'algorithme direct pour la programmation en nombres entiers. Quelques années plus tard, Young (1968) a introduit le concept du problème d'augmentation intégral. Les résultats de Balas (1989) ont contribué à l'essor des méthodes primales pour les SPPs. En fait, ses résultats théoriques ont démontré qu'un chemin reliant toute paire de points extrêmes entiers existe, ainsi l'optimalité pourrait être atteinte en parcourant ces solutions entières.

Simplex intégral. L'algorithme primal IS (IS : *Integral Simplex*) a été introduit par Thompson (2002), il opère en combinant deux méthodes : une recherche locale pour génèrer un optimum local, et une méthode globale pour résoudre un SP dans chaque nœud de l'arbre de branchement. L'avantage d'IS est de pouvoir arrêter la résolution à tout moment tout en assurant une solution entière. Cependant, des problèmes de dégénérescence sont survenus lors des expérimentations et les essais d'intégration avec la GC.

La structure particulière du SPP a induit plusieurs tentatives d'hybridation d'IS et de la GC. Les premiers schémas d'intégration ont été proposés par Rönnberg et Larsson (2009, 2014).

Ils ont utilisé le simplex intégral dans un contexte de génération de colonnes pour résoudre un SPP. Quant à l'optimalité, elle a été atteinte en employant adéquatement une stratégie de branchement. Cependant, face à la forte dégénérescence du SPP, seuls les problèmes avec petites instances ont été résolus. En absence de résultats expérimentaux, quelques résultats théoriques s'en sont découlés.

Simplex primal amélioré. Comme nous l'avons mentionné, DCA est une stratégie avancée pour gérer la redondance des contraintes. Mais au cours de la résolution, la question est comment s'assurer qu'une contrainte qui était redondante dans une itération passée ne l'est plus dans l'itération courante. Cela a conduit à classer les colonnes non générées dans des classes dites de compatibilité. Supposons que S est le support des colonnes de la solution courante, algébriquement, une colonne A_j est **compatible** avec S si A_j s'écrit comme une combinaison linéaire des colonnes de S, sinon elle est incompatible. Introduit par Elhallaoui et al. (2011), le simplex primal amélioré IPS (IPS : *Improved Primal Simplex*) permet de résoudre ce problème.

2.4.3 Méthodes primales pour SPP

Simplexe en nombres entiers avec décomposition. L'algorithme du simplexe en nombres entiers avec décomposition (ISUD : *integral simplex using decomposition*) a été introduit par Zaghrouti et al. (2014) dans le but de résoudre efficacement les SPPs de grande taille et faire face à leur dégénérescence prononcée. En termes généraux, ISUD divise le problème de manière dynamique en deux sous-problèmes, et essaie de trouver des directions de descente strictes à chaque itération, conduisant à une solution optimale entière. Soit $\boldsymbol{A}$ la matrice de contraintes, S, C et I désignent les supports des colonnes de la solution courante, des variables compatibles et incompatibles, respectivement. Les colonnes de $\boldsymbol{A}$ sont partitionnées en colonnes compatibles et incompatibles par rapport à la solution courante S, tel que $\boldsymbol{A} = [S \quad C \quad I]$. Partant du constat stipulant que les variables de base dégénérées sont incompatibles ; et que les variables compatibles de coût réduit négatif donnent un pivot non dégénéré en entrant dans la base, le PMR est décomposé dynamiquement en deux petits sous-problèmes. (i) Le problème réduit (PR) contient les colonnes compatibles et cherche à améliorer la solution courante S. (ii) D'autre part, le problème complémentaire (PC) contient les colonnes incompatibles, et vise à trouver des directions de descente entières menant vers des solutions entières améliorantes. Une direction de descente est une combinaison compatible des colonnes incompatibles. Des résultats expérimentaux conduits sur de grandes instances de SPP, atteignant 500,000 variables, ont démontré l'efficacité de la méthode.
Puisque l'algorithme n'offre aucune garantie d'intégralité des directions de descente, Rosat

et al. (2014) s'est proposé d'ajouter des plans sécants au niveau du PC pour pénaliser les directions fractionnaires. Une autre version améliorée d'ISUD, nommée *ZOOM*, a été introduite par Zaghrouti et al. (2020). Elle permet l'exploration du voisinage de la direction fractionnaire afin de trouver une nouvelle direction entière. Nous soulignons que ISUD a incité plusieurs contributions scientifiques s'intéressant à appliquer ou adapter l'algorithme.

Génération de colonnes en nombres entiers. Récemment, Tahir et al. (2019) ont introduit la génération de colonnes en nombres entiers (ICG : *integral column generation*). Les auteurs ont proposé une approche intéressante et réussie qui combine l'algorithme primal ISUD dans un schéma exact de GC. ICG est un algorithme séquentiel où le PMR est résolu à l'aide de ISUD, tandis que le SP est résolu à l'aide de la programmation dynamique. Cette approche tire parti des performances irréfutables de la GC, plus encore, la décomposition dynamique du PMR accélère sa réoptimisation. La méthode a connu un succès dans la résolution de grandes instances des problèmes de rotations d'équipages dans le domaine aérien (CPP : *crew pairing problem*) ainsi que des problèmes d'horaires de chauffeurs et d'autobus (VCSP : *vehicle and crew scheduling problem*). Les auteurs argumentent le fait que ICG est capable de résoudre des SPPs avec des contraintes supplémentaires.

2.5 Sommaire

À l'issue de cette étude de revue, nous avons essayé de cerner notre problématique et guider nos contributions. Les idées principales à retenir se déploient comme suit :

— Le VRP est un problème difficile. Outre sa complexité combinatoire théorique, il s'agit d'un problème épineux de management des opérations et des processus.

— Dans la littérature, jamais un VRP n'a été résolu moyennant un algorithme primal basé sur des méthodes exactes.

— Les méthodes primales appartiennent à une classe très retreinte et sont peu abordées dans la littérature. Le schéma global de leur fonctionnement garantit une solution réalisable à tout moment au cours de la résolution. Ce qui représente un atout majeur pour des solutions industrielles.

— La littérature locale de VRP est très riche, cependant, certaines variantes, omniprésentes dans l'industrie du fret, ne sont pas suffisamment abordées.

CHAPITRE 3 ORGANISATION DU TRAVAIL

Bien que les principes relatifs aux problèmes de tournées de véhicules soient connus à travers une large littérature, les réalisations concrètes et adaptées de ces approches restent marginales ; les méthodes exactes ayant contribué dans la résolution des VRPs étant essentiellement des méthodes duales fractionnaires. En pratique, ces méthodes souffrent d'une convergence lente et nécessitent des stratégies appropriées pour la stabilisation. Plus encore, il faut attendre la fin de la résolution pour obtenir une bonne solution réalisable ; ce qui peut entraver le transfert de la solution en milieu industriel. Concernant les méthodes primales exactes, elles sont généralement très peu abordées, et jamais utilisées pour résoudre un VRP. La littérature exprime une tendance apparente vers les méthodes heuristiques. En général, celles-ci ne garantissent pas la qualité de la solution.

Le premier objectif (Chapitre 4) se penche sur la résolution d'un VRP multi-attributs en recourant à un algorithme primal intégré dans un cadre de génération de colonnes. Le PMR est résolu à l'aide de ISUD, ensuite ICG utilise une solution duale correspondant à la solution entière courante afin de générer un grand nombre de colonnes. Le modèle mathématique est formulé comme un SPP, tandis que le SP est un problème de PCCCR sur un graphe acyclique, résolu par un algorithme de programmation dynamique. L'étude numérique conduite sur sept instances réelles a été concluante et a démontré (i) qu'un VRP riche peut être résolu efficacement en utilisant un nouveau paradigme primal très peu abordé dans la littérature, et que (ii) l'algorithme ICG peut être amélioré pour pouvoir accommoder des instances plus larges et des contraintes plus difficiles.

Le second objectif (Chapitre 5) aborde un problème de transport du dernier kilomètre avec des fenêtres de temps, une flotte hétérogène, des contraintes de compatibilité véhicule client, et des options de livraison avec transporteur externe. Nous améliorons la performance de ICG en implémentant au niveau du SP une stratégie d'accélération multiphase, et en modélisant le SP sur un graphe urbain partiellement cyclique. À l'aide d'une méthodologie de recherche aussi bien théorique qu'empirique, nous montrons que nous sommes en mesure de générer des solutions entières la plupart du temps, sans avoir recours au branchement. En effet, dans la majorité des cas, le PC a directement généré des direction entières. Pour le reste, la résolution d'un petit MIP permet d'atteindre rapidement une solution entière. Nous abordons aussi la notion et le rôle de la dualité dans un contexte de PPNE en nous basant sur des fondements théoriques et expérimentaux. Finalement, nous présentons quelques techniques utiles à l'implémentation avec une étude numérique conduite sur 30 instances

réelles comportant jusqu'à 199 clients.

Pour résoudre la problématique exposée dans ce manuscrit, nous avons entrepris quatre années de collaboration étroite avec un prestataire 3PL ; et un stage d'observation d'une année y a été accompli. Le troisième objectif de notre thèse, présenté dans le chapitre 6, est une étude de cas réelle qui expose la face compliquée de l'implémentation d'un VRP en pratique, dans le contexte enchevêtré des opérations logistiques. Nous décrivons la méthodologie suivie, les changements introduits, l'architecture de notre système ainsi que les résultats observés sur plusieurs niveaux.

Finalement, à travers la dernière contribution (Chapitre 7), nous proposons deux pistes d'amélioration de la version de base de ICG. Afin de trouver efficacement des voisinages potentiels d'amélioration, nous présentons deux approches d'intégration : ICG–(ZOOM) et ICG-(H-ALNS). Dans cette étude préliminaire, nous discutons les premiers résultats ainsi que les démarches prédisposées à réussir une toute première hybridation entre une heuristique et un algorithme primal exact.

CHAPITRE 4 ARTICLE 1: SOLVING A REAL-WORLD MULTI-ATTRIBUTE VRP USING A PRIMAL-BASED APPROACH

Cet article conférence a été écrit par M. Messaoudi, I. El Hallaoui, LM. Rousseau, et A. Tahir, et a été publié en Juillet 2020 dans *Combinatorial Optimization. ISCO 2020. Lecture Notes in Computer Science, vol 12176. Springer.*

4.1 Introduction

Supply chain encompasses several integrated activities and hand-offs allowing physical and information flows to be routed from the source (supplier) to the final destination (customer). Many believe that nearly two thirds of the supply chain total cost is related to transportation, more specifically, trucking is the dominant spend component.

Indeed, in a customer-centric era, transportation industry is becoming increasingly complex, and firms are facing a serious imperative: being able to deliver efficiently a whatever-whenever-wherever while respecting time, cost and quality. However, traditional networks, inefficient and fragile systems with limited computing performance are crippling firms to provide high-quality service, and maintain growth. In that respect, logistics providers are called upon to sharpen their practices through resilient tailor-made approaches.

Our main objective is to solve a real-world routing problem subject to a set of constraints and specific business rules commonly encountered in logistics industry markets. Our solution approach is based on a primal-based algorithm in column generation framework. The aim is not only to efficiently generate optimal dispatching plans, but also to shed light on the opportunity offered by such methods, especially when deployed on large-scale problems. In the remainder of this paper, note that all the mathematical formulations will be presented in their maximization form, but results also hold for a minimization scenario.

4.2 Related Literature

Vehicle Routing Problems (VRP) are so popular, and are the subject of an intensive literature due to their wide application in logistics and freight industry. Since its introduction by Dantzig and Ramser (1959); Clarke and Wright (1964), several approaches regarding modelling and solution methods have been proposed for many VRP variants. Classical VRP aims to design minimal cost routes for a fleet of identical vehicles such that each customer is served exactly once, the capacity is respected and each vehicle starts and ends its route at the depot. In

a-demand-driven-supply chain, many requirements and constraints arise, thus we switched from the classical VRP to new and more combinatorial and difficult models (Coelho et al., 2015) which combine not only the usual restrictions such as time windows and fleet structure, but also specific business rules that vary according to the context. A detailed study of those variants can be found in (Toth and Vigo, 2014). Since VRP is NP-hard (Lenstra and Kan, 1981), heuristics and metaheuristics are more suitable than exact methods which are difficult to implement on large-scale problems. According to the classification of Letchford and Lodi (2002), solution methods are classified in two main classes : dual fractional and primal methods. Dual fractional methods maintain iteratively optimality and feasibility until integrality is achieved. Whereas primal or augmentation methods maintain both feasibility and integrality and stop when optimality is reached.

One of the most known dual fractional methods is the branch-and-price (B&P) algorithm which combines branch-and-bound ($B\&B$) and column generation. The latter is an iterative process that solves the linear relaxation of the problem called a master problem (MP) for a restrictive subset of variables (columns), then called the restricted master problem (RMP). The duals related to the RMP's constraints are sent to a subproblem (SP) in order to generate positive-reduced cost variables, to be added into the RMP. The process stops when no such variables exist, and an optimal solution is obtained. If the latter is fractional, $B\&B$ is applied using a suitable branching strategy. However, despite its overall success, convergence problems can occur and affect the method's efficiency (Vanderbeck, 2005), they could be circumvented by using the stabilization strategies found in the literature and a fine-tuning strategy.

For VRPs, column generation is one of the notable exact methods that have successfully solved large and complex problems (Lübbecke and Desrosiers, 2005). In such a context, the master problem is often modelled as a Set Partitioning Problem (SPP) and the subproblem is the Shortest Path Problem with Resource Constraints (SPPRC) defined on a directed graph (Irnich and Desaulniers, 2005), and usually solved with the dynamic programming algorithm introduced by Desrochers and Soumis (1988).

Interestingly, exact primal methods have attracted very little interest in the literature, furthermore, concrete and adapted realizations of these approaches remain marginal. In a simple way, a primal procedure moves from one integer solution to an improving adjacent one until optimality, and such a sequence of moves leading to an improving solution is called a descent direction. With respect to the usual notation, a descent direction refers to augmentation direction in a maximization context as well. The first primal approach was introduced by Ben-Israel and Charnes (1963), then Young (1968) set the concept of the integral augmentation problem. An interesting combined approach was proposed by

Thompson (2002); Rönnberg and Larsson (2009) that showed how integral simplex can be properly embedded in column generation context. They used an adequate branching strategy to obtain an optimal solution. Although, these models are restricted to small instances as they haven't been able to overcome the high degeneracy of SPPs. Integral Simplex Using Decomposition (ISUD) is one of the most promising primal methods, it was introduced by Zaghrouti et al. (2014) to solve large-scale SPPs. Based on the improved primal simplex algorithm (Elhallaoui et al., 2011), ISUD takes advantage of degeneracy and finds strict descent directions at each iteration leading to an optimal solution. ISUD's performance has been enhanced by adding secant plans to penalize fractional descent directions (Rosat et al., 2014), and also by exploring neighborhood search (Zaghrouti et al., 2020). ISUD performed excellent computational results for SPPs with up to 500.000 variables. Recently, Tahir et al. (2019) introduced integral column generation (ICG). This three-stage sequential algorithm combines ISUD and column generation to solve SPPs. Experiments on large-scale instances of Vehicle and Crew Scheduling Problem (VCSP) and Crew Pairing Problem (CPP) showed that ICG exceeds two well-known column generation-based heuristics. The authors evoked the possibility to adapt ICG even on SPP with side constraints.

The remarkable performance of ISUD and ICG algorithms makes them worth pursuing. We believe that such methods have to be experimented on complex and well-known problems of literature such as rich routing problems.

Organization of the paper and contributions. To the best of our knowledge, it is the first time that a primal algorithm based on an exact method has been used to solve a rich vehicle routing problem. It is also an opportunity to discuss the procedure's performance on a real-world combinatorial problem. On the one hand, experimentation on real instances showed that the used primal method finds very good results in a short computing time. Indeed, it outperforms a well-known branch-and-price heuristic. On the other hand, the solution has led to a positive impact on the company's outcome indicators. This paper is organized as follows. In section 4.3, we describe our problem and give the related notation. In section 4.4, we give the mathematical formulation of master problem and subproblem. In section 5.4, we introduce some theoretical notions related to the primal approach. The solution method (ICG) is described in the section 4.6. The computational results are reported in the section 4.7. Finally, concluding remarks are presented in the section 4.8.

4.3 Problem Statement

In the remainder of the paper, we use the notation organized in the table 5.2 below.

Table 4.1 Definition of the parameters and variables

Notation	Type	Description
Ω	–	Set of feasible routes
N	–	Set of customers to visit
K	–	Set of heterogeneous k-type vehicles
a_{ir}	binary	Is equal to 1 if and only if customer i is served by route r
d_i	real	Demand associated with customer $i \in N$
l_k	real	Travelled distance between origin and destination by k-vehicle $\in K$
p_r	real	Profit collected by the route $r \in \Omega$
q_k	real	Capacity of vehicle $k \in K$
s_i	real	Service time at customer $i \in N$
w^k	real	Accumulated working time of k-vehicle $\in K$
θ_r	binary	Is equal to 1 if and only if route r is selected

Given a set of customers $i \in N$ geographically scattered, a logistic hub O handles their transport operations. The daily task is to ensure next-day deliveries within specific time frames imposed by either customer convenience or/and urban traffic regulation, using heterogeneous vehicles with different capacities, types and operating costs. In addition, specific business rules such as the driving hours set up by work unions, loading rate, and urban accessibility regulation must be satisfied.

The objective is to maximize the profit collected by routes, resulting in the difference between freight rates and freight costs, while satisfying the following constraints:

1. Each customer $i \in N$ is visited by a single route $r \in \Omega$
2. Each customer $i \in N$ is visited within the time window $[a_i, b_i]$
3. Each customer $i \in N$ is visited by an allowed k-vehicle
4. The total load $\sum_{i \in N} d_i$ on the route travelled by k-vehicle does not exceed the vehicle's capacity q_k
5. The total travelling time l_k, including service times s_i, does not exceed the allowed working time w^k of k-vehicle

4.4 Mathematical Models

The master problem and the subproblem are formulated as SPP and SPPRC, respectively.

4.4.1 Master Problem

With respect to the below-mentioned notation, we formulate the problem as follows:

$$(SPP): \quad \max_{\theta} \quad \sum_{r\in\Omega} p_r\theta_r \tag{4.1}$$

$$\text{s.t.} \quad \sum_{r\in\Omega} a_{ir}\theta_r = 1 \qquad \forall i \in N, \tag{4.2}$$

$$\theta_r \in \{0,1\} \qquad \forall r \in \Omega \tag{4.3}$$

Each variable θ_r is associated with a feasible route $r \in \Omega$ which specifies a sequence of customers $i \in N$ to be served. The objective function (4.1) aims to maximize the profit made by the feasible route r. The constraints (4.2) guarantee that each customer is delivered exactly once. The choice of binary variables is imposed by (4.3).

4.4.2 SPPRC on $G(V, A)$

The subproblem is modelized as a shortest path problem with resource constraints, and is solved using a labelling algorithm as shown in Feillet et al. (2004). We have one supbroblem for each $k-$vehicle, put simply we omit the k-index. The reduced cost of feasible route r travelled by $k-$vehicle, starting and ending at the depot O and visiting a sequence of customers $i \in N$ is computed as :

$$\overline{p}_r = p_r - \sum_{i\in N} a_{ir}\pi_i > 0$$

The dual variable π is associated to the partitioning constraints (4.2). If all columns have negative reduced cost, the algorithm stops and an optimal solution is obtained for the linear relaxation of (SPP) (4.1-4.2). For each k-vehicle, SPPRC is defined on a cyclic graph $G = (V, A)$. V contains $|N| + 2$ vertices, one vertex for each customer $i \in N$, and (s, t) pair where s and t both refer to the depot O. A contains departure arcs $(s, j),\ \forall j \in N$, arrival arcs $(i, t),\ \forall i\ N$ and connecting arcs $(i, j),\ \forall (i, j) \in N$ so that the client j can be reached after the

client i by a realistic route as indicated by the actual road map. Let $\mathbf{\Gamma} = \{\mu_1, \mu_2, \ldots, \mu_{|\Gamma|}\}$ be the set of resource constraints, and L_i^μ be the label related to the resource $\mu \in \mathbf{\Gamma}$ and associated to the vertex $i \in N$ such that the resource window $[a_i^\mu, b_i^\mu]$ is respected.

In our case, we consider the following resources :

— $L_i^T \in [a_i, b_i]$: The time resource indicates the arrival time at customer $i \in V$. L_i^T could be less than a_i, i.e. driver might arrive before the starting delivery period.

— $L_i^D \in [0, q_{max}]$: The demand resource specifies the accumulated load until customer $i \in V$.

— $L_i^W \in [0, 480\,\text{min}]$: The working time resource specifies the total time travelled by the driver, from $\{O\}$ to customer $i \in V$.

— L_i^c : The cost resource is unconstrained, and used to compute the reduced cost of every travel arc $(i, j) \in A$.

The label represents a feasible partial path if : $L_i^\mu \in [a_i^\mu, b_i^\mu],\ \forall\ \mu \in \mathbf{\Gamma}$. In such case, the label is feasible and extended along the $(i, j) \in A$ by calling for a resources-extension function, denoted $f_{ij}(L_i^\mu)$.
While the labelling algorithm solves the subproblem, the SPP calls for ICG algorithm as explained in the section (4.6).

4.5 Preliminaries

For the sake of clarity, we introduce some preliminaries concerning the primal approach principles, and the ISUD algorithm used in our solution procedure. We remind that detailed literature and examples could be found in (Tahir et al., 2018; Elhallaoui et al., 2011; Zaghrouti et al., 2014).

As mentioned earlier, the primal methods have paid particular attention to SPPs. In addition to their popularity in routing and scheduling, these problems satisfy the Trubin theory. If we denote X the SPP polytope and X_S the set of its integer solutions, Trubin (1969) shows that every edge of $Conv(X_S)$ is an edge of X, then SPP is said quasi-integral. This property makes it possible to use linear-programming pivots to reach integer extreme points.

Let consider the (SPP):

$$z^* = \max_{\boldsymbol{\theta}} \left\{ \mathbf{p}^\top \boldsymbol{\theta} \;\middle|\; A\boldsymbol{\theta} = \mathbf{e},\ \boldsymbol{\theta} \in \{0,1\}^n \right\} \tag{4.4}$$

Let $A \in \{0,1\}^{m \times n}$ be the binary constraint matrix, and let $A_1, A_2, \ldots A_n$ be the columns in

A indexed in $J = \{1, 2, \ldots, n\}$ such that A_j denotes the j^{th} column in A.

$$A = \begin{bmatrix} A_1 & A_2 & \ldots & A_n \end{bmatrix}$$

$\mathcal{F}_{SPP} \subseteq \{0,1\}^n$ denotes the feasible region of SPP, while $\mathcal{F}^R_{SPP}$ denotes the feasible region of the linear relaxation of SPP. $\theta^*_r \in \mathcal{F}_{SPP}$ denotes the optimal solution and z^*_{SPP} is the optimal solution value.

Definition 1. *A column A_j is said to be **compatible** with S if $A_j \in Span(S)$, i.e. it can be written as a linear combination of the columns in S, otherwise, it is said to be incompatible.*

Definition 2. *The **incompatibility degree** δ_j of a column A_j towards a given integer solution is a metric measure that represents a distance of A_j from the solution. We note that δ_j of a compatible column is zero.*

Based on the definition of compatibility, ISUD decomposes the columns in A into three sets such that:

$$A = \begin{bmatrix} S & C & I \end{bmatrix}$$

Where S, C and I denote respectively the working basis (the support of the current integer solution), compatible columns subset and incompatible columns subset.

As described in 3, the RMP is decomposed into two small subproblems: The complementary problem (CP) handles the incompatible columns and finds a descent direction $\mathbf{d}$ leading to an improved solution, while the reduced problem (RP) handles the compatible columns and seeks to improve the current solution.

Algorithm 1: $ISUD$ pseudo-code

```
Find initial solution θ^0 and set θ̄ ← θ^0
[S  C  I] ← Partition the binary matrix A into columns subsets
do
    Solve RMP(θ̄, C) to improve the current solution
    (Z^CP, d) ← Solve CP to find a descent direction
while Z^CP > 0 and d is integer
θ̄ = θ̄ + d
return θ̄
```

Given a current solution $\bar{\theta} \in \mathcal{F}_{SPP}$, let $\mathbf{d}$ be the direction leading to the next solution such that $A\mathbf{d} = \mathbf{0}$. In fact, the set of the decent directions generates the null space of A which

could be an infinite cone. Thus, normalization constraints $\mathbf{W}^\top \mathbf{d} = \mathbf{1}$ are added to bound the problem where $\mathbf{W}$ denotes the weight vector. The linear program CP finds, if possible, the combination of columns to obtain a feasible integer descent direction $\mathbf{d}$, i.e., that satisfies the following conditions:

1. $\mathbf{p}_r^\top \mathbf{d} > 0$ (improving)
2. $\overline{\boldsymbol{\theta}} + \alpha \mathbf{d} \in \mathcal{F}_{SPP}, \quad \alpha > 0$ (integer)

4.6 Solution Method

ICG is a three-stage sequential algorithm, which merges primal approach in a column generation context.

The algorithm 4 summarizes the major steps of the solution method:

1. The initialization step builds an artificial initial solution (θ^0, π^0). The initial primal solution θ^0 is built such that, in Step 1, each customer i is visited by a single vehicle k that bears a very large cost. In the initial dual solution π^0, each dual value is set to this large cost.
2. The first step starts by solving the subproblems $SP(\pi^t)$. Using the duals π^t corresponding to the current solution θ, positive-reduced cost routes are included in RMP. If no such routes are generated, we stop the algorithm and the best solution found θ^* is returned.
3. In the second step, ISUD solves the RMP to improve the solution. The criterion $minImp$ decides whether the improvement is sufficient or not. If so, neighborhood search is explored around θ^t by invoking a small MIP. This improvement step is iterated until the number of consecutive improvement failures $consFail$ reaches $maxConsFail$.

Algorithm 2: ICG pseudo-code

Parameters : $maxConsFail$, $minImp$.

Initialize : $t \leftarrow 0$; $(\theta, \pi) \leftarrow (\theta^0, \pi^0)$; $consFail \leftarrow 0$

Output : (z^*, θ^*)

repeat

  Step 1: CG

  $\Omega' \leftarrow$ Solve the $SP(\pi^t)$

  if $\Omega' = \varnothing$ **then**

    break

  end

  $\Omega \leftarrow \Omega' \cup \Omega$

  $t \leftarrow t + 1$

  Step 2: RMP

  $(\theta^t, z^t, \pi^t) \leftarrow$ Solve the RMP using ISUD

  if $\frac{z^{t-1} - z^t}{z^{t-1}} \leq minImp$ **then**

    $consFail \leftarrow consFail + 1$

    $\theta^t_{NS} \leftarrow$ search an improved solution around θ^t by solving a restricted MIP

    if $z^t_{NS} > z^t$ **then**

      $\theta^t \leftarrow \theta^t_{NS}$

      $(\theta^t, z^t, \pi^t) \leftarrow$ Resolve RMP using ISUD

    end

  else

    $consFail \leftarrow 0$

  end

  $(z^*, \theta^*) \leftarrow (z^t, \theta^t)$

until $consFail \geq maxConsFail$

return (z^*, θ^*)

Remark 1. *The theoretical observations and empirical study made by Rosat et al. (2014), concluded that there is a strong correlation between the choice of the normalization weights vector ($W^\top \boldsymbol{d} = \boldsymbol{1}$) and the descent direction obtained. In practice, we use incompatibility degree δ_j as a weight vector to favor integrality.*

4.7 Experimentation

Through this section, we discuss the results obtained by the ICG algorithm on real-life instances. ICG is compared to a well-know branch-and-price diving heuristic (DH) which

is a dual-fractional heuristic based on column generation (Joncour et al., 2010). DH uses a depth-first search by exploring a single branch of the search tree. At each node, candidate columns for selection are evaluated, and the most fractional variable is set to 1. The process stops when the solution of the master problem is integer. The computing was performed on 7 real-life instances using a C++ implementation, under Linux on workstations with 3.3GHz Intel Xeon E3-1226 processors, and 8 GB RAM. The algorithms were implemented using $IBM\ CPLEX$ commercial solver (version 12.4). The SPPRC was solved by dynamic programming using the *Boost* library version 1.54.

4.7.1 Instance Description

The real-life instances are provided by a major logistics provider. The instances correspond to home appliances' distribution of 7 weekdays and involve from $n = 34$ to $n = 140$ customers. We consider heterogeneous fleet of 6 types of vehicles. The fleet size is unlimited since the company can use external vehicles.

For each day, the order form indicates the customer index, his location, the quantity requested, the delivery deadline, and the allowed time frame. Following several tests, the ICG algorithm was implemented with the following parameters values: $minImp = 0.0025$, $maxConsFail = 5$. The table 4.2 shows the computational performance of both ICG and DH. Column 1 indicates the name of the instance ($Name$). Column 2 indicates the number of customers (n) in each instance. For both ICG and DH runs, columns 3 and 7 display the number of iterations ($Iter$). Columns 4 and 8 display the total computing time in seconds ($Time$). Column 6 indicates the total number of integer solutions found ($nb.Sol$). Finally, columns 5 and 9 report the optimality gap in percentage (Gap) between the cost of the best integer solution found and the linear relaxation optimal value.

4.7.2 Numerical Results

Table 4.2 Computational results on 7 realistic instances

Instances				**ICG**			**DH**	
Name	n	Iter	Time	Gap%	nb.Sol	Iter	Time	Gap%
J2	34	4	0.95	0	5	7	0.23	0
J3	40	7	4.48	0.94	8	8	1.14	1.3
J7	66	9	22.6	1.3	28	33	14.4	1.77
J18	106	7	32	0.14	16	17	31.6	1.3
J25	136	12	515	1.62	51	94	1181.4	2.3
J19	140	9	168	0	20	18	321	0.05
J23	126	6	126	0.09	66	48	804	1.03

One can observe that ICG clearly outperforms DH. Indeed, the primal-based method has successfully obtained a feasible solution for all instances, within a competitive computing time $\in [0.95\,\text{sec},\ 515\,\text{sec}]$. Optimality gap varies from 0.00% to 1.6%. For DH, the computing time $\in [0.23\,\text{sec},\ 1181.4\,\text{sec}]$, and optimality gap varies from 0.00% to 1.77%. ICG reduces the DH computing time by a factor of 2.7 on average. ICG performs fewer iterations (at most 12) than DH (between 7 and 94). This can be explained by the ISUD algorithm which generates a large set of columns at each iteration. The figure 4.1 displays the time evolution according to the number of customers. ICG shows a remarkable performance on large instances. In fact, DH time was sped up by a factor of 3.5 on average. The authors (Tahir et al., 2018) also noticed this finding while experimenting ICG on large VCSP and CPP instances.

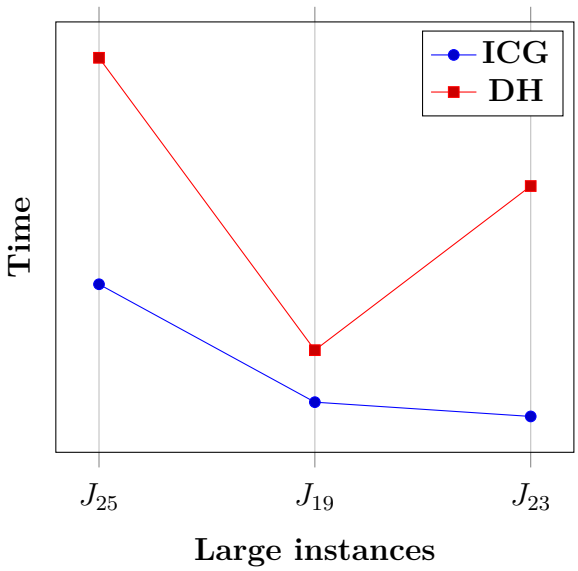


Figure 4.1 Comparative computing times on large instances $n = 126, 136, 140$

We easily notice that ICG yields a large number of integer solutions (from 5 to 66), furthermore, ISUD generates a large number of columns. Thus, this important behaviour helps to improve the objective function since the first iterations. We notice that, for all instances solved with ICG algorithm, no branching method has been activated to obtain integer solutions. One recall that branch-and-bound methods could obtain a good solution with a good tuning and a proper branching strategy.

4.8 Conclusion and Further Work

In this paper, we have experimented for the first time a primal algorithm (ICG) on a complex routing problem. ICG combines the primal algorithm ISUD into column generation framework, and considers a set partitioning formulation. The computational indicators have shown the effectiveness of ICG algorithm while tested on real data reaching 140 customers and a realistic network.

Since the subproblem is a time-consuming, we would like to propose a new intelligent network modelling based on realistic data analysis. We are testing another ICG version that dynamically augments the search space to quickly find integer solutions without any branching recourse.

CHAPITRE 5 ARTICLE 2: INTEGRAL COLUMN GENERATION FOR A REAL-WORLD LAST-MILE TRANSPORTATION PROBLEM

Cet article a été écrit par M. Messaoudi, A. Tahir, I. El Hallaoui, et LM. Rousseau, et a été soumis à la revue *OMEGA-The International Journal of Management Science*.

5.1 Introduction

The global supply chain is an involved network of integrated organizations, operations and functions. It facilitates the routing of physical, financial and informational flows from the first supplier to the final customer while closely monitoring cost, time and quality.

When it comes to physical distribution, freight transportation is probably the most challenging and expensive component of the supply chain. In a competitive environment, customers have high expectations which translate into complex constraints such as delivery time frames, compatible vehicles, customized packaging, etc. As a result, the transport ecosystem has undergone tremendous changes in terms of velocity, new practices and tech-enabled tools. Companies are increasingly involved in the acquisition of high-performance planning systems capable of meeting customer demand, and ensuring that resources move quickly and efficiently from one echelon of the complex distribution chain to another. Aware of this level of complexity, the trend towards outsourcing logistics operations has almost become a necessity for many companies. This has led to the rise of a specialized company known as a Third Party Logistics Provider (3PL), which provides a host of customized services including transportation, warehousing and freight forwarding. During seasonal peak periods, the 3PL itself may rely on external carriers to satisfy growing demand and avoid costly investments. This is just one example of how the global chain acquires a more complex and intertwined aspect.

We generally divide logistics operations into two main categories: inbound and outbound logistics. The difference between inbound trucking (*long-medium haul*) and outbound trucking (*short-haul*) is mainly one of distance travelled and delivery pattern. Outbound trucking involves the last mile delivery, ensuring that goods arrive at the final customer's location. Trucks make up to five deliveries per day on congested, narrow streets and in central business districts. A 3PL has the duty of coordinating delivery locations, times, and returns with busy schedules and numerous constraints. Despite its complexity, an efficient last mile delivery solution could potentially slash shipment time and plot efficient routes to increase customer satisfaction, outpace the competition, and further boost profitability. To do so, planners

must have efficient optimization tools and models with the ability to achieve good practical results in a very short time, while ensuring that each constraint is fulfilled. These transport planning patterns invoke complex and rich Vehicle Routing Problems (VRP). Considering their economic relevance, a large number of contributions have been made to the literature over a period of decades, involving several variants and resolution methods, and a cross between exact algorithms, heuristics and meta-heuristics.

In this paper, we solve a difficult real-world last mile delivery transport problem using a primal algorithm embedded in a column generation scheme. Beyond short computing times, our objective includes the production of efficient plans that respect technical and practical constraints, and allow for a quick response to any unforeseen situation. We believe that the content is particularly novel and provides an appropriate blend of theoretical background and practical insight for the VRP community.

5.2 Related Literature and Contributions

In this section, we first review the main literature on VRPs, in particular on their most relevant variants. Later, we examine the state of the art in both dual and primal paradigms.

5.2.1 Vehicle Routing Problem

VRP stands as one of the most studied topics in the operations research literature. It owes its popularity to its wide application in the tertiary sector, in particular the logistics and freight industry. It was introduced by Dantzig and Ramser (1959) and generalized five years later by Clarke and Wright (1964). The problems of vehicle routing consist of designing a set of routes for k vehicles that have to visit n customers. Each route starts and finishes at a depot and must satisfy a set of constraints such as delivery time windows, specific vehicle characteristics, drivers allowed working time, etc. In a real-world context and a demand-driven supply chain, multi-attribute routing problems have become more prevalent and several variants have emerged. Such problems are known to be challenging as they include technical and practical rules (Coelho et al., 2015), and are usually solved by means of heuristic methods. The authors (Vidal et al., 2013b) published an analysis of sixty-four meta-heuristics which successfully solved fifteen VRP variants. In practice, demand is fluctuating and seasonal peak periods often occur. As a result, total demand could exceed the available capacity of owned fleet, and the logistics provider may consider hiring an external carrier. This variant of VRP, called the Vehicle Routing Problem with Private Fleet and Common Carrier (VRPPC), was first addressed by Ball et al. (1983). Among the notable contributions, Chu (2005) put forward an

interesting model for the VRPPC and solved it with a heuristic approach.
To our knowledge, the proposed approaches are all based on heuristic methods, except for the recent work of Dabia et al. (2019). The latter proposed an exact branch-and-price-and-cut algorithm for rich VRPPCs that was tested on instances taken from benchmark literature. The best set partitioning problem (SPP) formulation solved the largest instance of 100 customers in a time of 238 sec. Although VRPPC problems are an appropriate illustration of the supply chain backdrop, this field is still largely unexplored, and invites researchers to propose more effective solutions to handle highly constrained and complex contexts. For further details concerning more VRP variants, Toth and Vigo (2014); Laporte and Martello (1990) discuss the most relevant ones.

5.2.2 Dual-fractional Paradigms

Several approaches have been developed for solving integer programming models. Following the classification of Letchford and Lodi (2002), the first main class of algorithms involves the dual-fractional methods. At each iteration, these methods maintain optimality and feasibility of the fractional solution until reaching the integrality. Another class of algorithms involves primal methods that consist in looking for optimality while maintaining integrality and feasibility.
The branch-and-price (B&P) algorithm is one of the well-known dual-fractional methods. It is based on a combined scheme of branch-and-bound (B&B) and column generation (CG). Over the past decades, CG has gained considerable popularity in solving practical decision-making problems. A wide variety of vehicle routing, aircrew rostering and general assignment applications can be found in Lübbecke and Desrosiers (2005). When solving large and difficult problems, it is almost impossible to consider all variables implicitly. Moreover, many variables are often non-basic, hence the idea of considering only a restricted set of columns, and generating the rest as needed. As displayed in Figure 5.1, CG is an iterative process that solves the linear relaxation of the column-reduced master problem (MP) called the restricted master problem (RMP). By solving the RMP, we obtain dual prices corresponding to each of the constraints. The duals are then used in the objective function of the subproblem (SP) to price out the non-basic variables, and identify one with a positive reduced cost which is added to the RMP (assuming w.l.o.g. that the problem is a maximization problem). The process stops when no such variables exist, and an optimal solution is retrieved. At this stage, if the solution is fractional, B&B is invoked using appropriate branching rules. Despite its success, a major drawback of this method is the slow convergence, especially when dealing with large-scale optimization problems (Vanderbeck, 2005). To avoid this issue, several acceleration and stabilization techniques have been proposed in the literature.

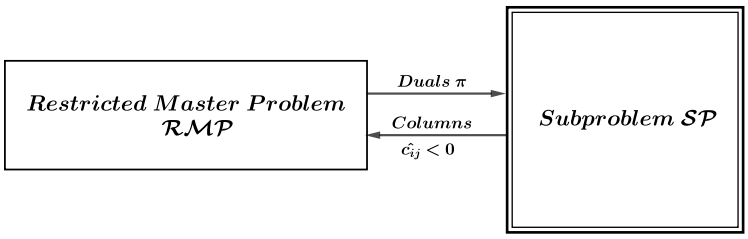


Figure 5.1 Column Generation Process

As extensively stated in the related literature, VRPs are commonly modelled according to: (i) a three-index vehicle flow formulation where the decision variable corresponds to arcs traversed by commodities; or (ii) a set partitioning formulation where the decision variables correspond to feasible routes. While the first model requires fewer variables and a large number of constraints, the second one is a compact and flexible model but requires an exponential number of variables. Hence, column generation arises as a natural methodology for this type of formulation (Toth and Vigo, 2014), and the SP is modelled as a Shortest Path Problem with Resource Constraints (SPPRC), commonly defined on a directed acyclic graph (Irnich and Desaulniers, 2005). The SPPRC consists of finding a least-cost path from the source to the destination, while respecting the constraints on resource consumption, and is generally solved by a dynamic labelling algorithm (Feillet et al., 2004). According to Desrosiers and Lübbecke (2005), CG owes its effectiveness to the SPPRC, which can handle the most difficult constraints that introduce a large integrality gap. SPP formulation is flexible and provides a stronger linear relaxation, while its main limitation lies in the high degeneracy. This is where primal methods become interesting, as they can turn this issue into a criterion of an intelligent decomposition and thus efficiently solve large-sized problems.

5.2.3 Primal Paradigms

In contrast to dual-fractional algorithms, primal methods have received marginal interest in theory and practice. The underlying idea of the primal method is to work on the original problem by searching for the optimal solution within a feasible region. Considering the fact that each point in the search procedure is feasible and integer, the algorithm moves through a sequence of improving pivots until optimality is reached. Such a sequence takes steps through the feasible region and is called a descent (ascent) direction. The underlying theory behind the primal approach was first given by Ben-Israel and Charnes (1963) and

called the direct algorithm for integer programming. Then, Young (1968) introduced the concept of the all-integer augmentation problem, which uses the Gomory cut generation procedure. Moreover, the author enhanced similarities and contrasts with existing integer programming techniques. The results of Balas (1989) were instrumental in the rise of most primal methods. The authors asserted that optimality could be achieved by improving integer solution sequences. Based on the improved primal simplex algorithm (Elhallaoui et al., 2011), a new and interesting primal algorithm was proposed by Zaghrouti et al. (2014), namely the Integral Simplex Using Decomposition (ISUD). The latter is designed to solve large-scale SPPs, and to eliminate their high degeneracy. In broad terms, ISUD splits the problem dynamically into two smaller SPs, and finds strict descent directions at each iteration, leading to an optimal solution. To penalize the fractional descent directions yielded by ISUD, Rosat et al. (2014) studied the impact of adding secant plans. then, Zaghrouti et al. (2020) proposed another version of ISUD which consists of exploring the neighbourhood around the current solution to achieve a better improvement. ISUD was tested on very large SPP instances with up to 500,000 variables, and performed very well computational results.

Recently, Tahir et al. (2019) introduced an Integral Column Generation (ICG) algorithm. The authors proposed an interesting approach that merges ISUD and CG. This primal-based method has successfully solved large-scale vehicle and crew-scheduling problem (VCSP) and airline crew pairing problem (CPP) instances, and outperformed two well-known column generation-based heuristics. The authors argue the validity of the algorithm in the case where the SPP has side constraints.

Overall, throughout the literature, we notice that branch-price are among the most commonly investigated approaches, whereas primal-based methods have not been dealt with in depth. We strongly believe that the theory behind the primal methods, as well as the striking efficiency of recent primal algorithms such as ISUD and ICG on large-sized problems, make them potential candidates to be tested on complex combinatorial problems.

5.2.4 Contributions and Organization of the Paper

This paper outlines a new paradigm to solve a difficult combinatorial problem. To our knowledge, this is the first work that shows how to solve a complex real-world vehicle routing problem by means of a primal-based algorithm embedded in an exact column generation scheme. Relevant contributions of this paper are:

— We formulated our model over a real cyclic city network, unlike most vehicle and crew scheduling problems. In addition, to date, Integral Simplex has been applied only when the tasks are ordered and can be easily aggregated. However, we tackled a generic

problem where the latter assumption is not fundamental.

— We explain how ICG does not rely on branching to generate integer solutions. Indeed, in most cases, the adopted dynamic space augmentation strategy makes it possible to achieve integrality. Otherwise, a small mixed integer problem (MIP) is used.

— We used a novel pricing procedure based on primal-driven duals. Through a theoretical and an empirical study, we stated that such *duals* favor the generation of columns that can potentially be part of an improved integer solution. Actually, on average, at each iteration, more than 70% of the generated columns belong to an improved solution.

— The overall algorithm was successful in generating a set of near-optimal solutions for 30 real-world instances with up to 199 customers. ICG succeeded in reducing the computation time by more than seven, when compared to the classical branch-and-price diving heuristic (DH).

— Finally, the ICG algorithm can be easily implemented in the practical context. In addition to its performance, ICG allows us to: (i) display numerous solutions so the planner can select the most viable one; (ii) stop the process at any time with a guarantee of obtaining a feasible solution, unlike B&P methods; and (iii) promote the capitalization of the user's know-how thanks to the effective use of the primal information.

This paper is organized as follows. Section 5.3 describes our problem statement. Section 5.4 gives a brief overview of the ISUD algorithm. Section 5.5 deals with the mathematical formulations. Section 5.6 details the methodology used to solve the problem. Section 5.7 describes the acceleration strategy. Section 5.8 reports the computational results and analyzes algorithm performance. Finally, some of our conclusions are drawn in Section 5.9.

5.3 Problem Description

The company that participated in this case study is a third-party logistics provider that offers a spectrum of tailored services from warehousing, freight forwarding, distribution, and cross-docking to IT solutions. Starting from a logistics service center, the operator schedules next-day deliveries to his nationwide final customers. The specific continental context implies several technical and business-rules constraints that every shipment must satisfy. The attributes are mainly related to customers, vehicle-specific characteristics, network structure, objective function and laws and regulations. This rich last mile delivery problem with time windows, a heterogeneous fleet and external courier service must satisfy every customer's request with exactly one visit while considering: vehicles with multiple capacities, types, costs and ownership; multiple time windows associated with customers; incompatibility

constraints between vehicles and customers; maximum driving time; transportation of goods and urban accessibility regulations; the possibility of using express courier services and the possibility of open routes that do not return to the depot. The network of customers can be separated into three classes depending on the related business rules. A customer is either a downstream or upstream user, which triggers additional constraints as shown in Table 5.1.

Table 5.1 Client characteristics

Index	**Class**	**Description**
$\boldsymbol{C^1}$	Large retailers	High turnover Strict time windows and large service time Set of customers to visit
$\boldsymbol{C^2}$	Stores and shopping malls	Located in metropolitan areas and residential districts Strict time windows and large service time Specific rules applied to vehicles and accessibility
$\boldsymbol{C^3}$	Distant retailers	Small deliveries Poorly connected areas

Note that the company uses its own vehicles and, if necessary, it calls for an outsourced fleet. The latter is owned by partners and the pricing is previously negotiated. Since a customer can be served by an internal or an external vehicle, this does not affect the transport rate.

Remark 2. *Sometimes the company finds it useful to deliver to some customers (C^3) using the express courier service (external fleet). This decision arises in two cases: when serving isolated customers, or when delivering small, fragmented orders known as lighter and high value consuming goods.*

5.4 Preliminaries

ISUD is a building block of our solution method. Thus, this section sheds light on the essential theory behind. See (Tahir et al., 2018; Elhallaoui et al., 2011; Zaghrouti et al., 2014) for an extensive study. We would like to solve the following SPP:

$$z^* = \max_{\boldsymbol{\theta}} \left\{ \mathbf{p}^\top \boldsymbol{\theta} \;\middle|\; \boldsymbol{A\theta} = \boldsymbol{e},\, \boldsymbol{\theta} \in \{0,1\}^n \right\} \tag{5.1}$$

In the remainder of this paper, we use lower-case (upper-case) bold symbols for vectors (matrices).

Let $\mathbf{A} \in \{0,1\}^{m \times n}$ be the binary constraint matrix. Let $A_1, A_2, \ldots A_n$ be the columns in $\mathbf{A}$ indexed in $\Omega = \{1, 2, \ldots, n\}$ such that A_j denotes the j^{th} column in $\mathbf{A}$. Let $\mathbf{A}'_{LJ}$ be the

submatrix of $\mathbf{A}$ with rows and columns indexed by L and J such that $L \subseteq N = \{1, 2, \ldots, m\}$, and $J \subseteq \Omega = \{1, 2, \ldots, n\}$. We note by C($\mathbf{A}$) or span($\mathbf{A}$) the subspace spanned by the linear combination of columns of $\mathbf{A}$.

$\mathcal{F}_{SPP} \subseteq \{0, 1\}^n$ denotes the feasible region of SPP, $\mathcal{F}^R_{SPP}$ denotes the feasible region of the linear relaxation of SPP. Let $\boldsymbol{\theta}$ be the current feasible solution, $\boldsymbol{\theta}^*$ the optimal solution and z^*_{SPP} the optimal solution value. We denote by $\boldsymbol{\theta^t}$ the current solution at t-th iteration, and by $\boldsymbol{\theta^{t+1}}$ the next solution that we want to reach. We denote by $\boldsymbol{d}$ the ascent direction in the sense that any move in such direction will increase the cost function value such that: $\boldsymbol{\theta^{t+1}} = \boldsymbol{\theta^t} + \rho\ \boldsymbol{d^t}$, where $\rho > 0$ is an integer step size. For a current solution $\boldsymbol{\theta}$, let S denote the index set of the basic non-degenerate (nonzero) variables. Likewise, let S^* be the index set of nonzero basic variables in an optimal solution $\boldsymbol{\theta}^*$. Throughout this paper, we use S, *the reduced basis* and *the current solution* interchangeably. Likewise, we use the words *column* and *variable* interchangeably. As outlined in the literature review, the primal methods focus primarily on SPPs, the greatest limitation of which lies in their high degeneracy. However, these problems satisfy the Trubin theory (Trubin, 1969) stating that every edge of $\mathcal{F}^R_{SPP}$ is an edge of $\mathcal{F}_{SPP}$. Consequently, SPP is said to be quasi-integral and it is possible to perform linear programming pivots to reach integer extreme points. Thus, there exists a sequence of augmenting solutions such that θ^{t+1} is an adjacent vertex of θ^t in $\mathcal{F}^R_{SPP}$.

ISUD is recognized as being one of the most important primal methods for SPPs. Its salient feature lies in the decomposition technique taking advantage of the SPP's degeneracy. The horizontal and the vertical decomposition of the RMP involves rows/constraints reduction and columns/variables reduction. Thus, the RMP is split into two smaller linear programs and its re-optimization is easily performed. To learn more about the decomposition mechanism, let us first introduce the following definitions:

Definition 3. *A column A_j is **compatible** with S if $A_j \in Span(S)$, i.e., it can be written as a linear combination of the columns in S, otherwise it is incompatible. A variable associated with a compatible column is compatible.*

Definition 4. *The **incompatibility degree** δ_j of a column A_j towards an integer solution is a metric measure that represents a distance of A_j from the solution. We note that $\delta_j = 0$ if the column A_j is compatible.*

Definition 5. *A compatibility matrix* $\mathbf{M}$ *is defined by:* $\boldsymbol{MA}_r = 0, \quad \forall r \in C.$

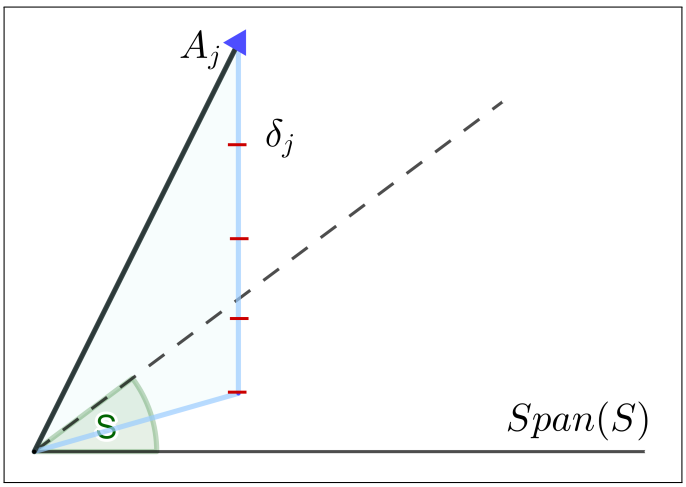


Figure 5.2 Geometric insight of the incompatibility degree of a column A_j

Among the several forms proposed in the literature, we have implemented the algorithm by using the compatibility matrix $\mathbf{M_2}$, proposed by Bouarab et al. (2017) to measure the deviation from the sequencing of tasks in the vehicle routes. ISUD makes use of those features and decomposes the columns in A into three sets such that:

$$A = \begin{bmatrix} S & C & I \end{bmatrix}$$

where S, C and I denote the index sets of the current basic integer solution, compatible and incompatible variables, respectively. This decomposition is mainly justified by the fact that (i) the degenerate basic variables are incompatible; and (ii) the compatible variables yield a non-degenerate pivot while entering into the basis. The ISUD algorithm performs towards the following structure:

Algorithm 3: *ISUD* pseudocode

```
1 Find initial solution θ^0 and set θ̄ ← θ^0
2 [S  C  I] ← Partition the binary matrix A into column subsets
3 do
4 |   Solve RMP(θ̄, C) to improve the current solution
5 |   (Z^CP, d) ← Solve CP to find an ascent direction
6 while Z^CP > 0 and d is integer
7 θ̄ = θ̄ + d
8 return θ̄
```

As described in Algorithm 3, the RMP is dynamically decomposed into two small subproblems:

the complementary problem (CP) and the reduced problem (RP). The linear program CP involves incompatible columns, and at each iteration, it aims to find an integer ascent direction $\mathbf{d}$. The latter is a compatible convex combination of incompatible columns leading to primal basic solutions with improved objective value. Moreover, the ascent direction yielded by CP is minimal in the sense that if only one of its variables is omitted, the convex combination is no longer compatible and cannot pivot into the basis. On the other hand, RP handles the compatible columns and seeks to improve the current solution.

Remark 3. *The ascent direction yielded by CP generates the null space of A such that $A\mathbf{d} = \mathbf{0}$. The latter could be an infinite cone. Thus, normalization constraints $\mathbf{w}^\top \boldsymbol{d} = \boldsymbol{1}$ are added to bound the complementary problem. More details on this will be given in Section 5.6.*

The CP formulation is based on the constraints aggregation methods first introduced by Elhallaoui et al. (2011), which implies linear transformations and substitutions of the constraint matrix $\mathbf{A}$. Thus, rows (constraints) and columns (variables) are eliminated from the linear program and the density of the CP is reduced. Let v_r, $r \in I$, λ_l, $l \in S$ be the weight variables defining the linear combination of the columns in I and S, respectively. And let w_r, $r \in I$ be the weight variables in the normalization constraint. Using the notation of Zaghrouti et al. (2014), the relaxed CP is formulated as follows:

$$\max_{v,\lambda} \quad \sum_{r\in I} p_r v_r - \sum_{l\in S} p_l \lambda_l \tag{5.2}$$

$$\text{s.t.:} \quad \sum_{r\in I} A_r v_r - \sum_{l\in S} A_l \lambda_l = 0 \tag{5.3}$$

$$\sum_{r\in I} w_r v_r = 1 \tag{5.4}$$

$$v_r \geq 0, \qquad \forall r \in I \tag{5.5}$$

The objective function (5.2) is the reduced cost, measuring the improvement in the objective value. The compatibility constraints (5.3) ensure that the linear combination of the pivoting columns is compatible with S, following Definition 3. The normalization constraint (5.4) bounds the problem. Finally, the non-negativity constraints are enforced through (5.5).

As mentioned above, the *RP* involves the compatible columns and keeps only the constraints corresponding to non-degenerate basic variables. Let A'_C be the submatrix of $\mathbf{A}$ of order $(|S| \times |C|) = (p \times n)$, formed of its first p-linearly independent rows and all its columns.

Observe that $S \subseteq C$ (each column is linearly dependant of itself), and the rows in A'_C are suitably re-ordered.

The RP is given by the following formulation:

$$\max_{C} \quad p_C \,.\, \theta_C \tag{5.6}$$

$$\text{s.t.:} \quad A_C \theta_C = \mathbf{e} \tag{5.7}$$

$$\theta_C \in \{0,1\}^{|C|} \tag{5.8}$$

Note that Bouarab et al. (2017) have proved that a pivot on any compatible column with a positive reduced cost is non-degenerate and strictly improves the objective function value.

In the next sections, we describe our solution method based on ISUD, and we show why it works for a complex routing problem.

5.5 Mathematical Formulation of the Problem

In the following section, we define the set partitioning master problem and the corresponding pricing subproblem regarding a real-world rich vehicle routing problem defined on a cyclic city network.

5.5.1 Notation

For the remainder of this paper, we use the following notation:

Table 5.2 Mathematical notation

Notation	Type	Description
Sets		
Ω	–	Set of feasible routes
N	–	Set of customers
K	–	Set of heterogeneous vehicles of type k
Parameters		
a_r^n	binary	Is equal to 1 if the customer n is visited by route r
ξ^n	real	Delivery demand of customer $n \in N$
η^k	real	Total travelled distance by the vehicle $k \in K$
p_r	real	Profit collected by the route $r \in \Omega$
ψ^k	real	Capacity of the vehicle $k \in K$
σ^n	real	Service time at the customer $n \in N$
ω^k	real	Total working time of the driver on vehicle $k \in K$
τ^k	real	Total travelling time of vehicle $k \in K$
Variables		
θ_r	binary	Is equal to 1 if the route r is selected

5.5.2 Master Problem

The MP of the problem outlined in Section 5.3 is formulated as a SPP which is fully compliant with a CG framework.

$$(SPP): \quad \max_{\theta} \quad \sum_{r \in \Omega} p_r \theta_r \tag{5.9}$$

$$\text{s.t.} \quad \sum_{r \in \Omega} a_r^n \theta_r = 1 \qquad \forall n \in N \tag{5.10}$$

$$\theta_r \in \{0, 1\} \qquad \forall r \in \Omega \tag{5.11}$$

Each row of the formulation represents a task to perform, while each column represents a feasible vehicle route. These routes occur as paths on suitable networks and are generated by an SPPRC pricing procedure (Irnich and Desaulniers, 2005). Each variable θ_r is associated with a feasible route $r \in \Omega = \{1, 2, \ldots .n\}$ which specifies a sequence of customers $n \in N = \{1, 2, \ldots .m\}$ to visit. The objective function (5.9) aims at maximizing the profit made by the feasible route r (see Section 5.5.2). The partitioning constraints (5.10) guarantee that each customer is delivered exactly once. The integrality restriction is formulated by means of (5.11).

Cost Function. Profit p_r describes the financial benefit generated from operating the trip r. It is calculated as revenue (the transport rate $\mathcal{P}(\xi^n)$) less expenses (the transport cost $\mathcal{C}^k$). Note that data processing was necessary to accurately model and predict the nonlinear cost function involved. Pricing the trip sustained by vehicle k implies fixed and variable costs. On the one hand, components making up fixed costs F^k are equipment costs, license fees, taxes, insurance, management and overhead. This amount is fixed whether the truck is used or not. On the other hand, variable costs include fuel, labor, tires, and maintenance and repair. This amount, denoted by V^k, changes in relationship to the ownership and the characteristics of the truck k, and is attributable to the unit of distance. That is, the total transport cost is given by:

$$\mathcal{C}_T^k \quad = \quad V^k \times \eta^k + F^k \qquad \forall k \in K \tag{5.12}$$

The transport rate is the price paid by the customer n to have his order ξ^n delivered. It is a function per piece, which varies according to a specific pricing grid. The rate is quoted by positioning ξ^n in the pricing matrix, the columns of which are the non-overlapping volume intervals, and the rows represent the cities (the source is always the depot). Let R_v^n be the rate associated with the v^{th} volume interval $[a_v,\ b_v]$, the transport rate is given by:

$$\mathcal{P}(\xi^n) \quad = \quad R_v^n \times \xi^n \qquad a_v \leq \xi^n \leq b_v \quad \forall v = 1, 2, \ldots, 30 \tag{5.13}$$

In the following section, we present the formal definition of the SP and the outline of the elements required for the SPPRC pricing procedure adapted to the rich vehicle routing problem.

5.5.3 Pricing Subproblems

First, we will provide an overview of the SPPRC, in particular the definition of the resources that were customized to suit all of the problem attributes. Next, we will give the mathematical formulation of the SP. Note that for simplicity, we omit the k-index.

SPPRC pricing procedure. We start with a current RMP for which the basis is iteratively enriched by entering non-basic variables. This is accomplished by solving one or several pricing subproblems. That is, we want to find columns with a positive reduced cost. If not, the current solution of the RMP solves the MP optimally as well. The subproblem amounts to an SPPRC, defined on a network $G(V, A)$ with regard to the vehicle k, where V is the subset of nodes and A is the subset of arcs. Let $\epsilon = s - i$ be a partial path from the depot

$s = O$ to the customer i. A feasible route $r = s - i_1 - i_2 - \cdots - i_n - t$ is a sequence of feasible partial paths, starting and finishing at the depot and fulfilling all of the resource constraints outlined in further detail below. Let $\mathbf{\Gamma} = \{\mu_1, \mu_2, \ldots, \mu_{|\Gamma|}\} = \{c, W, D, T\}$ be the set of resource constraints. As shown in Figure 5.3, each label L_i^μ is associated with node $i \in V$. The label is a 4-dimensional resource vector that computes the accumulated amount of resource μ across the arcs. Each arc $(i, j) \in A$ has a 4-dimensional resource consumption vector $\boldsymbol{\nu}_{ij}^\mu$. A resource frame $[a_i^\mu, b_i^\mu]$ is associated with each resource $\mu \in \mathbf{\Gamma}$.

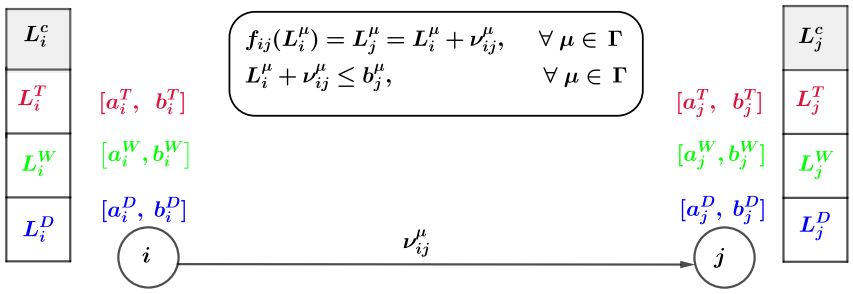


Figure 5.3 One step extension of SPPRC

The solution procedure is based on a dynamic programming algorithm as described in (Irnich and Desaulniers, 2005), whereas the resource constraints were carefully tailored with regard to the real-world constraints. In brief terms, the labelling algorithm explores the search space by associating labels or states to each node. This search is based on an extension function that allows us to create new labels from i to the successor nodes. A label is created only if the feasibility test is successful; the resource consumption respects its related window. The resources description is given in Table 5.3.

Table 5.3 Resources description

Resource	Window	Description
$L_i{}^T$	$[a_i, b_i]$	The time resource indicates the arrival time in minutes at customer $i \in V$. L_i^T could be less than a_i, i.e., driver might arrive before the start of the delivery period.
L_i^D	$[0, Q]$	The demand resource specifies the total volume carried, up to customer $i \in V$.
L_i^W	$[0, 480]$	The working time resource specifies the total time in minutes travelled by the driver, from the source s to customer $i \in V$.
L_i^c	–	The cost resource is unconstrained, and is used to compute the reduced cost $\bar{p}_{ij}$ of the travel arc $(i, j) \in A$.

Remark 4. *Since the network is cyclic, path-structural constraints are added in order to eliminate the cycles.*

At each iteration, the resource-extension function generates new updated labels at $j \in V$ as follows:

$$f_{ij}(L_i^T) = \max\left(L_i^T, a_i\right) + \tau_{ij} + \sigma_i \tag{5.14}$$

$$f_{ij}(L_i^D) = L_i^D + \xi_{ij} \tag{5.15}$$

$$f_{ij}(L_i^W) = L_i^W + \tau_{ij} \tag{5.16}$$

$$f_{ij}(L_i^c) = L_i^c + \bar{p}_{ij} \tag{5.17}$$

Subproblem formulation. The purpose of the subproblem is to find routes that are likely to improve the objective function, such that:

$$\bar{p}_r = p_r - \sum_{n \in N} a_r^n \pi^n \ > 0$$

where $\boldsymbol{\pi} = \{\pi^n |\ n \in N\}$ stands for the vector of the *duals* used to price out the columns and is associated with constraints (5.10). A route r is a set of arcs $(i, j) \in A$, thus, we introduce the arc variable x_{ij}^r which is equal to 1 if (i, j) is included in r, and 0 otherwise. That is, a cost of route r can be expressed as:

$$p_r = \sum_{n \in N} p_{ij} x_{ij}^r$$

The reduced cost of arc (i, j) can be expressed as:

$$\bar{p}_{ij} = p_{ij} - \sum_{n \in N} a_{ij}^n \pi^n$$

Finally, on combining the aforementioned considerations, the subproblem can be defined as follows:

$$(SP): \quad \max_{x} \quad \sum_{(i,j) \in A} \bar{p}_{ij} x_{ij} \tag{5.18}$$

$$\text{s.t.} \quad \sum_{i \in V} x_{ij} - \sum_{i \in V} x_{ji} = \begin{cases} -1 & \text{if} \quad j = s \\ 0 & \text{if} \quad j \in V \setminus \{s, t\} \\ 1 & \text{if} \quad j = t \end{cases} \tag{5.19}$$

$$x_{ij} \left(L_i^\mu + \nu_{ij}^\mu - L_j^\mu \right) \leq 0 \qquad \forall \mu \in \mathbf{\Gamma}, \quad \forall (i, j) \in A \tag{5.20}$$

$$a_i^\mu \leq L_i^\mu \leq b_i^\mu \qquad \forall \mu \in \mathbf{\Gamma}, \quad \forall i \in V \tag{5.21}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in A \tag{5.22}$$

The objective function (5.18) aims at maximizing the profit of the routes. The flow conservation constraints (5.19) ensure that the graph is connected from s to d. The resource variable values are updated if the arc (i, j) is a component of the route; this is expressed by the constraints (5.20). The resource constraints are ensured by (5.21). Finally, the binary requirement on the arc variables is expressed by (5.22). The detailed description of the expert-guided network is given in a dedicated section (see Section 5.7).

5.6 Methodology

Our solution method (ICG) is geared towards CG and ISUD algorithms. ICG exploits the degeneracy-based decomposition and the reliability of the CG framework. This section examines the methodology we undertook, and the improved measurements that were tailored in accordance with this last mile transportation problem.

5.6.1 Description of the ICG Algorithm

As shown in Figure 5.4, ICG is a three-stage sequential algorithm. The first two stages involve the RMP, solved using ISUD (see Algorithm 3), whereas the third stage involves the CG subproblems, as previously defined. ICG takes advantage of the irrefutable performance of

CG with the addition of the dynamic decomposition of the RMP, which is reduced vertically and horizontally. Indeed, experiments have shown that horizontal reduction of constraints has the greatest impact on the size of the current basis, thus allowing a faster re-optimization of the RMP.

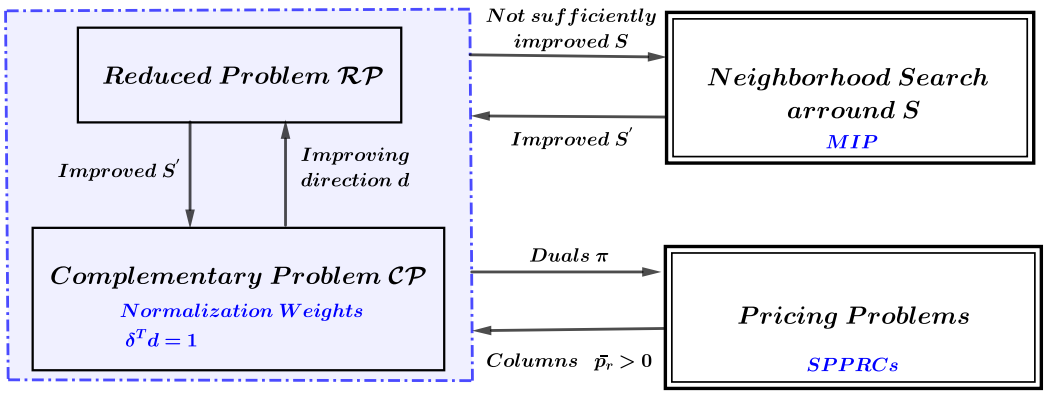


Figure 5.4 Integral Column Generation Algorithm

We pointed out earlier that normalization constraints $\mathbf{w}^\top \mathbf{d} = \mathbf{1}$ are used to bound the CP (see Section 5.4). In fact, the constraints structure strongly impacts an algorithm's performance. In the course of their research, Rosat et al. (2014) identified the existence of a normalization constraint that may lead to the optimal solution. However, no techniques to find these constraints were provided. Through experimentation, we use incompatibility degree δ_k as a weight variable. This seems to significantly penalize the fractional directions.

Algorithm 4: ICG pseudo-code

Parameters : $maxConsFail$, $maxDegree$, $minImp$.
Initialize : $t \leftarrow 0$; (θ^0, π^0); $consFail \leftarrow 0$
Output : (z^*, θ^*)

repeat
  Step 1: CG
  $\Omega' \leftarrow$ Solve the $SP(\pi^t)$
  if $\Omega' = \varnothing$ **then**
    break
  end
  $\Omega \leftarrow \Omega' \cup \Omega$
  $t \leftarrow t + 1$
  Step 2: RMP
  $(\theta^t, z^t, \pi^t) \leftarrow$ Solve the RMP using ISUD
  if $\frac{z^{t-1} - z^t}{z^{t-1}} \leq minImp$ **then**
    $consFail \leftarrow consFail + 1$
    $\theta^t_{NS} \leftarrow$ search an improved solution around θ^t by solving restricted MIP
    if $z^t_{NS} > z^t$ **then**
      $\theta^t \leftarrow \theta^t_{NS}$
      $(\theta^t, z^t, \pi^t) \leftarrow$ Resolve RMP using ISUD
    end
  else
    $consFail \leftarrow 0$
  end
  $(z^*, \theta^*) \leftarrow (z^t, \theta^t)$
until $consFail \geq maxConsFail$
return (z^*, θ^*)

Our steps proceed in the same way as described in Algorithm 4:

— Initialization step: The initial primal solution θ^0 is built such that, in Step 1, each customer i is served by one vehicle with a large cost M. The initial dual solution π^0 is built such that each dual value is set to M. Note that the quality of (θ^0, π^0) significantly impacts the quality of ISUD solutions, especially for the large-sized instances.

— Step 1: At each iteration, CG subproblems $SP(\pi^t)$ are solved using the *duals* π^t corresponding to the current solution θ^t. If $\bar{p_r} < 0$, there is no improving variable, the algorithm stops and the best solution found θ^* is returned. Otherwise, the promising variable found is added to the RMP and we repeat.

— Step 2: ISUD is used for optimizing the RMP in order to determine the current solution θ^t, its objective function value z^t, and the dual multipliers π^t. If the improvement test is

successful, i.e., $\frac{z^{t-1}-z^t}{z^{t-1}}$ attains the parameter $minImp$ (the level at which improvement is considered sufficient), we are done with the current solution. Otherwise, a neighborhood search, defined by the incompatibility degree δ_r, is explored around θ^t by means of a small MIP. If $z^t_{NS} > z^t$, then θ^t_{NS} becomes the current solution and is used to re-optimize the RMP. We repeat until the number of consecutive improvement failures $consFail$ reaches $maxConsFail$. In the next section, this step is given in more detail.

Note that in Step 1, the MIP is invoked only if the CP stops with a non-integer ascent direction. The latter is used to define a small-sized neighborhood around the current integer solution, which is easily solved using a commercial MIP to find an integer direction. Otherwise, solving with a MIP can be omitted.

Finally, we remind the reader that ICG is an exact algorithm using a heuristic stopping criterion that guarantees, in practice, an optimal or near-optimal solution.

5.6.2 Primal-driven Dual Solutions

It has been argued by Zaghrouti et al. (2014) that the performance of ISUD is mainly based on the dynamic decomposition of the RMP. However, this leads to a major downside: unlike B&P, it is impossible to calculate a complete dual solution. Indeed, we consider a smaller subset of variables and constraints; as a result, we get aggregated duals instead of one for each of the original tasks as required by the pricing subproblem.

To overcome this limitation, appropriate *duals* are calculated according to the procedure proposed by Bouarab et al. (2017) and adapted to routing problems. In this context, the *duals* correspond to the current solution and are called *primal-driven*. A dual solution $\boldsymbol{\pi}$ is *primal-driven* if the reduced cost of the non-degenerate variables in the basic current solution equals zero with respect to $\boldsymbol{\pi}$. Throughout the paper, we use the term *duals* in accordance with this definition.

We can see that $\boldsymbol{\pi}$ involves the primal information contained in the current solution, and thus exploits the foreknown practical routes and schedules. The notion of duality is crucial, and deserves greater interest especially in the primal context. The following two propositions assert that the appropriate use of the *duals* not only decides which variables will enter into the basis, but beyond that, they contribute to a reduction of the number of iterations as well as the running times, on average. Indeed, it is possible to generate several components of the optimal solution and the improving ascent direction in a single iteration. Through the following propositions, we provide compelling evidence for the validity of the procedure used to compute $\boldsymbol{\pi}$. The numerical results support our theoretical findings (see Section 5.8.4).

Proposition 1. *As long as we do not reach optimality, at least one dual solution* $\boldsymbol{\pi}$ *exists such that every entering variable in the integer ascent direction* $\boldsymbol{d}$ *has a positive reduced cost* $\bar{p_r} = p_r - \sum_{n \in N} a_r^n \pi^n > 0$ *with respect to* $\boldsymbol{\pi}$.

Proof. Let π and y be the *duals* associated to constraints (5.3) and (5.4), respectively. The dual problem of the CP (see Section 5.4) is formulated as follows:

$$(DCP): \quad y^* = \min_{y,\pi} \quad y \tag{5.23}$$

$$\text{s.t.} \quad p_r - \boldsymbol{\pi}^\top \cdot A_r = 0 \qquad \forall r \in S \tag{5.24}$$

$$p_r - \boldsymbol{\pi}^\top \cdot A_r \leq y \; w_r \qquad \forall r \in I \tag{5.25}$$

Let I_Q denote the index set of columns in the minimal integer ascent direction d. If the linear program (DCP) is restricted to $I_Q \subset I$, then the constraints (5.25) are saturated, i.e., $p_r - \pi^T \cdot A_r = y^* w_r, \; \forall r \in I_Q$. On the other hand, by construction, I_Q involves a minimal combination of incompatible columns yielding an ascent direction, which means that $y^* > 0$, and $w_r, \; r \in I$ are the weight variables in the normalization constraints and assumed to be positive. Therefore, the reduced cost $(p_r - \pi^T \cdot A_r) > 0, \; \forall r \in I_Q$ which completes the proof. □

Proposition 2. *For a current solution* $\boldsymbol{\theta}$*, let* S *be the set of nonzero variable indices. Likewise, let* S^* *be the set of nonzero variable indices in an optimal solution* $\boldsymbol{\theta^*}$*. At least one dual solution* $\boldsymbol{\pi}$ *exists such that every variable* $\theta_r^*, \; r \in S^* \setminus S$ *has a positive reduced cost* $\bar{p_r} = p_r - \sum_{n \in N} a_r^n \pi^n > 0$ *with respect to* $\boldsymbol{\pi}$.

Proof. Let $\boldsymbol{\pi}$ be a dual solution corresponding to the current solution $\bar{\boldsymbol{\theta}}$ and $S := \left\{ r \,|\, \bar{\theta_r} \neq 0 \right\}$ the set of nonzero variable indices in $\bar{\boldsymbol{\theta}}$. Let $D^q = \{ j \mid d_j^q > 0 \}$ be the set of minimal ascent directions, such that d^q is the ascent direction leading from θ^q to $\theta^{q+1} := \theta^q + \rho^q \; d^q, \; \rho > 0$. Let θ^q be the current feasible integer solution at iteration q and θ^{q+1} is its adjacent solution. Let us consider the following linear program:

$$(DCP): \quad y^* = \min_{y,\pi} \quad y \tag{5.26}$$

$$\text{s.t.} \quad p_r - \boldsymbol{\pi}^\top \cdot A_r = 0 \qquad \forall r \in S \tag{5.27}$$

$$p_r - \boldsymbol{\pi}^\top \cdot A_r \leq y \; w_r \qquad \forall r \in S^* \setminus S \tag{5.28}$$

1. If $y^* > 0$, then all variables $v_r, \; r \in S^* \setminus S$ of the primal problem have a positive reduced

cost. Hence, the assertion is true.

2. If $y^* \leq 0$, then $D^q \in S^* \setminus S$ exists such as $p_r - \boldsymbol{\pi}^{\boldsymbol{T}} \cdot A_r = y^* w_r,\ \forall r \in D^q$. This is contrary to assumption because d^q is a minimal ascent direction (i.e., the reduced cost of variables $v_r,\ \forall r \in D^q$ must be positive) and the weight variables w_r are positive, which completes the proof.

□

We have provided a theoretical support for the validity of the duality notion in this primal context. ICG relies on a dual solution which favors the generation of promising columns that are likely to improve the current integer solution. We remind the reader that the RMP can handle a very large number (pool) of columns, which attenuates the outcome of the *duals* quality. The tracking of this impact is further detailed in a dedicated section (see Section 5.8.4).

5.7 Acceleration Strategy

Given the complexity of the real context, we proposed invoking two different techniques to improve the overall performance of the solution mechanism. The relevance of our choice is proven through the empirical results in Section 5.8.

5.7.1 Dynamic Augmentation of the Search Space (DASS)

In the ICG algorithm, branching is never performed to obtain an integer solution. Our last relaxed CP could yield a non-integer direction (leading to a non-integer solution), hence, we recommend using the multi-phase strategy. This technique was proposed by Zaghrouti et al. (2014) to improve the performance of the standard ISUD algorithm. Broadly, the aim of DASS is to dynamically augment the search space by including new incompatible columns in the CP, as needed. As explained in Example 5.7.1, solving a CP could require a sequence of $k_1 < k_2 < \cdots < k_p$ phases. In each Φ_k, we solve a restricted CP, involving only columns that are at distance $||\boldsymbol{M} A_j|| \leq \delta_k, \forall j \in I$ from the current solution.

Note that considerable attention must be paid at the subproblem level. In fact, an additional resource constraint must be added such that $\delta_j \leq k$ for every feasible column j. In Bouarab et al. (2017), the authors demonstrated that δ_k influences the density of the CP's constraint matrix. In fact, in phase Φ_k, every column has at most $k+1$ nonzero coefficients. Hence, we recommend starting with a low δ_k and increasing it as needed until until k exceeds a parameter value *maxDegree*.

DASS strategy: illustrative example. As put forward, DASS enables us to avoid the branching procedure. To illustrate this proof-of-concept, we apply the ICG algorithm on the following set partitioning problem. To ensure accuracy of our example, it was implemented as a linear programming model and solved by Cplex.

$$
\begin{array}{lccccccccccccccccccc}
\min\limits_{\theta} & 10\,\theta_1 & + & 8\,\theta_2 & + & 16\,\theta_3 & + & 12\,\theta_4 & + & 10\,\theta_5 & + & 8\,\theta_6 & + & 16\,\theta_7 & + & 16\,\theta_8 & & \\
 & \theta_1 & & & & & + & \theta_4 & & & & & + & \theta_7 & + & \theta_8 & = & 1 \\
 & \theta_1 & & & & & + & \theta_4 & & & & & + & \theta_7 & + & \theta_8 & = & 1 \\
 & & & \theta_2 & & & & & + & \theta_5 & & & + & \theta_7 & + & \theta_8 & = & 1 \\
 & & & \theta_2 & & & + & \theta_4 & & & & & & & + & \theta_8 & = & 1 \\
 & & & \theta_2 & & & + & \theta_4 & & & & & + & \theta_7 & & & = & 1 \\
 & & & & & \theta_3 & & & + & \theta_5 & & & & & & & = & 1 \\
 & & & & & \theta_3 & & & & & + & \theta_6 & & & & & = & 1 \\
 & & & & & & & & & & & & & & & & \theta\in\{0,1\}^8 &
\end{array}
$$

1. Let $\theta_1 = \theta_2 = \theta_3 = 1,\ \theta_4 = \cdots = \theta_8 = 0$ be an initial solution with a cost of 34. Therefore, the index sets are: $S = \{1, 2, 3\}$, $I = \{4, 5, 6, 7, 8\}$, $C = \emptyset$ (no compatible columns with Swere found).
2. The word *cluster* refers to a feasible subset of tasks (vehicle routes); that is, incompatibility degree $\delta_j, j \in I$ is the number of additional clusters needed to make a column compatible. In a concrete way, it is the minimum number of times that task subsets in the current solution S must be divided to make the column compatible with S. For example, $\delta_r = 1$ for $r = 4, 6, 7, 8$ while $\delta_5 = 2$. Indeed, the current solution S must be split into two task subsets to make column v_5 compatible.
3. Since $C = \emptyset$, we cannot build the RP, therefore, we proceed directly to the CP.
4. We form the CP from incompatible columns, using incompatibility degrees as a weight vector in normalization constraints. Solving the CP requires a sequence of two phases $k_1 = 1 < k_2 = 2$, where $k = \delta_j, j \in I$.

$$
\begin{array}{lccccccccccccccccccc}
z^{CP^{\phi_1}} = \min\limits_{v,\lambda} & 12\,v_4 & + & 8\,v_6 & + & 16\,v_7 & + & 16\,v_8 & - & 10\,\lambda_1 & - & 8\,\lambda_2 & - & 16\,\lambda_3 & & \\
 & v_4 & & & + & v_7 & + & v_8 & - & \lambda_1 & & & & & = & 0 \\
 & v_4 & & & + & v_7 & + & v_8 & - & \lambda_1 & & & & & = & 0 \\
 & & & & + & v_7 & + & v_8 & & & - & \lambda_2 & & & = & 0 \\
 & v_4 & & & & & + & v_8 & & & - & \lambda_2 & & & = & 0 \\
 & v_4 & & & + & v_7 & & & & & - & \lambda_2 & & & = & 0 \\
 & & & & & & & & & & & & - & \lambda_3 & = & 0 \\
 & & & v_6 & & & & & & & & & - & \lambda_3 & = & 0 \\
 & v_4 & + & v_6 & + & v_7 & + & v_8 & & & & & & & = & 1 \\
 & & & & & & & & & & & & & & v\geqslant 0, \lambda \geqslant 0 &
\end{array}
$$

— Phase $\phi_{k=1}$: solving the CP for columns in $I^{k=1} = \{j \in I, \delta_j \leq 1\} = \{4, 6, 7, 8\}$ yields the optimal solution: v_4=v_7=v_8=$\frac{1}{3}$, v_6=0,$\lambda_1 = 1$ λ_2=$\frac{2}{3}$, $\lambda_3 = 0$ at a cost of $z^{CP^{\phi_1}}$=$-\frac{2}{3}$. However, this solution does not yield an integer ascent direction (no disjoint columns exist), hence increasing the current phase is required.

$$
\begin{array}{lllllllllllllllllll}
z^{CP^{\phi_2}}= & \underset{v,\lambda}{min} & 12\ v_4 & + & 10\ v_5 & + & 8\ v_6 & + & 16\ v_7 & + & 16\ v_8 & - & 10\ \lambda_1 & - & 8\ \lambda_2 & - & 16\ \lambda_3 & & \\
 & & v_4 & & & & & + & v_7 & + & v_8 & - & \lambda_1 & & & & & = & 0 \\
 & & v_4 & & & & & + & v_7 & + & v_8 & - & \lambda_1 & & & & & = & 0 \\
 & & & & v_5 & & & + & v_7 & + & v_8 & & & - & \lambda_2 & & & = & 0 \\
 & & v_4 & & & & & & & + & v_8 & & & - & \lambda_2 & & & = & 0 \\
 & & v_4 & & & & & + & v_7 & & & & & - & \lambda_2 & & & = & 0 \\
 & & & & v_5 & & & & & & & & & & & - & \lambda_3 & = & 0 \\
 & & & & & + & v_6 & & & & & & & & & - & \lambda_3 & = & 0 \\
 & & v_4 & + & \mathbf{2}\ v_5 & + & v_6 & + & v_7 & + & v_8 & & & & & & & = & 1 \\
 & & & & & & & & & & & & & & & & & v \geqslant 0, \lambda \geqslant 0 &
\end{array}
$$

— Phase $\phi_{k=2}$: solving the CP for columns in $I^{k=2} = \{j \in I, \delta_j \leq 2\} = \{4, 5, 6, 7, 8\}$, yields the integer solution : v_4=v_5=v_6=$\frac{1}{4}$, v_7=$v_8 = 0$, λ_1=$\lambda_2 = \lambda_3 = \frac{1}{4}$ and a more interesting cost of $z^{CP^{\phi_2}} = -1$. This induces an integer ascent direction.

— The algorithm stops with the optimal integer solution $S = \{4, 5, 6\}$.

5.7.2 Expert-guided SPPRC Network Model

The multi-attribute VRP invokes real, complex city networks; consequently, the subproblem is a time-consuming one. To this end, we used a clever expert-guided network which helped us to decrease the running time. As displayed in Figure 5.5, SPPRC is defined on a partially cyclic directed graph $G(A_{SD_d}, V_{SD_d})$, where $\widehat{SD_d}$ represents the shipping direction, V_{SD_d} represents the super nodes (cities or cluster of customers $n \in \widehat{SD_d}$), and A_{SD_d} represents the connecting arcs towards the direction $\widehat{SD_d}$. Cycles are permitted inside the super nodes such that the subgraph induced by vertices (customers) of city i is complete.

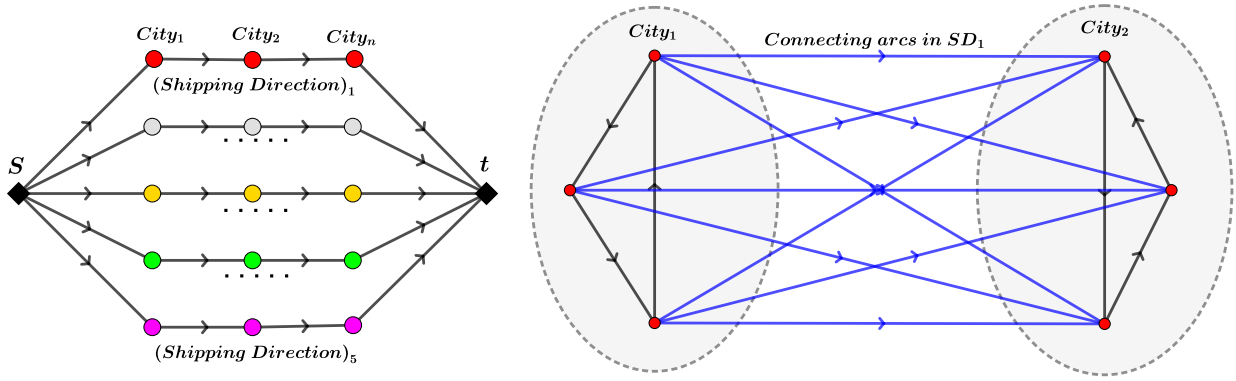


Figure 5.5 Network Modelling

Based on data analytic techniques, the following modifications were made:

— The distribution network was clustered into 5 zones.

— Shipping directions were separated into five non-overlapping delivery axes $\widehat{SD_d}, d = 1, \ldots, 5$ such that if $i \in V_{SD_d}$, hence i can exclusively be reached by arcs in A_{SD_d}.

— Infeasible arcs were discarded with respect to the real road network and traffic regulations.

— Customers sharing near locations were aggregated, for instance at shopping malls and supermarkets.

In the following section, we outline the statistical metrics which reveal the enhancing impact of our strategy.

5.8 Computational Study

Through this section, we report the numerical results obtained by the ICG algorithm on real-life rich VRP instances. The primal method ICG is tested against a basic version of a diving heuristic (DH): a column generation-based B&P heuristic. DH uses a depth-first search by exploring a single branch of the tree. At every node, among the candidate columns, the most fractional variable is set to 1. The process stops when the solution of the MP is integer. (For more details on DH algorithm, see (Joncour et al., 2010)). The experiments were realized using a C++ implementation on Linux, on workstations with 3.3GHz Intel Xeon E3-1226 processors, and 8 GB RAM. The algorithms were implemented using *IBM CPLEX*, a commercial solver (version 12.4). The SPPRC was solved by a DP algorithm using the *Boost* library (version 1.54).

5.8.1 Instance Description

The computational studies were carried out on three sets of real-life instances provided by a third-party logistics provider. The data covers orders reaching 199 customers, over a 30-day period. The fleet is heterogeneous in terms of characteristics, payload, ownership and cost. The fleet size is unlimited since the company can invoke external vehicles. The warehouse management system combines the daily shipping orders until 5:00pm. Then, the planners schedule the deliveries manually within 3 to 5 hours. The total planning horizon between the order request and the delivery is one operating day.

We used a geocoding API to convert the physical addresses to longitude and latitude coordinates. Travel time and travel distance matrices were to some extent determined using Dijkstra's shortest path algorithm on a real road map. After the initial test series, we set the parameter values as: $minImp = 0.0025,\ maxConsFail = 5,\ maxDegree = 7$.

5.8.2 Numerical Results

Table 5.4 reports the experimental data on ICG compared to DH. Instances are classified into three sets: small, medium and large. Column 2 is for the name of the instance ($Name$). Column 3 is for the number of customers ($|N|$). For both ICG and DH runs, columns 4 and 10 display the number of column generation iterations ($Iter$); columns 5 and 11 display the total computing real time in seconds ($Time$); and columns 6 and 12 display the percentage optimality gap between the cost of the best solution found and the linear relaxation optimal value (Gap). For ICG, column 7 displays the total number of integer solutions found ($No.Sol$); column 8 displays the number of times that a MIP was solved (K_{MIP}); and column 9 displays the time spent solving the MIP in seconds (T_{MIP}).

Table 5.4 Numerical results on 30 real-life instances comparing ICG and DH

Instances			**ICG**						**DH**		
Set	Name	|N|	Iter	Time(sec)	Gap (%)	No.Sol	K_{Mip}	T_{Mip}(sec)	Iter	Time(sec)	Gap (%)
	S-01	27	14	1.15	0.37	19	1	0.25	15	0.28	0.40
	S-02	32	10	0.80	0.00	7	0	0.03	7	0.11	0.00
	S-03	34	7	0.62	0.00	5	0	0.07	7	0.24	0.00
Small	S-04	40	9	2.20	0.86	7	0	0.13	8	1.02	1.30
	S-05	45	11	2.29	2.60	16	1	0.34	28	1.38	4.71
	S-06	45	11	2.11	0.53	13	2	0.54	16	0.85	0.86
	S-07	49	7	1.08	1.89	21	0	0.15	11	0.53	3.79
	M-01	62	7	4.40	0.00	5	0	0.16	11	7.67	0.00
	M-02	63	8	7.48	0.00	17	0	0.57	11	7.45	0.00
	M-03	66	8	6.61	0.24	20	0	0.63	16	6.74	0.02
Medium	M-04	66	9	7.36	1.35	25	0	0.83	39	13.99	1.74
	M-05	83	9	24.46	1.84	22	1	1.56	28	54.44	6.61
	M-06	97	9	70.84	0.00	18	0	0.88	39	253.72	1.19
	M-07	97	9	21.98	0.10	26	1	1.13	30	53.02	0.46
	M-08	98	15	67.61	2.37	40	3	5.40	48	301.89	4.45
	L-01	106	9	19.32	0.15	17	1	1.48	17	30.17	1.29
	L-02	114	11	84.74	0.47	28	1	3.96	71	618.43	0.69
	L-03	116	14	129.19	0.88	45	3	10.87	67	450.50	1.94
	L-04	123	12	12.96	0.24	24	1	2.96	40	10.47	1.22
	L-05	125	15	195.83	1.80	44	3	11.91	99	1066.80	4.86
	L-06	126	7	99.39	0.06	31	0	1.35	42	699.34	0.07
	L-07	131	9	114.31	0.00	8	1	0.52	22	466.51	1.18
Large	L-08	134	11	120.41	0.00	19	1	3.15	65	1143.15	1.82
	L-09	136	13	387.87	1.77	48	3	188.30	134	1775.68	3.87
	L-10	136	10	117.33	0.00	21	1	2.83	21	349.18	0.17
	L-11	140	8	24.16	0.37	31	0	1.67	54	91.14	1.59
	L-12	140	9	113.69	0.00	27	0	2.25	16	270.07	0.01
	L-13	152	11	277.14	0.00	31	3	5.79	121	3207.29	0.03
	L-14	160	10	89.42	0.08	46	0	3.87	21	210.42	0.03
	L-15	199	12	540.33	0.40	66	1	40.16	158	8106.62	1.55
Avg		**98**	**10.1**	**85**	**0.61**	**24.9**	**0.93**	**9.79**	**42.06**	**639.96**	**1.52**

The most remarkable result to emerge from the data is that all instances were successfully solved by ICG and DH. However, for DH, some solutions cannot be considered feasible with regard to the practical context; for example, the largest instance was solved in more than two hours. Of the 30 instances, ICG resolved half of them in less than a minute. It is easy to notice that ICG yielded an average time reduction of up to 86% compared to DH. The optimality gap varies between 0.00% and 2.6% which supports the quality of the solutions. In addition, compared to DH, we observe that the number of iterations performed by ICG is minor. This is not surprising, since ICG generates a large number of columns that are well-managed at the RMP level by: (i) reducing the density of the CP's constraint matrix using the $\boldsymbol{M_2}$ compatibility matrix; and (ii) selecting the columns to be considered in the CP using the multi-phase strategy DASS. The latter is mainly used to promote the generation of

a lot of integer solutions in few iterations.

When ISUD fails to sufficiently improve the solution, we call on a small MIP. For instance, in the first row, $K_{MIP} = 1$ $No.Sol = 19$, $T(\text{sec})_{MIP} = 0.25$ sec means that of the 19 integer solutions found, only one was obtained by solving one MIP, and the remaining 18 solutions were yielded by the multi-phase process. By examining the accurate statistical data displayed by entries $No.Sol$, K_{MIP} and T_{MIP}, we come out with some very pertinent and conclusive remarks. DASS is so effective that we rarely need to solve a MIP. For all instances, K_{MIP} is less than or equal to three. Moreover, in 43% of the instances, integer solutions were exclusively found by increasing the phase. Finally, the success rate of DASS is roughly at 96%, meaning that most of the integer solutions are obtained through the DASS, whereas the rest are found by MIP. The latter requires an average time of 1.41 sec (median value) for half of the instances.

It is worthwhile to note that tested on the largest sets, the superiority of ICG performance is striking. Indeed, Figure 5.6 shows that ICG is always faster than DH and can yield remarkable time reductions of up to 87% on average. Similar behaviour was observed by Tahir et al. (2018) while experimenting ICG on large VCSP and CPP instances.

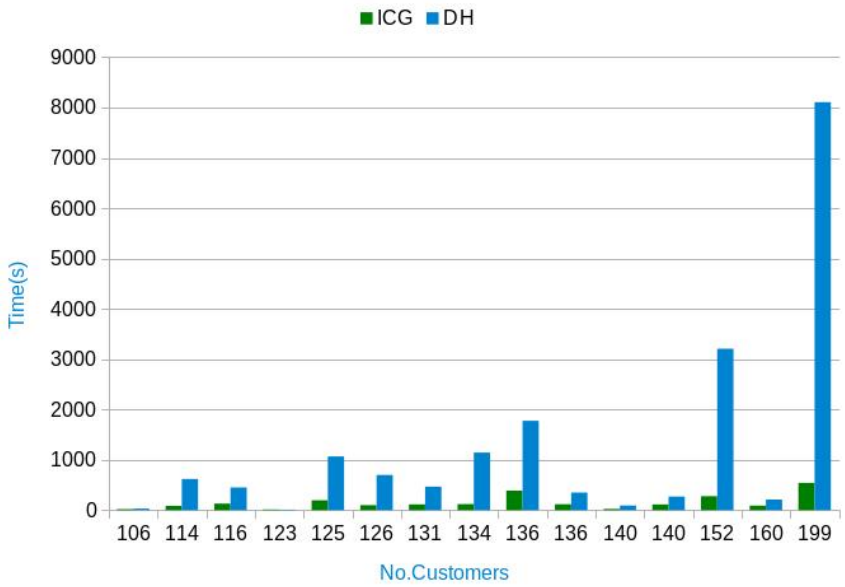


Figure 5.6 Comparing computing time of large-sized instances

Figure 5.7 and Figure 5.8 illustrate the variation of the objective value over the computing time, for two large instances (L-09; |N|=136) and (L-15; |N|=199), respectively. The two graphs reveal that ICG performs a marginal number of iterations compared to DH, and it

is able to improve the objective value drastically from early iterations. For instance, on the graph corresponding to L-09, it is easy to notice that the objective value found by DH in 1740 sec, was obtained by ICG in just 100 sec. The same behaviour was observed for the other instances. In fact, a common issue with B&P methods is the typical *tailing-off* effect, which results in slow convergence and an unnecessarily large number of iterations. Despite this, it remains possible that applying a stabilization strategy, fine tuning, and using appropriate branching rules could attenuate this issue.

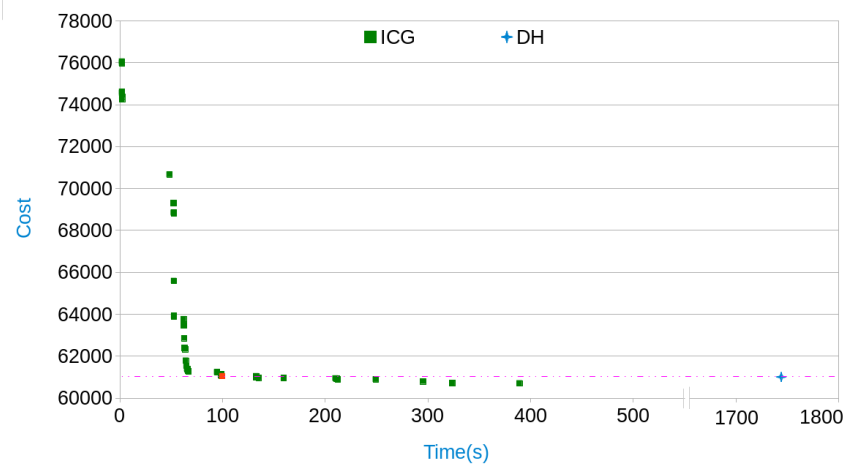


Figure 5.7 The evolution of the objective value over time for instance L-09

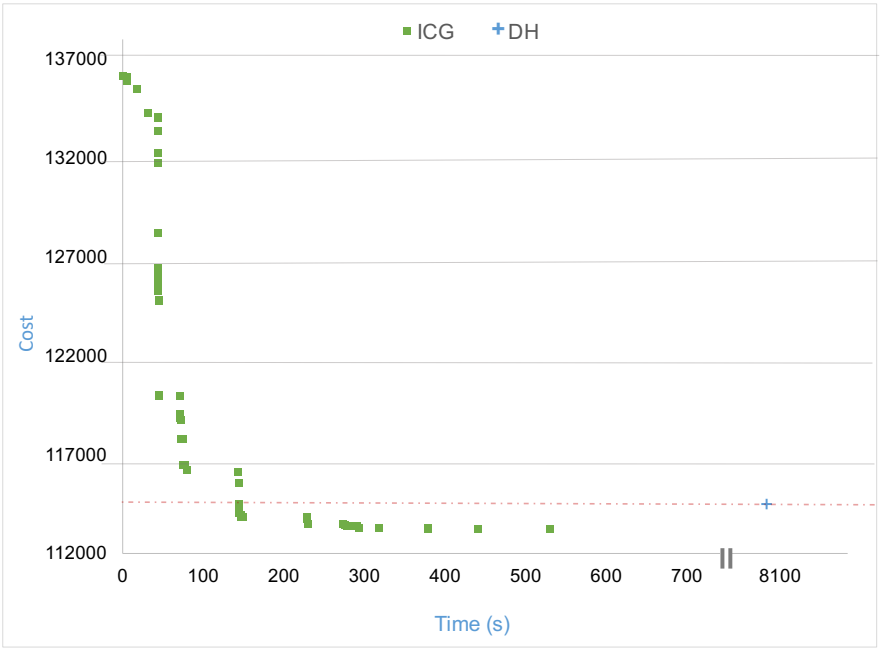


Figure 5.8 The evolution of the objective value over time for instance L-15

To the same end, Figure 5.9 features the number of columns that are a part of the final solution for both ICG (green marks) and DH (blue marks). It is remarkable that ICG needed 13 iterations to roughly yield all the solution columns, while the DH required 121 iterations (21 branching nodes) to do so. Indeed, this is not surprising since ICG generates many more columns. Also, the *duals* used exploit the primal information,thereby increasing the opportunity to obtain promising columns that that help improve solutions.

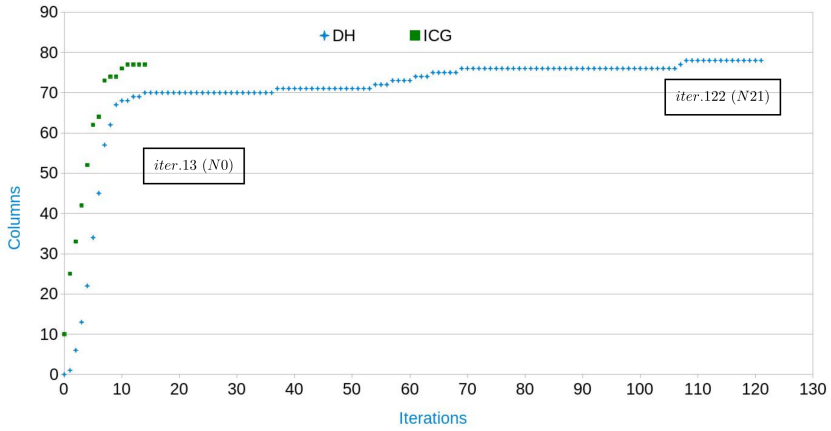


Figure 5.9 The evolution of improving columns over iterations

Overall, the numerical results and metrics clearly reveal the efficiency of ICG in solving difficult, real-world instances.

5.8.3 Acceleration Strategy Impact

Let us now look at the impact of the acceleration strategy. We conducted a set of experiments on the same problem outlined in Messaoudi et al. (2020). The authors used a basic version of the algorithm (ICG^0), where the multi-phase strategy was not implemented and the SPPRC was defined on an unprocessed road map.

Table 5.5 Acceleration strategy impact on seven real-world instances

Instances		**ICG^0**			**ICG**		
Name	|N|	Iter	Time(sec)	Gap(%)	Iter	Time(sec)	Gap(%)
S-03	34	4	0.95	0	7	0.62	0.00
S-04	40	7	4.48	0.94	8	2.20	0.86
M-04	66	9	22.6	1.3	9	7.36	1.35
L-01	106	7	32	0.14	9	19.32	0.15
L-06	126	6	126	0.09	7	99.39	0.06
L-09	136	12	515	1.62	13	387.87	1.77
L-12	140	9	168	0	9	113.69	0.00

Table 5.5 compares the performance of ICGs implemented with and without an acceleration strategy: ICG and ICG^0, respectively. The improved version yielded a substantial time decrease, reaching 27%. Thus, we can argue that the strategy favours integrality and accelerates the computing time as well.

5.8.4 Empirical Analysis of Duals

The purpose of this subsection is twofold: (i) to understand how the dual solution contributes to the ICG's performance; and (ii) to argue that our proposals put forward in the previous section are still valid in practice. We have theoretically demonstrated the existence of a dual solution such that all entering variables in the optimal solution have a negative reduced cost. To date, there is no systematic method for generating all the missing columns in the current solution (the set $S^* \setminus S$). However, we show that the dual solution we use guarantees a high percentage of missing columns with a negative reduced cost. Thus, the chances that they are generated during the resolution are enhanced. For this empirical study, we needed to implement a suitable script to track the columns of improving and optimal solutions. Given an optimal solution, we track its missing columns. At each iteration, we identify those with a negative reduced cost associated with the dual solution used in the current iteration. For a greater accuracy, it should be noted that: (i) this part used an appropriate $primal - driven - dual$ solution; and (ii) the model was transformed into a minimization problem (for implementation purposes) but the results remain valid for the maximization scenario.

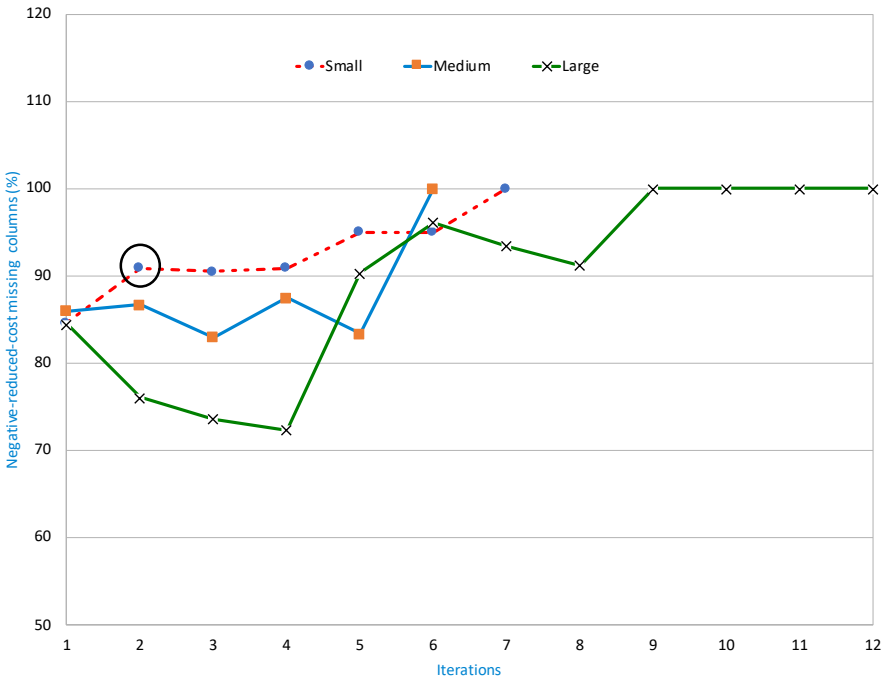


Figure 5.10 Quality of the primal-driven dual solutions tested on real-world instances

Figure 5.10 gives an overview of the quality of the primal-driven duals used. The three-line graph was obtained by conducting a series of experiments on the 30 real instances, and represents the typical behavior of the three classes of instances. The x-axis indicates the iterations, while the y-axis indicates the percentage of missing columns which have a negative reduced cost. Each line of the graph represents one instance class. For example, the circled dot (2;92%) indicates that, 92% of missing columns have a negative reduced cost associated with the dual solution given by RMP at iteration 2. In summary, our theoretical assessments seem to be well-supported by the results. At each iteration, at least 70% of the missing columns have a negative reduced cost associated with a primal-dual solution. We note that for the large-instances line, all columns were fully generated at iteration 9, but reaching iteration 12 is due to the stopping conditions of the algorithm. We remind the reader that we cannot compare to DH, for which such information is not available. As previously mentioned, B&P does not take advantage of the current solution and is purely based on the dual information. The numerical results obtained from experiments conducted on a real-life rich VRP, have led us to conclude that the ICG algorithm is theoretically and empirically valid. There is evidence to suggest that the method could be used when complex and large-sized routing problems occur, especially since its efficiency can be even further enhanced by using a few practical techniques.

5.8.5 Hands-on Tricks

The purpose of this section is to give some practical tips regarding the implementation of ICG on a complex routing problem.

First of all, we remind the reader that the routes are stored in an available pool of columns for RMP. In the pool, every column is associated with its incompatibility degree; and its reduced cost with respect to the dual solution. The backup of all columns is crucial. Indeed, the degrees and reduced costs vary dynamically. In other words, a column that was incompatible in one iteration is likely to be compatible subsequently.

Another important recommendation concerns the pricing procedure. Theoretically, considering a minimisation problem and given a primal-driven dual, only promising negative-reduced-cost columns are added to RMP. Hence, columns that would possibly be part of a better integer solution could be eliminated. In practice, it is useful to customize the pricing criteria by allowing those with a slightly positive cost to enter the basis. As noted earlier, an ascent direction is a linear combination of incompatible columns. Hence, positive reduced cost columns could be selected as they contribute indirectly to the function value improvement. This technique increases the chance of obtaining several improving directions in fewer iterations.

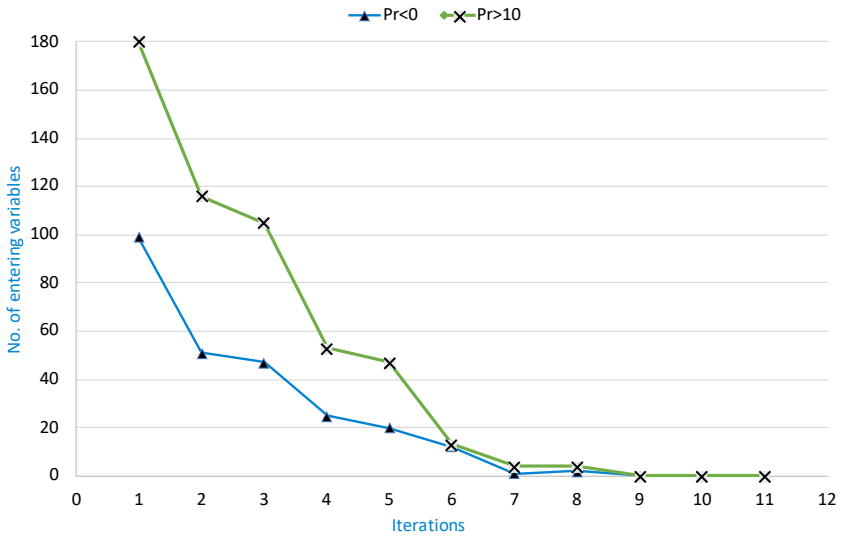


Figure 5.11 Practical pricing procedure based on combined costs

Figure 5.11 illustrates this concept. The x-axis indicates the number of iterations, while the y-axis indicates the number of variables that were part of an ascent direction. The first line (cross marks) corresponds to the negative reduced cost variables, the second line (triangle

marks) corresponds to the (slightly) positive reduced cost variables. Each point plots the number of columns that improved the solution. It is easy to notice that towards the end of the process ($it = 7$), the two lines merge, meaning that both the positive and negative reduced cost variables managed to combine. Naturally, the number of columns of $\bar{p_r} > 0$ drops over iterations as the cost value decreases, and thus cannot contribute to the improvement.

Overall, we showed that in this context, the cost error is self-correcting, which allows us to anticipate the generation of useful columns, and thus accelerate the process.

5.9 Conclusion

In this paper we presented the first application of a primal-based method (ICG) to solve a real-world rich VRP variant, namely the last mile delivery. ICG combines the primal algorithm ISUD with a column generation framework, where we consider a set-partitioning master problem, and an SPPRC subproblem. The efficiency of the ICG algorithm was tested on 30 real data sets with up to 199 customers served through a complex urban network. ICG was compared to a classical B&P heuristic DH, and has largely obtained better solutions in terms of time and quality. Indeed, 30 mid-sized and large-sized real instances were successfully solved in less than two minutes on average and a small number of iterations. We also mention that the algorithm has proven its effectiveness in a complex environment by providing feasible, flexible and economical solutions.

Since ICG takes advantage of the clever vertical and horizontal decomposition of the RMP, several search paths can be explored. We strongly recommend exploring novel hybridization schemes embedding heuristics and/or machine learning models. Research in these areas is already underway.

CHAPITRE 6 ARTICLE 3: DESIGNING AN INTEGRATED DELIVERY ROUTING OPTIMIZER FOR A LOGISTICS SERVICE PROVIDER: KEY REQUIREMENTS, TECHNIQUES AND LESSONS LEARNED

Cet article a été écrit par M. Messaoudi, I. El Hallaoui, et LM. Rousseau, et a été soumis à la revue *INFORMS Journal on Applied Analytics*.

6.1 Introduction

With its 180 worldwide connections and proximity to the Strait of Gibraltar, Tangier Med Port has become the 35^{th} largest container port in the world (*Source: Med Port Authority*). Surrounded by a multi-sector industrial hub including automotive, aeronautics, logistics, textiles, food processing and trade, this world-world port has elevated Morocco to a truly global scale of logistics. Since 2010, an ambitious national logistics strategy has been launched to consolidate this dynamic and to upgrade the logistics sector in Morocco. Indeed, the success of this transformation requires the emergence of integrated and efficient logistics service actors offering adapted and sophisticated services. Several 3PL operators on the market offer an integrated range of logistics services including transport, warehousing, order preparation and other value-added operations. The courier activity is growing by 15% annually with more than 6 million parcels shipped annually, however, 40% of the market is held by the informal sector. On another level, the market for IT equipment for logistics has grown by 30 to 40% over the last 5 years. Nevertheless, the adoption rate of IT remains low in the logistics sector in Morocco (*Source: MLD Agency, National Strategy report, June 2016*). Overall, the multiplication of operators on the market and the diversification of the offer of logistics services had a beneficial impact on logistics costs. Despite these efforts, logistics activities are severely penalized by: (i) unfair competition from the informal sector, (ii) competition from new international actors, (iii) lack of transparency, and (iv) scarcity of standardized regulations. This said, the implementation of the related action plan must be the subject of close collaboration between the ministerial departments, local authorities and stakeholders.

The present paper is a part of a project aiming at developing a new vehicle routing optimizer for a 3PL provider which operates in an extremely complex landscape. This case study summarizes the pre-implementation phase of our system. It is aimed at (i) testing the performance of our solution on several real business configurations at a 3PL company, and (ii) verifying that the developed system's functionality meets specification and business requirements.

The company that participated in this case study is SNTL Group, a leading Moroccan logistics provider since 1937, offering a range of value-added services involving four business activities: supply chain; infrastructure and urban logistic; fleet management; finance and insurance. With the aim of contributing to the foundation of African resilient supply chain solutions, the company has established a research centre, focusing on research, consulting, certification and training activities.

The main **contribution** of this paper is an integrated approach able to provide practical and efficient vehicle routing solutions that integrate both the sourcing level (carrier selection and outsourcing decision) and the control of operations, for long-haul and last-mile deliveries. The proposed models and algorithms are based on a novel primal-based algorithm that produces better solutions in smaller times compared to state-of-the-art metaheuristics. In contrast with classical branch-and-price methods, the proposed approach can display numerous alternatives to the planner, who can then select the most suitable solution and it guarantees feasible solutions wherever we stop the process.

This paper is organized as follows: first, we give an insight into the VRP systems. Next, we describe the problem and the scope of our study. Then, we describe the overall process of data preparedness, and the adoption of new technical and managerial patterns. Given the unavailability of relevant data, managerial accounting techniques were used to construct reliable cost models. After that, we describe our system's capabilities. Finally, we report the results achieved and the lessons learned.

6.2 Relevant Work

Logistics services providers (LSP), commonly known as the third party logistics provider (3PL), are the backbone of logistics operations, and act as a link between the shipper and its operational solutions by handling logistics and transportation operations and developing both standardized and customized services.

In the operations research (OR) literature, planning transportation activities is defined as a vehicle routing problem (VRP), often claimed to be easy to understand but hard to solve. VRP has been extensively covered in the operations research literature in the sense that it props up the logistics and transportation industry as well as the decision-making process. In fact, in the years between 2009 and 2015, 277 papers have been published, they are analyzed and classified in the VRP taxonomy of Braekers et al. (2015). More recent study (Konstantakopoulos et al., 2020) revealed that 263 papers on the subject of freight transportation have been published; from 2010 to 2020. We summarized some of their results in Figure 6.1, which represents

the classification of algorithms for the VRP. For each class, the related number of published papers during the last decade is indicated between parentheses.

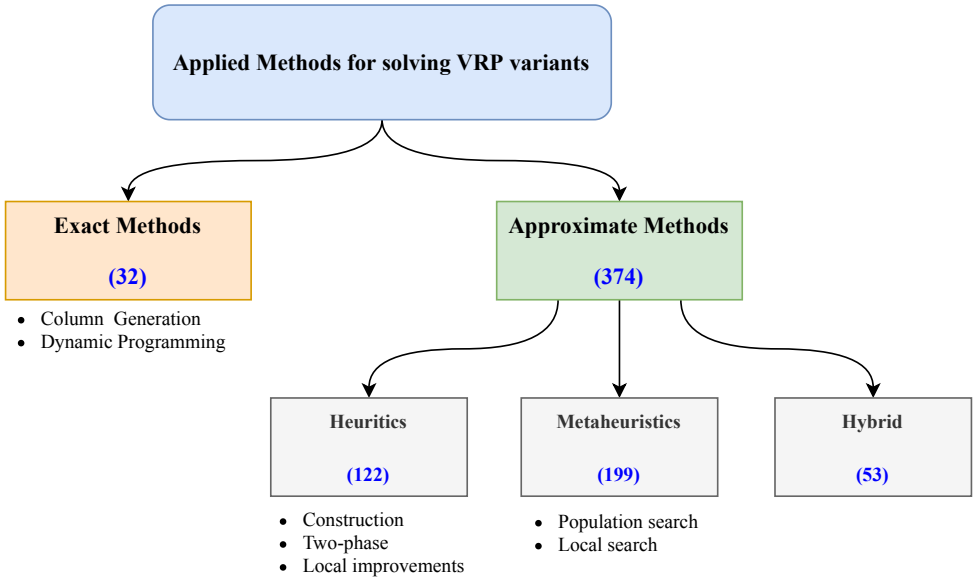


Figure 6.1 Classification of algorithms for solving VRP variants, adapted from Lin et al. (2014) and Konstantakopoulos et al. (2020)

The VRP literature shows a predominant trend of heuristics and metaheuristics, which do not guarantee optimality, but they do find improving solutions for large-sized problem instances within a reasonable amount of time. On the other hand, exact methods are rarely applied to solve rich VRPs as they are computationally demanding.

3PLs handle logistics operations of different customers; hence, constraints atomize into hundreds of complex ones, making route planning even more challenging and making it impossible to solve the problem manually. Hence, using VRP software (VRPS) systems helps them control costs and maintain customer service levels. The main capabilities of commercial VRPS are presented in Figure 6.2; adapted from the studies of Rincon-Garcia et al. (2018); Bräysy and Hasle (2014); Crainic et al. (2009); Drexl (2012).

According to Drexl (2012); Bräysy and Hasle (2014); Rincon-Garcia et al. (2018); Crainic et al. (2009), commercial VRPS systems commonly use a large set of heuristics, derived from published research, internally customized methods, and in-house algorithms (Holland et al., 2017). Certainly, heuristic methods are fast and generally provide a good compromise between efficiency and quality of the solution. However, some heuristic implementations are tailored to work well in specific test instances, by carefully tuning parameters, these might be a barrier to their practical take-up. Particularly, 3PLs market segment is known for its

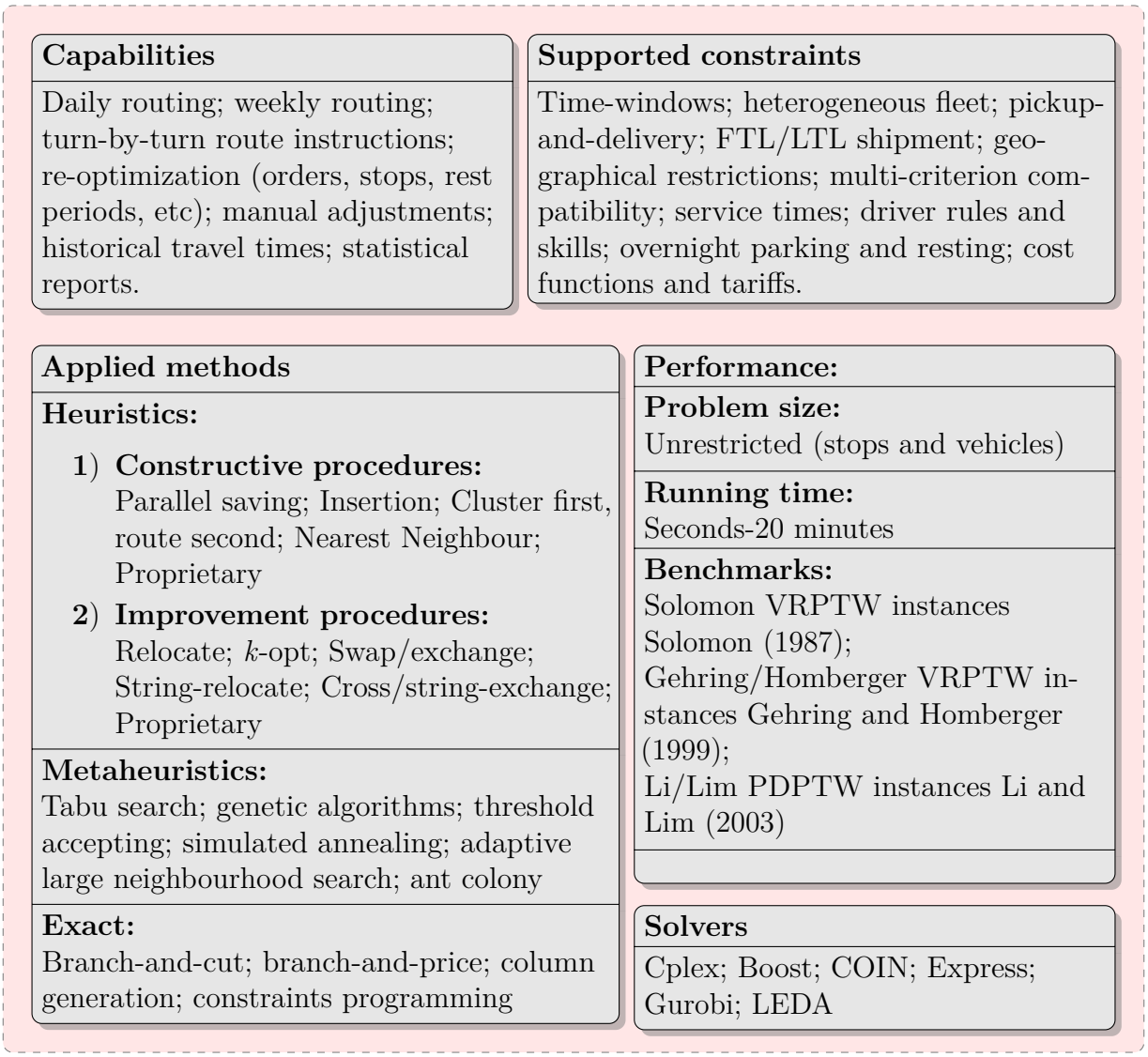


Figure 6.2 VRPS in a nutshell

dynamic constraints (new clients requirements, regulations amendments, etc). Therefore, only agile and resilient VRP systems (Aka., algorithms) can keep these businesses profitable.

6.3 Problem Statement

Through this section, we will present the general framework of our case study. We will first introduce the scope of our mission, and then we will bring the reader closer to the operational context of the company.

6.3.1 Scope and Mission

SNTL Supply Chain is a 3PL operating a very large central mixed flows logistics platform, located in the vibrant economic centre of the country and benefiting from a multimodal logistic connectivity. The company is the exclusive logistics subcontractor for a wide range of customers from different industries: household appliances (eg.,*Samsung*, *Whirlpool*, ...), home furniture (eg.,*Ikea*), automotive (*Renault*), textile (Government) and food products. Transport operations fall into three categories: urban last-mile delivery, regional delivery and national delivery.

As part of its 2025 strategic vision, the company wants to strengthen and develop its transport offer both for inbound (long haul) and outbound (last mile) activities. On a further level, the company is expanding rapidly in West Africa, hence the need for automated, generic and resilient transport planning solutions that are effective in practice, scalable with the dynamic environment of the 3PL, and that come as close as possible to the real business process.

The study we are presenting in this paper is the result of a research project carried out jointly with the company's research centre over a period of 3 years. The elicitation phase was the subject of a one-year observation internship within the company. Prior implementation was carried out at the Samsung warehouse. The next step is to deploy to production.

6.3.2 Template of an Operational Day

LSP manages the logistics activities of its contractors, by ensuring all operations from the receiving of goods, through the storage, to the delivery of products as required by the final customers. As illustrated in Figure 6.3, on a daily basis, the contractors receive purchase orders (POs), on their enterprise resource planning software (ERP), from customers spread throughout the country. Then, POs are transferred to the 3PL's warehouse management system (WMS). They also forward the arrival plannings (AP) in preparation for the receiving of goods at the warehouse. After the cut-off time, POs are extracted to Excel files to proceed to a manual planning according to the cluster first, customer second strategy. Later, the assignment of vehicles is carried out by a dedicated department. Once the orders are prepared for the dispatch, the loading operators proceed to place the parcels inside the vehicles following the delivery orders (DOs). Manual loading operation is often efficient, as it is carried out according to field experience, whereas the sequence of visits is the driver's decision.

Delivery time agreements vary according to the transport type. Urban destinations must be served within one day, while the lead time for regional and national deliveries is up to two days. Plus, the company outsources some deliveries to an external courier service in three

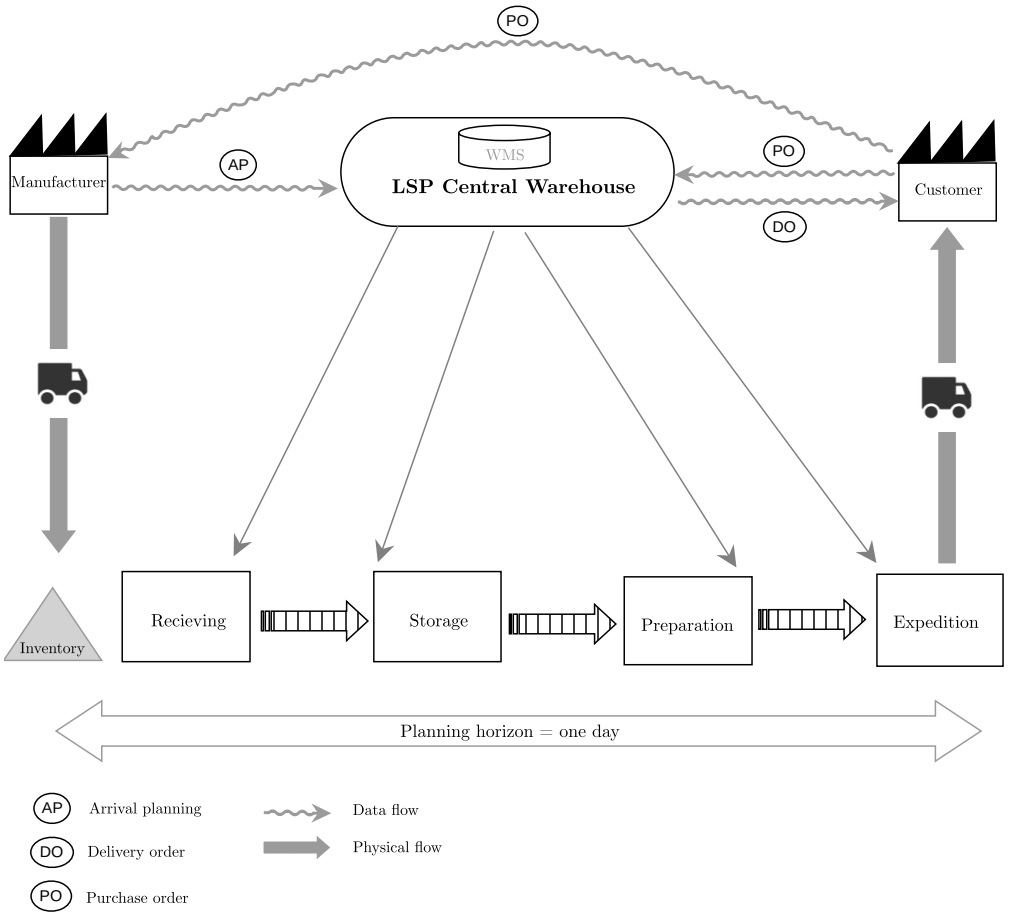


Figure 6.3 Value Stream Mapping of the 3PL

cases: (i) the destination is poorly connected; (ii) the parcels are light but have high value; and (iii) for facing urgent events. The in-house fleet consists of 300 vehicles of various types and capacities (see Table 6.1), in addition to the external fleet, chartered as required. The

Table 6.1 Vehicles types and network classes

Acronym	**GCW***(ton)	**Road Network**
T_1	3.5	Street class 1
T_2	7	Urban area
T_3	14	Road class 2
T_4	19	Road class 3
T_5	25	Road class 3

*Gross Combination Weight.

3PL orchestrates very heterogeneous flows in terms of customer segmentation, product types

and sectors. This amplifies the complexity of delivery planning, each one subject to a different set of constraints as shown in Table 6.2.

Once we have introduced the global context of this study, we can observe that planning transport activity for a 3PL is a major challenge, regulated by a set of endogenous and exogenous constraints. In the following sections, we will describe the fundamental steps and tools undertaken to solve this difficult problem.

Table 6.2 The relevant attributes related to the 3PL context

Attribute	**Related context**
Time windows	Customers
Heterogeneous fleet	Customers, profession
Min/Max fill rate	Business-rules, regulation
Compatibility (site/vehicle)	Business-rules
Compatibility (product/vehicle)	Business-rules
Restricted urban accessibility	Regulation
Driving hours	Work-Unions

6.4 Data Preparedness

In this section, we will outline a key stage for solving a VRP, namely the modelling and the data preparedness. In what follows, we will present how to successfully conduct this process.

6.4.1 Requirements Elicitation

In practice, things are not as neat and linear as they appear in theory. To gather, evaluate and process accurate requirements, we have followed the following steps:

1) Requirements Gathering.

Modelling a real-world problem involves hierarchical sources of information, multi-echelon levels of flows, field-related practices, etc. We used the following techniques:

- Observation: a one-year observation internship was carried out within the company. The direct follow-up of the operations allowed us to identify the main delivery incidents which involve order reject, order return and delivery failure. These incidents are mainly triggered by wrong addresses, wrong orders, unavailable customers, reported requested delivery dates, truck delays, technical problems, expired POs, and lost DOs.

- Interviews: involving executive managers, stakeholders, users, operators, drivers, customers, and authorities.
- Analysis of internal documents: including customer database, carrier database, POs history, DOs history, contracts agreements, financial reports, and legal documents.

Note that diversifying the sources of information contributes to building a mathematical model that is close to reality and prevents obtaining solutions that are not feasible in practice.

2) Evaluation.

A good analysis of activities and flows allowed us to determine which factors drive margins at the end-user node, which can be used as guidelines to best improve future profits. Also, this step provided opportunities to examine new supply chain and distribution approaches. Indeed, the profit margin losses and the rise in operating costs are not only the result of the distribution plans, but also the result of the global transport master plan. That said, the mathematical program had to meet operational and strategic objectives. At the end of this step, we have identified the weakest legs affecting both the operational level (transport planning) and the strategic level (contract negotiation and bidding). These are summarized in Table 6.3.

Table 6.3 The main inherent weaknesses in the process

Operational level	**Strategic level**
Faulty design of the transportation network and schedules.	Low customer service level and responsiveness.
Poor data accuracy.	Complex business scenarios.
Manual data entry.	Cannot assign the best rate.
Non-optimized planning capabilities across different modes of deliveries.	No constraints are checked before the proposal.
Non-compliance with regulation.	

3) Prioritization.

Since the context is extremely complex and entangled, incidents and problems are recorded every hour and every day. Their global management is impossible, however, the Pareto principle comes in handy in this context since that 80% of incidents come from 20% of factors, the importance of prioritizing becomes therefore obvious. We have identified the sources of the most penalizing problems which we will expand in the following section; covering data correction.

4) Consolidation.

Once the problem is understood, analyzed and separated, it was then possible to consolidate the gathered requirements to set the mathematical formulation of the rich VRP and the definition of its specific attributes. A careful study of the literature investigating close VRP variants provided the guideline for the solution methods that might be successful. Then, the next step is to analyze and prepare the key data.

6.4.2 Data Correction

The company suffers from a major challenge: many of the essential data components were either incorrect, incomplete or unstructured. In what follows, we will describe how this issue has been addressed. We will also provide the statistical analysis techniques that we have used to model the cost functions. This crucial step, entailing correcting and building data, required over 6 months of work.

Customer address. Over a quarter of the customer addresses were incorrect. Physical addresses have been corrected and then transformed into geocodes (i.e. latitudes and longitudes). For a dynamic adjustment, we utilized the internal tracking service to discard long, problematic and risky roads. We will show later how we were able to make accurate delivery locations by introducing the new service address concept.

Service Time. Delivery stop times are heavily dependent on the processes that occur at each delivery. These processes are often dictated by the customer receiving the delivery and the sidewalk distance from the park point to the actual delivery point. The service time was established with regard to the customer segment. It took into account the loading time per unit of volume, the confirmation orders time, the average wait. These elements have been developed based on the experience of the drivers, business rules, the geographical area of the customers and the delivery reports which indicate the delay factors.

Customer segmentation. Customer analysis is a crucial step and must be done in a loop to maintain control and be sure to integrate all constraints and relevant details. The 3PL manages a wide range of subcontractors from different industries. As a result, downstream customers are very heterogeneous in terms of geographical locations (urban, rural areas), order characteristics (volume, declared value, type), and constraints (multiple time windows, vehicle compatibility, service time). Thus, the customers and related constraints had to be split into subsets.

Fleet of vehicles. The fleet K contains 300 owned vehicles, and when required, these are chartered from external carriers. Each vehicle $k \in K$ is characterized by the type (rear opening, side opening, swap body, semi-trailer); the volume capacity in cubic meters (CBM); the Vehicle Fill Rate (VFR); the ownership (owned or chartered vehicle); the fixed cost F^k, and the variable cost V^k. Note that the VFR is the ratio of the actual capacity used in a vehicle to the total capacity available in terms of weight and volume. To secure loads against movements, and to maximise efficiency, each vehicle must satisfy $\text{VFR}_{min} = 60\%$ and $\text{VFR}_{max} = 90\%$.

Business rules. It is extremely difficult to accommodate exogenous constraints for the 3PL as customers and flows come from different sectors. The business rules file was created, where all rules and constraints from the profession, transport agreements, national transport agency, the ministry, and work union were translated into appropriate readable rules.

6.4.3 The Need for New Paradigms

Following the elicitation phase, we came up with the conclusion that financial inefficiency comes both from operations and business models, highlighting the need not only to optimize operations, but above all to implement new solutions paradigms based on best practices.

Service address vs. physical address. The thorough analysis of the data allowed us to state that the addresses of the customers appearing in the official files, represent physical addresses and do not correspond to the accurate location where the deliveryman will park to make the deliveries (delivery door or the loading dock). This results in errors of several kilometers and hours. Indeed, in urban areas, stops and U-turns might cause security incidents, tickets and delays. We have therefore come to an agreement that a new customer must communicate his exact delivery address. For existing customers, the information was provided by their agents and also by the drivers.

Courier service delivery option. We have already mentioned that, in predefined cases, some deliveries are outsourced; they are operated by an external courier operator. By analyzing transport costs, as well as the operational flows, we have noticed that parcel delivery is a useful transport solution offering a good compromise between cost, efficiency and safety. That said, we have integrated the courier option as a new vehicle type, and the solver decides when to outsource a delivery according to the profitability of the route. Note that additional assignment constraints have been considered, including the maximum number of daily requests,

or the order CBM range. The impact of this strategy will be outlined in Table 6.6.

Inter-regional delivery. In order to facilitate planning, the internal business processes banned inter-regional deliveries. As explained earlier, the operator consolidates orders so that each vehicle visits one city. This decision was the result of a weak information system and a lack of planning resources. Based on data analytic techniques, the distribution network was clustered into five zones, connected by non-overlapping delivery axes. Also, infeasible arcs regarding the real road network and traffic regulations were discarded. Finally, nodes located in the same shopping malls and supermarkets were aggregated. The distribution network was modelled on an *expert-guided* graph $G(A_{SD_d}, V_{SD_d})$ where A_{SD_d} represents the connecting arcs towards the five shipping direction $\widehat{SD_d}$, and V_{SD_d} represents the cluster of customers where cycles are permitted. This modelling, efficiently solved by the dynamic programming algorithm, made it possible to tackle both inter and intra-regional deliveries, which proved to be very practical.

Order planning. Loading and unloading operations benefit from the human know-how and the field- experience, thus arrangement of the goods in the vehicles is often optimized. Our concern was rather the automation of loading orders. This process must comply with several constraints, mostly practical: (i) a stock keeping unit (SKU) cannot be split; (ii) prioritize least-cost vehicles assignment; (iii) respect the permitted VFR intervals; (iv) respect the customer contractual terms (full truckload and less than truckload modes); and (v) minimize empty space.

After going through the crucial stages of modelling and preparedness, in the following section, we will present the implemented solution, its components and advantages.

6.5 Managerial Accounting Techniques

In practice, the cost function is not always structured and yet further analytical processing is required. Our objective function aims at maximizing the profit generated from operating deliveries. It is calculated as revenue (transport rate less transport cost). In the following, we will describe the statistical analysis tools we employed to model the transport rate and the transport cost functions.

6.5.1 Transport Rate Model

The transport rate is the price paid by the customer n to receive the order ξ^n. For a better comprehension, Figure 6.4 displays a sample of the pricing grid used by the 3PL. The columns framed in red indicate thirty CBM volume intervals of the demand ξ^n, ranging from 0.1 CBM to 40 CBM. The rate is quoted in accordance with the destination and the volume interval. Here for example, a customer n, located in *Boujdour*, with demand $\xi^n = 0.26$ CBM, will pay the rate circled in green. Note that for higher volumes, pricing is switched to FTL; a flat rate depending on the destination and the shipment type (20"FN, 40"FH, 40"FN). The transport rate is a piecewise linear function, given by:

$$\mathcal{P}(\xi^n) \quad = \quad R_v^n \times \xi^n \qquad a_v \leq \xi^n \leq b_v \quad \forall v = 1, 2, \ldots, 30 \tag{6.1}$$

where R_v^n stands for rate associated with the v^{th} volume interval $[a_v,\ b_v]$.

Departure site	Destination City	CBM Range			.....			20FN	40FH	40FN
		0	0.1	0.2	.....	35	40			
		0.1	0.2	0.3	.....	40	FTL			
Casa Warehouse	Agadir	x	x	x	.....	x	x	x	x	x
Casa Warehouse	Al Hoceima	x	x	x	.....	x	x	x	x	x
Casa Warehouse	Azilal	x	x	x	.....	x	x	x	x	x
Casa Warehouse	Ben Slimane	x	x	x	.....	x	x	x	x	x
Casa Warehouse	Béni mellal	x	x	x	.....	x	x	x	x	x
Casa Warehouse	Boujdour	x	x	x	.....	x	x	x	x	x
Casa Warehouse	Boulemane	x	x	x	.....	x	x	x	x	x
Casa Warehouse	Casablanca	x	x	x	.....	x	x	x	x	x
Casa Warehouse	Chefchaouen	x	x	x	.....	x	x	x	x	x

Figure 6.4 Example of the transport rate (tariffs) data file

6.5.2 Transport Cost Model

In what follows, we describe the steps that allowed us to model the transport cost function.

Linear Interpolation.

To analyze the total transport cost, we examined first the relative database, an extract of which is presented in Figure 6.5.

On the one hand, we can notice that for each destination and for each vehicle type, there are several price quotations offered by third-party carriers. For the sake of simplicity, and since these offers are generally similar, we have considered an average transportation cost per

Destination	Total transport cost				
	T1 (3.5t)	T2 (7t)	T3 (14t)	T4 (19t)	T5 (25t)
	Carriers offer	Carriers offer	Carriers offer	Carriers offer	Carriers offer
City 1	carrier 1	_	carrier 1	carrier 1	carrier 1
	:	_	:	:	:
	carrier m	_	carrier m	carrier m	carrier m

Figure 6.5 Example of the transport cost data file

destination and for each vehicle type. On the other hand, as it is shown in Figure 6.5, entries for truck type T_2 are not available. Therefore, we used the linear interpolation technique to predict the missing transport cost from data of trucks T_1 and T_3, according to the following interpolation equation:

$$y_2 = y_1 + (y_3 - y_1)\frac{(x_2 - x_1)}{(x_3 - x_1)}$$

where x_i (resp., y_i) is the volume capacity (resp., the transport cost) of vehicle T_i, $i = 1, 2, 3$.

Linear Regression Analysis.

In this section, we explain how regression analysis was used to estimate fixed and variable costs of each vehicle type. First, in the input data file (see Figure 6.5), we added a column that displays the distance travelled between the destination (city) and the central warehouse. Then, linear regression was calculated to predict variable cost V^k of vehicle k (dependent variable) based on the travelled distance η^k by vehicle k (independent variable). In fact, linear regression uses a series of mathematical equations to find the best possible fitting line to the data points (see example in Figure 6.6).

Regression analysis calculates: $\mathcal{C}_T^k$ the total cost; F^k the fixed cost; V^k the variable cost per travelled distance units η^k, where distance is measured in kilometers and costs are measured in Moroccan currency (MAD). The total transport cost is given by:

$$\mathcal{C}_T^k = V^k \times \eta^k + F^k \qquad \forall k \in K \tag{6.2}$$

Regression analysis in *Excel ToolPak* add-in provided the summarized output presented in Table 6.4. In the column marked "Coeff", the number labeled F^k is a statistical estimate of

the fixed cost, and the number labeled V^k is a statistical estimate of the variable cost per distance unit. The column labeled "$p - value$" expresses the level of statistical significance. Finally, columns 4 and 5 display the confidence interval values.

Table 6.4 Regression analysis calculations summary for each vehicle type

(a) Vehicle type $K_1 = 3.5t$

Costs	*Coeff*	*p-value*	*Lower 95%*	*Upper 95%*
F^k	510.74	1.4e-11	396.15	625.33
V^k	5.45	3.2e-41	5.23	5.67

(b) Vehicle type $K_2 = 7t$

Costs	*Coeff*	*p-value*	*Lower 95%*	*Upper 95%*
F^k	528.27	4.1e-09	381.79	674.75
V^k	6.73	1.4e-40	6.45	7.01

(c) Vehicle type $K_3 = 14t$

Costs	*Coeff*	*p-value*	*Lower 95%*	*Upper 95%*
F^k	545.82	1.1e-06	349.91	741.73
V^k	8.01	2.3e-38	7.64	8.38

(d) Vehicle type $K_4 = 19t$

Costs	*Coeff*	*p-value*	*Lower 95%*	*Upper 95%*
F^k	458.96	6e-08	316.15	601.78
V^k	9.59	7e-48	9.32	9.86

(e) Vehicle type $K_5 = 25t$

Costs	*Coeff*	*p-value*	*Lower 95%*	*Upper 95%*
F^k	499.97	1e-05	292.50	707.44
V^k	12.21	2e-45	11.81	12.60

6.6 From Project to Reality

After a period of testing on several configurations, we were able to successfully implement the solution. For extensive details on the primal-based solution approach, we refer the reader to the following papers Tahir et al. (2019); Messaoudi et al. (2020); Zaghrouti et al. (2014); Bouarab et al. (2017).

6.6.1 Applied Methods

The main objective aimed at solving a complex vehicle routing problem, with time windows, a heterogeneous fleet, site compatibility, delivery options and a set of business constraints. The master problem (MP) was modelled as a set partitioning problem (SPP). The subproblem (SP) was modelled as the shortest path with resource constraints (SPPRC), defined on a customized cyclic urban network, and solved using a dynamic programming (DP) algorithm

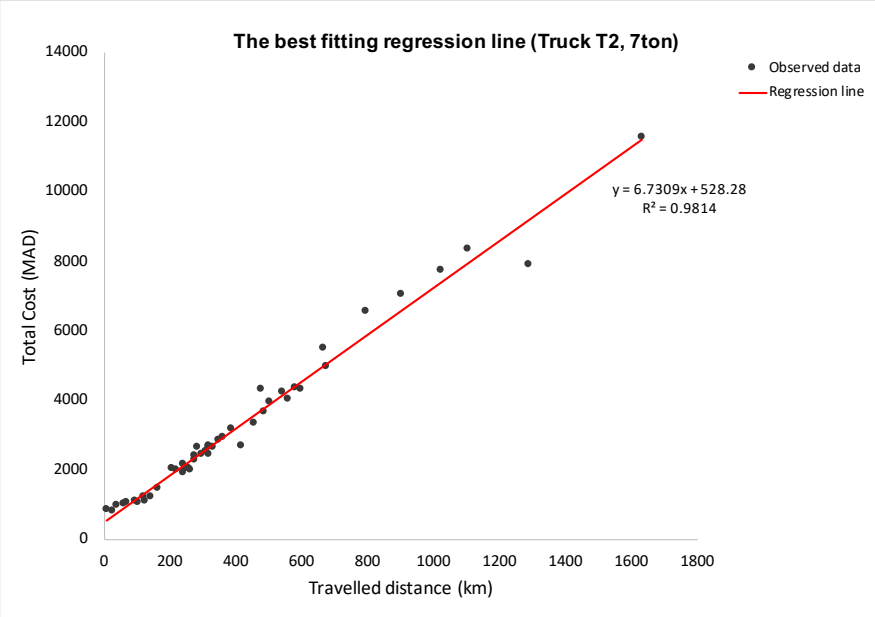


Figure 6.6 Regression line, vehicle type T_2

(Feillet et al., 2004). To provide feasible and efficient solutions, we utilized a primal integer programming algorithm, called integral column generation (ICG) (Tahir et al., 2019). ICG is a sequential algorithm, embedding an integral primal algorithm in a column generation scheme. As shown in Figure 6.7, the column generation (CG) process splits the problem into two problems: SP and restricted MP (RMP). The latter is solved using the integral simplex using decomposition (ISUD) algorithm (Zaghrouti et al., 2014) and decomposed into two subproblems: (i) the reduced problem (RP) contains a very small number of variables and constraints. (ii) The complementary problem (CP) finds a direction which improves the actual solution by updating the set of variables and constraints.

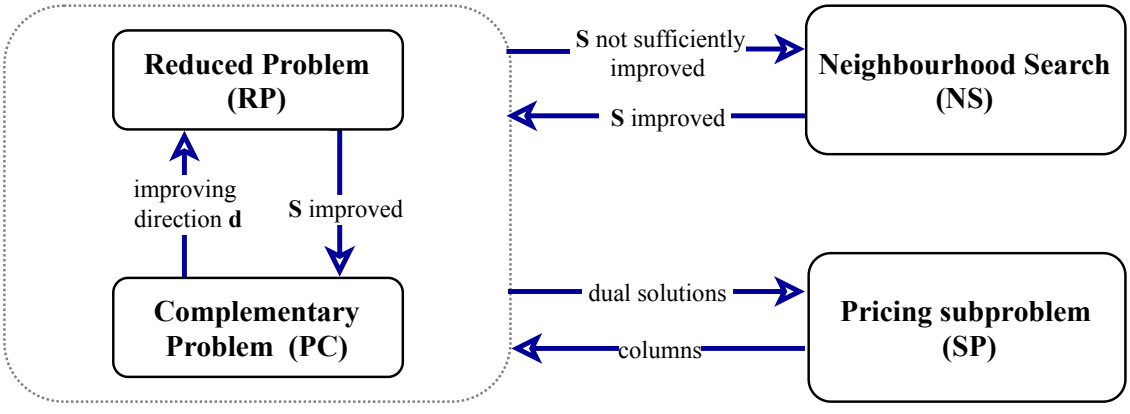


Figure 6.7 ICG algorithm process

ICG takes advantage of the performance of CG and the dynamic decomposition of the RMP allowing its fast re-optimization. Also, the primal programming framework guarantees that the feasible solutions can be obtained at any moment of the solution process; no necessity to wait until the final iteration. The four ICG modules collaborate in a coordinated approach to find fast and feasible solutions, without time-consuming tuning procedures, as only three easy parameters are used. Mainly, the latter define, according to the situation, the improvement rate of the output solutions as well as the stopping conditions of the process.

6.6.2 Key Features

The main features of our systems are summarized in Figure 6.8. The integrated delivery routing optimizer (IDRO) provides many advantages:

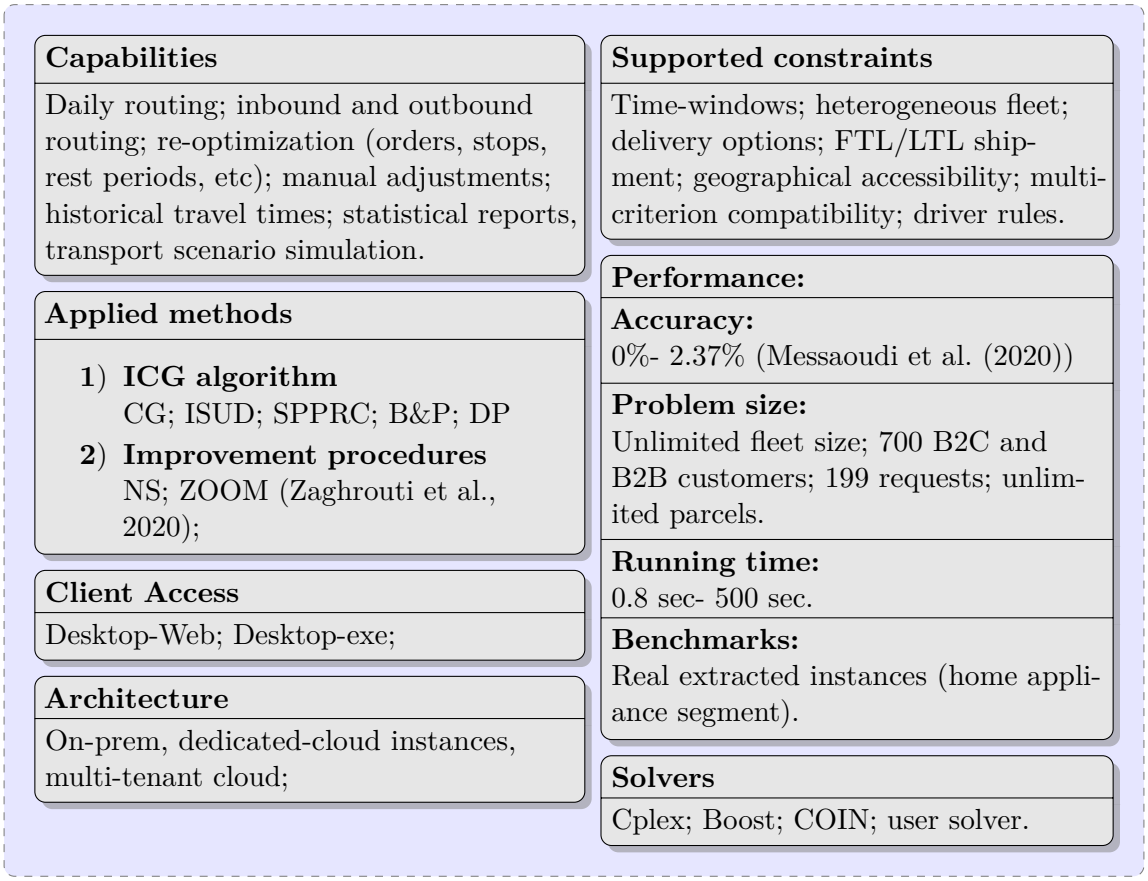


Figure 6.8 IDRO key capabilities

Algorithm. The solution approach is a fair compromise between the efficiency of heuristic methods and the reliability of exact methods, which results in a neat and generic scientific

code; thus, the system does not become obsolete if it is confronted with new constraints or new customers. Indeed, SPPRC makes it possible to accommodate complex and non-linear constraints, while the neat decomposition of RMP provides fast re-optimization, required in case of a vehicle breakdown, insertion or cancellation of orders, road closures, etc.

Architecture. In business-based applications, business rules and data processing logic vary according to the market landscape. It is crucial that the system can be customized by adding plugin modules to extend the functionality of the core system providing extensibility, flexibility, and isolation of application features. The overall architecture of our solution is illustrated in Figure 6.9.

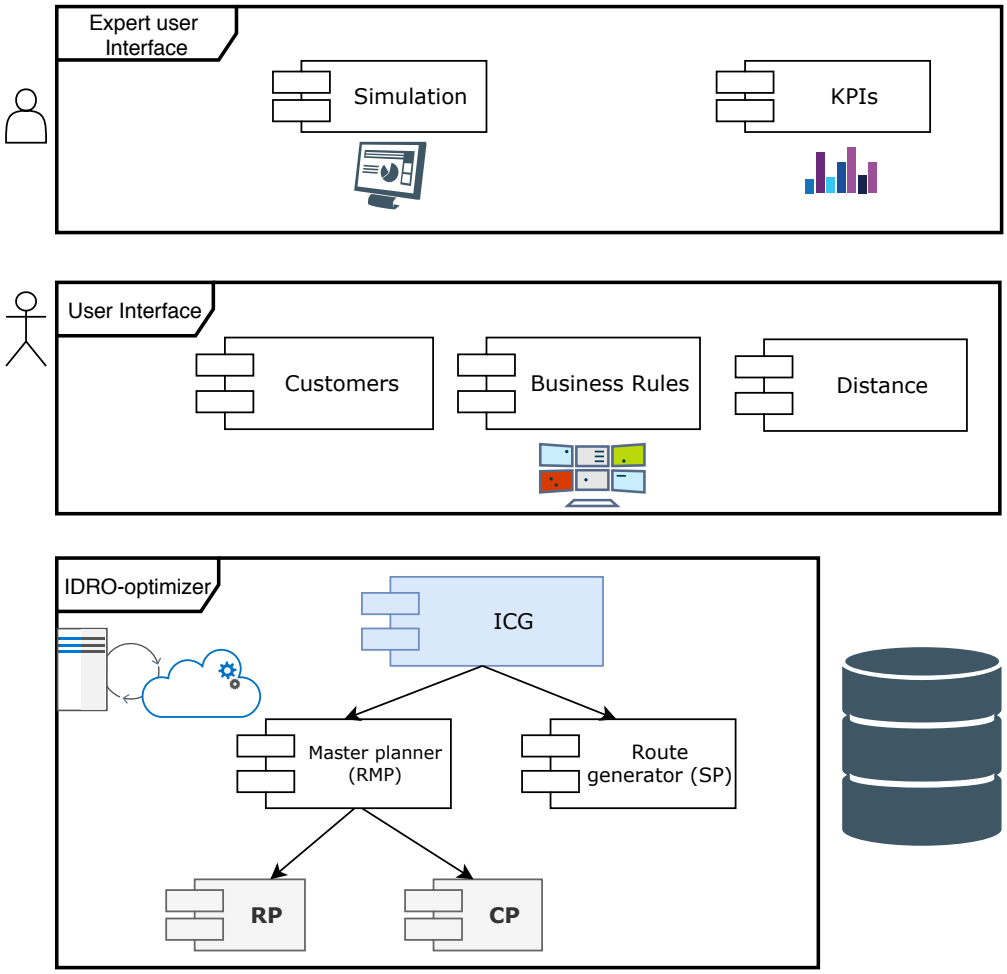


Figure 6.9 IDRO architecture overview

Decision-support. There are two different interfaces for two personas users: managers and dispatchers. Via the first interface, the user has access to the production modules which

enables the optimization of transport operations. The second interface gives access to the decision support modules, which ensures operations control, strategic bid and award process.

System parameters. We use five parameters that are easy to configure and understand, without requiring that the user should have technical or theoretical knowledge. In addition, it is possible to set up default settings by type of business or contractors.

Human knowledge. The system accommodates manual interventions for example set up priorities, discard some constraints, activate and deactivate vehicles, etc. Besides, a feasible solution is always available, means that the dispatcher could stop the process whenever he deems it beneficial. This is a particularly required asset in breakdown situations.

Available information. It is crucial for a primal algorithm to start from a good initial point to enhance its performance. Thus, we take advantage of available information such as delivery history to build feasible visit patterns. The algorithm is primal insofar as it optimizes the problem itself and uses the currently available information to build the improving solutions. In such a competitive environment, governed by bidding and contracting, it is essential to capitalize on the information at hand.

6.7 Results and Impact

Through this section, we will present the impact that the solution has achieved. The realized benefit is roughly divided into financial and nonfinancial components.

6.7.1 Financial Results

The finance department has estimated a 5% increase in overall profit, and is projecting 15% growth within 5 years, with roll-out to all other seven warehouses. Table 6.5 reports a small insight on the performance of the integrated delivery route optimizer (IDRO); compared to the in-house solution of the 3PL. For a meaningful, unbiased results, the instances correspond to nine operational days from different periods. Column 1 is for the name of the instance (Name). Column 2 is for the number of customers ($|N|$). Column 3 displays the percentage of customers served through a last-mile network (% L.Mile). Columns 4 and 7 display the financial profit generated by the route in Moroccan currency (Profit); columns 5 and 8 display the number of trucks (Nb.Trucks). Finally, column 6 displays the total computing real time in seconds (Time). IDRO successfully solved all the instances in just a few seconds. This applies

even to large instances. The gain is calculated as the difference between the profit made by our solution and the profit made by the in-house solution. The *partial* profit achieved by IDRO showed a substantial improvement compared to the in-house solution (50%). The *total* projected gain was estimated considering: (i) roll-out to the other seven warehouses, (ii) coverage of all customers, and (iii) the average number of daily requests. Finally, we note that no profit could be achieved for the instance S-03. Indeed, this is the result of poorly negotiated selling rates.

Table 6.5 Numerical results comparing IDRO and the in-house solution on 9 real instances

Instance			**IDRO**			**In-house.Sol**	
Name	$\|N\|$	% L.Mile	Profit(MAD)	Nb.Trucks	Time(sec)	Profit(MAD)	Nb.Trucks
S-01	27	78	3768	21	1.15	2505	8
S-03	34	76	-1207	26	0.62	-6027	6
S-04	40	90	2100	36	2.2	510	4
S-05	45	38	5780	17	2.29	3700	13
M-02	63	65	4005	41	7.48	1711	9
M-03	66	52	3708	34	6.61	2600	18
M-04	66	89	2080	59	7.36	594	18
M-05	83	64	6200	53	24.46	5500	10
M-08	98	41	7048	40	67.61	5390	23
Total 9-day profit			33,482			16,483	
9-day gain (MAD)			16,999				
Annual *partial* gain (US)			49,485.98				
Annual *total* gain (US)			519,277,5				

6.7.2 Nonfinancial Results

Service Address. The new geocoding paradigm we have introduced made it possible to locate the exact address of the unloading, avoiding delays, U-turns and stops. This has resulted in considerable savings in terms of delivery times. A minimum of 15 minutes per route visiting 3 customers can be saved. For an average of 45 vehicles per day, this results in monthly savings of 20,250 minutes.

Courier Service delivery option. Let us now look at the impact of the courier service integration. Table 6.6 compares the performance of the algorithm implemented with and without the courier delivery option: IDRO and DRO, respectively. The improved version yielded a daily substantial cost decrease, reaching 1,600 US on average.

Table 6.6 Quantitative impact of the Courier Service delivery options on six real-world instances

Instances		**IDRO**			**DRO**		
Name	$\|N\|$	Time(sec)	Cost(MAD)	No.Veh	Time(sec)	Cost(MAD)	No.Veh
S-03	34	0.9	6854	26	1.1	27891	10
S-04	40	4.4	5294	36	5.6	23209	10
S-07	49	1.1	27099	21	2.2	30676	14
M-01	62	4.4	9366	50	709	35752	14
M-03	66	6.6	18775	34	19.8	31979.8	14
M-04	123	12.9	116130	59	12.87	129783	52
Avg	62	4.6	30586	38	125	46548.9	19

Customer service. An analysis of the results obtained over a period of 30 days showed that delivery incidents were reduced by 70%, particularly those caused by address errors, delays, order errors and lost documents. Incidents intrinsic to the customers and to the fleet's management are still persistent, and are the result of the holistic environment.

From operational, tactical and strategic angles, the indirect repercussions of the solution are summarized in Figure 6.10.

Figure 6.10 Operational, tactical and strategic impact.

6.7.3 Lessons Learned

The sharing of success and failure experiences is of great benefit to both researchers, students and the business community. Based on our experience, lessons learned worth sharing and capitalizing on are as follows:

— In contrast to theory, the correlation between the performance of the algorithm and the quality of the solution is not obvious in practice. Several factors from operational, organizational or technical backgrounds can limit performance and efficiency, those include data accuracy, the business process, the transport master plan, and the geographical representation of the network.

— The elicitation phase is a key step, which contributes to outlining the problem and to develop a good model supporting the entire distribution network. It is necessary to succeed in federating all the stakeholders, of all hierarchical levels.

— Data processing is the most fastidious stage, whereby it is necessary to correct and build the needed data.

— Allow two to three attempts to reach a first suitable solution. Each attempt should serve as a prototype, which will be used to gain shareholder support when validating decisions or new investments.

— The risk entailed by organizational changes, and business process reevaluation should be expected. Solutions must be smoothly implemented close to the drivers and operators to reduce and manage their resistance to change.

— Separating customers into sub-segments enables clustering customers with similar needs, which leads to a more effective analysis.

— Develop the skill to transform complex mathematical models and business processes into simple configuration such that the user does not see the complexity.

6.8 Conclusion

In this paper, we presented a summary of a real case study conducted within a 3PL. The objective was twofold: (i) to show what the implementation of VRP looks like in practice, and (ii) to highlight the needs of 3PLs in terms of planning and optimization tools to face the great challenges they face. In this study, we presented the various stages that led to the implementation of the solution. The direct follow-up of the operations on the field and the close collaboration with the actors gave us a good understanding of the flows, bottlenecks, sources of losses and also the potential areas for improvement. To summarize, the solution was able

to achieve interesting results on the operational, tactical and strategic levels. With the aim of creating solutions capable of solving more general VRPs, our system is currently undergoing improvement, by the integration of additional improving and accelerating approaches, from heuristics to Machine Learning techniques.

CHAPITRE 7 VERS UN SCHÉMA D'HYBRIDATION DES METHODES PRIMALES EXACTES ET HEURISTIQUES

Dans ce chapitre, nous proposons deux pistes d'amélioration de la version de base de ICG. Nous présentons un aperçu des techniques employées ainsi que les résultats numériques préliminaires ; obtenus sur des instances réelles d'un VRP riche.

7.1 Introduction

La force de l'algorithme primal ICG réside dans le fait que ses quatre modules coopèrent afin de trouver une suite de solutions entières menant à une solution optimale ou presque optimale.

- **Sous-problème (SP)** : génère des colonnes prometteuses en termes de l'information primale (succession des tâches couvertes) et l'information duale (le coût réduit). Cette dernière est calculée en utilisant une solution duale correspondant à la solution entière courante.
- **Problème réduit (PR)** : cherche une solution entière améliorante en effectuant des pivots sur des variables correspondantes à des colonnes compatibles avec la solution entière courante.
- **Problème complémentaire (PC)** : cherche une direction de descente, définie par une combinaison compatible des colonnes incompatibles. Cette direction peut être entière ou fractionnaire.
- **Recherche de Voisinage (NS)** : ce module intervient lorsque l'amélioration de la solution, à l'aide de l'algorithme ISUD, est jugée non suffisante. *NS* cherche une solution améliorante dans le voisinage de la solution entière courante. Ce voisinage est défini par des colonnes qui se trouvent à une certaine distance mesurable par rapport au sous-espace vectoriel généré par les colonnes de la solution entière courante. Cette distance correspond au degré d'incompatibilité.

En scrutant les résultats que nous avons précédemment présentés dans le Tableau 5.4 (cf. Article 5), nous avons constaté que le module *Recherche de Voisinage (NS)* peut parfois être coûteux. À titre d'exemple, pour résoudre l'instance (L-09 ; $|N| = 136$), *NS* a consommé approximativement 50% du temps total de la résolution. Ce constat est susceptible de se reproduire sur d'autres instances, d'où le besoin d'implémenter des mécanismes performants

permettant l'amélioration locale de la solution entière courante.
À travers ce chapitre, nous proposons un schéma initial d'hybridation entre l'algorithme primal ICG et une heuristique d'amélioration qui viendrait remplacer le module *NS*. L'objectif ultime serait de renforcer le taux de succès pour atteindre une solution optimale ou quasi optimale sans branchement et sans augmenter la durée de la résolution. Une telle amélioration serait particulièrement avantageuse dans le cas des problèmes réels avec de grandes instances. Afin de trouver efficacement de tels voisinages, nous allons présenter deux approches distinctes : *ZOOM* et H-ALSN, comme le montre la Figure 7.1.

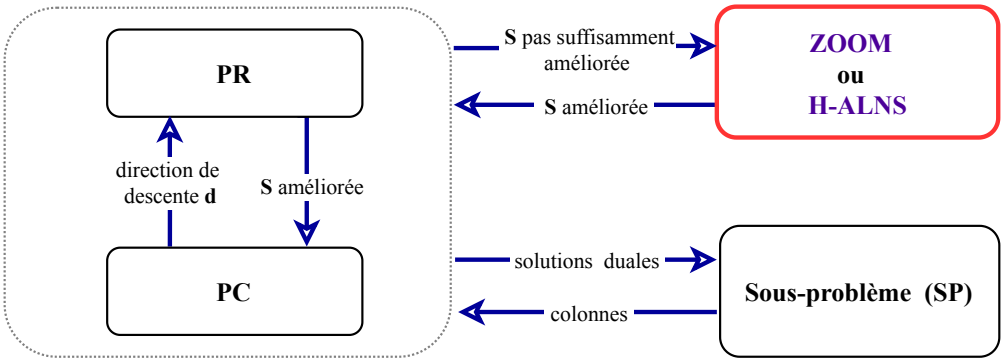


Figure 7.1 La structure de ICG

La première approche, appelée *ZOOM*, consiste à chercher une solution améliorante dans le voisinage d'une direction de descente fractionnaire. La deuxième approche ferait l'appel d'une heuristique adaptative H-ALNS pour la recherche d'une solution entière dans son voisinage.

Dans la Section 7.2, nous présenterons l'algorithme *ZOOM*, ainsi que les résultats préliminaires obtenus sur 30 instances réelles d'un VRP riche. Dans la Section 7.3, nous donnerons d'abord un aperçu de l'heuristique H-ALNS, ensuite nous présenterons les résultats préliminaires obtenus jusqu'à présent sur 15 instances réelles d'un VRP riche. Enfin, nous proposerons certaines techniques d'adaptation pour reussir une premiere hybridation entre un algorithme primal exact et une heuristique. La section 7.4 conclura le présent chapitre en mettant en avant les futures pistes d'amélioration.

7.2 Approche ICG-ZOOM

Cette approche a été introduite par Zaghrouti et al. (2020) dans le but de pallier aux faiblesses de la version de base de ISUD. L'idée maîtresse de *ZOOM* consiste à explorer le voisinage d'une direction de descente fractionnaire afin de trouver une direction de descente entière, ceci compenserait l'éventuel échec du PC à trouver des directions de descente entières. À travers

cette section, nous introduirons d'abord l'algorithme, ensuite nous discuterons les résultats numériques obtenus suite à l'intégration de ICG et *ZOOM*.
Au long de ce chapitre, nous adoptons la même notation utilisée dans le chapitre 5. Soit $\bar{\boldsymbol{\theta}}$ la solution entière courante réalisable ; $\boldsymbol{\theta}^*$ la solution optimale ; z^* la valeur de la solution optimale et $\boldsymbol{d}$ la direction de descente menant vers une solution améliorante. Nous notons par S l'ensemble d'indices des colonnes de la solution entière courante, par C l'ensemble d'indices des colonnes compatibles avec S et par I l'ensemble d'indices des colonnes incompatibles avec S. Nous utilisons S et *la solution courante* de manière interchangeable.

7.2.1 Un Aperçu de ZOOM

ZOOM utilise une décomposition dynamique où l'intégralité est uniquement considérée au niveau du PR. Quant au PC, il opère pour fournir les directions de descente. Quand celles-ci sont fractionnaires, ZOOM opère différemment, dans le sens où il n'a recours ni au branchement, ni aux coupes. Comme le montre la Figure 7.2, *ZOOM* utilise la direction fractionnaire afin de guider la recherche des solutions améliorantes dans une région d'amélioration potentielle.

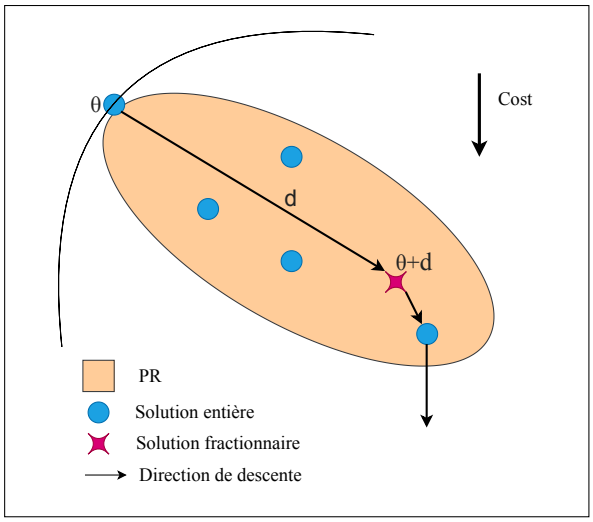


Figure 7.2 Recherche d'une solution entière dans le voisinage d'une direction $\boldsymbol{d}$

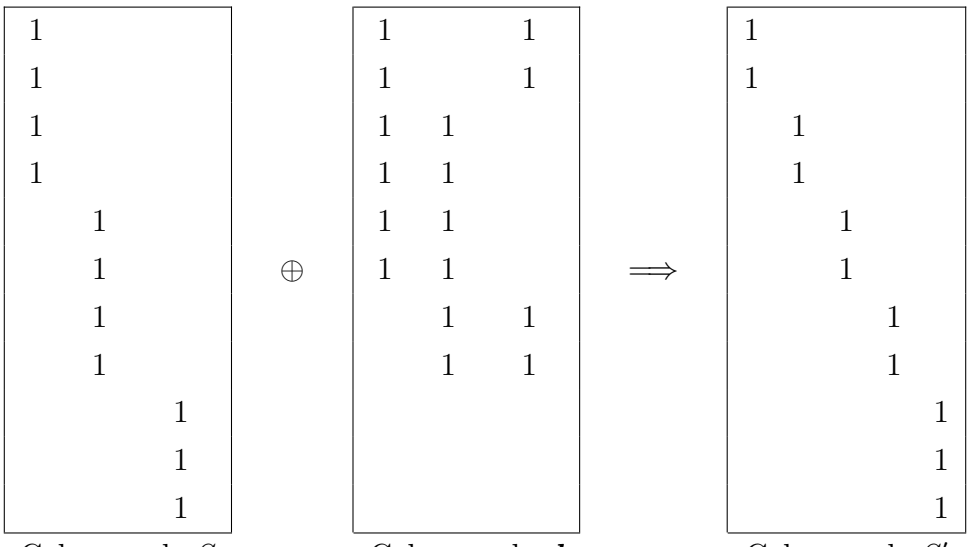
L'algorithme 5 décrit la procédure de *ZOOM* scindée en deux étapes principales :

Algorithm 5: *ZOOM*

Trouver une solution initiale $\boldsymbol{\theta^0}$, $\bar{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta^0}$, $\boldsymbol{d} \leftarrow 0$
$[S \quad C \quad I] \leftarrow$ partitionner la matrice des contraintes A en sous-ensemble de colonnes
répéter
  1 : obtenir une direction de descente *d*
  répéter
    Résoudre le $PR(\bar{\boldsymbol{\theta}}, S, C)$
    $(Z^{PC}, \boldsymbol{d}) \leftarrow$ résoudre $PC(\bar{\boldsymbol{\theta}}, S, I)$ pour trouver une direction de descente
  tant que $Z^{PC} < 0$ ***et d*** *est entière*
  2 : chercher une solution entière dans un voisinage défini par $\boldsymbol{d} \neq 0$
  $S' \leftarrow$ désagréger S selon $\boldsymbol{d}$.
  $[I'] \leftarrow$ trouver les colonnes de A compatible avec S'
  Résoudre le $PR(\bar{\boldsymbol{\theta}}, S', C')$
tant que $\boldsymbol{d} \neq 0$ ***et*** $|Z^{PC}|$ *n'est pas suffisamment petite*
Retourner $\bar{\boldsymbol{\theta}}$

*Les ensembles S, C et I sont mis à jour une fois qu'une nouvelle solution entière est trouvée.

Tant que la direction de descente est entière, **l'étape 1** consiste à résoudre une suite itérative de PR et PC. Lorsque PC trouve une direction de descente fractionnaire, **l'étape 2** cherche à trouver une solution entière dans le voisinage de la dernière direction de descente trouvée. D'abord, les colonnes de la solution entière courante S sont désagrégées de telle sorte que les colonnes incompatibles, correspondantes aux variables entrantes dans la direction $\boldsymbol{d}$, deviennent compatibles avec la nouvelle solution S'. L'exemple ci-dessous illustre le principe de désagrégation, où l'opération est notée par $\oplus$.



Après désagrégation, les nouvelles colonnes de A deviennent compatibles avec la nouvelle solution S'. Ces colonnes définissent un voisinage autour de la direction $\boldsymbol{d}$ dans lequel s'effectue une recherche d'une solution améliorante. Cette recherche se fait en résolvant un PR construit à l'aide des nouvelles colonnes compatibles. On note que le PR est généralement de très petite taille et possède de bonnes propriétés, il peut être résolu par n'importe quel solveur commercial MIP (Cplex).

7.2.2 Résultats Numériques

L'objectif est de comparer les performances des deux versions de ICG avec et sans *ZOOM*. Dans la version de base de ICG (ICG-(NS)), lorsque l'amélioration de la solution est jugée non suffisante, nous faisons appel au module *NS* pour rechercher une solution améliorante dans le voisinage de la solution courante. Dans la version améliorée ICG-ZOOM ; *NS* est remplacé par le module *ZOOM*. Celui-ci, rappelons-le, consiste à chercher une solution améliorante dans le voisinage d'une direction de descente fractionnaire. Nous avons mené les expérimentations sur les mêmes séries d'instances réelles que nous avons auparavant décrit dans le chapitre 5, ainsi que les mêmes valeurs des paramètres ($minImp = 0.0025,\ maxConsFail = 5,\ maxDegree = 7.$). Nous avons utilisé une implémentation C++ sous Linux, sur des stations de travail équipées de processeurs Intel Xeon E3-1226 à 3,3 GHz et de 8 Go de RAM.

Les résultats numériques préliminaires sont présentés dans le Tableau 7.1. Les instances sont classifiées en trois groupes : petites, moyennes et grandes. La colonne 1 désigne la classe de l'ensemble (Classe) ; la colonne 2 indique le nom de l'instance (Nom) et la colonne 3 indique le nombre de clients (|N|). Pour ICG-(NS) et ICG-(ZOOM) les colonnes 4 et 8 affichent le nombre d'itérations de génération des colonnes (Iter.) ; les colonnes 5 et 9 indiquent le temps de calcul total en secondes (Temps(sec)) ; et les colonnes 6 et 10 indiquent le pourcentage d'écart d'optimalité entre le coût de la meilleure solution trouvée et la valeur optimale de la relaxation linéaire (Gap(%)) et finalement les colonnes 7 et 11 indiquent le nombre total de solutions entières trouvées (No.Sol.).

Tableau 7.1 Résultats numériques de ICG-(NS) et ICG-(ZOOM) sur 30 instances réelles

Instances			**ICG-(NS)**				**ICG-(ZOOM)**			
Classe	Nom	|N|	Iter.	Temps(sec)	Gap(%)	No.Sol	Iter.	Temps(sec)	Gap(%)	No.Sol.
	S-01	27	14	1.15	0.37	19	14	1.52	0.37	19
	S-02	32	10	0.80	0.00	7	11	1.30	0.00	7
	S-03	34	7	0.62	0.00	5	7	0.50	0.00	5
Petite	S-04	40	9	2.20	0.86	7	10	2.36	0.86	7
	S-05	45	11	2.29	2.60	16	11	2.28	2.60	16
	S-06	45	11	2.11	0.53	13	9	2.05	0.54	17
	S-07	49	7	1.08	1.89	21	7	1.24	1.89	21
	M-01	62	7	4.40	0.00	5	8	4.44	0.00	5
	M-02	63	8	7.48	0.00	17	8	6.63	0.00	17
	M-03	66	8	6.61	0.24	20	8	6.36	0.24	20
Moyenne	M-04	66	9	7.36	1.35	25	9	7.76	1.31	26
	M-05	83	9	24.46	1.84	22	9	19.45	1.93	21
	M-06	97	9	70.84	0.00	18	12	74.09	0.00	25
	M-07	97	9	21.98	0.10	26	10	24.38	0.14	30
	M-08	98	15	67.61	2.37	40	9	35.52	2.29	31
	L-01	106	9	19.32	0.15	17	10	20.46	0.15	21
	L-02	114	11	84.74	0.47	28	12	87.95	0.43	29
	L-03	116	14	129.19	0.88	45	13	113.80	1.42	43
	L-04	123	12	12.96	0.24	24	8	8.17	0.24	23
	L-05	125	15	195.83	1.80	44	10	97.83	1.42	44
	L-06	126	7	99.39	0.06	31	7	92.63	0.06	31
	L-07	131	9	114.31	0.00	8	6	71.69	0.02	8
Grande	L-08	134	11	120.41	0.00	19	11	131.88	0.00	19
	L-09	136	13	387.87	1.77	48	9	271.67	1.77	50
	L-10	136	10	117.33	0.00	21	11	134.39	0.00	22
	L-11	140	8	24.16	0.37	31	8	22.18	0.37	31
	L-12	140	9	113.69	0.00	27	9	105.71	0.00	27
	L-13	152	11	277.14	0.00	31	15	303.27	0.00	46
	L-14	160	10	89.42	0.08	46	10	82.38	0.02	47
	L-15	199	12	540.33	0.40	66	10	332.60	0.43	40
Moyenne			10.1	85.00	0.61	24.9	9.7	68.88	0.62	24.9

Les résultats présentés dans le Tableau 7.1 montrent que les deux versions de ICG trouvent en moyenne des solutions entières de même gap. De plus, ICG-(ZOOM) est plus rapide et trouve, en moyenne, le même nombre de solutions entières. Pour la grande instance (L-15 ; n= 199), *ZOOM* était en mesure de réduire le temps d'exécution de 38%. Pour certaines instances, le temps de résolution augmente, ceci n'est pas surprenant vu qu'il s'agit des résultats préliminaires sans aucun réglage particulier des paramètres. Globalement, ces résultats témoignent que *ZOOM* peut significativement améliorer la performance d'ICG en cherchant des solutions améliorées dans le voisinage d'une direction fractionnaire. Une amélioration potentielle de cette approche consisterait à définir des voisinages parallèles autour

de plusieurs directions de descente.

7.3 Approche ICG-Heuristique

Comme nous l'avons mentionné, notre objectif ultime serait de proposer une nouvelle approche hybride combinant l'algorithme primal ICG et une méthode heuristique dans le but d'une amélioration locale performante et peu coûteuse. Dans une première étape, nous avons examiné le potentiel de cette idée en implémentant d'abord ICG avec *ZOOM*, un algorithme où la recherche des solutions améliorées est guidée par la direction de descente. Les premières séries d'essais (cf. 7.1) ont montré une performance persuasive de l'algorithme. Dans cette section, nous présentons une étude préliminaire de l'approche hybride que nous souhaitons développer, mettant en lien l'algorithme ICG et une heuristique dans un schéma hybride.

En premier lieu, nous donnerons un aperçu général sur le fonctionnement de l'heuristique H-ALNS. Ensuite, afin d'évaluer la performance de l'heuristique désignée à résoudre notre problème, nous présenterons les résultats des expérimentations sur une série de nos instances (cf. Tableau 7.1). Finalement, nous discuterons les schémas d'hybridations et les techniques d'adaptation prévues entre ICG et H-ALNS.

7.3.1 Un Aperçu de H-ALNS

Introduit par Belhaiza (2019), H-ALNS (H-ALNS : *Hybrid Adaptive Large Neighborhood Search* est une heuristique hybride entre ALNS et un algorithme génétique. À chaque itération, un ou plusieurs opérateurs de réparation et de destruction R&D (R&D : *repair and destroy*) sont utilisés. La sélection des opérateurs est basée sur la mise à jour continue des taux de réussite des opérateurs. H-ALNS présente trois caractéristiques principales : (i) elle conserve une trace des meilleures solutions courantes (recherche de diversification), (ii) elle améliore ALNS grâce aux nouveaux opérateurs (recherche d'intensification), (iii) elle utilise les opérateurs de croisement génétique *crossover* pour s'échapper de l'optimum local. Le fonctionnement global de l'algorithme est donné dans la Figure 7.3 ci-dessous.

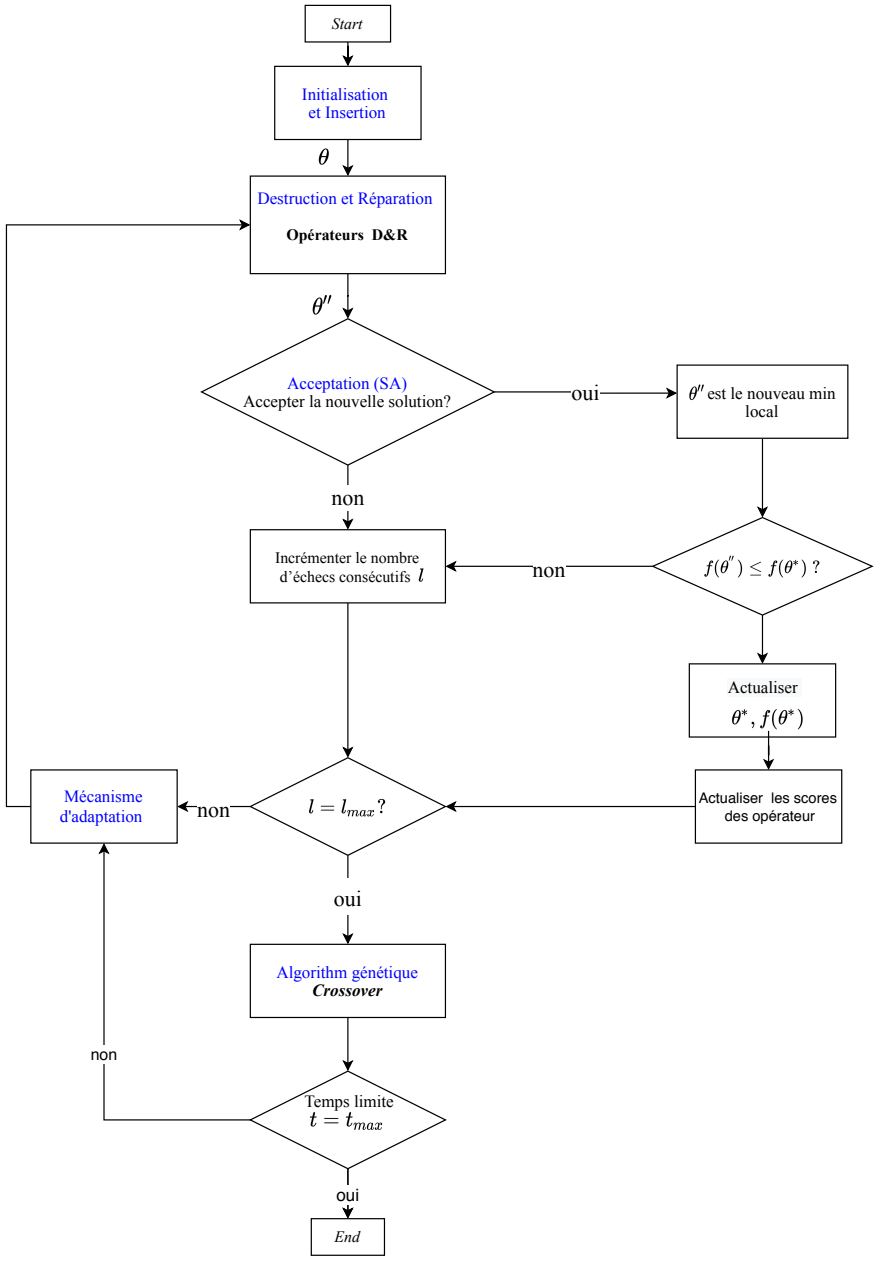


Figure 7.3 Un aperçu de l'algorithme H-ALNS ; adapté de Belhaiza (2019)

Comme le montre la Figure 7.3, H-ALNS opère selon une boucle itérative d'intensification et de diversification comme suit :

Initialisation et Insertion. Cette étape construit une première solution θ selon une version modifiée de l'algorithme de Clarke & Wright. Si une solution ne couvre pas nécessairement toutes les tâches, les opérations d'insertion sont effectuées avant chaque itération de la boucle de réparation et destruction. La fonction *fitness* $f(\theta)$ minimise la durée totale de la route $h(\theta)$ ainsi que les pénalités sur la violation des contraintes en introduisant les termes T, Q, O

correspondant à la violation des fenêtres de temps, la capacité des véhicules et le dépassement des délais, respectivement.

$$f(\theta) = h(\theta) + T + Q + O$$

Réparation et destruction. Cette procédure itérative applique plusieurs opérateurs R&D sur θ afin de trouver une solution améliorée $\theta^{"}$. Après chaque itération, les poids τ_g de chaque opérateur g heuristique sont mis à jour de manière adaptative en fonction de la qualité de la nouvelle solution. Si de bonnes solutions sont trouvées en utilisant certains de ces opérateurs, leur poids relatif est augmenté, ce qui les rend plus susceptibles d'être sélectionnées lors des itérations suivantes.
Pour une optimisation séparée des routes, la méthode utilise des opérateurs basés sur la suppression et la réinsertion des paires d'arcs. Ces mouvements peuvent réduire le coût d'une route et la rendre également réalisable. Pour chaque véhicule, la phase de destruction et de réparation applique jusqu'à trois opérateurs : R&D-relocaliser, R&D-sommet et R&D-arc.
Pour une optimisation multiple des routes, la méthode fait appel à des opérateurs basés sur les échanges d'arcs et les déplacements de visites au sein des séquences. Ces mouvements permettent d'améliorer plusieurs routes simultanément en utilisant trois opérateurs différents : R&D- interroutes-relocaliser, R&D-interroutes-échanger et R&D-interroutes-croiser.

Mécanisme de sélection. La sélection proportionnelle ou de roulette (*roulette wheel*), est un opérateur génétique utilisé pour sélectionner des solutions potentiellement utiles. Initialement, tous les poids sont initialisés à 1 et la probabilité τ_g de sélectionner l'opérateur g est égale à $\frac{1}{G}$, où G est l'effectif des opérateurs utilisés. En effet, si un opérateur g est appliqué, son poids τ_g est mis à jour en fonction de son succès ou de son échec à améliorer la solution globale θ^* ou la solution courante $\theta^{"}$. En effet, $\varrho_g = 2$ si la solution voisine améliore θ^* ; 1 si elle améliore $\theta^{"}$. Sinon, $\varrho_g = 0$.

Si l'opérateur améliore la solution globale ou la solution existante, son poids est augmenté : $\tau_g = [(1-\epsilon)\tau_g + \epsilon\varrho_g]/o_g$, où $\epsilon \in [0,1]$ est le facteur de réaction, et o_g est le nombre de fois que l'opérateur g a été sélectionné. Par exemple, si un opérateur g réussit lors de son premier appel à améliorer θ^*, son poids τ_g passe de $\frac{1}{G}$ à $\frac{\epsilon(2G-1)+1}{G}$. Alors que s'il réussit lors de son premier appel à améliorer $\theta^{"}$, son poids augmente de $\frac{1}{G}$ à $\frac{\epsilon(G-1)+1}{G}$. Sinon, s'il ne réussit pas à améliorer, son poids décroît de $\frac{1-\epsilon}{G}$.

Mécanisme d'acceptation (SA). Le recuit simulé performe le mécanisme d'acceptation et de rejet et met à jour les probabilités de sélection des opérateurs. Le processus de refroidissement de la température considère une fonction exponentielle standard à deux paramètres, à savoir la température de départ T^0, et la vitesse de refroidissement $\alpha \in [0,1]$. La température

courante T suit le schéma de refroidissement $T = T^0\alpha^l$ à chaque cycle de température l. Pour un meilleur compromis entre lenteur et rapidité du recuit simulé, un nouveau cycle de température est incrémenté après chaque 1000 itérations. La solution voisine $\theta^{"}$ est acceptée si $f(\theta^{"}) \leq f(\theta)$, sinon, elle est acceptée avec une probabilité $e^{\frac{f(\theta^{"})-f(\theta)}{T^0}} \geq 0,5$. Si le mouvement de réparation est accepté, le poids de l'opérateur correspondant τ_g est ajusté, et le nombre d'appels o_g est augmenté.

Croisement génétique. Pour échapper aux vallées sous-optimales, l'opérateur de croisements génétiques *Crossover* est appelé chaque fois que la meilleure solution ne peut être améliorée pendant un nombre $l_{max} = 5$ de cycles de température consécutifs, soit 5000 itérations. Pour un aperçu plus approfondi sur la méthode, nous conférons le lecteur aux travaux suivants : Belhaiza et al. (2014); Belhaiza (2019); Belhaiza et al. (2019).

7.3.2 ZOOM et les Opérateurs de Destruction

On ne doit pas perdre de vue que l'opération de désagrégation, que nous avons décrite dans la Section 7.2, soumet une analogie nette avec les opérateurs heuristiques de destruction ; par exemple ceux de la méthode LNS (LNS : *Large Neighbourhood search*). Concrètement, désagréger les colonnes d'une solution revient à les détruire pour en construire de nouvelles colonnes compatibles. Néanmoins, à l'encontre de LNS, le choix des tâches ou des clients à enlever de la solution est une décision rationnelle, guidée par une direction de descente. Celle-ci est obtenue en résolvant un programme linaire précisant la zone d'amélioration potentielle dans le domaine réalisable. De même, nous notons que la destruction effectuée par *ZOOM* est intérimaire dans le sens où les tâches enlevées d'une solution ne sont pas complètement éjectées, mais elles restent présentes comme des variables agrégées au niveau du PC.

7.3.3 Résultats Numériques

Afin de tester la performance de l'heuristique H-ALNS et la comparer avec celle de ICG-NS, nous avons conduit une série d'expérimentations préliminaires sur un sous-ensemble d'instances qui ont été présentées dans le Tableau 7.1. Les résultats de l'implémentation de H-ALNS sont présentés dans le Tableau 7.2. Les instances sont classifiées en trois groupes : petites, moyennes et grandes. La colonne 1 indique la classe de l'instance (Classe) ; la colonne 2 correspond au nom de l'instance (Nom) ; la colonne 3 correspond au nombre de clients (|N|) ; les colonnes 4 et 6 affichent le temps de calcul total en secondes (Temps(sec)) et les colonnes 5 et 7 affichent le pourcentage d'écart d'optimalité entre le coût de la meilleure solution trouvée et la valeur optimale de la relaxation linéaire (Gap(%)).

Tableau 7.2 Résultats numériques de la comparaison entre ICG-(NS) et H-ALNS sur 15 instances réelles

Instances			**ICG-(NS)**		**H-ALNS**	
Classe	Nom	|N|	Temps(sec)	Gap(%)	Temps(sec)	Gap(%)
	S-01	27	1.2	0.37	75.0	0.31
	S-02	32	0.8	0.00	11.0	0.00
Petite	S-03	34	0.6	0.00	54.0	0.00
	S-04	40	2.2	0.86	182.0	0.86
	S-05	45	2.3	2.60	106.0	2.57
	M-01	62	4.4	0.00	282.0	0.00
	M-02	63	7.5	0.00	2594.0	0.00
Moyenne	M-03	66	6.6	0.24	108.0	0.23
	M-04	66	7.4	1.35	68.0	1.61
	M-05	83	24.5	1.84	1604.0	1.86
	L-01	106	19.3	0.15	5720.0	1.38
	L-02	114	84.7	0.47	2760.0	2.11
Grande	L-03	116	129.2	0.88	8098.0	3.48
	L-04	123	13.0	0.24	5873.0	0.83
	L-05	125	195.8	1.80	7825.0	5.28
Moyenne			33.3	0.72	2357.3	1.4

Comme on peut le constater, H-ALNS était incapable de traiter efficacement toutes les instances, et la performance de ICG dépasse nettement celle de H-ALNS. Cette dernière ne serait pas en mesure d'être implémentée dans la pratique. En effet, le milieu industriel exige que l'algorithme soit en mesure de fournir des solutions de bonne qualité dans un temps raisonnable. Si pour les petites instances le temps de calcul est relativement rapide, les grandes instances ont été résolues dans un temps moyen qui dépasse deux heures. Ceci ouvre des pistes d'amélioration et la recherche des stratégies d'adaptation de H-ALNS dans notre contexte pour un schéma coopératif gagnant-gagnant.

7.3.4 Techniques d'Hybridation

Les résultats présentés dans le Tableau 7.2 montrent que la rapidité de H-ALNS dépend principalement de la taille de l'instance à résoudre, de ce fait, les petites instances ont été résolues dans un temps temps raisonnable.

En partant de ce constat, nous présumons que l'idée d'incorporer H-ALNS à ICG reste tout de même une option réaliste. En effet, lorsque l'amélioration de la solution courante à l'aide d'ISUD n'est pas suffisante, nous résolvons une petite instance à l'aide de H-ALNS. Ces

instances correspondront aux sous-graphes définissant des voisinages autour de la solution courante. La définition de ces voisinages serait guidée par trois techniques :

— **Le coût réduit :** en utilisant la solution duale correspondant à la solution entière courante, calculer les coûts réduits des arcs entre chaque paire de clients (i, j). Ensuite, supprimer les arcs dont le coût réduit dépasse une certaine valeur. En d'autres termes, ceci revient à enlever les arcs qui ont de rares chances d'être empruntés dans une solution améliorante.

— **La longueur de détour :** supposons que le client i est suivi par le client j dans la solution courante, et que t_{ij} est le temps de parcours menant de i à j. Enlever tous les arcs (i, k) tel que $k \neq j$ dont le temps de parcours t_{ik} dépasse une limite t_{max}. L'idée ici est d'éliminer les longs détours par rapport à la solution courante.

— **les arcs inactifs :** éliminer les arcs qui n'ont jamais été ou très peu empruntés dans les solutions entières trouvées dès le début de la résolution. Ce critère peut être utilisé lorsque le nombre de solutions entières trouvées devient important.

En appliquant ces techniques, d'une façon séparée ou combinée, le graphe réduit résultant est ensuite résolu à l'aide de H-ALNS dans l'espoir de trouver une solution améliorante sans résoudre un programme linéaire en nombre entier (MIP ou *ZOOM*). Ce dernier est, parfois, difficile à résoudre lorsque le voisinage considéré est de grande taille. Nous notons que les idées présentées dans cette section ne sont pas encore testées et feront objet d'extension et d'affinement dans le futur proche.

7.4 Conclusion et Discussion

À travers ce chapitre nous avons discuté deux approches susceptibles d'améliorer la performance de ICG. L'objectif ultime serait de réussir une toute première hybridation entre une méthode primale et une méthode heuristique dans un contexte exact. Dans sa version de base, ICG a recours au module *NS* qui cherche une solution améliorante dans le voisinage de la solution entière courante.

À travers une première approche, nous avons remplacé ce module par *ZOOM*. Celui-ci détermine des voisinages potentiels d'amélioration autour d'une direction de descente fractionnaire. Par conséquent, la recherche dans une telle région favorise l'obtention rapide d'une solution entière de meilleur coût. La portée de cette amélioration a été corroborée et démontrée par les résultats numériques. Ceci nous a encouragés à intensifier la recherche afin de trouver une technique encore plus performante, notamment en présence de problèmes industriels avec de grandes instances.

À cette fin, la seconde approche que nous avons proposée consiste à remplacer le module *NS* par une heuristique hybride à large voisinage adaptatif dite H-ALNS. Les tests préliminaires de la version autonome de H-ALNS sur nos instances ont montré que la méthode est performante uniquement sur les petites instances.

Ainsi, fallait-il réfléchir à des techniques judicieuses qui permettraient de réussir l'incorporation de l'heuristique à ICG. Parmi les démarches potentielles, est le fait de considérer qu'une petite instance correspond à un sous-graphe définissant un voisinage autour de la solution courante. De tels voisinages seront construits par le biais des techniques exploitant aussi bien l'information primale que duale.

En analysant l'opération de désagrégation effectuée par *ZOOM*, nous avons repéré une grande analogie avec les opérateurs de destruction (utilisés par l'heuristique LNS par e.g.). De ce fait, nous présumons que : (i) l'intégration de ICG avec une heuristique est une approche réaliste et relativement simple à concrétiser ; (ii) l'opération de désagrégation de *ZOOM* pourrait être remplacée par les opérateurs de destruction connus dans la littérature ; finalement (iii) toutes ces techniques d'amélioration pourraient s'incorporer dans un même algorithme, dont la sélection serait gérée par un mécanisme adaptatif. Celui-ci attribuerait un score à chacune de ces composantes en se basant sur leurs succès à trouver une solution améliorante entière dans le voisinage désigné.

Nous notons que suite à des incidents d'ordres techniques et administratifs, ce projet a dû être temporairement suspendu avant de finaliser l'étape de l'intégration entre ICG et H-ALNS. Une piste fructueuse que nous entendons poursuivre subséquemment.

CHAPITRE 8 DISCUSSION GÉNÉRALE

Les travaux de recherche conduits dans le cadre de cette thèse portent sur la résolution efficace des VRPs riches dans un milieu pratique, en introduisant un nouveau paradigme de solution basé sur la programmation primale en nombres entiers. À notre connaissance, il s'agit de la première contribution qui propose une telle approche dans le contexte des VRPs. Les différents chapitres de ce manuscrit ont relaté le fil de l'histoire qui a conduit à la conception d'un outil d'optimisation de la distribution, destiné à un prestataire de service logistique intégré 3PL. Pour aussi bien la communauté scientifique que le milieu industriel y trouvent intérêt, nous avons présenté successivement, les fondements théoriques, en passant par les expérimentations sur un jeu d'instances réelles pour aboutir à l'implémentation en milieu pratique.

Nous avons réussi une première application de l'algorithme primal ICG à un problème de transport difficile présent dans toutes les configurations logistiques ; un VRP riche qui respecte les exigences techniques et pratiques du métier, et qui fournit des solutions réalisables en pratique. Grâce à ISUD, ICG permet de contourner la dégénérescence engendrée par la structure du SPP qui est une formulation courante des VRPs. ISUD résout efficacement le PMR et permet de trouver une séquence de solutions entières réalisables, de coûts décroissants et menant à des solutions optimales ou quasi optimales.

Pour trouver des solutions entières, un branchement allégé est délégué à un solveur MIP. L'intégralité est favorisée par le biais d'une stratégie dite *multiphase*. Implémentée aux niveaux du PMR et du SP, elle consiste à résoudre le PC uniquement pour les colonnes qui se trouvent à une certaine distance de la solution courante. Dans la majorité des itérations, le PC réussit à générer directement des directions qui mènent à des solutions entières. Un exemple théorique est présenté dans le chapitre 5 pour en illustrer le fonctionnement.

Les méthodes de B&P sont connues d'être plus efficaces sur les problèmes de transport aérien, où les graphes sont généralement acycliques. En revanche, le transport routier fait appel à des réseaux de distribution complexes et cycliques ce qui rend la résolution extrêmement difficile. Dans cette optique, nous avons exploité les données de livraison pour confectionner un réseau de distribution réaliste, modélisé comme un graphe orienté sur les grands axes de distribution, et cyclique à l'intérieur des villes. Quant aux contraintes d'élimination des cycles, elles ont été gérées au niveau du SP. L'impact de cette amélioration a été mesuré et validé grâce à des expérimentations.

La performance de ISUD est principalement basée sur la décomposition dynamique du PMR et l'agrégation des contraintes. Toutefois, ceci engendre une limitation majeure puisqu'il

n'est pas possible de calculer une solution duale complète. Pour surmonter cette limitation, nous utilisons une solution duale correspondant à la solution entière courante. Par une étude théorique et empirique, nous avons démontré que ces solutions duales favorisent la génération de colonnes qui peuvent potentiellement faire partie d'une solution entière améliorante et contribuant ainsi à réduire le temps de calcul. La validité de la procédure utilisée pour calculer les solutions duales a été également supportée par les résultats numériques.

Lorsque l'amélioration de la solution, à l'aide de l'algorithme ISUD, est jugée non suffisante, une recherche d'une solution améliorante dans le voisinage de la solution entière courante est effectuée. Ce voisinage est défini par des colonnes qui se trouvent à une certaine distance mesurable par rapport au sous-espace vectoriel généré par les colonnes de la solution entière courante ; appelée degré d'incompatibilité. Cette recherche, effectuée par un solveur, peut être coûteuse des fois. Nous avons alors proposé de renforcer la performance de ICG à travers son intégration avec deux approches ; *ZOOM* et une heuristique.

Quand le problème complémentaire trouve une direction de descente fractionnaire, aucun branchement ou ajout de coupes n'est opéré. Plutôt c'est le problème réduit qui se charge de trouver une solution entière ; dans un voisinage guidé par cette direction. Celle-ci intervient pour définir une région potentielle où la recherche d'une solution entière de meilleur coût a de fortes chances d'aboutir. Nous avons expérimenté cette première approche, dont les résultats numériques ont montré que ICG a clairement bénéficié de cette intégration. Ceci nous a incités à soulever la question suivante : ne serait-il pas judicieux de remplacer *ZOOM* par une heuristique d'amélioration performante ? L'objectif serait de guider efficacement la recherche des solutions entières améliorantes sans toutefois prolonger le temps de calcul.

Cette réflexion a ouvert une nouvelle piste de recherche, susceptible de mener vers un premier schéma d'hybridation entre une heuristique et une approche primale exacte. La recherche a été initiée en testant tout d'abord l'heuristique H-ALNS sur nos instances. L'heuristique a trouvé de la difficulté à résoudre les grandes instances, toutefois, elle a réussi à résoudre assez rapidement les petites instances et à générer des solutions améliorantes. En gardant ceci en vue, nous avons proposé des techniques adéquates susceptibles de réussir l'intégration souhaitée. Des contraintes techniques ont entravé la poursuite de la deuxième phase de l'hybridation, un objectif que nous présumons explorer postérieurement.

En conduisant ce projet de recherche, nous avons réalisé que dans le monde réel, la modélisation et la résolution d'un VRP est une mission difficile qui nécessite plusieurs étapes et techniques, ainsi que la fédération de plusieurs ressources hiérarchiques. D'ailleurs, dans le contexte complexe et enchevêtré des opérations, les exigences sont très spécifiques, les contraintes ne sont ni conceptualisées ni formulées, et les données relatives aux adresses et aux coûts sont souvent

erronées. À travers une étude de cas réalisée auprès d'un prestataire logistique, nous avons décrit et élicité la méthodologie et les techniques nécessaires pour réussir l'implémentation d'un VRP dans un milieu pratique.

Nous présumons que les caractéristiques de l'algorithme primal ICG lui octroieraient un transfert facile vers la pratique. La solution est basée sur une (i) une architecture modulaire ce qui permet à l'algorithme de gérer efficacement les nouvelles extensions du problème et d'accommoder les règles du métier au niveau du sous-problème. L'algorithme permet de (ii) générer de nombreuses solutions afin que l'utilisateur puisse en sélectionner la plus viable. Un autre avantage indéniable est celui de (iii) pouvoir arrêter le processus à tout moment avec la garantie d'obtenir une solution réalisable.

Nous croyons que les contributions présentées dans cette thèse sont susceptibles d'encourager et de faciliter l'implémentation de la méthode primale dans des contextes industriels similaires ou connexes.

CHAPITRE 9 CONCLUSION ET RECOMMANDATIONS

Au fil de notre thèse, nous nous sommes portés sur la résolution des VRPs riches qui ne peut efficacement aboutir sans tenir compte de la facette réaliste et complexe des opérations ; justement en intégrant avec fidélité les contraintes endogènes et exogènes. À cette fin, et pour la première fois, un algorithme issu de la programmation primale en nombres entiers est utilisé sur le terrain d'essai des VRPs. Dans ce qui suit, nous passerons en revue la synthèse de nos recherches, leurs limitations ainsi que les futues pistes d'amélioration.

9.1 Synthèse des travaux

Dans le premier sujet, nous avons fait appel à la méthode ICG afin de résoudre un problème de VRP multi-attributs avec fenêtres de temps et flotte hétérogène. Pour évaluer la viabilité de l'algorithme, nous avons conduit une étude quantitative préliminaire sur sept instances réelles atteignant 140 clients. La qualité et la rapidité des solutions obtenues ont mis à jour aussi bien le potentiel que les éventuelles pistes d'améliorations de la méthode.

Le second objectif s'est focalisé sur l'amélioration de l'algorithme confirmant en même temps sa capacité à résoudre efficacement des VRPs riches. Nous avons confronté l'algorithme au problème difficile de la livraison du dernier kilomètre sans omettre d'intégrer des contraintes plus difficiles, bien entendu toutes issues de la pratique. Nous avons implémenté deux stratégies d'accélération qui nous ont permis de réduire considérablement les temps de calcul et d'éviter le recours au branchement classique pour atteindre l'intégralité. Aussi, nous avons introduit deux propositions qui supportent la qualité des solutions duales que nous utilisons ; un fondement que nous avons également supporté empiriquement. Enfin, nous avons conduit des expérimentations sur une trentaine d'instances réelles atteignant 199 clients. Il s'est avéré que le temps de calcul moyen a été de l'ordre d'une dizaine de secondes, et que la livraison de 160 clients a été planifiée en moins de deux minutes.

Dans le troisième objectif, nous avons étudié la méthodologie de résolution du VRP selon une perspective pratique. Nous avons exposé une étude de cas réelle conduite chez un prestataire logistique opérant dans une économie émergente. Nous avons détaillé toutes les étapes qui précèdent l'implémentation de la solution en pratique. Pour favoriser la performance de l'algorithme, nous avons aussi introduit de nouveaux paradigmes techniques et managériaux. Par ailleurs, des techniques analytiques comme l'interpolation et la régression linéaire ont été employées afin de modéliser les fonctions des coûts. L'étude a également discuté de

l'architecture de notre système, de son impact et des leçons apprises.

Dans le dernier chapitre, nous avons présenté une étude préliminaire sur l'hybridation d'une heuristique avec un algorithme primal exact. Nous avons proposé deux méthodes d'amélioration de ICG ; toutes les deux basées sur l'amélioration locale efficace de la solution courante. En premier lieu, nous avons incorporé *ZOOM* à ICG, dans le but de trouver efficacement des voisinages potentiels d'amélioration. Les résultats numériques ont confirmé ce constat : la performance de ICG a été améliorée. Ensuite, nous avons testé une heuristique hybride H-ALNS sur nos instances. Sa performance a été limitée aux petites instances, nous avons alors proposé des techniques susceptibles de faciliter l'hybridation entre ICG et H-ALNS, ouvrant au passage toute une nouvelle piste de recherche.

9.2 Limitations des solutions proposées

En conduisant ces projets de recherche, nous avons assimilé aussi bien les avantages que les limitations du paradigme primal exact. (i) Nous avons considéré une formulation du SPP où les variables sont binaires. Or, dans d'autres contextes, il est possible que la formulation ait besoin des variables entières. Un VRP dont la fonction objectif minimise le nombre de véhicules utilisés en est un exemple. (ii) Comme nous l'avons précédemment mentionné, implémenter la stratégie multiphase au niveau du SP permet de favoriser l'intégralité en sélectionnant à chaque phase les colonnes qui se trouvent à une certaine distance de la solution courante. Cette distance, dite degré d'incompatibilité, est pratiquement calculée à partir de la succession des clients couverts par chaque route (colonne). Cependant, cette information n'est parfaite qu'en absence des contraintes supplémentaires. Or, plusieurs applications se modélisent réellement comme des SPPs avec contraintes supplémentaires. (iii) Tester la performance de l'algorithme sur des instances réelles est certes important, mais il faudrait le confirmer sur des benchmarks classiques de VRP pour bien se situer par rapport à l'état de l'art.

9.3 Améliorations futures

Le paradigme primal exact pourrait faire l'objet de plusieurs applications et ouvrir plusieurs pistes de recherche. (i) Nous envisageons de poursuivre l'étude présentée dans le dernier chapitre visant à réussir une première hybridation entre une méthode primale exacte et une heuristique. (ii) Comme nous l'avons discuté dans notre deuxième article, les contraintes de normalisation contribuent à favoriser l'intégralité. En présence de cette preuve de concept, nous présumons que les techniques d'apprentissage profond pourraient être judicieusement employées pour estimer les poids de ces contraintes de telle sorte que les directions de descente

fractionnaires soient pénalisées. (iii) ICG permet de proposer des solutions, et à tout moment l'utilisateur peut les vérifier, les adopter ou les rejeter. Il serait pertinent que l'algorithme apprenne de ces décisions par les moyens d'apprentissage profond. Ce mécanisme permettrait de calquer l'expérience pratique, d'intégrer dynamiquement les contraintes exogènes, et d'améliorer les performances de l'algorithme.

RÉFÉRENCES

E. Balas, “The prize collecting traveling salesman problem”, *Networks*, vol. 19, no. 6, pp. 621–636, 1989.

M. O. Ball, B. Golden, A. Assad, et L. Bodin, “Planning for truck fleet size in the presence of a common-carrier option”, *Decision Sciences*, vol. 14, no. 1, pp. 103–120, 1983.

V. M. M. Barbosa, “Bus driver rostering by hybrid methods based on column generation”, Thèse de doctorat, Universidade de Lisboa (Portugal), 2018.

S. Belhaiza, “A hybrid adaptive large neighborhood heuristic for a real-life dial-a-ride problem”, *Algorithms*, vol. 12, no. 2, p. 39, 2019.

S. Belhaiza, P. Hansen, et G. Laporte, “A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows”, *Computers & Operations Research*, vol. 52, pp. 269–281, 2014.

S. Belhaiza, R. M’Hallah, G. B. Brahim, et G. Laporte, “Three multi-start data-driven evolutionary heuristics for the vehicle routing problem with multiple time windows”, *Journal of Heuristics*, vol. 25, no. 3, pp. 485–515, 2019.

A. Ben-Israel et A. Charnes, “Contributions to the theory of generalized inverses”, *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 3, pp. 667–699, 1963.

M. Blocho, “Heuristics, metaheuristics, and hyperheuristics for rich vehicle routing problems”, dans *Smart Delivery Systems*. Elsevier, 2020, pp. 101–156.

M.-C. Bolduc, J. Renaud, et F. Boctor, “A heuristic for the routing and carrier selection problem”, *European Journal of Operational Research*, vol. 183, no. 2, pp. 926–932, 2007.

M.-C. Bolduc, J. Renaud, F. Boctor, et G. Laporte, “A perturbation metaheuristic for the vehicle routing problem with private fleet and common carriers”, *Journal of the Operational Research Society*, vol. 59, no. 6, pp. 776–787, 2008.

H. Bouarab, I. El Hallaoui, A. Metrane, et F. Soumis, “Dynamic constraint and variable aggregation in column generation”, *European Journal of Operational Research*, vol. 262, no. 3, pp. 835–850, 2017.

S. Boussier, D. Feillet, et M. Gendreau, “An exact algorithm for team orienteering problems”, *4or*, vol. 5, no. 3, pp. 211–230, 2007.

K. Braekers, K. Ramaekers, et I. Van Nieuwenhuyse, “The vehicle routing problem : State of the art classification and review”, *Computers & Industrial Engineering*, 2015.

O. Bräysy et G. Hasle, “Chapter 12 : Software tools and emerging technologies for vehicle routing and intermodal transportation”, dans *Vehicle Routing : Problems, Methods, and Applications, Second Edition*. SIAM, 2014, pp. 351–380.

S. E. Butt et D. M. Ryan, “An optimal solution procedure for the multiple tour maximum collection problem using column generation”, *Computers & Operations Research*, vol. 26, no. 4, pp. 427–441, 1999.

I.-M. Chao, “A tabu search method for the truck and trailer routing problem”, *Computers & Operations Research*, vol. 29, no. 1, pp. 33–51, 2002.

I.-M. Chao et T.-S. Liou, “A new tabu search heuristic for the site-dependent vehicle routing problem”, dans *The next wave in computing, optimization, and decision technologies*. Springer, 2005, pp. 107–119.

C.-W. Chu, “A heuristic algorithm for the truckload and less-than-truckload problem”, *European Journal of Operational Research*, vol. 165, no. 3, pp. 657–667, 2005.

G. U. Clarke et J. W. Wright, “Scheduling of vehicles from a central depot to a number of delivery points”, *Operations research*, vol. 12, no. 4, pp. 568–581, 1964.

L. C. Coelho, J. Renaud, et G. Laporte, “Road-based goods transportation : a survey of real-world applications from 2000 to 2015”, Technical report, Technical Report FSA-2015-007, Québec, Canada, Rapp. tech., 2015.

J.-F. Cordeau et G. Laporte, “A tabu search algorithm for the site dependent vehicle routing problem with time windows”, *INFOR : Information Systems and Operational Research*, vol. 39, no. 3, pp. 292–298, 2001.

J.-F. Cordeau, G. Laporte, et A. Mercier, “A unified tabu search heuristic for vehicle routing problems with time windows”, *Journal of the Operational research society*, vol. 52, no. 8, pp. 928–936, 2001.

J. Cordeau, G. Desaulniers, J. Desrosiers, M. Solomon, et F. Soumis, “The vrptw. the vehicle routing problem”, *SIAM Monographs on Discrete Mathematics and Applications*, pp.

157–194, 2001.

J.-F. Côté et J.-Y. Potvin, “A tabu search heuristic for the vehicle routing problem with private fleet and common carrier”, *European Journal of Operational Research*, vol. 198, no. 2, pp. 464–469, 2009.

T. G. Crainic, M. Gendreau, et J.-Y. Potvin, “Intelligent freight-transportation systems : Assessment and the contribution of operations research”, *Transportation Research Part C : Emerging Technologies*, vol. 17, no. 6, pp. 541–557, 2009.

S. Dabia, D. Lai, et D. Vigo, “An exact algorithm for a rich vehicle routing problem with private fleet and common carrier”, *Transportation Science*, 2019.

G. B. Dantzig et J. H. Ramser, “The truck dispatching problem”, *Management science*, vol. 6, no. 1, pp. 80–91, 1959.

M. Dell’Amico, F. Maffioli, et P. Värbrand, “On prize-collecting tours and the asymmetric travelling salesman problem”, *International Transactions in Operational Research*, vol. 2, no. 3, pp. 297–308, 1995.

M. Desrochers et F. Soumis, “A generalized permanent labelling algorithm for the shortest path problem with time windows”, *INFOR : Information Systems and Operational Research*, vol. 26, no. 3, pp. 191–212, 1988.

M. Desrochers, J. Desrosiers, et M. Solomon, “A new optimization algorithm for the vehicle routing problem with time windows”, *Operations research*, vol. 40, no. 2, pp. 342–354, 1992.

J. Desrosiers et M. E. Lübbecke, “A primer in column generation”, dans *Column generation*. Springer, 2005, pp. 1–32.

M. Drexl, “Rich vehicle routing in theory and practice”, *Logistics Research*, vol. 5, no. 1, pp. 47–63, 2012.

I. Elhallaoui, D. Villeneuve, F. Soumis, et G. Desaulniers, “Dynamic aggregation of set-partitioning constraints in column generation”, *Operations Research*, vol. 53, no. 4, pp. 632–645, 2005.

I. Elhallaoui, A. Metrane, G. Desaulniers, et F. Soumis, “An improved primal simplex algorithm for degenerate linear programs”, *INFORMS Journal on Computing*, vol. 23, no. 4, pp. 569–577, 2011.

J. Euchi, “The vehicle routing problem with private fleet and multiple common carriers : Solution with hybrid metaheuristic algorithm”, *Vehicular Communications*, vol. 9, pp. 97–108, 2017.

D. Feillet, P. Dejax, M. Gendreau, et C. Gueguen, “An exact algorithm for the elementary shortest path problem with resource constraints : Application to some vehicle routing problems”, *Networks*, vol. 44, no. 3, pp. 216–229, 2004.

D. Feillet, P. Dejax, et M. Gendreau, “Traveling salesman problems with profits”, *Transportation science*, vol. 39, no. 2, pp. 188–205, 2005.

H. Gehring et J. Homberger, “A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows”, dans *Proceedings of EUROGEN99*, vol. 2. Citeseer, 1999, pp. 57–64.

B. E. Gillett et L. R. Miller, “A heuristic algorithm for the vehicle-dispatch problem”, *Operations research*, vol. 22, no. 2, pp. 340–349, 1974.

F. Glover, “Tabu search—part i”, *ORSA Journal on computing*, vol. 1, no. 3, pp. 190–206, 1989.

M. Gmira, M. Gendreau, A. Lodi, et J.-Y. Potvin, “Tabu search for the time-dependent vehicle routing problem with time windows on a road network”, *European Journal of Operational Research*, 2020.

B. L. Golden, L. Levy, et R. Vohra, “The orienteering problem”, *Naval Research Logistics (NRL)*, vol. 34, no. 3, pp. 307–318, 1987.

P. Hansen et N. Mladenović, “Variable neighborhood search : Principles and applications”, *European journal of operational research*, vol. 130, no. 3, pp. 449–467, 2001.

C. Holland, J. Levis, R. Nuggehalli, B. Santilli, et J. Winters, “Ups optimizes delivery routes”, *Interfaces*, vol. 47, no. 1, pp. 8–23, 2017.

T. Ibaraki, S. Imahori, M. Kubo, T. Masuda, T. Uno, et M. Yagiura, “Effective local search algorithms for routing and scheduling problems with general time-window constraints”, *Transportation science*, vol. 39, no. 2, pp. 206–232, 2005.

T. Ibaraki, S. Imahori, K. Nonobe, K. Sobue, T. Uno, et M. Yagiura, “An iterated local search algorithm for the vehicle routing problem with convex time penalty functions”, *Discrete Applied Mathematics*, vol. 156, no. 11, pp. 2050–2069, 2008.

S. Ichoua, M. Gendreau, et J.-Y. Potvin, “Vehicle dispatching with time-dependent travel times”, *European journal of operational research*, vol. 144, no. 2, pp. 379–396, 2003.

S. Irnich et G. Desaulniers, “Shortest path problems with resource constraints”, dans *Column generation*. Springer, 2005, pp. 33–65.

S. Irnich, P. Toth, et D. Vigo, “Chapter 1 : The family of vehicle routing problems”, dans *Vehicle Routing : Problems, Methods, and Applications, Second Edition*. SIAM, 2014, pp. 1–33.

C. Joncour, S. Michel, R. Sadykov, D. Sverdlov, et F. Vanderbeck, “Column generation based primal heuristics”, *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 695–702, 2010.

S. Kirkpatrick, C. D. Gelatt, et M. P. Vecchi, “Optimization by simulated annealing”, *science*, vol. 220, no. 4598, pp. 671–680, 1983.

Ç. Koç, T. Bektaş, O. Jabali, et G. Laporte, “Thirty years of heterogeneous vehicle routing”, *European Journal of Operational Research*, vol. 249, no. 1, pp. 1–21, 2016.

Ç. Koç, G. Laporte, et İ. Tükenmez, “A review on vehicle routing with simultaneous pickup and delivery”, *Computers & Operations Research*, p. 104987, 2020.

N. Kohl, J. Desrosiers, O. B. Madsen, M. M. Solomon, et F. Soumis, “2-path cuts for the vehicle routing problem with time windows”, *Transportation Science*, vol. 33, no. 1, pp. 101–116, 1999.

G. D. Konstantakopoulos, S. P. Gayialis, et E. P. Kechagias, “Vehicle routing problem and related algorithms for logistics distribution : a literature review and classification”, *Operational Research*, pp. 1–30, 2020.

Y. A. Koskosidis, W. B. Powell, et M. M. Solomon, “An optimization-based heuristic for vehicle routing and scheduling with soft time window constraints”, *Transportation science*, vol. 26, no. 2, pp. 69–85, 1992.

J. Kytöjoki, T. Nuortio, O. Bräysy, et M. Gendreau, “An efficient variable neighborhood search heuristic for very large scale vehicle routing problems”, *Computers & operations research*, vol. 34, no. 9, pp. 2743–2757, 2007.

A. H. Land et A. G. Doig, “An automatic method for solving discrete programming problems”, dans *50 Years of Integer Programming 1958-2008*. Springer, 2010, pp. 105–132.

G. Laporte et S. Martello, “The selective travelling salesman problem”, *Discrete applied mathematics*, vol. 26, no. 2-3, pp. 193–207, 1990.

J. K. Lenstra et A. H. G. Kan, “Complexity of vehicle routing and scheduling problems”, *Networks*, vol. 11, no. 2, pp. 221–227, 1981.

A. N. Letchford et A. Lodi, “Primal cutting plane algorithms revisited”, *Mathematical Methods of Operations Research*, vol. 56, no. 1, pp. 67–81, 2002.

A. N. Letchford, J. Lysgaard, et R. W. Eglese, “A branch-and-cut algorithm for the capacitated open vehicle routing problem”, *Journal of the Operational Research Society*, vol. 58, no. 12, pp. 1642–1651, 2007.

H. Li et A. Lim, “A metaheuristic for the pickup and delivery problem with time windows”, *International Journal on Artificial Intelligence Tools*, vol. 12, no. 02, pp. 173–186, 2003.

J. Li et W. Lu, “Full truckload vehicle routing problem with profits”, *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 1, no. 2, pp. 146–152, 2014.

C. Lin, K. L. Choy, G. T. Ho, S. H. Chung, et H. Lam, “Survey of green vehicle routing problem : past and future trends”, *Expert systems with applications*, vol. 41, no. 4, pp. 1118–1138, 2014.

S. Lin, “Computer solutions of the traveling salesman problem”, *Bell System Technical Journal*, vol. 44, no. 10, pp. 2245–2269, 1965.

M. Lübbecke et C. Puchert, “Primal heuristics for branch-and-price algorithms”, dans *Operations Research Proceedings 2011*. Springer, 2012, pp. 65–70.

M. E. Lübbecke et J. Desrosiers, “Selected topics in column generation”, *Operations Research*, vol. 53, no. 6, pp. 1007–1023, 2005.

M. Messaoudi, I. El Hallaoui, L.-M. Rousseau, et A. Tahir, “Solving a real-world multi-attribute vrp using a primal-based approach”, dans *International Symposium on Combinatorial Optimization*. Springer, 2020, pp. 286–296.

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

E. Osaba, X.-S. Yang, et J. Del Ser, “Is the vehicle routing problem dead ? an overview through bioinspired perspective and a prospect of opportunities”, dans *Nature-Inspired Computation in Navigation and Routing Problems*. Springer, 2020, pp. 57–84.

J.-Y. Potvin et M.-A. Naud, “Tabu search with ejection chains for the vehicle routing problem with private fleet and common carrier”, *Journal of the operational research society*, vol. 62, no. 2, pp. 326–336, 2011.

N. Rincon-Garcia, B. J. Waterson, et T. J. Cherrett, “Requirements from vehicle routing software : perspectives from literature, developers and the freight industry”, *Transport Reviews*, vol. 38, no. 1, pp. 117–138, 2018.

E. Rönnberg et T. Larsson, “Column generation in the integral simplex method”, *European Journal of Operational Research*, vol. 192, no. 1, pp. 333–342, 2009.

E. Rönnberg et T. Larsson, “All-integer column generation for set partitioning : basic principles and extensions”, *European Journal of Operational Research*, vol. 233, no. 3, pp. 529–538, 2014.

S. Ropke et D. Pisinger, “A unified heuristic for a large class of vehicle routing problems with backhauls”, *European Journal of Operational Research*, vol. 171, no. 3, pp. 750–775, 2006.

S. Rosat, I. Elhallaoui, F. Soumis, et A. Lodi, “Integral simplex using decomposition with primal cuts”, dans *International Symposium on Experimental Algorithms*. Springer, 2014, pp. 22–33.

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

D. Soler, J. Albiach, et E. MartíNez, “A way to optimally solve a time-dependent vehicle routing problem with time windows”, *Operations Research Letters*, vol. 37, no. 1, pp. 37–42, 2009.

M. M. Solomon, “Algorithms for the vehicle routing and scheduling problems with time window constraints”, *Operations research*, vol. 35, no. 2, pp. 254–265, 1987.

A. Stenger, D. Vigo, S. Enz, et M. Schwind, “An adaptive variable neighborhood search algorithm for a vehicle routing problem arising in small package shipping”, *Transportation Science*, vol. 47, no. 1, pp. 64–80, 2013.

A. Tahir, G. Desaulniers, et I. El Hallaoui, “Integral column generation for the set partitioning problem”, *EURO Journal on Transportation and Logistics*, pp. 1–32, 2018.

A. Tahir, G. Desaulniers, et I. El Hallaoui, “Integral column generation for the set partitioning problem”, *EURO Journal on Transportation and Logistics*, vol. 8, no. 5, pp. 713–744, 2019.

É. Taillard, P. Badeau, M. Gendreau, F. Guertin, et J.-Y. Potvin, “A tabu search heuristic for the vehicle routing problem with soft time windows”, *Transportation science*, vol. 31, no. 2, pp. 170–186, 1997.

G. L. Thompson, “An integral simplex algorithm for solving combinatorial optimization problems”, *Computational Optimization and Applications*, vol. 22, no. 3, pp. 351–367, 2002.

P. Toth et D. Vigo, *Vehicle routing : problems, methods, and applications*. Siam, 2014, vol. 18.

V. Trubin, “On a method of solution of integer linear programming problems of a special kind”, dans *Soviet Mathematics Doklady*, vol. 10, 1969, pp. 1544–1546.

F. Vanderbeck, “Implementing mixed integer column generation”, dans *Column generation*. Springer, 2005, pp. 331–358.

T. Vidal, T. G. Crainic, M. Gendreau, et C. Prins, “Heuristics for multi-attribute vehicle routing problems : a survey and synthesis”, *European Journal of Operational Research*, vol. 231, no. 1, pp. 1–21, 2013.

T. Vidal, T. G. Crainic, M. Gendreau, et C. Prins, “A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows”, *Computers & Operations Research*, vol. 40, no. 1, pp. 475–489, 2013.

J. Widuch, “Current and emerging formulations and models of real-life rich vehicle routing problems”, dans *Smart Delivery Systems*. Elsevier, 2020, pp. 1–35.

L. A. Wolsey et G. L. Nemhauser, *Integer and combinatorial optimization*. John Wiley & Sons, 1999, vol. 55.

R. D. Young, “A simplified primal (all-integer) integer programming algorithm”, *Operations Research*, vol. 16, no. 4, pp. 750–782, 1968.

E. E. Zachariadis et C. T. Kiranoudis, “An open vehicle routing problem metaheuristic for examining wide solution neighborhoods”, *Computers & Operations Research*, vol. 37, no. 4,

pp. 712–723, 2010.

A. Zaghrouti, F. Soumis, et I. El Hallaoui, “Integral simplex using decomposition for the set partitioning problem”, *Operations Research*, vol. 62, no. 2, pp. 435–449, 2014.

A. Zaghrouti, I. El Hallaoui, et F. Soumis, “Improving set partitioning problem solutions by zooming around an improving direction”, *Annals of Operations Research*, vol. 284, no. 2, pp. 645–671, 2020.

Y. Zhao, T. Larsson, et E. Rönnberg, “An integer programming column generation principle for heuristic search methods”, *International Transactions in Operational Research*, vol. 27, no. 1, pp. 665–695, 2020.