2016-25
Samira Mirzaei
PhD Dissertation

Optimization Algorithms for Multi-Commodity Routing and Inventory Routing Problems

DEPARTMENT OF ECONOMICS AND BUSINESS ECONOMICS
AARHUS BSS • AARHUS UNIVERSITY • DENMARK

Optimization Algorithms for Multi–Commodity Routing and Inventory Routing Problems

A PhD dissertation

by

Samira Mirzaei

Aarhus BSS, Aarhus University
Department of Economics and Business Economics.

September 2016

Optimization Algorithms for Multi–Commodity Routing and Inventory Routing Problems

Dissertation advisors
Sanne Wøhlk
Anders Thorstenson

"No matter where you're from, your dreams are valid."

Lupita Nyong'o

Table of contents

Figures

Tables

Resumé

Denne afhandling består af tre kapitler, der hvert udgør en selvstændig videnskabelig artikel. De tre artikler relaterer sig alle til flerkammer ruteplanlægningsproblemer, hvor kunderne kræver flere forskellige varer, hvor en flåde af flerkammerinddelte køretøjer servicerer kunderne, og hvor hvert kammer er indrettet til en bestemt vare.

I den første artikel præsenterer vi to strategisk forskellige scenarier på et *Multi-Compartment Vehicle Routing Problem*. I det første scenarie kan forskellige varer blive leveret til kunden af forskellige køretøjer, men den totale mængde af hver enkelt varegruppe, som kunden skal have, skal leveres af ét enkelt køretøj. I det andet scenarie må hver kunde kun serviceres af et enkelt køretøj, og det køretøj skal levere den totale mængde af alle de varer, som kunden har bestilt. Vi præsenterer en Branch-and-Price algoritme til at løse de to scenarier i vores problemstilling optimalt, og ved at sammenligne de optimale omkostningsniveauer i hvert af de to scenarier analyserer vi effekten af den strategiske beslutningsregel, der bestemmer, om der skal tillades flere besøg hos den samme kunde eller ej. Beregningsresultaterne viser, at algoritmen kan løse instanser med op til 50 kunder og 4 varegrupper optimalt.

I den anden artikel introducerer vi *the Periodic Multi-Compartment Vehicle Routing Problem*, som er en udvidelse af *the Periodic Vehicle Routing Problem*, hvor hver kunde ofte har brug for gentagne leverancer af forskellige varegrupper inden for en given tidsperiode. Vi fremlægger en matematisk formulering af dette problem og udvikler en metaheuristisk ramme baseret på en *adaptive large neighborhood search* for at opnå gode løsninger på denne type af problemstillinger. Vi tester vores algoritme på i alt 120 enkeltkammer- og flerkammerinstanser. Beregningsresultaterne for instanserne med enkeltkammerkøretøjer indikerer, at algoritmen kan finde gode løsninger inden for en rimelig tid på instanser med op til 100 kunder og 10 tidsperioder. Nye benchmark instanser laves til *the Periodic Multi-Compartment Vehicle Routing Problem*, og beregningsresultaterne fremlægges.

I den sidste artikel undersøger vi et *Multi-Compartment Inventory Routing Problem* i et scenarie, hvor varerne opbevares separat både på lagrene og i køretøjerne. Ofte har hver enkelt kunde brug for gentagne leverancer af forskellige varer inden for en given tidsperiode, og kunden kan modtage dem fra forskellige køretøjer. Ifølge leveringsplanen for hver eneste vare i hver periode skal den totale mængde af en bestemt vare dog stadig leveres af ét køretøj. Vi foreslår to matematiske formuleringer af dette problem, og vi udvikler en heuristisk Branch-and-Price algoritme til at opnå gode løsninger på denne

type af problemstillinger. Beregningseksperimenterne viser, at den foreslåede heuristiske Branch-and-Price algoritme inden for en rimelig tid kan opnå løsninger af høj kvalitet for de fleste af instanserne i testen.

Summary

This dissertation consists of three chapters, each of which constitutes a self-contained research paper. The three papers are all related to Multi-Compartment Routing Problems in which customers require multiple commodities and a fleet of multi-compartment vehicles services the customers and each vehicle compartment is dedicated to one commodity.

In the first paper, we present two strategically different versions of the Multi-Compartment Vehicle Routing Problem. In the first version, different commodities may be delivered to the customer by different vehicles, but the full amount of each commodity must be delivered by a single vehicle. In the second version, each customer may only be serviced by a single vehicle, which must deliver the full amount of all commodities demanded by that customer. We present a Branch-and-Price algorithm for solving the two versions of the problem to optimality and we analyze the effect of the strategic decision of whether or not to allow multiple visits to the same customer by comparing the optimal costs of the two versions. Computational results show that the algorithm can solve instances with up to 50 customers and 4 commodities to optimality.

In the second paper, we introduce the Periodic Multi-Compartment Vehicle Routing Problem, which is an extension of the Periodic Vehicle Routing Problem in which each customer often requires repeated delivery of multiple commodities over a time horizon. We present a mathematical formulation for this problem and develop a meta-heuristic framework based on adaptive large neighborhood search to obtain good solutions for this class of problem. We test our algorithm on a total of 120 single-compartment and multi-compartment instances. Computational results for the single-compartment instances indicate that the algorithm can find good solutions within a reasonable time for instances with up to 100 customers and 10 periods. New benchmark instances are presented for the Periodic Multi-Compartment Vehicle Routing Problem and computational results are presented.

In the last paper, we investigate a Multi-Compartment Inventory Routing Problem in a setting where the commodities are kept separated both in the inventories and in the vehicles. Each customer often requires repeated deliveries of multiple commodities over a time horizon and may receive them from different vehicles. However, based on the delivery plan of each commodity in each period, the full amount of the commodity must still be delivered by a single vehicle. We propose two mathematical formulations for this problem and we develop a heuristic Branch-and-Price algorithm to find good solutions for this class of

problem. Computational experiments show that the proposed heuristic Branch-and-Price algorithm can obtain high quality solutions within a reasonable amount of time for most of the test instances.

Preface

This dissertation has been prepared at the School of Business and Social Sciences, Aarhus University in partial fulfillment of the requirements for acquiring the PhD degree in Economics and Business. The work was carried out at the Department of Economics and Business Economics.

The dissertation extended the literature by investigating three different variants of routing problems in a multi-compartment setting. The dissertation considers exact methods, heuristics, metaheuristics and matheuristics, to solve the problems.

Three research papers are included in the dissertation. The first paper has been accepted and is available on-line, the second paper has been submitted to a peer-reviewed operations research journal, and the last paper is a working paper which is expected to be submitted in December 2016.

Acknowledgments

My deepest gratitude goes first and foremost to my supervisor, Sanne Wøhlk, for her untiring encouragement, guidance, and support. She walked with me through all the steps of my PhD studies and without her advice and contributions, this dissertation could not have reached its present form. I am so fortunate to have her as my supervisor. I would also like to thank my co-supervisor Prof. Anders Thorstenson for supporting me and providing me with insightful suggestions.

I would like to thank all the members of CORAL for providing a very pleasant research environment. I am grateful to the head of the CORAL group, Prof. Kim Allan Andersen, for his support and his advice during my employment at Aarhus University.

I offer my gratitude to the Department of Economics and Management at University of Brescia for their hospitality during my stay as visiting PhD student from February until July 2016. I want to especially thank Dr. Claudia Archetti who is co-author of my third paper. I learned a lot from her during my exchange program and she has always been there when I needed support.

I am also thankful to the following administrative staff of the department for their various forms of support during my PhD studies: Ingrid Lautrup, Karin Vinding, Christel Mortensen, and Susanne Christensen.

I would like to thank all my friends and PhD colleagues for their friendship and support during these three years. I specially want to thank Reza and Mahboobe Pourmayed, Parisa Bagheri Tookanlou, Marjan Mohammadreza, Maryam Ghoreishi, Ata Jalili Marand, Camilla Pisani, Viktoryia Buhayenko, Lukas Bach, Agus Darmawan, Sune Lauth Gadegaard, Maria Elbek Andersen, Hani Zbib, and Lone Kiilerich, for many stimulating discussions support and fun throughout these three years.

Last, but definitely not least, I want to give a special thank to my family who always support me and believe in me: to my parents Maryam and Ebrahim, who let me find my own way and taught me to be patient, strong, and never give up, to my sisters and brothers Saeedeh, Sarah, Saeed, Hamid, and Nader, who always are my best friends no matter how far apart we are, and to my lovely niece Ava, who fills our hearts with lots of smiles and abundant joy.

Samira Mirzaei
September 2016

Introduction

A supply chain is a network of organizations that are involved in moving a product or service from a supplier to a customer. Supply chain activities include the transformation of raw materials and components into a finished good that is delivered to a final customer. The fundamental idea in supply chain management is to enhance the collaboration at various stages of the supply chain. Therefore, decisions relating to the different stages should be integrated to allow examination of possible trade-offs for improved overall performance. Supply chain management involves decisions at the strategic, tactical, and operational levels as follows: The strategic level determines the site and purpose of business facilities, production technologies to be employed at each facility, and the inventory and product management of each plant. Strategic decisions determine the network through which production, assembly, and distribution serve the marketplace. The strategic level creates a network of reliable suppliers, transporters, and logistics handlers and thereby provides the environment in which tactical and operational actions must be performed. The tactical level determines material flow management policies and common concerns at this level include procurement contracts for the necessary materials and services, production schedules and guidelines to meet quality at all plants, transportation and warehousing solutions, and inventory levels. The operational level arranges operations in such a way as to guarantee that final products are delivered to customers in time and the logistic network is organized to be responsive to customer demands. The operational level includes day-to-day processes, decision-making, and planning that are necessary to keep the supply chain active. Some well-known activities at this level are daily and weekly forecasting to satisfy demand, making schedule changes to production, monitoring logistics activities, and managing incoming and outgoing materials and products.

Improvement of the supply chain at the strategic level is potentially complex and costly, but by better decision-making at the tactical and the operational levels, the performance of the supply chain can be improved and the overall cost of the supply chain may be decreased by lower investment. For instance, the major components of logistic costs are transportation costs, representing approximately one third, and inventory costs, representing one fifth of logistic costs, Tseng et al. (2005). The improvement of transportation and inventory management activities can improve operational efficiencies and increase productivity. Vehicle Routing Problems, Periodic Vehicle Routing Problems, and Inventory Routing Problems are important and well-known combinatorial optimization problems which have been investigated by different researchers in

order to minimize logistic costs while satisfying service level requirements.

The Vehicle Routing Problem (VRP) is the most studied combinatorial optimization problem and occurs in many transportation and distribution systems with considerable economic benefits. Numerous companies and organizations in the area of logistics and transportation deal with variations of this problem every day.

In many supply chains such as e.g. grocery distribution and waste collection where periodic pickups or deliveries are made to a set of customers over a time horizon, optimizing these periodic operations can be a cause of significant cost savings. This problem is known as the Periodic Vehicle Routing Problem (PVRP).

An interesting example of supply chain collaboration is the concept of Vendor Managed Inventory (VMI). Under VMI, customers make their inventory data available to their suppliers (distributors) who take over the responsibility of making decisions to replenish customers. This integrated inventory and distribution management offers a large degree freedom for designing efficient vehicle routes, while optimizing inventory across the supply chain. The Inventory Routing Problem (IRP) is based on this idea. The aim of the IRP is to integrate the transportation activities and inventory management along the supply chain and avoid inefficiency by solving the underlying vehicle routing and inventory sub-problems separately. Thus the IRP is an important problem when studying Supply Chain integration.

In practice, supply chains such as petroleum distribution, waste collection, and grocery distribution are often faced with a situation where customers request delivery of multiple commodities, and these commodities need to be keep separated during transport and storage. In these supply chains, multi-compartment vehicles deliver commodities to customers and they have dedicated storage for different commodities. In the academic literature, there is, however, a tendency to simplify problems by considering single-commodity models or sometimes multi-commodity models with joint storage in the vehicles and in the inventories. Thus most models assume that each customer requires only one type of commodity and there are few papers in this area that investigate multi-commodity problems. The main contribution of this dissertation is to study the VRP, the PVRP, and the IRP in a multi-compartment setting where different commodities are delivered to customers by a fleet of multi-compartment vehicles. Each compartment of the vehicle is dedicated to a specific commodity and the commodity is stored in a dedicated storage at the customer location. We investigate this problem for a single period as the Multi-Compartment Vehicle Routing Problem (MCVRP) in the first chapter and for multiple periods as the Periodic Multi-Compartment Vehicle Routing Problem (PMCVRP) and the Multi-Compartment Inventory Routing Problem (MCIRP) in the second and third chapters, respectively.

We consider two different delivery policies for multiple commodities. In the first policy, which we refer to as Commodity-Split (hereafter C-Split), a customer may receive different commodities from different vehicles. However, the full amount of each commodity must still be delivered by a single vehicle in each period of the planning horizon, i.e. we do not allow split delivery as regards the individual commodity. In the second policy, which we refer to as No-Split delivery, each customer may only be serviced by a single vehicle in each period of the planning horizon and that vehicle must deliver the full amount of all

commodities demanded by that customer. As customers often prefer to have all commodities supplied by one vehicle and managers prefer to decrease the overall cost, the cost difference between these policies is an interesting research subject. If the cost difference between these policies is more or less insignificant, considering No-Split delivery will be a cause of higher customer satisfaction.

The most commonly used techniques for solving routing problems are exact solution methods, heuristic, and metaheuristic techniques. Exact solution methods such as e.g. Branch-and-Bound, Branch-and-Price, etc. can find optimal solutions, but they are often extremely time-consuming when solving real-world problems. However, applying exact methods is still valuable for solving new variants of routing problems or new benchmark instances because the exact methods can obtain optimal solutions to be used for future evaluation of the quality of heuristic methods. Heuristic and metaheuristic techniques are powerful search algorithms that are able to produce good solutions in a reasonable time for difficult problems. In many cases, however, no proof is given as to the quality of these solution methods. Recently, matheuristic techniques have been applied to several different routing problems. Matheuristics often make use of mathematical programming models combined with a heuristic framework. All of the above-mentioned techniques have successfully been applied to a wide range of single-commodity routing and transportation problems. However, due to the limited existing work on multi-compartment problems, the transformation of the techniques needed when extending from one to multiple commodities and from one compartment to multiple has not yet been fully explored. In this dissertation, we are applying a wide range of solution approaches that are specially designed for solving these multi-compartment routing problems. This is the second contribution of the dissertation. In the first paper, we apply a Branch-and-Price algorithm, which is an exact method. In the second paper a metaheuristic framework is presented, and in the last paper a heuristic Branch-and-Price algorithm is implemented which is a matheuristic technique.

This research is the first study on multi-compartment routing problems, but not the last one. Additional research is needed on the topic and future research could focus on solving the problems with real data, investigating the problems by flexible compartment size, considering the problems by heterogeneous vehicles, adding real life constraints such as time windows, deliveries under specific rest rules, traffic conditions, etc. to different variants of the problems. Furthermore, much more research is needed to develop fast algorithms for solving large instances of the problem.

We also contribute by developing new sets of benchmark instances for the MCVRP, PMCVRP, and MCIRP. All new test sets will be made available online for future research on
http://www.optimization.dk

Chapter 1: A Branch-and-Price Algorithm for Two Multi-Compartment Vehicle Routing Problems

The first chapter of this dissertation investigates the Multi-Compartment Vehicle Routing Problem. This problem is a variation of the VRP in which the customers request delivery of different commodities and

these commodities need to be kept segregated during transport. There are different applications for this problem, e.g. petroleum and animal feed distribution, waste collection, home delivery of groceries, etc. but there are few papers that investigate the Multi-Compartment Vehicle Routing Problem (MCVRP). Most of these papers investigate this problem for a special application. For instance Avella et al. (2004) and Cornillier et al. (2007) investigate delivery of petroleum products and El Fallahi et al. (2008) only consider animal feed distribution. In this research, we investigate the No-Split and the C-Split MCVRP in a general setting. We propose mathematical models based on a flow formulation and a set portioning formulation for these two strategically different versions of the MCVRP.

Furthermore, most of the papers that consider MCVRP apply heuristic based solution approaches. In many cases, neither lower bounds nor optimal solutions exist and therefore, the quality of the obtained solutions is not known. There are some papers that present exact solution methods for solving the MCVRP (see Avella et al. (2004), Cornillier et al. (2007), Lahyani et al. (2015)). However, Avella et al. (2004) and Cornillier et al. (2007) consider some constraints that simplify the analysis and facilitate applying exact methods and Lahyani et al. (2015) do not investigate the problem in the No-Split setting. Therefore, we develop a Branch-and-Price algorithm for solving the No-Split and C-Split MCVRP to optimality and perform a comparison between the two.

When we started our work, we found that many of the instances used by the heuristics are too large in terms of capacities for our exact algorithm to tackle. We have therefore made 5 new datasets containing a total of 189 new data instances. We also use part of the data of Muyldermans and Pang (2010). In total we have tested our algorithm on 274 instances.

The computational results show that the algorithm can solve instances with up to 50 customers and 4 commodities for the No-Split MCVRP. The algorithm is able to solve 112 of the instances to optimality for the No-Split strategy. We analyzed 105 instances for the C-Split strategy and the algorithm can solve 45 of them to optimality. We also compared two delivery policies for 105 instances and in 43 percent of the instances the cost of the C-Split MCVRP was lower than that of the No-Split MCVRP. For 27 percent of the instances the costs of both delivery policies were the same and for the rest of the instances we were not able to draw a conclusion.

Chapter 2: An Adaptive Large Neighborhood Search Heuristic for a Periodic Multi-Compartment Vehicle Routing Problem

In many supply chains, multi-compartment vehicles service the customers. Considering periodic delivery of commodities to a set of customers over a planning horizon (instead of planning for single period) can lead to significant cost savings. Therefore, in the second chapter, we introduce the Periodic Multi-Compartment Vehicle Routing Problem (PMCVRP), which is an extension of the Periodic Vehicle Routing Problem.

For this problem we consider a distribution network with a depot, a set of customers, and a fleet of homogenous multi-compartment vehicles. Each compartment of the vehicles is dedicated to one commodity.

Each customer has a known demand for a set of commodities that must be delivered a known number of times during the planning horizon. Each customer may only be serviced by a single vehicle in each period of the planning horizon. We investigate this problem in the No-Split setting, which means that the vehicle must deliver the full amount of all commodities ordered for that period by the customer. The possible delivery days for each commodity are given by a commodity-schedule that is chosen from a commodity-schedule menu and the possible days of visits for each customer are given by a customer-schedule that is chosen from a customer-schedule menu.

We have chosen to solve the PMCVRP by iteratively considering three steps. At first, a customer-schedule is selected for each customer from the menu of customer-schedules. Then, a commodity-schedule is chosen for each commodity requested by the customer from the commodity-schedule menu such that the visit schedule of the customer is satisfied. Finally, the daily routes are designed with the objective of minimizing the total cost. This is integrated in an adaptive large neighborhood search (ALNS) algorithm.

Since there are no test problems for the PMCVRP in the literature, we used benchmark instances of the PVRP to compare the performance of the proposed algorithm for just a single commodity with those obtained by the three most recent algorithms in the literature. We have also created a total of 90 new, multi-compartment instances for the PMCVRP.

Our algorithm is able to find good solutions within a reasonable time for the instances with up to 100 customers and 10 periods. The instances with long time horizons and high number of customers are harder to solve, although the average gap for all PVRP instances is less than 4.6 percent. When we look at our results for the new datasets we see that instances with short time horizons, low number of customers, and only few commodities are the easiest to solve. Because this is the first paper to study the PMCVRP, no direct comparison can be performed for these instances.

Chapter 3: A Heuristic Branch-and-Price Algorithm for the Multi-Compartment Inventory Routing Problem

In the last chapter, we consider the Multi-Compartment Inventory Routing Problem (MCIRP). Although Multi-Commodity Inventory Routing Problems are investigated by different researchers, there are only few papers (related to fuel distribution) that consider dedicated storage for each commodity at the supplier's and customers' locations as well as in the vehicles (see Popović et al. (2012), Vidović et al. (2014), and Coelho and Laporte (2015)). We are not aware of any research addressing the Multi-Compartment Inventory Routing Problem in a general setting.

For this problem, we consider a two-echelon supply chain involving a supplier, a set of customers, and a fleet of homogenous multi-compartment vehicles. The supplier produces and stores a set of commodities. The commodities are delivered to the customers by a homogeneous fleet of multi-compartment vehicles where each compartment of the vehicle is dedicated to one commodity. Each customer often requires repeated delivery of multiple commodities over a time horizon and may receive them from different vehicles.

However, based on the delivery plan of each commodity in each period, the full amount of the commodity must still be delivered by a single vehicle (C-Split MCIRP). Each customer has a dedicated storage location with limited capacity for each commodity. The aim of this problem is to specify replenishment plans for each period of the planning horizon that minimize total inventory and routing cost in such a way that out-of-stock situations never occur for the supplier and the customers.

We present a mathematical model based on flow formulation for the problem. We also propose a set partitioning model for the MCIRP which is used for solving the problem by a heuristic Branch-and-Price algorithm. The set partitioning model contains a sub-problem and a master problem. We apply three heuristic labeling algorithms named No-Semi, Friends, and Dominance heuristics for solving the sub-problem.

Since there are no test problems for the MCIRP, we used the IRP benchmark instances and compared the performance of the MCIRP for a single commodity with the two most recent algorithms. We have also created a total of 90 new data instances for the MCIRP by modifying the IRP benchmark instances.

Computational results for the IRP dataset indicate that for the instances with up to 20 customers and 3 periods, the algorithm can find good solutions within a reasonable time with an average optimality gap of 1.14 percent. From the results for the IRP and the MCIRP instances in general, we see that the capability of our algorithm decreases when the number of nodes, the capacity of vehicles, the number of commodities, and the length of the planning horizon increase. Because this is the first paper to study the MCIRP, no direct comparison can be performed for these instances.

Discussion of Different Solution Approaches

A number of techniques can be implemented for solving various Multi-Compartment Routing Problems. These techniques include exact solution methods, heuristics as well as metaheuristics. The exact methods are generally relatively slow, but they can find optimal solutions. Heuristic and metaheuristic methods are usually faster than exact methods, but they will often provide sub optimal solutions and we need other ways to evaluate the quality of their solutions.

In recent years, researchers have implemented heuristic and meta-heuristics for solving the MCVRP, but in many cases, neither lower bounds nor optimal solutions exist and therefore the quality of the obtained solutions is unknown.

Exact algorithms for solving Vehicle Routing Problems include Branch-and-Cut and Branch-and-Price algorithms. A Branch-and-Cut algorithm is a Branch-and-Bound algorithm that uses cutting planes to improve the lower bound and thereby tighten the duality gap. This algorithm has recently been implemented for the VRP by Lysgaard et al. (2004), Ralphs et al. (2003), and Achuthan et al. (2003). However, as mentioned by Fukasawa et al. (2006), for instances with more than 7 feasible routes, adding several classes of cuts does not guarantee tight lower bounds. Closing the resulting duality gap and finding the optimal solution usually requires exploring several nodes in the branch-and-cut tree. When we compare the latest

papers in this area (see Lysgaard et al. (2004) and Ralphs et al. (2003)), we note that despite of substantial theoretical and implementation efforts in Lysgaard et al. (2004), the results are only marginally better than Ralphs et al. (2003). Petroleum distribution is an important example of an MCVRP problem but each vehicle can usually only service a maximum of 4 customers and hence the number of feasible routes in this problem is often greater than seven. The same applies to grocery distribution as the number of feasible routes usually also exceeds seven. This solution approach is therefore not efficient for our problem.

Branch-and-Price is a Branch-and-Bound algorithm in which the LP-relaxation of each node in the branching tree is solved by dynamically adding columns with negative reduced costs to a master problem. Due to its many successes, Branch-and-Price has rapidly become an approach that cannot be ignored when solving Vehicle Routing Problems. Furthermore, the most successful exact solution approach for solving the VRP is the Branch-and-Cut-and-Price algorithm presented by Pecin et al. (2014) that can solve all VRP instances from the literature with up to 199 nodes to optimality.

We also implemented the No-Split MCVRP model based on flow constraints presented in section 1.3.2 in Lingo 14, created an MPS data file, and ran the resulting model in a CPLEX environment. The CPLEX could solve instances with at most 25 nodes and 2 commodities to optimality, but for larger instances, CPLEX could not find optimal solution within 24 hours. Based on the limitations of the other solution approaches, we decided to apply the Branch-and-Price for the MCVRP.

The PMCVRP is a variation of the PVRP in which multiple commodities are delivered by multi-compartment vehicles during a planning horizon. Although the PVRP has been studied by several authors, we are not aware of any research addressing the generalization of this problem with the inclusion of multi-compartment vehicles.

PVRP is NP-hard and considering multi-commodities and multi-compartment vehicles add complexity to this problem. Due to the difficulty of the PVRP, heuristics and meta-heuristics have been the main approach to solving this problem. For instance, Hemmelmayr et al. (2009) presented a variable neighborhood search and Alegre et al. (2007) proposed an adaptation of scatter search for the problem. Due to the complexity of the PMCVRP, we decided to approach it by a meta-heuristic.

We reviewed existing PVRP literature and found high diversity among the presented meta-heuristics for solving this problem, but no solution approach seemed immediately superior for our PMCVRP. On the other hand, adaptive large neighborhood search has been very successful within the area of routing problems and has provided state-of-the-art results at the time of publication. A number of researchers implemented this solution approach for the VRP, VRP with time window, pickup and delivery problem with time window, and so on. For instance, Røpke and Pisinger (2006) presented the adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows and their paper has been cited 634 times, which shows the quality of their solution approach. Therefore, we decided to implement adaptive large neighborhood search for the PMCVRP.

The Inventory Routing Problem is a combination of the VRP and an Inventory Management problem. Due to the underlying VRP subproblem, this problem is NP-hard. Its complexity grows exponentially with

the number of customers and time periods. In addition, considering the multi-compartment vehicles in this research adds complexity to the problem.

A number of researchers have presented heuristics and meta-heuristics for solving the IRP. The running time of meta-heuristics is shorter than for exact methods and matheuristic. However, due to the use of random numbers in most of the meta-heuristics, the quality of the solution changes in different runs. Furthermore, there is no proof of quality of the solution.

Few papers present exact algorithms for the IRP (see Coelho et al. (2013)). One of the most successful solution approaches is a Branch-and-Price-and-Cut algorithm presented by Desaulniers et al. (2015) that used a set of 640 benchmark instances involving between two and five vehicles. Their solution approach performed well on the instances with four and five vehicles. They also proved optimality for 54 of 238 instances which were still open before their work.

Based on the experience from the first project, we knew that implementation of the Branch-and-Price algorithm without considering different types of cuts was not likely to be efficient for our problem that also included multi-commodity and multi-compartment vehicles. We also wanted to have a solution approach that could be adapted to finding an optimal solution to the problem, or at least to finding a lower bound in the future. Therefore we decided to implement the heuristic Branch-and-Price algorithm which is faster than the Branch-and-Price algorithm but includes the possibility to modify it in order to obtain lower bounds for the problem in the future. An additional benefit of this decision was that the present dissertation can present three different solution approaches.

Chapter 1

A Branch–and–Price Algorithm for Two Multi–Compartment Vehicle Routing Problems

History: *This paper was prepared in collaboration with Sanne Wøhlk. The research was conducted from the autumn of 2013 to the Summer of 2015. The paper was submitted to EURO Journal on Transportation and Logistics in August 2015, and was accepted in June 2016. The work was presented at the OR 2015, Vienna, Austria, and at the ICCL 2015, Delft, The Netherlands.*

A Branch-and-Price Algorithm for Two Multi-Compartment Vehicle Routing Problems

Samira Mirzaei[†], Sanne Wøhlk[†]

[†]Department of Economics and Business Economics, Aarhus University, Denmark, {mirzaei, sanw}@econ.au.dk

Abstract

Despite the vast body of literature on vehicle routing problems, little attention has been paid to multi-compartment vehicle routing problems that investigate transportation of different commodities on the same vehicle, but in different compartments. In this project, we present two strategically different versions of the MCVRP in general settings. In the first version, different commodities may be delivered to the customer by different vehicles, but the full amount of each product must be delivered by a single vehicle. In the second version, each customer may only be serviced by a single vehicle, which must deliver the full amount of all commodities demanded by that customer. We present a Branch-and-Price algorithm for solving the two versions of the problem to optimality and we analyze the effect of the strategic decision of whether or not to allow multiple visits to the same customer by comparing the optimal costs of the two versions. Computational results are presented for instances with up to 100 customers and the algorithm can solve instances with up to 50 customers and 4 commodities to optimality.

1.1 Introduction

Vehicle routing problems have been the object of numerous studies and a very large number of papers propose solution methods for solving these problems, see Laporte et al. (2000) and Toth and Vigo (2014). Yet, despite the progress in the field, many challenges still exist and new ones are emerging. In practice, for petroleum and animal feeds distribution, waste collection, home delivery of groceries, etc. the distribution systems are often challenged by a situation where the customers request delivery of different commodities, and these commodities need to be kept segregated during transport. For bulk material, such separation into compartments is necessary to avoid mixing different types of material. For home delivery of groceries, the need for multiple compartments originates from the different temperature requirements for different products. When different types of waste are co-collected, it is important for further treatment that the waste is kept separated during transport. Models and algorithms for solving such multi-compartment problems are thus highly relevant in practice.

In the academic literature, there is, however, a tendency to simplify problems by considering single-commodity models. Most studies assume that each customer requires only one type of commodity, whereas

few papers in this area investigate Multi-Compartment Vehicle Routing Problems (MCVRP). Furthermore, most of the papers that do consider multiple compartments, apply a heuristic based solution approach.

In this project, we present two strategically different versions of the MCVRP. In the first version, which we refer to as Commodity-Split (hereafter C-Split) MCVRP, a customer may receive different commodities from different vehicles. However, the full amount of each commodity must still be delivered by a single vehicle, i.e. we do not allow split delivery as regards the individual commodity. In the second version, which we refer to as No-Split MCVRP, each customer may only be serviced by a single vehicle, which must deliver the full amount of all commodities demanded by that customer. From a customer perspective, a No-Split service will often be perceived as the best service. This is particularly the case for private citizens who prefer that all ordered groceries arrive in one delivery and that all waste is collected jointly by a single vehicle.

This project has two purposes. Firstly, we develop a Branch-and-Price algorithm for solving the two versions of the MCVRP to optimality. Our algorithm uses heuristic as well as exact column generation and is able to solve instances with up to 50 nodes and 4 commodities to optimality within an hour. Not all instances up to this size could be solved, however. Secondly, we analyze the effect of the strategic decision of whether or not to allow deliveries to a single customer to be shared among several vehicles by comparing the costs of the C-Split and the No-Split versions of the MCVRP.

1.2 Literature Review

The most important application of MCVRP is related to petroleum distribution and is known as the petrol station replenishment problem. Brown and Graves (1981) consider a petroleum products dispatching problem from a single terminal to a set of customers. They use optimization routines for automation of this problem and in order to reduce manual operations and operating costs. In addition to optimization algorithms, they propose a heuristic sequential network assignment for this problem. Brown et al. (1987) explain the computer system of Mobil Oil Corporation for centralized control of petroleum distribution to their customers in the United States. This system exploits the knowledge of humans and assists Mobil to reduce its costs and staff and improve the customer service.

Van der Bruggen et al. (1995) redesign the distribution structure of a large oil company in the Netherlands. They choose a hierarchical approach and decompose the problem into three sub-problems including assignment of clients to depots, assignment of products to truck compartments, and route planning. They apply a heuristic for solving the VRP sub-problem. Ben Abdelaziz et al. (2002) solve the petroleum delivery problem of a Tunisian company. They present a mathematical model for this problem. They design a set of least-cost vehicle routes subject to several constraints and they apply the Variable Neighborhood Search heuristic to reach a near-optimal solution.

Avella et al. (2004) investigate the problem of delivering different types of fuel to a set of pumps. They formulate this problem as a set partitioning model and solve it by a Branch-and-Price algorithm. The

initial columns for the Branch-and-Price algorithm are provided by an innovative combinatorial heuristic. Cornillier et al. (2007) consider delivery of petroleum products to petrol stations and decompose this problem into a truck loading problem and a routing problem. They first use a heuristic for solving the loading problem. Due to the structure of the problem, there are three possible outcomes: 1) An optimality test proves that the resulting solution is optimal; 2) A solution was produced but the optimality test failed; and 3) The heuristic did not obtain a feasible solution. In the latter two cases, the algorithm proceeds by solving an ILP model to obtain an optimal solution. They use two different algorithms, based on a matching approach or a column generation scheme for solving the routing sub-problem. Cornillier et al. (2008) investigated the same problem in a multi-period setting. They present a mathematical model and a heuristic for this problem. The heuristic includes procedures for route construction, truck loading, route packing, and postponement of deliveries. Inspired by the same type of problem, Coelho and Laporte (2015) study a multi-period inventory routing problem. They present a classification of this problem, which, on one hand, considers the option of allowing the content of each vehicle compartment to be split between multiple customers (or multiple tanks at the same customer), and on the other hand, considers the option of allowing the delivery to each customer tank to come from multiple vehicles. They present an exact algorithm for solving the four types of problems originating from this classification. They also consider the problems in a single period setting, which, due to the splitting definition, are variations of multi-commodity VRP problems that differ from the ones studied in the present paper.

Distribution of animal feed is another application of MCVRP. El Fallahi et al. (2008) consider a fleet of multi-compartment vehicles where each compartment is assigned to one commodity and the full demand of each customer for a commodity must be delivered by a single vehicle. It is, however, possible to deliver different commodities by different vehicles to the same customer. This is what we refer to as the C-Split MCVRP in this paper. They present a mathematical model for this problem and propose a heuristic approach based on a Memetic algorithm and Tabu search for solving it.

Due to the increased focus on sorting, waste collection is a more recent application of MCVRP. Muyldermans and Pang (2010) investigate this problem with assumptions similar to El Fallahi et al. (2008). They propose a local search procedure using 2-opt, cross, exchange, and relocate for finding efficient moves. They apply the mechanisms of neighbor lists and marking for speed improvement. To enhance the solution quality, they combine this procedure with a Guided Local Search heuristic. Reed et al. (2014) use an Ant Colony System to solve CVRP and MCVRP for waste collection from households. Henke et al. (2015) present a model for this problem with flexible compartment sizes and solve the problem by a Variable Neighborhood Search.

Chajakis and Guignard (2003) investigate MCVRP for distribution to convenience stores. They propose mathematical models for two possible cargo space layouts and solve this problem by a Lagrangian Relaxation algorithm. Lahyani et al. (2015) investigate the olive oil collection process in Tunisia as a multi-period MCVRP. They present a mathematical model along with a set of known and new valid inequalities. They use this in a Branch-and-Cut algorithm and evaluate the performance of the algorithm on real data sets

under different transportation scenarios.

MCVRP has also been investigated in more general settings: Repoussis et al. (2007) present a hybrid of a Greedy Randomized Adaptive Search procedure and a Variable Neighborhood Search for the heterogeneous fixed fleet No-Split MCVRP. Derigs et al. (2011) present an integer mathematical model for the C-Split MCVRP and propose a range of algorithms that use different approaches for construction, including Local Search, Large Neighborhood Search, Simulated Annealing, Record-to-Record Travel, and Tabu Search. Wang et al. (2014) formalize the heterogeneous fixed fleet multi-compartment vehicle routing problem in the C-Split setting and present a Reactive Guided Tabu Search for solving this problem. Archetti et al. (2014) study different strategies for distributing a set of commodities to customers. They compare the effect of using vehicles dedicated to one type of commodity to using vehicles that carry different commodities in a single compartment. They also consider the option of splitting orders in various ways. They solve small instances with up to 15 customers and 3 compartments to optimality within 30 minutes by a Branch-and-Cut algorithm. Additional instances with up to 100 customers are solved heuristically by creating multiple copies of each customer based on the commodities requested by the customer, and applying a heuristic for the VRP. Archetti et al. (2015) apply a Branch-and-Price-and-Cut algorithm on the C-Split MCVRP. They consider two different acceleration heuristics based on a state space relaxation technique and 2-cycle elimination for speeding up the Label Setting Algorithm. The proposed algorithm solves all of the instances with 15 customers, and some instances with up to 40 customers and 3 commodities to optimality within two hours.

In most of the related papers, heuristics and meta-heuristics are used, but in many cases, neither lower bounds nor optimal solutions exist and therefore, the quality of the obtained solutions is not known. In contrast, Avella et al. (2004), Cornillier et al. (2007), Lahyani et al. (2015), and Archetti et al. (2015) present exact solution methods for solving the MCVRP. However, Avella et al. (2004) assume that truck compartments are either empty or full and that the number of customers on each route is at most four. Cornillier et al. (2007) assume that the content of each compartment cannot be split between customers and that the number of customers visited by each truck on any given route will not exceed two. Each of these constraints simplifies the analysis and facilitates applying exact methods. Lahyani et al. (2015) present a Branch-and-Cut algorithm for the multi-period MCVRP, but they have not investigated the problem in the No-Split setting. Archetti et al. (2015) apply a Branch-and-Price-and-Cut algorithm on the C-Split MCVRP. They have not studied this problem in No-Split setting.

1.3 Mathematical Models

In this section, we present mathematical models for the MCVRP. We first consider the C-Split MCVRP and present a model related to this problem in Section 1.3.1. In Section 1.3.2 we then investigate the No-Split MCVRP. The main difference between these models is related to the possibility of multiple visits. In the C-Split delivery, each of the commodities demanded by a customer may be serviced by separate vehicles

but in the No-Split delivery, the full set of commodities demanded by a customer should be serviced by a single vehicle.

Consider a complete graph $G(N,E)$ where $N = \{0,1,...,n\}$ is a set of nodes including one depot (node 0) and a set N' of n customers and E is a set of edges. The depot stores a set M of commodities which must be delivered by a homogeneous fleet K of vehicles, each with $|M|$ compartments. Compartment m of each vehicle is dedicated to commodity m and has a known capacity Q_m. Each customer $i \in N'$ has a known demand $q_{im} \leq Q_m$ for each commodity m, possibly null for some commodities not ordered by the customer. We use M_i to denote the set of commodities demanded by customer i, i.e. $M_i = \{m \in M | q_{im} > 0\}$. The travel cost from $i \in N$ to $j \in N$ is given by c_{ij} and is symmetric.

1.3.1 C-Split MCVRP

In the following, we will focus on the C-Split MCVRP and define

$$y_{imk} = \begin{cases} 1 & \text{if customer } i \in N' \text{ receives commodity } m \in M \text{ from vehicle } k \in K \\ 0 & \text{otherwise.} \end{cases}$$

$$x_{ijk} = \begin{cases} 1 & \text{if vehicle } k \in K \text{ travels directly from } i \in N \text{ to } j \in N \\ 0 & \text{otherwise.} \end{cases}$$

The C-Split MCVRP can then be stated as follows:

$$\min \quad \sum_{i \in N} \sum_{j \in N} \sum_{k \in K} c_{ij} x_{ijk} \tag{1.1}$$

$$\text{st} \quad \sum_{k \in K} y_{imk} = 1 \quad \forall i \in N', \forall m \in M_i \tag{1.2}$$

$$\sum_{i \in N'} y_{imk} q_{im} \leq Q_m \quad \forall m \in M, \forall k \in K \tag{1.3}$$

$$y_{imk} \leq \sum_{j \in N} x_{ijk} \quad \forall i \in N', \forall m \in M_i, \forall k \in K \tag{1.4}$$

$$\sum_{j \in N} x_{0jk} \leq 1 \quad \forall k \in K \tag{1.5}$$

$$\sum_{j \in N} x_{ijk} = \sum_{j \in N} x_{jik} \quad \forall i \in N', \forall k \in K \tag{1.6}$$

$$\sum_{i \in N} x_{i0k} \leq 1 \quad \forall k \in K \tag{1.7}$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1 \quad \forall S \subseteq N', |S| \geq 2, \forall k \in K \tag{1.8}$$

$$x_{ijk} \in \{0,1\} \quad \forall i,j \in N, \forall k \in K \tag{1.9}$$

$$y_{imk} \in \{0,1\} \quad \forall i \in N', \forall m \in M, \forall k \in K \tag{1.10}$$

The objective function represents the total cost of the routes to be minimized. Constraints (1.2) specify that each commodity ordered by a customer is brought by a single vehicle. Constraints (1.3) ensure that the compartment capacities are respected. Constraints (1.4) indicate that a vehicle $k \in K$ can only deliver

a commodity $m \in M_i$ to customer $i \in N'$ if the vehicle visits that customer. Constraints (1.5) ensure that each vehicle $k \in K$ leaves the depot at most once. Constraints (1.6) ensure the continuity of each route. Constraints (1.7) ensure that the depot is not visited more than once by each vehicle. Constraints (1.8) are the classical sub-tour elimination constraints and finally, constraints (1.9) and (1.10) are the domain constraints.

1.3.2 No-Split MCVRP

In the No-Split MCVRP, each customer must be serviced by a single vehicle and the vehicle must therefore have enough capacity to service all commodities to be delivered to that customer. The No-Split MCVRP model is proposed as follows:

$$\min \quad \sum_{i\in N}\sum_{j\in N}\sum_{k\in K} c_{ij}x_{ijk} \tag{1.11}$$

$$\text{st} \quad \sum_{j\in N}\sum_{k\in K} x_{ijk} = 1 \quad \forall\, i \in\ N' \tag{1.12}$$

$$\sum_{i\in N'} q_{im}\sum_{j\in N} x_{ijk} \leq Q_m \quad \forall m \in M, \forall k \in K \tag{1.13}$$

$$\text{Constraints } (1.5) - (1.9)$$

The objective function represents the total routing cost to be minimized. Constraints (1.12) ensure that each customer is visited exactly once. Constraints (1.13) define upper bounds on the delivery amounts determined by the capacities of the vehicle compartments.

1.3.3 Set Partitioning Model

Because we solve the two problems by Branch-and-Price, we reformulate them in terms of set partitioning models. We first consider the C-Split MCVRP. A set partitioning model can be obtained by applying Dantzig-Wolfe decomposition to (1.1) - (1.10) (Dantzig and Wolfe, 1960, 1961). Due to the structure of the problem, a natural choice is to keep constraints (1.2) in the master problem resulting from the decomposition as these are the only constraints which include all vehicles. This leaves constraints (1.3) - (1.10) for the so-called pricing problem of the decomposition.

Formally, for each vehicle $k \in K$, we consider the polytope $\mathscr{P}_k$ given by constraints (1.3) - (1.10) with fixed k. Because the vehicles are identical, these polytopes are identical and we can therefore consider a single polytope defined as $\mathscr{P} = \mathscr{P}_k, \forall k \in K$. With this notation, $\mathscr{P}$ defines the feasible region of the pricing problem of the decomposition and we define Ω to be the set of extreme points of $\mathscr{P}$. Such an extreme point corresponds to a route for a single vehicle, that starts and ends in the depot node, visits a number of customers where it delivers some of the requested commodities (possibly all) while respecting the compartment capacities. As Branch-and-Price is based on delayed column generation, the complete set of routes in Ω will not be considered. Rather, routes will be created dynamically in the pricing problem based on their ability to improve the solution.

Let c_r be the cost of such a route $r \in \Omega$ and let a_{imr} be a parameter taking the value 1 if route $r \in \Omega$ delivers commodity $m \in M$ to customer $i \in N'$, and zero otherwise. In order to construct the master problem for the C-Split MCVRP, we define for each route $r \in \Omega$ the decision variable λ_r as 1 if route r is used in the solution, and zero otherwise. The C-Split MCVRP can then be formulated as the following set partitioning model, where the convexity constraint (1.16) is relaxed from equality because we do not need to use all vehicles and (1.15) ensure that all requested commodities are serviced at all customers.

$$\min \quad \sum_{r\in\Omega} c_r\lambda_r \tag{1.14}$$

$$\text{st} \quad \sum_{r\in\Omega} a_{imr}\lambda_r = 1 \quad \forall i \in N', \forall m \in M_i \tag{1.15}$$

$$\sum_{r\in\Omega} \lambda_r \leq |K| \tag{1.16}$$

$$\lambda_r \in \{0,1\} \quad \forall r \in \Omega \tag{1.17}$$

For the LP-relaxation of this model, let $\pi_{im}, \forall i \in N', \forall m \in M_i$ be the dual variables corresponding to constraints (1.15) and let μ be the dual variable of constraint (1.16). Then, letting y_{im} and x_{ij} be simplifications of y_{imk} and x_{ijk}, respectively, the objective function of the pricing problem becomes

$$\min \sum_{i\in N}\sum_{j\in N} c_{ij}x_{ij} - \sum_{i\in N'}\sum_{m\in M_i} \pi_{im}y_{im} - \mu \tag{1.18}$$

For the No-Split MCVRP, we use a similar approach. Here, constraints (1.12) are the only constraints that include multiple vehicles. Hence, these constraints take the place of (1.2) in the master problem while (1.5) - (1.9) along with (1.13) are treated in the pricing problem. For the No-Split MCVRP, we remove the commodity-index because all commodities requested by a customer must be serviced by the same vehicle. The parameter a_{ir} therefore takes the value 1 if route $r \in \Omega$ services customer $i \in N'$, and zero otherwise, and the master problem is as follows:

$$\min \quad \sum_{r\in\Omega} c_r\lambda_r \tag{1.19}$$

$$\text{st} \quad \sum_{r\in\Omega} a_{ir}\lambda_r = 1 \quad \forall i \in N' \tag{1.20}$$

$$\sum_{r\in\Omega} \lambda_r \leq |K| \tag{1.21}$$

$$\lambda_r \in \{0,1\} \quad \forall r \in \Omega \tag{1.22}$$

Letting $\pi_i, \forall i \in N'$ be the dual variables corresponding to constraints (1.20) and μ be the dual variable of constraint (1.21) and using a similar simplified notation for x_{ij}, we get the following objective function for the No-Split MCVRP pricing problem:

$$\min \sum_{i\in N}\sum_{j\in N} c_{ij}x_{ij} - \sum_{i\in N'}\sum_{j\in N} \pi_i x_{ij} - \mu \tag{1.23}$$

1.4 Branch-and-Price Algorithm

This section describes our Branch-and-Price algorithm in detail. The overall process is outlined in Figure 1.1. The algorithm is initialized by a feasible solution obtained from a construction heuristic and improved by a simulated annealing. This is included in the *start* part of the algorithm and is described in Section 1.4.1.

The lower left part of the figure shows our node selection strategy which is explained in Section 1.4.2. To control the node selection, the algorithm uses two counters: *it* counts the total number of iterations of the full algorithm, and *cnt* counts the number of iterations since the last application of the *depth-first* search. In Section 1.4.2, we also explain our branching strategy. The remainder of the figure follows a relatively classical Branch-and-Bound structure. In the figure, *best solution* refers to the best feasible integer solution found so far.

The *pricing* part of the algorithm comes into play when a node in the branching tree is being solved. We explain our overall strategy for solving a node in Section 1.4.3, and Figure 1.2 provides an overview of the process, which includes an exact column generation described in Section 1.4.4 and a number of heuristic column generation algorithms that are detailed in Section 1.4.5.

1.4.1 Initialization

We initialize our Branch-and-Price algorithm by defining an LP model containing a small number of rows and columns. For the C-Split MCVRP, the LP is initialized by $n|M|$ constraints of type (1.15) to ensure that all commodities are serviced at all customers, in addition to the constraint (1.16) to limit the fleet of vehicles. For the No-Split MCVRP, the $n|M|$ constraints of type (1.15) are replaced by n constraints of type (1.20) to ensure that all customers are serviced.

To ensure that the LP model has a feasible solution, even when we delete some columns due to branching rules, we add a number of dummy columns to the problem. These columns are never deleted, but each dummy column has a large cost to ensure that it will not become part of the final solution. However, a feasible LP solution can always be obtained by setting the variables corresponding to the dummy columns to one. In the C-Split MCVRP, to ensure feasibility for every customer i-commodity m combination, we create a dummy column that represents a route which starts at the depot, travels along a shortest path to i, services commodity m, and returns to the depot. The number of dummy columns for the No-Split MCVRP, decreases to n and each column represents a route that starts at the depot, services customer $i \in N'$, and returns to the depot. The coefficient of constraint (1.16) and constraint (1.21), respectively, is 0 for the dummy columns to ensure feasibility throughout the algorithm.

In addition to these dummy columns, we use a heuristic for initialization of the master problem. The heuristic starts by applying a nearest neighbor strategy that works as follows. Each route starts at the depot and repeatedly extends from its current point to the nearest unvisited customer subject to the vehicle compartment capacities, where nearest is measured in terms of travel cost. For the No-Split version, we have

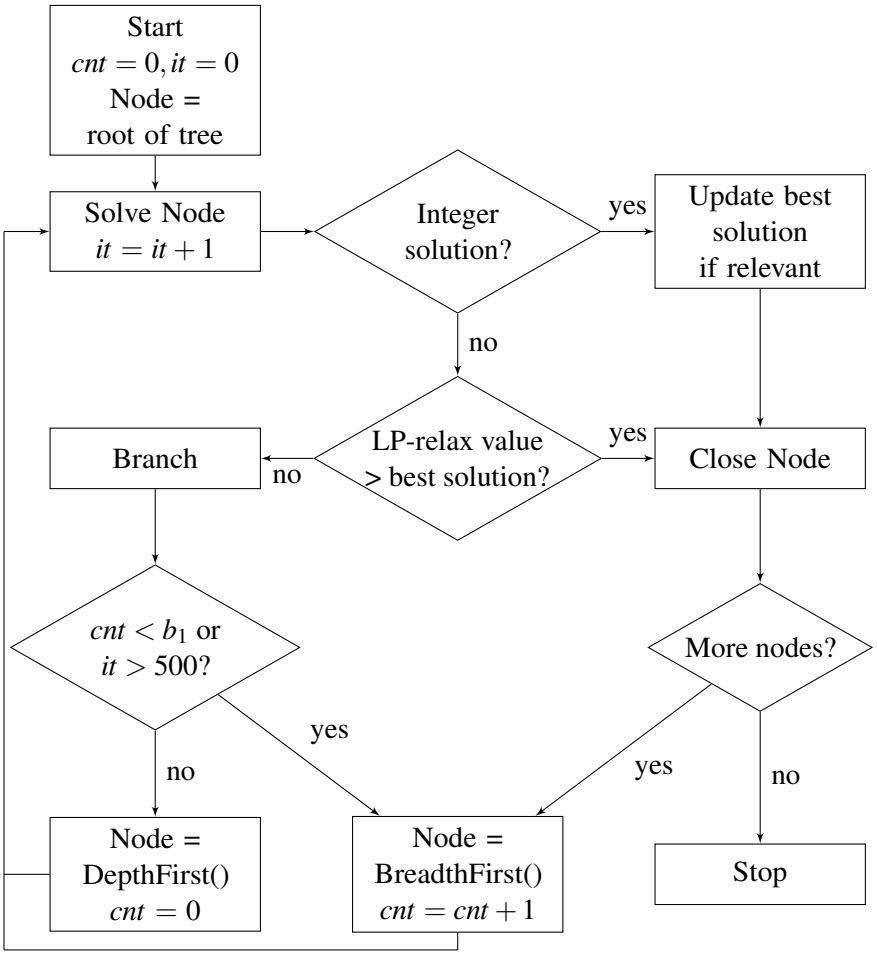

Figure 1.1: Flow of the overall process.

to deliver all commodities ordered by the customer and consequently we only consider customers whose total orders do not exceed the remaining vehicle capacity. For the C-Split we do not need to service all commodities, and the remaining capacity therefore only needs to be sufficient for the customer's demand for one of the commodities. However, when a customer is visited, the vehicle services the customer as regards all the commodities for which the capacity is sufficient. In case some commodities are not serviced at a customer, that customer will still be regarded as unserviced with respect to the non-delivered commodities. The route extension continues until there is no more capacity for serving customers or until all customers have been visited, at which point the vehicle returns to the depot. If there is at least one unvisited customer, a new route is initialized.

Once a feasible solution is obtained by this approach, it is improved by a Simulated Annealing using three neighbourhood structures: Reallocate, which moves a customer from one position to another; exchange, which swaps two customers; and 2-opt. All three neighborhoods are used in random versions, i.e. the customer nodes to be involved in the neighborhoods are selected randomly. In each iteration of the algorithm, one of the three neighborhoods is chosen randomly with equal probability and is used in that iteration. The Simulated Annealing is run for approximately 30 seconds, and the columns corresponding to

the routes of the final solution are used in the initialization of the master problem.

1.4.2 Branching and Node Selection

In our Branch-and-Price algorithm, we apply the branching strategy known as follow-on branching, originally based on Ryan-Foster branching. The idea is to select a pair of required customer nodes, i and j, and create two branches in the branching tree. In the left branch, the connection from i to j is fixed, and j must be serviced immediately after i. In the right branch, this connection is forbidden, and j may not be serviced immediately after i. See Ryan and Foster (1981) and Vanderbeck (2000) for more on follow-on branching.

However, if the number of vehicles used in the optimal solution of the LP-relaxation of the master problem is significantly fractional, we give priority to branching on the number of vehicles. Let v be the number of vehicles used in the solution. We branch on the number of vehicles if $\lfloor v \rfloor + 0.2 < v$ and $v < \lceil v \rceil - 0.2$, where the threshold of 0.2 is found experimentally to be a reasonable choice. In this case, we tighten constraint (1.16) to require that the number of vehicles is $\leq \lfloor v \rfloor$ in the left branch and $\geq \lceil v \rceil$ in the right branch.

Due to the limitations of the standard node selection algorithms, i.e. breadth-first or depth-first search, we have used a combination of the two that allows a single search algorithm to have the complementary strengths of both. The algorithm starts in a breadth-first search fashion where the node to be explored next is selected among unexplored nodes as the one whose parent's LP-relaxation has the lowest value. In case of a tie we select an arbitrary node. This is repeated for b_1 iterations. Next, a depth-first search explores nodes along a left branch until it either reaches a node that has an integer solution or a node that has an LP-relaxation value that is worse than the current best integer solution. At this point the depth-first search stops. Then the algorithm proceeds with breadth-first search and the process is repeated until there are no more nodes for branching or the global lower bound exceeds the value of the best feasible solution. However, after 500 iterations, the algorithm shifts to pure breath-first search. To optimize the performance of the program we tested different values and chose $b_1 = 18$. The node selection strategy is outlined in the lower left corner of Figure 1.1. In this algorithm, the left branch is chosen before the right branch.

1.4.3 Node Solving Procedure

As is usual in column generation approaches, we create a copy t of the depot node and search for routes starting in node 0, ending in node t, which respect all resource constraints and have negative reduced cost with respect to the current dual prices of the master problem. The column generation is initialized with a resource constraint for each commodity. During the process, additional resource constraints are added as explained below. We let S be the set of all such feasible routes with negative reduced cost and define $S_2 \subseteq S$ as the set of feasible elementary routes with negative reduced costs. Finally, we set $S_1 = S \setminus S_2$, i.e. the set of non-elementary routes. This means that S_2 includes only the routes that visit each customer at most once,

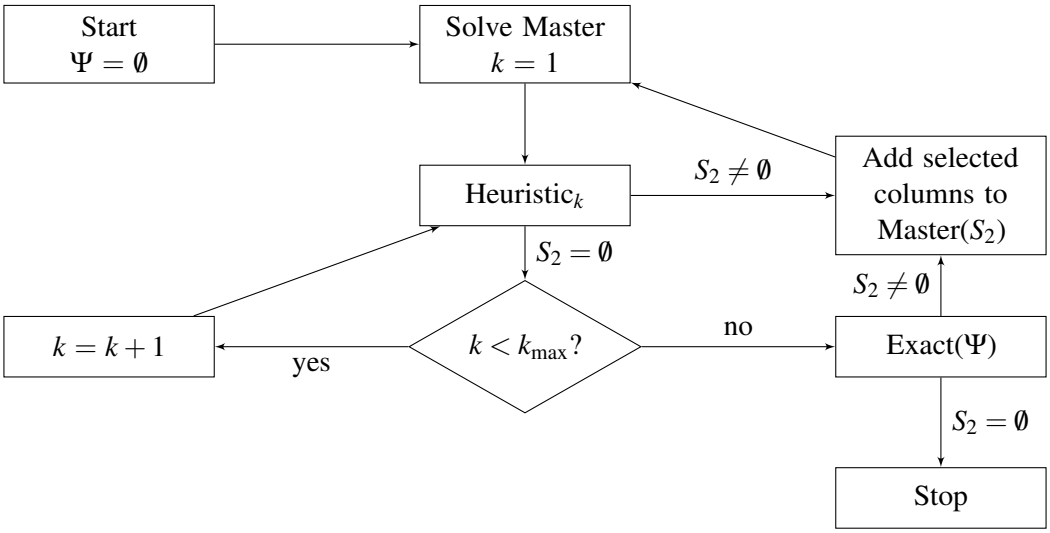

Figure 1.2: Flow of node solution process.

whereas routes in S_1 visit at least one customer more than once. Hence, routes in S_1 are not accepted as part of the final solution, but they may be created by the column generation algorithms nonetheless.

We define $\Psi \subseteq N'$ as a set of nodes which will not be allowed to appear on a route more than once in our exact algorithm. As will be explained in Section 1.4.4, the nodes $i \in \Psi$ will be treated as resources and therefore, we seek to keep $|\Psi|$ small.

Figure 1.2 shows an outline of our procedure for solving a node in the branching tree. Throughout the process, we only add elementary routes to the master problem. First, we iteratively apply heuristic column generation based on $k_{\max}$ heuristics. These are explained in Section 1.4.5. Next, we apply exact column generation where the nodes in Ψ are treated as resources and nodes are added to Ψ when necessary. The exact algorithm is described in Section 1.4.4. At any time, if columns are added to the master problem, the procedure starts with the first heuristic again.

Given a set S_2 of feasible elementary routes with negative reduced costs created by one of the heuristics or by the exact algorithm, we add columns to the master problem. If $|S_2| \leq 150$, we add all routes from S_2 to the master problem. Otherwise, we add the 150 routes from S_2 with the highest negative reduced cost to the master problem.

1.4.4 Exact Column Generation

We use a Label Setting Algorithm to solve the pricing problem. By means of this algorithm we find new columns that can improve the master problem or we prove that no such column exists. The algorithm identifies the full set of non-dominated columns which are partitioned into S_1 and S_2. If $S_2 \neq \emptyset$, we add up to 150 of these columns representing elementary routes to the master problem. The columns added are those with highest negative reduced cost. If the algorithm did not identify any elementary routes, i.e. $S_2 = \emptyset$, but found non-elementary routes with negative reduced cost, i.e. $S_1 \neq \emptyset$, the algorithm proceeds by adding

additional resource constraints to the problem. This process is explained below.

We refer the reader to Irnich and Desaulniers (2005) for a thorough description of labeling algorithms for solving shortest path problems with resource constraints and to Desaulniers et al. (2005) for a general introduction to column generation.

The algorithm uses a set $\mathscr{L}_i$ of labels for each node $i \in N \cup \{t\}$. Each label L_i^k in $\mathscr{L}_i$ represents a partial route R_i^k from node 0 to node i. Associated with L_i^k is a capacity resource vector σ_i^k, where σ_{im}^k represents the capacity consumption of commodity m along R_i^k and a 'node' resource vector W_i^k, where W_{ij}^k gives the number of visits to node j on R_i^k. Note that $W_{ii}^k \geq 1$ because the label is in node i, and $W_{ij}^k \leq 1 \ \forall j \in \Psi$. Finally, we use $C(R_i^k)$ to denote the reduced cost of the partial route R_i^k. These are determined by applying (1.18) and (1.23) to the partial route R_i^k for C-Split and No-Split, respectively.

A label L_i^a dominates a label L_i^b if any feasible extension into a complete route that can be made from L_i^b can also be made from L_i^a, and if routes resulting from extensions of L_i^b can not have smaller reduced costs than the routes obtained from the same extensions of L_i^a. In this case, the label L_i^b is superfluous and can be dominated. Formally, a label L_i^a dominates a label L_i^b if and only if $\sigma_i^a \leq \sigma_i^b$, $C(R_i^a) \leq C(R_i^b)$, and $W_{ij}^a \leq W_{ij}^b, \ \forall j \in \Psi$. To explain the latter, assume that $\Psi = \{1,2,4,6,8\}$ and label L_i^a services route $R_i^a = [1,2,3,8]$. Consider label L_i^b and assume that $\sigma_i^a \leq \sigma_i^b$, $C(R_i^a) \leq C(R_i^b)$. If the route of L_i^b is $R_i^b = [1,2,6,8]$, then L_i^a can dominate L_i^b because $R_i^a \cap \Psi = [1,2,8] \subseteq [1,2,6,8] = R_i^b$. Here, node 3 is not relevant because it is not in Ψ. But if the route of L_i^b is $R_i^b = [1,2,4,8]$, then L_i^a cannot dominate L_i^b because $W_{i6}^a = 1$ but $W_{i6}^b = 0$.

To handle the branching rules described in Section 1.4.2, we define an $n(n+1)$ *tabu matrix* T and a *fix* vector F of size $n+1$

We first consider the right-branch rule, forbidding the service of a required node j or a return to the depot to follow immediately after another required node i. This is recorded in T as follows:

$$T_{ij} = \begin{cases} 1 & \text{if } \ j \in \ N' \cup \{t\} \text{ may not be visited immediately after } \ i \in N \\ 0 & \text{otherwise.} \end{cases}$$

In the left branch, we set $F_i = j$, thereby forcing j to follow immediately after i. In the labeling algorithm, before extending L_i^k to j, T_{ij} and F_i are checked and appropriate action is taken.

Algorithm 1 shows the details of the proposed Label Setting Algorithm. The algorithm starts with an empty set of labels, except for a zero-label for the depot. It then repeatedly selects an unprocessed label L_i^k with smallest resource consumption with respect to commodity 1, σ_{i1}^k. For all $j \in N'$ a new label L_j' is created as an extension of L_i^k to j if it is resource feasible with respect to the capacity and 'node' resources, and allowed by the branching rules. Similarly, all labels are extended to node t, if feasible, representing a return to the depot. This continues until there are no unprocessed labels.

To strengthen the formulation, we only consider elementary routes in the master problem. Since the problem of finding elementary routes is strongly NP-hard, we use the idea of Boland et al. (2006) and employ the Label Setting Algorithm for solving the Resource Constrained Shortest Path Problem with node resources for only some nodes. We set $\Psi = \emptyset$ and solve the non-elementary problem using the Label

Algorithm 1 Label Setting Algorithm

Create L_0^1 with $R_0^1 = 0, \sigma_0^1 = 0, W_0^1 = 0$, and $C(R_0^1) = 0$.
Set $\mathscr{L}_0 = \{L_0^1\}$
Set $\mathscr{L}_i = \emptyset, \forall i \in N' \cup \{t\}$.
while $\bigcup_{i \in N} \mathscr{L}_i \neq \emptyset$ **do**
  Choose a label $L_i^k \in \bigcup_{i \in N} \mathscr{L}_i$ and set $\mathscr{L}_i = \mathscr{L}_i \setminus \{L_i^k\}$
  for all $j \in N' \cup \{t\}$ **do**
    if $T_{ij} \neq 1$ and ($F_i = j$ or $F_i = 0$) **then**
      Let $L_j^{'}$ denote the label obtained by extending L_i^k to j.
      if $L_j^{'}$ is feasible with respect to resources **then**
        if $L_j^{'}$ is not dominated by any label in U_j **then**
          Create $L_j^{'}$ and set $\mathscr{L}_j = \mathscr{L}_j \cup \{L_j^{'}\}$.
          Remove any dominated labels from $\mathscr{L}_j$
        end if
      end if
    end if
  end for
end while
Set $S_1 = \{L_t^k \in \mathscr{L}_t \mid \exists\, j \in N' \text{ with } W_{tj}^k > 1 \text{ and } C(R_t^k) < 0\ \}$
Set $S_2 = \{L_t^k \in \mathscr{L}_t \mid W_{tj}^k \leq 1\ \forall j \in N' \text{ and } C(R_t^k) < 0\}$

Algorithm 2 Exact Column Generation

$it = 1$,
while $it = 1$ or $S \neq \emptyset$ **do**
  Run the Label Setting Algorithm (Algorithm 1),
  if $S_2 \neq \emptyset$ **then**
    return S_2
  else if $S_1 \neq \emptyset$ **then**
    Set $L_t^{k^*} = \text{argmin}_{L_t^k \in S_1}\{C(R_t^k)\}$.
    Set $j^* = \text{argmax}_{j \in N'}\{W_{tj}^{k^*}\}$
    $\Psi = \Psi \cup \{j^*\}$.
  end if
  $it = it + 1$
end while
return $\emptyset$

Setting Algorithm. If no elementary route is created, we examine the non-elementary route with the highest negative reduced cost, R_t^k. In this route, we identify a customer j with $W_{tj}^k > 1$, i.e. customer j is visited more than once. Based on this, we set $\Psi = \Psi \cup \{j\}$. We repeat the procedure until there are no more non-elementary routes ($S_1 = \emptyset$), at which point the node in the branching tree has been solved to optimality. This exact algorithm is outlined in Algorithm 2.

1.4.5 Heuristic Column Generation

The main weakness of the exact algorithm is that the running time increases as the number of nodes treated as resources grows, because it becomes increasingly more difficult to dominate labels. This is the motivation for only running the exact column generation to ensure that there are no more elementary routes with negative reduced cost and hence prove that the node has been solved to optimality. For the No-Split problem, we use $k_{\max} = 4$ different heuristic techniques in order to speed up the Branch-and-Price algorithm, whereas we use $k_{\max} = 5$ for the C-Split version of the problem. Each of these heuristics produces elementary routes, but not necessarily the routes with the most negative reduced cost.

The first heuristic is a simple heuristic column generation based on the nearest neighbor heuristic which works as follows. We start a route at the depot and iteratively extend the route from its current end point to the nearest unvisited customer subject to the vehicle compartment capacity in the same way as we did in the construction heuristic described in Section 1.4.1. However, for this heuristic column generation, 'nearest' is measured in terms of travel cost minus the cost π_{im} for each delivery of commodity m to customer i (π_i for No-Split). The route is extended as long as the reduced cost is negative and the remaining capacity of the vehicle is large enough to service at least one of the unvisited customers.

The second heuristic is based on scaling of the capacity of the compartments. The exact algorithm is executed with modified vehicle capacities $Q'_m = g_1 Q_m,\ (\forall\, m \in M, 0 < g_1 \leq 1)$. In effect, this means that a vehicle can service fewer customers. Even though such shorter routes do not fully use the vehicle capacity, they can still be useful as supplements to longer routes. This heuristic is efficient for instances with high capacity of the vehicle compartments compared to the demand of customers because these are the instances where the labeling algorithm is generally time consuming. After testing different values, $g_1 = 0.5$ was selected.

The third heuristic is also a truncated version of our labeling algorithm and is based on extending each label from each node to its neighbors within a predefined distance. Each label in node i is only extended to the nodes j to which the distance c_{ij} is below a specified length D. The length of D is calculated by $max_{i,j \in N}\{c_{ij}\ g_2\}, 0 < g_2 \leq 1$. By considering only edges within this maximum length D, the algorithm will be limited to a restricted set of neighbors when processing each node. If the algorithm fails to identify negative reduced cost routes, g_2 is increased to αg_2 and the search is repeated while $g_2 \leq 1$. After computational testing, the initial values $g_2 = 0.6$ and $\alpha = 1.3$, were chosen.

The fourth heuristic is a heuristic Label Setting Algorithm with modified dominance rules. In this algorithm, a label L_i^a can dominate label L_i^b if it dominates with respect to the remaining capacity of compartments and the cost of the route, independent of 'node' resource vector. In other words, L_i^a can dominate L_i^b if and only if $\sigma_i^a \leq \sigma_i^b$, and $C_i^a \leq C_i^b$. By applying this heuristic, the valid routes are created with better speed.

The fifth heuristic is only used for the C-Split problem. In this heuristic, the labeling algorithm is executed for the C-Split problem but with No-Split settings. This means that we only extend a label L_i^k from i to customer j, if the remaining capacity is sufficient to service all commodities at customer j, i.e. if

$\sigma^k_{im} + q_{jm} \leq Q_m \; \forall m \in M_j$. This heuristic creates valid routes similar to routes in the No-Split setting and helps us to reduce the running time because the commodities are not treated separately.

1.5 Computational Experiments

We have tested our algorithm on a total of 274 instances. In this section, we present our computational results and analysis, which form the basis of our conclusions. The algorithm is implemented in C++ in Microsoft Visual Studio 2010 with the use of IBM ILOG CPLEX 12.6.1 callable library. All tests were made on a laptop with a Pentium core i5 processor and a clock speed of 1.8 GHz and 8 GB of RAM.

Details about our test sets are presented in Section 1.5.1. We have generated five sets of test instances which differ in type of demand and vehicle capacity. They are indexed 1 through 5. Furthermore, we have used three sets of instances from Muyldermans and Pang (2010). These are referred to as M4, M5, and M6, referring to the authors' own numbering.

Our results are structured as follows. In Section 1.5.2, we analyze the performance of our algorithm as regards its ability to solve problems using the No-Split strategy, and Section 1.5.3 is devoted to the performance as regards the C-Split strategy. From a computational perspective, the No-Split strategy is the easier of the two strategies for a Branch-and-Price algorithm due to the reduced flexibility for service. Therefore, we only present results for the C-Split strategy up to the point where the instances become too large for the algorithm to handle. For the C-Split, the size is perceived as a combination of nodes and commodities. This is further explained in Section 1.5.3. Finally, in Section 1.5.4, we compare the two strategies and analyze if allowing C-Split can lead to reductions in cost.

Table 1.1 provides a summary of the main results and an overview of Tables 1.2 through 1.15. For each of the tables, Table 1.1 provide the name of the test set along with the type of demand in the instances and the vehicle capacity for each commodity. The center part of the table states the number of instances in the set (NO.), the number of instances solved to optimality within a time limit of one hour (Opt.), and the average and maximal gaps after one hour of computation. Finally, the right-most column gives information about the strategy used in the tests presented in the tables.

Tables 1.2 through 1.15 give the details of our results. In these tables, we report the following: The instance name, which is composed of four numbers: The number of customers in the graph, the number of commodities, the vehicle capacity for each commodity, and finally an ID number to distinguish instances with similar characteristics. Under the *Root* header, we report the initial feasible solution obtained by running a simulated annealing algorithm for approximately 30 seconds (*SA*), the lower bound obtained as the LP-relaxation of the root of the branching tree (*LB*), and the percentage gap between this lower bound and the final best solution obtained (*Gap*). Under the *Best known* header, we report the value of the best feasible solution found (*UB*), the global lower bound (*LB*), and the percentage gap between the two (*Gap*). The gap is replaced by an asterisks ($*$) if optimality has been proven. Finally, we report the number of nodes in the branching tree (*Nodes*), the maximum number of customers on a route in the best solution (*Cust*),

the number of vehicles used in the best solution (*Veh*), and the total running time in seconds (*Time*). If the time limit of 1 hour is reached, we allow the algorithm to finish solving the current node in the branching tree and use TL to denote the time.

Table 1.1: Summary of computational results.

Table	Set	Demand	Capacity	No	Opt	Av gap	Max gap	Strategy
1.2	1	u(1,5)	10	44	29	0.4	3.1	No-Split
1.3	2	u(1,5)	15	24	2	2.9	16.6	No-Split
1.4	3	u(1,5)	20	24	1	7.2	22.7	No-Split
1.5	M5	u(1,5), u(6,10)	6-12,12-36	35	16	1.5	7.0	No-Split
1.6	M6	u(1,5), u(6,10)	6-12,12-36	35	15	0.8	6.3	No-Split
1.7+1.8	4	bin	3	73	48	0.7	4.8	No-Split
1.9	5	bin	4	24	1	7.0	20.3	No-Split
1.10	M4	bin	3	15	0	7.7	24.0	No-Split
1.11	1	u(1,5)	10	20	14	0.6	3.4	C-Split
1.12	M5	u(1,5), u(6,10)	6,12-18	15	1	10.1	29.0	C-Split
1.13	M6	u(1,5), u(6,10)	6,12-18	15	2	5.5	13.2	C-Split
1.14+1.15	4	bin	3	55	28	1.2	6.4	C-Split

1.5.1 Data Instances

We have generated a total of 189 new data instances which have been grouped into 5 sets. In addition, we have performed tests on 85 instances from Muyldermans and Pang (2010), grouped into 5 sets. We provide the details of the data in the following.

We consider the 5 new sets of data first. All customer locations are based on the data of Muyldermans and Pang (2010) and are scattered in a square of size 100×100 distance units where the depot is located in the middle of the square. Muyldermans and Pang (2010) use 100 customers in all instances, but we vary the number of customers in the new sets from 10 to 100. The number of commodities in each instance varies between 2 and 4.

For data sets 1, 2, and 3, the customer's demand for each commodity is drawn uniformly between 1 and 5. This means that each customer has a positive demand for every commodity, i.e. $M_i = M \forall i \in N'$. The vehicle capacity is 10, 15, and 20 for sets 1, 2, and 3, respectively. Therefore, with n fixed, instances in set 1 are expected to be more tractable than those in set 2, and instances in set 3 are, in general, the hardest. Set 1 contains instances down to 10 customers of ensure some relatively easy test instances in the pool, for comparison purposes. Sets 2 and 3 start with 40 customers.

In data sets 4 and 5, the customer's demand for each commodity is binary and follows the following patterns: If there are 2 commodities in the system, the probability that a customer orders both commodities is 50%. The remaining customers only order one commodity with each one being evenly represented.

If there are 3 commodities in the system, the probability of ordering 1, 2, and 3 commodities is 33%, respectively. Again, once the number has been determined, the demand is distributed evenly among the different commodity combinations. For 4 commodities, the probability of ordering 1, 2, 3, and 4 commodities, respectively, is 25%. The vehicle capacity is 3 for each commodity in set 4 and 4 in set 5. All new data are available at http://www.optimization.dk/MCVRP.

One of the purposes of this paper is to compare the two strategies, but due to the difficulties of solving the problems with the C-Split strategy, we need relatively small instances for this purpose. We have therefore made the instances with 10-25 customers in set 1, all demanding every commodity. Some customers in set 4 only request one commodity and the algorithm can handle up to 75 customers and 3 commodities reasonable well. For these combinations, we have created 5 instances of each size to provide sufficient instances for our comparison. For all other combinations, we have created 2 instances of each size to favour variation in the test sets.

In our experiments, we also use part of the data of Muyldermans and Pang (2010), which all have 100 customers. As that paper presents a heuristic, many of the instances were too large in terms of capacities for our exact algorithm to tackle. But the instances where the capacity is relatively large compared to the vehicle compartment capacities are included in our tests. The instances originate from three sets, which we refer to as M4, M5, and M6. The 15 instances of set M4 have binary demand, two commodities, and a capacity of three. A number is added to the name of these instances (0, 33, or 66) indicating the percentage of the customers requesting both commodities. The first 10 instances in sets M5 and M6 have demand drawn uniformly between 1 and 5 and compartment capacities of 6 and 12, respectively. These instances are marked with an 'A' in our tables. The remaining 25 instances in each of the two sets have demand drawn uniformly between 6 and 10 and compartment capacities between 12 and 36. These instances are marked with an 'B' in our tables. The depot is located centrally in sets M4 and M5, and remotely in set M6. All instances in sets M4, M5, and M6 have two commodities.

1.5.2 Results for the No-Split Strategy

In this section, we consider the results obtained with the No-Split strategy. We first consider Table 1.2, which reports results for the 44 instances of set 1 having demands between 1 and 5 and vehicle capacity 10. We see, that our algorithm is able to solve most of the instances with up to 50 nodes and that this is often done in less than 10 minutes. For instances with 75 and 100 nodes, even though we do not prove optimality, the obtained gaps are very small, with a maximum of 3.1%. In total, 29 out of these 44 instances are solved to optimality.

We now take a look across the three tables 1.2, 1.3, and 1.4, which report results obtained with the No-Split strategy for instances of sets 1-3, having demand between 1 and 5 and vehicle capacity 10, 15, and 20, respectively. As expected, our ability to solve the problems to optimality decreases when the vehicle capacity increases and with a capacity of 20, we only prove optimality in one instance. Note, however, that the smallest instances are not present in Tables 1.3, and 1.4. Similarly, both the average and the maximum

gap increase with the vehicle capacity. The explanation for this is found in the labeling algorithm which creates significantly more labels when capacity is high. This can be seen by noting that the maximum number of customers in the best feasible solution increases from 5 to 8 when the vehicle capacity increases from 10 to 20. As a result, the solution time for each node in the branching tree increases and the number of nodes that can be investigated within the time limit decreases.

We see a similar pattern when we turn our attention to Tables 1.5 and 1.6 that present the results for the M5 and M6 sets of Muyldermans and Pang (2010) with similar type of demand. The instances with relatively few customers on each route are solved to optimality within a limited time, whereas the time limit is reached when compartment capacities increase. We observe no significant difference in computation times across the two sets. It is worth noting, that for instances 11 through 15 in each of these tables, the combination of demand and compartment capacities is such that the optimal solutions almost uses separate vehicles for each customer.

Tables 1.7, 1.8, and 1.9 show the results obtained with the No-Split strategy for instances of sets 4 and 5, having binary demand and vehicle capacities 3 and 4, respectively. The results for the case of a vehicle capacity of 3 are very similar to those seen in Table 1.2 as we are able to solve instances with up to 50 nodes to optimality within a short time. However, not all instances with 40 and 50 nodes could be solved within the time limit. 48 out of the 73 instances in set 4 were solved to optimality. Note that due to the nature of the binary data, there are up to 8 customers on a route even though the vehicle capacity is only 3 because all commodities are not ordered by all customers. When we look at instances with vehicle capacity 4 as shown in Table 1.9, we see that only one instance is solved to optimality and the average gap increases from 0.7% to 7.0% with a maximum gap of about 20% for the hardest instances with 100 customers and 4 commodities. Table 1.10 presents the results of set M4 with the same type of demand. We note that these instances seem harder than the instances with 100 customers in set 4. Our algorithm is particularly challenged by the first five instances in M4 where all customers demand only one commodity. This is due to the large number of labels in the exact column generation.

1.5.3 Results for the C-Split Strategy

The results for the C-Split strategy are shown in Tables 1.11 through 1.15 and it is evident that solving this problem is harder. Because the C-Split is harder to solve, we only present results for instances up to the size that could be handled by the algorithm. For set 1 (Table 1.11), with U(1,5) demand with capacity 10, even some instances with 25 customers could not be solved within the time limit whereas the same instances were solved in less than two minutes with the No-Split strategy. Tables 1.12 and 1.13, showing results for M5 and M6, indicate a similar pattern. Out of the total of 30 instances in these tables, only three could be solved to optimality within an hour, whereas with the No-Split strategy, all 30 instances could be solved to optimality. Tables 1.14-1.15 show the C-Split strategy for the binary data with vehicle capacity of 3 and when we compare these tables to Tables 1.7-1.8, we see the same general picture of the C-Split being harder for our algorithm to solve than the No-Split as we have seen for the other sets. For this set, we can

solve about half on the instances to optimality. For both sets 1 and 4, the C-Split cannot consistently solve problems with more than 20 nodes within an hour.

The explanation for this is to be found in the labeling algorithm for the two strategies. When allowing C-Split, the labeling algorithm will treat every commodity for every customer as a separate customer because any combination of including or excluding a commodity in the route is possible. Therefore, for the C-Split, the labeling algorithm will be burdened by this high number of artificial customers. Note that for the C-Split strategy, the *Cust* column reports the number of these artificial customers rather than the number of physical customers.

1.5.4 Comparison of the No-Split and the C-Split Strategies

In this section, we compare the costs resulting from each of the two strategies in order to determine whether companies can obtain savings in transportation cost by applying a C-Split strategy rather than a No-Split strategy. We therefore consider the 105 instances that were solvable by both algorithms and summarize our findings in Table 16. The table shows the results for the full set of 105 instances and for the four sets individually.

Out of the 105 instances, 45 were solved to optimality for both strategies and based on the results obtained for those 45 instances we can conclude that 1) the two strategies led to the same objective value for 28 of the instances (27%), thus allowing C-Split for these instances did not result in cost savings; and 2) for the remaining 17 instances (16%) cost savings were obtained by allowing C-Split at an average rate of 0.7%.

As regards the remaining 60 instances, we note that if an instance is solved to optimality for the No-Split problem but not for C-Split, and if the best solution obtained for C-Split is lower than the optimal solution for No-Split, then there will be a certain cost saving by applying C-Split. There are 28 such instances and the average cost saving for these instances is 1.8%. We point out that the cost saving is under-estimated because the optimal C-Split value may be lower than the best known. We cannot make any conclusions as to possible benefits from C-Split based on the obtained values for the remaining 32 instances.

In summary: for 27% of the instances, the solution for the two strategies are the same, for 43% of the instances there is a some cost saving, and for the remaining 30%, no conclusion can be drawn from our analysis.

Among instances where we obtain a certain saving, the instances in set M6 generally result in the highest savings. The explanation for this is likely to be found in the remote location of the depot which means that a significant travel distance can be saved if C-Split results in a decease in the number of vehicles used. However, it should be noted that for the M6 instances the demand is such that most customers are serviced by a separate vehicle in the No-Split scenario and hence, these instances are not representative.

We have performed additional tests on 11 instances of set 1 with modified demand using an objective of minimizing the number of vehicles used. These tests, which are not shown here, indicate that it is generally

not possible to reduce the number of vehicles by allowing C-Split. However, additional analysis is needed in order to draw more precise conclusions.

1.6 Concluding Remarks

In this paper, we have presented an exact algorithm based on Branch-and-Price for solving two versions of the Multi-Commodity Vehicle Routing Problem, one that allows different commodities to a customer to be delivered by different vehicles (C-Split) and one that does not allow such splitting (No-Split). The performance of our algorithm is analyzed based on a total of 274 instances. We are able to solve instances with up to 50 nodes and 4 commodities for No-Split. 112 of the instances have been solved to optimality for the No-Split strategy. Our performance limit is slightly lower as regards the more complex C-Split problem; we analyzed 105 instances and solved 45 of them to optimality.

Our comparison of the two strategies is based on 105 instances. We found that routing cost benefits in 45 of these instances by using commodity splitting, but in 28 instances, splitting did not lead to cost savings. As for the remaining 32 instances we were not able to draw a conclusion.

For instances that did get a cost saving from the rather more complicated C-Split strategy as opposed to the No-Split strategy, the saving rate averaged 1.4%. It would be interesting to see whether the advantage of using C-Split persists for large instances.

Table 1.2: No-Split strategy for U(1,5) demand and a capacity of 10.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
10-2-10-1	503.3	495.8	1.5	503.3	503.3	*	3	3	4	1
10-2-10-2	432.2	410.2	*	410.2	410.2	*	0	4	3	11
10-2-10-3	461.2	461.2	*	461.2	461.2	*	0	3	4	1
10-2-10-4	451.7	436.8	3.4	451.7	451.7	*	7	4	4	2
10-2-10-5	420.1	396.1	*	396.1	396.1	*	0	4	3	1
15-2-10-1	641.7	626.6	0.6	630.2	630.2	*	3	4	5	13
15-2-10-2	594.4	573.8	1.1	588.6	588.6	*	23	4	5	15
15-2-10-3	553.1	479.6	5.5	501.3	501.3	*	84	3	5	17
15-2-10-4	656.9	650.5	1.0	656.9	656.9	*	7	3	7	2
15-2-10-5	572.3	525.3	*	525.3	525.3	*	0	3	6	2
20-2-10-1	878.3	856.2	0.2	857.6	857.6	*	3	3	8	2
20-2-10-2	636.2	623.4	0.8	628.5	628.5	*	13	4	6	4
20-2-10-3	666.1	652.5	2.1	666.1	666.1	*	73	4	7	14
20-2-10-4	900.8	883.1	1.8	899.4	899.4	*	9	3	9	3
20-2-10-5	684.1	674.7	1.4	684.1	684.1	*	5	4	8	2
25-2-10-1	1022.8	992.6	3.0	1008.5	1008.5	*	859	4	10	113
25-2-10-2	768.8	743.4	2.0	753.8	753.8	*	101	4	7	25
25-2-10-3	882.2	848.2	2.0	855.4	855.4	*	68	5	10	17
25-2-10-4	1073.0	1045.3	0.9	1055.2	1055.2	*	23	3	11	5
25-2-10-5	852.3	839.5	1.5	852.3	852.3	*	17	4	9	6
40-2-10-1	1510.7	1416.3	1.8	1441.9	1441.9	*	1111	4	13	1526
40-2-10-2	1352.4	1233.6	2.0	1258.3	1258.3	*	1966	4	13	1616
40-3-10-1	1559.4	1489.5	0.6	1497.9	1497.9	*	183	4	15	579
40-3-10-2	1315.0	1234.2	1.4	1251.4	1251.4	*	783	4	15	1571
40-4-10-1	3383.1	2913.6	0.1	2917.4	2917.4	*	55	4	12	307
40-4-10-2	3328.7	3063.0	0.2	3069.2	3069.2	*	75	4	13	36
50-2-10-1	1898.1	1771.4	1.4	1795.5	1783.6	0.7	4219	5	16	TL
50-2-10-2	1571.8	1438.5	2.2	1470.7	1451.8	1.3	940	4	16	TL
50-3-10-1	1842.9	1701.5	3.1	1755.1	1716.4	2.3	7590	4	17	TL
50-3-10-2	1700.1	1576.9	0.5	1584.9	1584.9	*	525	4	18	490
50-4-10-1	4196.0	3607.2	2.2	3686.0	3686.0	*	1571	4	16	1213
50-4-10-2	4172.3	3663.4	0.5	3683.2	3683.2	*	888	4	16	976
75-2-10-1	2639.7	2447.3	2.6	2511.3	2455.3	2.3	1935	4	25	TL
75-2-10-2	2528.6	2269.1	2.1	2317.0	2276.2	1.8	629	5	25	TL
75-3-10-1	2830.6	2489.2	1.6	2528.5	2500.5	1.1	2976	4	26	TL
75-3-10-2	2729.9	2409.2	0.7	2426.2	2418.4	0.3	2660	4	26	TL
75-4-10-1	5915.2	5356.2	0.6	5389.0	5362.2	0.5	754	4	23	TL
75-4-10-2	6144.7	5252.5	0.1	5259.2	5259.1	0.0	1338	4	23	TL
100-2-10-1	3413.1	2996.8	1.6	3043.8	3001.6	1.4	620	4	31	TL
100-2-10-2	3230.1	2812.8	2.2	2874.4	2821.1	1.9	718	5	32	TL
100-3-10-1	3588.1	3187.5	1.0	3219.5	3190.7	0.9	475	4	34	TL
100-3-10-2	3531.8	3101.9	3.1	3199.5	3102.2	3.1	273	4	35	TL
100-4-10-1	8040.5	7122.0	0.4	7150.8	7126.3	0.3	1221	4	31	TL
100-4-10-2	8206.2	7267.0	0.4	7295.3	7277.6	0.2	276	4	32	TL

Table 1.3: No-Split strategy for U(1,5) demand and a capacity of 15.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-2-15-1	1064.8	945.3	3.9	982.6	955.2	2.9	719	7	8	TL
40-2-15-2	1073.0	1004.6	4.1	1045.4	1020.0	2.5	1126	6	9	TL
40-3-15-1	1208.2	1008.0	4.8	1056.5	1022.5	3.3	860	6	10	TL
40-3-15-2	1111.7	999.7	5.0	1050.2	1014.6	3.5	871	6	9	TL
40-4-15-1	2511.9	2260.0	2.5	2317.2	2317.2	$*$	677	5	10	697
40-4-15-2	2304.1	2077.8	1.4	2107.9	2107.9	$*$	173	5	9	193
50-2-15-1	1268.9	1167.0	2.0	1189.9	1171.4	1.6	329	7	11	TL
50-2-15-2	1276.7	1176.9	3.3	1215.9	1180.6	3.0	245	6	11	TL
50-3-15-1	1370.6	1220.2	3.1	1258.2	1226.0	2.6	230	5	12	TL
50-3-15-2	1464.4	1324.3	1.1	1338.3	1329.5	0.7	289	5	13	TL
50-4-15-1	3082.4	2589.7	1.2	2620.7	2598.2	0.9	244	5	11	TL
50-4-15-2	2714.7	2484.3	2.1	2536.1	2513.6	0.9	114	6	11	TL
75-2-15-1	1902.7	1647.4	4.5	1722.3	1653.1	4.2	766	6	16	TL
75-2-15-2	1913.4	1704.7	5.2	1794.0	1705.3	5.2	138	6	17	TL
75-3-15-1	1896.5	1703.0	4.1	1772.0	1704.4	4.0	330	6	17	TL
75-3-15-2	1997.9	1761.1	2.6	1807.7	1763.4	2.5	201	6	18	TL
75-4-15-1	4067.5	3640.6	0.7	3665.5	3643.8	0.6	330	5	16	TL
75-4-15-2	4197.1	3748.3	0.4	3761.7	3753.2	0.2	414	5	16	TL
100-2-15-1	2604.7	2134.3	6.3	2267.7	2136.1	6.2	306	7	23	TL
100-2-15-2	2404.4	2061.6	16.6	2404.4	2061.6	16.6	55	7	22	TL
100-3-15-1	2530.4	2167.2	4.8	2272.1	2172.2	4.6	382	5	23	TL
100-3-15-2	2728.9	2214.3	1.6	2250.3	2214.5	1.6	218	5	23	TL
100-4-15-1	5447.2	4929.6	1.1	4985.6	4950.6	0.7	102	6	22	TL
100-4-15-2	5405.0	4928.7	1.2	4989.5	4949.9	0.8	69	6	22	TL

Table 1.4: No-Split strategy for U(1,5) demand and a capacity of 20.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-2-20-1	907.8	775.6	6.8	828.7	783.2	5.8	354	7	7	TL
40-2-20-2	997.4	781.9	0.4	785.5	785.5	$*$	81	6	7	3030
40-3-20-1	922.8	782.7	8.7	850.8	785.8	8.3	488	7	7	TL
40-3-20-2	977.9	838.7	2.3	858.2	845.7	1.5	461	7	7	TL
40-4-20-1	1748.7	1635.1	6.6	1743.0	1700.1	2.5	739	6	7	TL
40-4-20-2	1857.0	1538.1	8.7	1671.6	1641.3	1.8	799	7	7	TL
50-2-20-1	977.5	896.4	9.0	977.5	897.8	8.9	67	8	8	TL
50-2-20-2	995.7	942.5	4.0	980.1	947.3	3.5	331	7	9	TL
50-3-20-1	1170.4	1009.1	2.0	1029.1	1012.1	1.7	230	7	9	TL
50-3-20-2	1055.0	954.1	3.9	991.7	957.4	3.6	197	6	10	TL
50-4-20-1	2186.9	1955.1	1.2	1978.4	1956.9	1.1	201	7	8	TL
50-4-20-2	2099.9	1879.1	0.8	1893.5	1880.7	0.7	126	7	8	TL
75-2-20-1	1616.7	1349.2	6.7	1440.3	1349.7	6.7	122	8	12	TL
75-2-20-2	1551.2	1331.5	16.5	1551.2	1331.5	16.5	51	7	14	TL
75-3-20-1	1619.9	1446.8	9.7	1586.9	1450.7	9.4	264	7	13	TL
75-3-20-2	1507.0	1324.9	13.7	1507.0	1326.0	13.6	64	7	13	TL
75-4-20-1	3323.9	3141.1	0.3	3150.6	3145.6	0.2	240	7	13	TL
75-4-20-2	3204.2	2751.5	1.2	2784.9	2755.0	1.1	111	7	12	TL
100-2-20-1	2116.0	1725.2	22.7	2116.0	1725.2	22.7	66	8	16	TL
100-2-20-2	1809.5	1590.9	3.9	1652.7	1590.9	3.9	64	8	16	TL
100-3-20-1	2036.0	1768.8	15.1	2036.0	1768.8	15.1	13	8	17	TL
100-3-20-2	1912.0	1629.0	17.4	1912.0	1629.0	17.4	85	8	17	TL
100-4-20-1	4288.7	3757.9	14.1	4288.7	3757.9	14.1	9	7	18	TL
100-4-20-2	4110.2	3637.5	13.0	4110.2	3637.5	13.0	31	7	18	TL

Table 1.5: No-Split strategy for instances of Figure 5 of Muyldermans and Pang (2010).

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-6-1-A	5062.3	4891.5	0.1	4897.0	4897.0	*	7	3	59	24
100-2-6-2-A	5021.8	4878.9	0.2	4888.8	4888.8	*	25	2	59	28
100-2-6-3-A	4963.6	4868.8	0.2	4878.6	4878.6	*	99	3	61	57
100-2-6-4-A	5130.1	4927.1	*	4927.1	4927.1	*	0	3	59	27
100-2-6-5-A	5050.4	4766.6	0.3	4778.9	4778.9	*	85	3	55	94
100-2-12-1-A	2785.4	2516.4	4.2	2623.3	2518.5	4.2	263	5	26	TL
100-2-12-2-A	2890.5	2489.1	4.2	2592.6	2490.7	4.1	174	5	28	TL
100-2-12-3-A	2835.3	2470.4	4.1	2570.7	2472.0	4.0	445	6	27	TL
100-2-12-4-A	2900.8	2549.5	4.7	2668.9	2550.0	4.7	148	5	28	TL
100-2-12-5-A	2815.8	2495.9	5.0	2620.3	2496.4	5.0	1170	5	27	TL
100-2-12-1-B	7673.5	7673.5	*	7673.5	7673.5	*	0	2	98	2
100-2-12-2-B	7194.6	7194.6	*	7194.6	7194.6	*	0	2	97	2
100-2-12-3-B	7172.0	7172.0	*	7172.0	7172.0	*	0	2	97	2
100-2-12-4-B	7665.4	7657.8	*	7657.8	7657.8	*	0	2	98	2
100-2-12-5-B	7531.5	7521.2	0.1	7531.5	7531.5	*	7	2	99	2
100-2-18-1-B	4458.0	4309.0	0.6	4334.5	4334.5	*	167	3	50	110
100-2-18-2-B	4145.3	4001.9	0.3	4015.5	4015.5	*	95	2	50	58
100-2-18-3-B	4295.9	4085.5	0.2	4091.7	4091.7	*	67	2	50	100
100-2-18-4-B	4575.6	4252.7	0.1	4256.1	4256.1	*	27	2	50	95
100-2-18-5-B	4349.3	4198.4	0.2	4207.1	4207.1	*	321	2	50	403
100-2-24-1-B	3858.8	3486.2	0.6	3506.2	3496.6	0.3	2272	3	38	TL
100-2-24-2-B	3547.8	3177.0	1.8	3235.7	3179.9	1.8	1218	3	36	TL
100-2-24-3-B	3684.2	3223.6	0.1	3226.7	3226.7	*	119	3	38	565
100-2-24-4-B	3654.9	3277.1	1.6	3330.6	3283.0	1.4	454	3	36	TL
100-2-24-5-B	3779.9	3460.0	1.3	3505.6	3462.8	1.2	440	3	39	TL
100-2-30-1-B	3153.4	2844.6	1.8	2894.4	2844.6	1.8	75	4	30	TL
100-2-30-2-B	2915.5	2622.4	1.9	2673.5	2624.6	1.9	507	4	30	TL
100-2-30-3-B	2998.9	2667.0	3.6	2763.7	2667.1	3.6	67	4	30	TL
100-2-30-4-B	2998.3	2761.9	2.6	2833.2	2763.6	2.5	459	4	30	TL
100-2-30-5-B	3058.8	2818.6	1.6	2863.5	2820.5	1.5	975	4	31	TL
100-2-36-1-B	2616.0	2377.3	2.5	2437.8	2381.4	2.4	98	5	24	TL
100-2-36-2-B	2371.6	2215.8	7.0	2370.2	2215.8	7.0	24	5	26	TL
100-2-36-3-B	2389.3	2250.5	1.0	2273.3	2251.5	1.0	234	5	25	TL
100-2-36-4-B	2546.1	2353.1	1.7	2392.5	2354.5	1.6	254	5	24	TL
100-2-36-5-B	2592.2	2328.9	4.1	2423.6	2328.9	4.1	114	5	25	TL

Table 1.6: No-Split strategy for instances of Figure 6 of Muyldermans and Pang (2010).

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-6-1-A	12297.2	11260.4	0.1	11273.0	11273.0	*	13	3	50	30
100-2-6-2-A	12963.1	12528.9	0.4	12580.1	12580.1	*	3	2	58	28
100-2-6-3-A	12351.3	11708.2	0.1	11718.6	11718.6	*	3	3	52	40
100-2-6-4-A	12693.9	11695.0	0.6	11760.6	11760.6	*	9	3	54	60
100-2-6-5-A	13701.0	13254.7	0.5	13318.8	13318.8	*	3	2	62	40
100-2-12-1-A	6119.2	5515.8	3.5	5709.3	5615.8	1.7	249	5	25	TL
100-2-12-2-A	6353.2	5880.3	2.2	6011.3	5948.1	1.1	209	5	27	TL
100-2-12-3-A	6314.1	5623.5	2.1	5740.8	5634.7	1.9	90	6	25	TL
100-2-12-4-A	6063.2	5508.8	1.7	5604.8	5563.0	0.8	196	5	25	TL
100-2-12-5-A	6654.2	6155.0	1.9	6270.4	6174.1	1.6	85	5	28	TL
100-2-12-1-B	21014.1	21014.1	*	21014.1	21014.1	*	0	2	98	2
100-2-12-2-B	20784.2	20784.2	*	20784.2	20784.2	*	0	2	97	2
100-2-12-3-B	20840.2	20840.2	*	20840.2	20840.2	*	0	2	97	2
100-2-12-4-B	21151.8	21151.8	*	21151.8	21151.8	*	0	1	100	1
100-2-12-5-B	20647.9	20647.9	*	20647.9	20647.9	*	0	2	98	1
100-2-18-1-B	11146.9	10879.1	0.7	10951.4	10951.4	*	865	2	50	259
100-2-18-2-B	10887.7	10708.8	0.1	10716.1	10716.1	*	127	3	49	113
100-2-18-3-B	10947.6	10779.0	0.1	10784.9	10784.9	*	347	3	49	344
100-2-18-4-B	11206.4	10814.8	*	10818.7	10818.7	*	97	2	50	169
100-2-18-5-B	11010.2	10638.2	0.8	10723.3	10723.3	*	21	3	50	140
100-2-24-1-B	8837.6	8092.7	0.1	8099.7	8097.7	0.0	1393	3	36	TL
100-2-24-2-B	8258.2	7593.3	0.4	7626.4	7625.0	0.0	921	3	34	TL
100-2-24-3-B	8657.6	7696.1	0.2	7714.4	7698.6	0.2	390	3	34	TL
100-2-24-4-B	9131.2	8154.9	0.2	8174.0	8172.1	0.0	400	3	37	TL
100-2-24-5-B	8790.0	7717.1	0.8	7776.1	7743.7	0.4	275	3	35	TL
100-2-30-1-B	7165.1	6484.8	1.0	6550.5	6516.8	0.5	198	4	29	TL
100-2-30-2-B	6901.5	6171.5	1.5	6264.1	6234.5	0.5	102	4	28	TL
100-2-30-3-B	6936.3	6274.0	1.0	6337.4	6295.7	0.7	111	4	28	TL
100-2-30-4-B	7024.8	6513.2	1.8	6632.6	6600.4	0.5	284	4	30	TL
100-2-30-5-B	7022.2	6289.5	1.4	6376.8	6289.5	1.4	49	4	28	TL
100-2-36-1-B	5626.2	5291.1	6.3	5626.2	5291.1	6.3	17	5	25	TL
100-2-36-2-B	5627.5	5093.9	1.7	5180.0	5151.1	0.6	104	5	23	TL
100-2-36-3-B	5641.2	5164.3	1.9	5263.2	5197.3	1.3	10	5	23	TL
100-2-36-4-B	5585.9	5302.3	5.3	5585.9	5302.3	5.3	80	5	25	TL
100-2-36-5-B	5547.6	5175.4	3.9	5376.6	5176.1	3.9	296	5	24	TL

Table 1.7: No-Split strategy for binary demand and a capacity of 3.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
10-2-3-1	411.5	398.9	3.0	410.9	410.9	*	7	4	3	2
10-2-3-2	431.6	424.2	1.7	431.6	431.6	*	3	3	4	1
10-2-3-3	357.0	350.2	1.9	357.0	357.0	*	3	4	3	1
10-2-3-4	421.4	393.2	7.2	421.4	421.4	*	29	4	3	6
10-2-3-5	386.9	386.9	*	386.9	386.9	*	0	4	3	1
10-3-3-1	389.8	375.8	3.7	389.8	389.8	*	5	5	2	2
10-3-3-2	398.3	372.3	1.0	376.0	376.0	*	3	5	3	1
10-3-3-3	421.4	393.2	7.2	421.4	421.4	*	19	4	3	3
10-3-3-4	401.2	393.7	1.2	398.4	398.4	*	15	4	3	3
10-3-3-5	396.5	396.5	*	396.5	396.5	*	0	5	3	1
15-2-3-1	531.8	507.3	3.3	523.8	523.8	*	137	5	4	24
15-2-3-2	550.0	508.5	2.1	518.9	518.9	*	3	4	5	2
15-2-3-3	524.6	507.0	3.4	524.1	524.1	*	37	5	4	8
15-2-3-4	453.2	447.7	1.2	453.2	453.2	*	3	5	4	2
15-2-3-5	554.3	528.4	4.3	550.9	550.9	*	43	4	4	11
15-3-3-1	466.3	460.8	1.2	466.3	466.3	*	3	5	4	2
15-3-3-2	498.1	482.9	1.0	487.5	487.5	*	13	5	4	4
15-3-3-3	518.5	482.8	5.1	507.4	507.4	*	619	5	4	123
15-3-3-4	451.0	451.0	*	451.0	451.0	*	0	5	4	1
15-3-3-5	404.3	380.4	3.5	393.6	393.6	*	85	7	4	19
20-2-3-1	665.2	650.8	0.2	651.9	651.9	*	9	5	5	5
20-2-3-2	613.9	588.6	1.7	598.6	598.6	*	3	4	6	2
20-2-3-3	709.9	704.5	0.8	709.9	709.9	*	17	4	6	5
20-2-3-4	604.7	584.4	2.0	595.6	595.6	*	57	5	6	13
20-2-3-5	568.3	557.3	1.5	565.6	565.6	*	21	5	6	6
20-3-3-1	640.8	599.6	2.1	612.2	612.2	*	275	5	5	65
20-3-3-2	564.1	551.5	2.3	564.1	564.1	*	17	5	5	5
20-3-3-3	691.6	617.6	2.7	634.1	634.1	*	17	4	6	7
20-3-3-4	573.0	562.0	1.9	572.4	572.4	*	29	5	5	23
20-3-3-5	602.1	587.5	1.5	596.1	596.1	*	47	6	5	24
30-2-3-1	867.0	845.6	*	845.6	845.6	*	0	5	9	7
30-2-3-2	892.9	837.4	2.1	853.8	853.8	*	122	4	9	46
30-2-3-3	1027.4	1006.3	1.5	1021.4	1021.4	*	3024	4	9	1615
30-2-3-4	857.5	837.4	2.0	853.8	853.8	*	321	4	9	111
30-2-3-5	826.0	794.1	1.3	804.6	804.6	*	135	5	8	45
30-3-3-1	945.6	882.5	2.9	907.8	907.8	*	4353	5	8	2702
30-3-3-2	859.0	792.2	2.0	808.4	808.4	*	397	5	8	398
30-3-3-3	992.8	947.0	1.6	962.4	962.4	*	91	5	9	66
30-3-3-4	810.2	797.5	1.0	805.1	805.1	*	167	6	8	73
30-3-3-5	918.2	870.5	2.3	890.3	890.3	*	1378	5	8	2905
40-2-3-1	1197.1	1137.7	*	1138.0	1138.0	*	3	5	10	8
40-2-3-2	1278.2	1074.0	0.6	1079.9	1079.9	*	743	6	11	392
40-2-3-3	1212.0	1144.7	1.8	1165.0	1164.6	0.0	1675	5	11	TL
40-2-3-4	1221.3	1120.0	1.2	1134.0	1134.0	*	55	4	12	220
40-2-3-5	1128.5	1062.2	1.0	1073.0	1073.0	*	306	5	11	326

Table 1.8: No-Split strategy for binary demand and a capacity of 3.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-3-3-1	1075.9	1044.0	0.9	1053.3	1053.3	*	31	7	10	217
40-3-3-2	1069.5	973.1	2.2	994.5	987.8	0.7	2527	7	10	TL
40-4-3-1	1110.2	1074.0	1.4	1088.9	1087.2	0.2	799	6	9	TL
40-4-3-2	1144.3	1069.8	0.8	1078.0	1078.0	*	125	5	11	204
50-2-3-1	1535.8	1422.7	1.4	1442.7	1436.0	0.5	1528	5	14	TL
50-2-3-2	1351.7	1237.7	0.2	1240.6	1240.6	*	21	5	14	70
50-2-3-3	1434.7	1353.5	2.3	1384.1	1361.0	1.7	1401	5	14	TL
50-2-3-4	1329.4	1286.7	0.7	1296.2	1296.2	*	163	5	14	132
50-2-3-5	1385.9	1300.8	4.8	1363.9	1305.9	4.4	2017	6	13	TL
50-3-3-1	1485.7	1364.9	1.1	1380.5	1373.0	0.5	546	5	12	TL
50-3-3-2	1321.5	1207.7	3.1	1244.5	1217.6	2.2	938	6	12	TL
50-4-3-1	1428.8	1314.6	2.0	1340.7	1319.5	1.6	726	6	11	TL
50-4-3-2	1253.0	1175.1	1.1	1188.6	1186.1	0.2	990	7	13	TL
75-2-3-1	2015.4	1844.7	4.3	1923.5	1847.3	4.1	1087	5	18	TL
75-2-3-2	2059.3	1880.5	0.9	1897.3	1887.2	0.5	369	5	20	TL
75-2-3-3	2142.6	1825.6	4.1	1900.3	1828.9	3.9	631	6	19	TL
75-2-3-4	1954.0	1691.1	2.2	1727.5	1692.4	2.1	192	5	18	TL
75-2-3-5	2157.2	1952.9	4.8	2047.4	1953.8	4.8	341	6	20	TL
75-3-3-1	2218.9	1875.8	3.6	1942.4	1880.6	3.3	656	6	18	TL
75-3-3-2	1924.2	1756.6	3.1	1811.9	1757.4	3.1	180	5	18	TL
75-4-3-1	2188.8	1841.2	3.0	1897.2	1844.6	2.8	415	7	17	TL
75-4-3-2	1875.4	1674.6	0.7	1686.2	1677.7	0.5	104	7	18	TL
100-2-3-1	2867.1	2605.5	1.3	2638.9	2609.4	1.1	686	5	28	TL
100-2-3-2	2761.1	2361.1	0.8	2381.0	2365.5	0.7	579	5	27	TL
100-3-3-1	2803.7	2374.6	3.2	2451.2	2374.7	3.2	162	6	25	TL
100-3-3-2	2603.6	2210.5	2.3	2261.5	2210.6	2.3	328	7	24	TL
100-4-3-1	2820.2	2379.5	2.2	2431.5	2380.0	2.2	205	7	23	TL
100-4-3-2	2592.5	2187.7	2.1	2234.4	2187.7	2.1	101	8	24	TL

Table 1.9: No-Split strategy for binary demand and a capacity of 4.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-2-4-1	1009.7	904.0	7.8	974.3	915.7	6.4	2420	6	9	TL
40-2-4-2	1132.4	969.2	3.6	1004.3	986.7	1.8	1639	5	9	TL
40-3-4-1	984.3	864.8	7.4	928.4	872.6	6.4	453	7	8	TL
40-3-4-2	927.3	853.4	2.3	873.2	854.0	2.2	65	6	7	TL
40-4-4-1	998.6	873.1	0.1	873.6	873.6	$*$	5	7	8	34
40-4-4-2	971.8	851.9	2.9	876.4	861.7	1.7	137	7	7	TL
50-2-4-1	1203.5	1121.9	1.3	1136.8	1130.7	0.5	331	6	11	TL
50-2-4-2	1207.0	1110.8	8.7	1207.0	1126.4	7.2	1401	6	11	TL
50-3-4-1	1276.5	1096.8	4.4	1145.4	1099.2	4.2	696	6	10	TL
50-3-4-2	1088.0	1026.0	6	1088.0	1028.2	5.8	69	7	9	TL
50-4-4-1	1203.4	1090.3	2.3	1115.6	1091.7	2.2	309	6	10	TL
50-4-4-2	1120.1	943.1	8.1	1019.2	943.1	8.1	8	9	8	TL
75-2-4-1	1684.0	1518.8	4.7	1590.5	1520.6	4.6	671	6	15	TL
75-2-4-2	1721.3	1495.1	2.8	1536.4	1495.4	2.7	248	7	14	TL
75-3-4-1	1865.7	1490.7	3.6	1544.2	1491.9	3.5	465	7	14	TL
75-3-4-2	1629.3	1449.6	3.2	1495.6	1453.4	2.9	133	9	14	TL
75-4-4-1	1582.5	1368.0	15.7	1582.5	1368.0	15.7	13	10	12	TL
75-4-4-2	1647.2	1431.4	3.4	1480.3	1435.4	3.1	82	8	14	TL
100-2-4-1	2107.4	1888.5	6.1	2003.7	1888.5	6.1	180	7	19	TL
100-2-4-2	2366.6	2018.5	5.7	2133.3	2018.5	5.7	69	7	20	TL
100-3-4-1	2106.2	1771.7	18.9	2106.2	1771.7	18.9	27	8	17	TL
100-3-4-2	2107.7	1763.0	19.6	2107.7	1763.0	19.6	39	8	17	TL
100-4-4-1	2179.4	1811.1	20.3	2179.4	1811.1	20.3	7	7	18	TL
100-4-4-2	2241.0	1882.3	19.1	2241.0	1882.3	19.1	39	7	19	TL

Table 1.10: No-Split strategy for instances of Figure 4 of Muyldermans and Pang (2010).

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-3-1-0	2257.9	1948.8	15.9	2257.9	1954.1	15.5	83	6	18	TL
100-2-3-2-0	2244.5	1806.7	24.2	2244.5	1809.6	24.0	9	6	18	TL
100-2-3-3-0	2221.6	1846.5	20.2	2219.3	1847.3	20.1	10	6	18	TL
100-2-3-4-0	1937.0	1716.4	12.9	1937.0	1718.9	12.7	19	6	17	TL
100-2-3-5-0	2227.8	1866.2	19.3	2226.9	1866.2	19.3	3	6	20	TL
100-2-3-1-33	2674.8	2352.1	4.2	2449.9	2352.3	4.1	324	6	24	TL
100-2-3-2-33	2455.1	2237.0	3.1	2307.1	2237.3	3.1	163	6	24	TL
100-2-3-3-33	2534.2	2245.3	3.3	2319.0	2246.6	3.2	1019	6	23	TL
100-2-3-4-33	2540.4	2141.2	3.6	2218.3	2142.1	3.6	497	6	22	TL
100-2-3-5-33	2485.4	2221.2	2.5	2276.8	2222.3	2.5	312	6	23	TL
100-2-3-1-66	2869.4	2748.3	0.6	2765.6	2751.5	0.5	861	4	31	TL
100-2-3-2-66	2956.9	2674.1	2.8	2748.7	2674.7	2.8	218	5	29	TL
100-2-3-3-66	3205.2	2836.6	1.3	2872.8	2840.3	1.1	277	5	30	TL
100-2-3-4-66	2994.4	2593.8	2.2	2649.7	2595.0	2.1	663	5	28	TL
100-2-3-5-66	3056.7	2809.0	0.6	2827.1	2811.0	0.6	119	5	30	TL

Table 1.11: C-Split strategy for U(1,5) demand and a capacity of 10.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
10-2-10-1	524.1	495.8	1.5	503.3	503.3	*	11	6	4	7
10-2-10-2	477.6	410.2	*	410.2	410.2	*	0	8	3	117
10-2-10-3	461.9	461.2	*	461.2	461.2	*	0	6	4	5
10-2-10-4	463.8	435.6	3.7	451.7	451.7	*	63	8	4	42
10-2-10-5	477.0	392.7	0.9	396.1	396.1	*	35	8	3	30
15-2-10-1	737.4	616.4	*	616.4	616.4	*	0	7	6	9
15-2-10-2	662.3	564.9	*	564.9	564.9	*	0	8	5	14
15-2-10-3	553.1	479.6	5.5	506.0	506.0	*	736	6	5	835
15-2-10-4	675.3	640.7	2.5	656.9	656.9	*	513	6	7	419
15-2-10-5	619.1	520.7	0.8	524.7	524.7	*	134	7	5	1477
20-2-10-1	924.7	833.6	2.0	849.9	849.9	*	258	7	8	946
20-2-10-2	692.6	613.7	2.4	628.5	628.5	*	69	8	6	350
20-2-10-3	668.3	636.8	4.6	666.1	643.9	3.4	81	9	7	TL
20-2-10-4	943.3	876.6	0.8	883.5	883.5	*	141	7	8	981
20-2-10-5	771.0	670.2	2.1	684.0	671.7	1.8	283	8	7	TL
25-2-10-1	1049.7	967.8	0.8	975.3	975.3	*	59	9	8	3206
25-2-10-2	821.9	733.2	3.0	755.2	736.7	2.5	86	8	7	TL
25-2-10-3	887.3	824.7	0.1	825.9	825.3	0.1	20	10	9	TL
25-2-10-4	1143.7	1026.5	2.1	1047.8	1031.9	1.5	127	7	10	TL
25-2-10-5	975.8	827.4	3.0	852.3	833.2	2.3	56	8	9	TL

Table 1.12: C-Split strategy for instances of Figure 5 of Muyldermans and Pang (2010).

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-6-1-A	5668.8	4628.9	22.5	5668.8	4628.9	22.5	14	5	54	TL
100-2-6-2-A	5805.7	4764.9	0.1	4768.5	4767.0	0.0	9	5	58	TL
100-2-6-3-A	5818.1	4511.8	29.0	5818.1	4511.8	29.0	2	6	58	TL
100-2-6-4-A	5841.7	4738.0	23.3	5841.7	4738.0	23.3	2	5	57	TL
100-2-6-5-A	5583.5	4563.2	22.4	5583.5	4563.2	22.4	24	6	52	TL
100-2-12-1-B	7812.8	7501.4	0.1	7510.7	7503.1	0.1	352	4	94	TL
100-2-12-2-B	7561.0	6912.1	0.1	6921.3	6917.4	0.1	315	4	92	TL
100-2-12-3-B	7409.8	6908.7	0.1	6917.3	6911.3	0.1	171	4	91	TL
100-2-12-4-B	7989.5	7422.0	0.4	7448.5	7428.3	0.3	209	4	91	TL
100-2-12-5-B	7868.9	7245.8	0.3	7270.5	7254.1	0.2	250	4	92	TL
100-2-18-1-B	4853.8	4309.5	12.6	4853.8	4309.5	12.6	13	5	51	TL
100-2-18-2-B	4482.6	3969.2	12.9	4482.6	3969.2	12.9	12	5	50	TL
100-2-18-3-B	4803.5	4067.9	18.1	4803.5	4067.9	18.1	18	5	51	TL
100-2-18-4-B	4834.5	4255.5	0.0	4255.5	4255.5	*	15	6	49	1271.7
100-2-18-5-B	5057.4	4155.6	21.7	5057.4	4155.6	21.7	13	5	51	TL

Table 1.13: C-Split strategy for instances of Figure 6 of Muyldermans and Pang (2010).

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-6-1-A	11644.3	10786.7	8.0	11644.3	10786.7	8.0	19	6	51	TL
100-2-6-2-A	12936.6	12066.8	7.2	12930.8	12066.8	7.2	7	7	57	TL
100-2-6-3-A	12452.6	11250.7	10.7	12452.6	11250.7	10.7	47	5	54	TL
100-2-6-4-A	12190.7	10767.1	13.2	12190.7	10767.1	13.2	5	7	54	TL
100-2-6-5-A	13490.0	12459.3	8.2	13476.1	12459.3	8.2	24	5	61	TL
100-2-12-1-B	20518.8	20206.6	0.0	20206.6	20206.6	*	0	4	94	292
100-2-12-2-B	19802.2	19452.2	0.5	19543.6	19530.2	0.1	177	4	91	TL
100-2-12-3-B	19713.3	19454.8	0.3	19522.7	19455.8	0.3	182	4	90	TL
100-2-12-4-B	20196.1	19875.7	0.3	19943.3	19943.3	*	3	3	94	164
100-2-12-5-B	19542.6	19171.2	0.4	19256.3	19245.5	0.1	204	3	91	TL
100-2-18-1-B	11332.5	10650.1	6.4	11332.5	10650.1	6.4	26	5	51	TL
100-2-18-2-B	11023.6	10310.8	6.9	11023.6	10310.8	6.9	11	5	50	TL
100-2-18-3-B	11112.7	10343.9	7.4	11112.7	10343.9	7.4	25	5	50	TL
100-2-18-4-B	11223.7	10475.8	7.1	11223.7	10475.8	7.1	11	5	51	TL
100-2-18-5-B	10938.8	10217.3	7.1	10937.8	10228.5	6.9	38	5	50	TL

Table 1.14: C-Split strategy for binary demand and a capacity of 3.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
10-2-3-1	411.5	398.9	3.0	410.9	410.9	*	21	5	3	6
10-2-3-2	458.9	424.2	1.7	431.6	431.6	*	3	5	4	4
10-2-3-3	357.0	350.2	1.9	357.0	357.0	*	3	5	3	2
10-2-3-4	441.9	393.2	7.2	421.4	421.4	*	290	6	3	84
10-2-3-5	390.6	386.9	*	386.9	386.9	*	0	6	3	1
10-3-3-1	389.8	375.8	3.7	389.8	389.8	*	3	9	2	5
10-3-3-2	397.3	372.3	1.0	376.0	376.0	*	3	9	3	2
10-3-3-3	460.6	393.2	7.2	421.4	421.4	*	81	9	3	35
10-3-3-4	427.1	392.9	0.8	396.2	396.2	*	147	9	3	141
10-3-3-5	422.3	396.5	*	396.5	396.5	*	0	8	3	4
15-2-3-1	530.7	507.3	3.3	523.8	523.8	*	989	6	4	290
15-2-3-2	536.3	508.5	2.1	518.9	518.9	*	3	6	5	3
15-2-3-3	556.4	507.0	3.4	524.1	524.1	*	59	6	5	22
15-2-3-4	462.4	444.4	2.0	453.2	453.2	*	17	6	4	7
15-2-3-5	554.7	526.8	4.6	550.9	550.9	*	455	6	4	372
15-3-3-1	470.9	460.8	1.2	466.3	466.3	*	3	9	4	17
15-3-3-2	532.1	480.2	1.5	487.5	487.5	*	41	8	4	46
15-3-3-3	526.9	478.5	2.5	490.4	490.4	*	346	8	4	1123
15-3-3-4	470.5	444.7	1.4	451.0	451.0	*	37	9	4	358
15-3-3-5	415.9	380.1	3.5	393.4	387.8	1.4	483	9	4	TL
20-2-3-1	687.8	650.1	0.3	651.9	651.9	*	5	6	5	21
20-2-3-2	606.1	588.6	1.7	598.6	598.6	*	21	6	6	215
20-2-3-3	702.6	690.9	1.3	700.1	700.1	*	263	6	6	376
20-2-3-4	600.5	579.5	0.4	581.9	581.9	*	55	5	6	38
20-2-3-5	592.6	556.6	1.6	565.6	565.6	*	301	5	6	498
20-3-3-1	632.5	592.7	3.1	611.2	611.2	*	89	9	5	572
20-3-3-2	601.0	551.4	2.3	564.1	560.2	0.7	545	9	5	TL
20-3-3-3	695.2	617.8	2.7	634.4	631.3	0.5	293	9	6	TL
20-3-3-4	670.3	562.0	1.9	572.4	564.0	1.5	70	9	5	TL
20-3-3-5	700.7	562.6	4.3	587.0	570.9	2.8	155	9	5	TL
30-2-3-1	972.8	841.4	0.1	842.3	842.3	*	5	6	9	36
30-2-3-2	942.4	829.4	2.0	845.9	845.6	0.0	296	6	9	TL
30-2-3-3	1199.8	990.4	1.3	1003.5	1001.1	0.2	549	6	9	TL
30-2-3-4	946.5	831.8	2.1	849.6	834.7	1.8	197	6	9	TL
30-2-3-5	850.3	784.9	1.6	797.5	791.5	0.8	1207	6	8	TL
30-3-3-1	1014.6	870.8	6.4	927.0	870.8	6.4	41	9	7	TL
30-3-3-2	1022.9	786.8	4.0	817.9	796.4	2.7	143	9	8	TL
30-3-3-3	1136.1	938.9	2.3	960.2	952.4	0.8	26	9	9	TL
30-3-3-4	848.3	797.5	1.6	809.9	798.1	1.5	110	9	8	TL
30-3-3-5	1095.3	869.0	3.4	899.0	872.5	3.0	317	9	9	TL
40-2-3-1	1211.3	1120.7	0.6	1127.0	1127.0	*	75	6	10	267
40-2-3-2	1218.4	1055.2	2.0	1076.1	1056.1	1.9	429	6	11	TL
40-2-3-3	1268.6	1136.0	2.5	1165.0	1144.7	1.8	721	6	11	TL
40-2-3-4	1350.3	1113.2	1.9	1134.0	1120.2	1.2	265	6	12	TL
40-2-3-5	1237.2	1049.3	1.6	1065.9	1049.4	1.6	349	6	11	TL

Table 1.15: C-Split strategy for binary demand and a capacity of 3.

	Root			Best known			Info			
ID	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
50-2-3-1	1589.7	1409.0	1.5	1429.8	1412.3	1.2	340	6	14	TL
50-2-3-2	1339.0	1229.1	0.1	1229.8	1229.8	*	23	6	14	854
50-2-3-3	1529.5	1341.6	3.1	1383.8	1347.9	2.7	48	6	14	TL
50-2-3-4	1458.7	1284.6	1.6	1305.2	1287.0	1.4	44	6	14	TL
50-2-3-5	1494.2	1295.9	3.4	1340.4	1298.0	3.3	156	6	13	TL
75-2-3-1	2183.4	1837.3	5.7	1942.2	1840.7	5.5	290	6	18	TL
75-2-3-2	2130.0	1862.0	4.7	1949.7	1862.0	4.7	186	6	20	TL
75-2-3-3	2040.8	1806.4	5.2	1899.8	1806.4	5.2	302	6	18	TL
75-2-3-4	2009.3	1683.9	3.7	1745.6	1683.7	3.7	257	6	17	TL
75-2-3-5	2299.5	1947.3	5.0	2044.7	1947.4	5.0	186	6	20	TL

Table 1.16: Comparison of C-Split to No-Split.

	Set 1	Set 4	Set M5	Set M6	Total
Number of instances	20	55	15	15	105
Number: Both strategies solved opt	14	28	1	2	45
Number: $OPT_{C-Split} = OPT_{No-Split}$	8	20	0	0	28
Number: $OPT_{C-Split} > OPT_{No-Split}$	6	8	1	2	17
Average saving, %	0.88	0.36	0.02	4.78	0.73
Number: $Best_{C-Split} > OPT_{No-Split}$	6	13	6	3	28
Average saving, %	0.93	0.54	3.02	6.34	1.76
Number: $OPT_{C-Split} \geqq ? \; OPT_{No-Split}$	0	14	8	10	32

Chapter 2

An Adaptive Large Neighborhood Search Heuristic for a Periodic Multi-Compartment Vehicle Routing Problem

---

History: *The research was conducted from the autumn of 2015 to the Summer of 2016, and was submitted to a peer-reviewed operations research journal in September 2016.*

An Adaptive Large Neighborhood Search Heuristic for a Periodic Multi-Compartment Vehicle Routing Problem

Samira Mirzaei[†],

[†]Department of Economics and Business Economics, Aarhus University, Denmark, mirzaei@econ.au.dk

Abstract

In this paper, we introduce the Periodic Multi-Compartment Vehicle Routing Problem which is an extension of the Periodic Vehicle Routing Problem in which each customer often requires repeated delivery of multiple commodities over a time horizon. A fleet of multi-compartment vehicles services the customers and each compartment of the vehicles is dedicated to one commodity. We present a mathematical formulation for this problem and develop a meta-heuristic framework based on adaptive large neighborhood search to obtain good solutions for this class of problem. We test our algorithm on a total of 120 PVRP and PMCVRP instances. Computational results for the PVRP instances indicate that the algorithm can find good solutions within a reasonable time for the instances with up to 100 customers and 10 periods. For these instances, an average optimality gap of 1.8 percent is obtained. New benchmark instances are presented for the PMCVRP and computational results are presented.

2.1 Introduction

In practice, for petroleum and animal feeds distribution, delivery of groceries, waste collection, etc., distribution systems are often faced with a situation where the customers request delivery of different commodities and these commodities need to be kept segregated during transport. In the academic literature, there is, however, a tendency of simplifying problems by considering single-commodity models. The majority of studies assume that each customer requires only one type of commodity, whereas there are few papers in this area that investigate the Multi-Compartment Vehicle Routing Problem (MCVRP).

In many supply chains such as e.g. grocery distribution and waste collection the customers often also require repeated visits over a time horizon. This problem is known as the Periodic Vehicle Routing Problem (PVRP). The PVRP is a variation of the Vehicle Routing Problem in which vehicle routes are constructed for more than one period (e.g. 6 days). In this paper, we suppose each period corresponds to a working day. In each day, a vehicle starts from a depot, services some customers, and returns to the depot. Customers are visited a known number of times during the planning horizon following a schedule selected from a menu of schedule options, each of which is a collection of days for visits. For instance, if a customer is serviced twice during 5 days, the menu options may be $\{(Mon., Wed.);\ (Tue., Thu.);\ (Wed., Fri.)\}$.

Considering delivery of multiple commodities by multi-compartment vehicles during the planning horizon is an interesting variant of the PVRP, which we refer to as the Periodic Multi-Compartment Vehicle Routing Problem (PMCVRP). We assume that each customer can only be serviced by a single vehicle on each scheduled visiting day, and the vehicle must deliver the full amount of all commodities ordered for this period by the customer. In line with the definition of Mirzaei and Wøhlk (2016), we refer to this problem as No-Split PMCVRP.

The PVRP has been studied for many years by different researchers, and the interested reader are referred to a comprehensive survey on the PVRP and its extension written by Francis et al. (2008). The PVRP was introduced in 1974 by Beltrami and Bodin. Russell and Igo (1979) provide a formal definition for the problem and Christofides and Beasley (1984) propose the first formulation of the PVRP. Gaudioso and Paletta (1992) present a heuristic for minimization of the number of vehicles. Chao et al. (1995) present a two-phase heuristic. Cordeau et al. (1997) propose a procedure based on a tabu search heuristic for solving different routing problems, including the PVRP. Drummond et al. (2001) implement a parallel genetic algorithm by combining genetic algorithm concepts and local search heuristics. Alegre et al. (2007) consider collection of commodities from different suppliers by a manufacturer of parts for automobiles as a PVRP and investigate this problem in a long planning horizon (30, 60, 90 days). They present a two-phase solution approach for this problem and they develop an adaptation of scatter search for solving the problem. In 2009, Hemmelmayr et al. presented a variable neighborhood search for solving PVRP and the Periodic Traveling Salesman Problem. Baldacci et al. (2011) propose an exact algorithm for solving PVRP. The exact algorithm is based on a set-partitioning-like formulation and uses five types of bounding procedures in order to reduce the number of variables in the formulation. The resulting problem is solved by an integer programming solver. Rahimi-Vahed et al. (2015) propose a new modular heuristic algorithm for solving three vehicle routing problems: multi-depot VRP, PVRP, and multi-depot PVRP. Norouzi et al. (2015) present a new mathematical model for PVRP and they propose an improved particle swarm optimization algorithm for solving the problem. The real world applications of PVRP have been investigated by Hadjiconstantinou and Baldacci (1998), Angelelli and Speranza (2002), Blakeley et al. (2003), and Hamzadayi et al. (2013). Most of these researches provide a two-phase approach for solving the problem. In the first phase, each order is assigned to a set of days in the planning horizon and in the second phase, the VRP sub-problem is solved for each day of the planning horizon (see for instance Alegre et al. (2007)). In this research we use a similar solution approach for solving the No-Split PMCVRP.

The most important application of MCVRP is related to petroleum distribution and is known as the petrol station replenishment problem and has been investigated by Brown and Graves (1981), Brown et al. (1987), Van der Bruggen et al. (1995), Ben Abdelaziz et al. (2002), Avella et al. (2004), and Cornillier et al. (2007). Distribution of animal feed is another application of MCVRP and has been studied by El Fallahi et al. (2008). Due to increased focus on sorting of waste, the waste collection problem is a more recent application of MCVRP and has been considered by Muyldermans and Pang (2010), Reed et al. (2014), and Henke et al. (2015). Chajakis and Guignard (2003) investigate MCVRP for distribution to convenience

stores. For a comprehensive survey, the interested reader can refer to Derigs et al. (2011). MCVRP in more general settings has also been investigated by Repoussis et al. (2007), Wang et al. (2014), and Mirzaei and Wøhlk (2016). Due to the complexity of MCVRP, in most of the related papers, heuristics and meta-heuristics are used and there are few papers that present exact solution methods for solving this problem (see Avella et al. (2004), Cornillier et al. (2007), and Mirzaei and Wøhlk (2016)). However, exact solutions are able to solve instances with up to 50 nodes and 4 commodities for No-Split MCVRP (see Mirzaei and Wøhlk (2016)).

Multi-Period Multi-Compartment Vehicle Routing Problems (MPMCVRP) have been studied by Cornillier et al. (2008), and Lahyani et al. (2015). Cornillier et al. (2008) consider delivery of petroleum products to petrol stations. They present a mathematical model and a heuristic for this problem. The heuristic includes procedures for route construction, truck loading, route packing, and anticipation or the postponement of deliveries. Lahyani et al. (2015) investigate the olive oil collection process in Tunisia. They present a mathematical model, along with a set of known and new valid inequalities. They use this in a branch-and-cut algorithm and evaluate the performance of the algorithm on real data sets under different transportation scenarios. Like PMCVRP, MPMCVRP investigates VRP in a multi-period setting. In PMCVRP, each customer is visited on a set of days based on a schedule chosen from schedule menu options. But in MPMCVRP, each customer can be visited on each day of the planning horizon and the decision on whether or not a customer is visited in each period of the planning horizon is a decision variable.

While the PVRP has been studied by several authors, we are not aware of any research addressing the generalization of this problem with the inclusion of multi-compartment vehicles. Further, we have not found any work addressing the generalization of MCVRP with periodic delivery. In this paper, we introduce the PMCVRP, where each customer often requires repeated delivery of multiple commodities over a time horizon. The multi-compartment vehicles service the customers and each compartment of the vehicle is dedicated to one commodity. We present a mathematical formulation for this problem and we develop a meta-heuristic framework based on adaptive large neighborhood search (ALNS) to find good solutions for this class of problems.

The rest of this paper is organized as follows. In Section 2.2, we formally define the PMCVRP and present a mathematical model for this problem. The metaheuristic algorithm is described in Section 2.3, followed by the results of computational experiments in Section 2.4. Conclusions and some directions for future research are described in Section 2.5.

2.2 Problem Definition

In this section, we describe the No-Split Periodic Multi-Compartment Vehicle Routing Problem in detail and provide a mathematical model for the problem. Even though the problem we consider is symmetric, for ease of reading, we define it on a directed complete graph $G(N,E)$ with node set $N = \{0,1,\ldots,n\}$, where node zero represents the depot and $N' = \{1,\ldots,n\}$ is the set of n customers. Each arc (i,j) in

$E = \{(i,j) : i,j \in N\}$ represents the possibility to travel from node $i \in N$ to node $j \in N$ at non-negative travel cost c_{ij}. The depot stores a set of commodities $M = \{1,2,...,|M|\}$, which must be delivered by the fleet K of identical vehicles each with $|M|$ compartments over a finite and discrete planning horizon $T = \{0,1,2,...,|T|\}$. Compartment $m \in M$ of each vehicle is dedicated to commodity m and has a known capacity Q_m. Each customer may only be serviced by a single vehicle in each period of the planning horizon, i.e. split deliveries are not allowed. The vehicle must deliver the full amount of all commodities ordered for that period by the customer. The fleet of vehicles is available in every period of the planning horizon.

We use M_i to denote the set of commodities demanded by customer $i \in N'$ over the planning horizon. Each customer $i \in N'$ has a known demand $q_{im} \leq Q_m$ for each commodity $m \in M_i$, which must be delivered a known number of times (delivery frequency) f_{im} during the planning horizon. The possible delivery days are given by a commodity-schedule that is chosen from a menu of commodity-schedule options. Each commodity-schedule is a set of days in the planning horizon, in which a commodity is delivered to a customer. For example, if a commodity is delivered to a customer twice during 5 days, one such commodity-schedule could be *{Mon., Thur.}*.
Let D be the set of all commodity-schedules, and index this set by $d \in D$. We refer to D as the commodity-schedule menu options. Each commodity-schedule can be fully described by a vector $\mathbf{e_d}$ such that:

$$e_{dt} = \begin{cases} 1 & \text{if day } t \in T \text{ is in commodity-schedule } d \in D \\ 0 & \text{otherwise.} \end{cases}$$

To fulfill the delivery frequency requirements of each customer i for commodity m, the PMCVRP must choose a commodity-schedule from a non-empty subset of candidate commodity-schedules $D_{im} \subseteq D$ such that:

$$D_{im} = \{d \in D, \sum_{t \in T} e_{dt} = f_{im}\}$$

D_{im} can be seen as the set of commodity-schedules where the number of delivery days equals the delivery frequency of commodity m for customer i.

The number of visits to each customer (visit frequency) Γ_i during the planning horizon is based on the maximum delivery frequency of different commodities ordered by the customer, i.e. $\Gamma_i = \{\max f_{im}, \forall m \in M_i\}$. For example, if two commodities a and b are ordered by the customer $i \in N'$ while $f_{ia} = 2$, and $f_{ib} = 1$, the customer is visited two times during the planning horizon under a customer-schedule that is chosen from a menu of customer-schedule options. Each customer-schedule is a set of days in the planning horizon in which a customer is visited. For example, if a customer is visited twice during 5 days, the customer-schedule menu options may be *{{Mon., Wed.}; {Tue., Thu.}; {Wed., Fri.}}*. Based on the customer-schedule of the customer, the commodity-schedule menu options for the requested commodities of the customer are updated.
We illustrate this by means of an example. The commodity-schedule menu options for commodity a may be *{{Mon., Wed.}; {Tue., Thu.}; {Wed., Fri.}}*, and for commodity b it may be *{{Mon.};{Wed.}}*, *{{Tue.};{Thu.}}*, *or {{Wed.};{Fri.}}*. If the customer-schedule *{{Mon., Wed.}}* is chosen for visiting the

customer, the commodity-schedule of commodity a will be $\{\{Mon., Wed.\}\}$, and the commodity-schedule of commodity b will be either $\{\{Mon.\}\}$ or $\{\{Wed.\}\}$.
Let S be the set of all customer-schedules, and index this set by $s \in S$. Each customer-schedule can be fully described by a vector $\mathbf{o_s}$ such that:

$$o_{st} = \begin{cases} 1 & \text{if day } t \in T \text{ is in customer-schedule } s \in S \\ 0 & \text{otherwise.} \end{cases}$$

To fulfill the visit frequency requirements of each customer i during the planning horizon, the PMCVRP must select a schedule from a non-empty subset of candidate customer-schedules $S_i \subseteq S$ such that:

$$S_i = \{s \in S, \sum_{t \in T} o_{st} = \Gamma_i\}$$

The model uses the variables y_{ij}^{kt} as the number of times the arc $(i, j) \in E$ is traversed by vehicle $k \in K$ in period $t \in T$. The variable z_i^s is one if customer $i \in N'$ is visited on the customer-schedule $s \in S_i$, and zero otherwise. The variable h_{im}^d is one if the commodity $m \in M_i$ is delivered to the customer $i \in N'$ by the commodity-schedule $d \in D_{im}$, and zero otherwise. The variable v_i^t is a binary aggregate decision variable which takes value 1 if customer $i \in N'$ is serviced on day $t \in T$, and value 0 otherwise. The variable $\bar{v}_{im}^t$ is a binary aggregate decision variable which takes value 1 if commodity $m \in M_i$ is delivered to customer

$i \in N'$ on day $t \in T$, and value 0 otherwise.

$$\min \quad \sum_{(i,j)\in E}\sum_{k\in K}\sum_{t\in T} c_{ij}\, y_{ij}^{kt} \tag{2.1}$$

$$\text{st} \quad \sum_{s\in S_i} z_i^s = 1 \quad \forall i \in N' \tag{2.2}$$

$$\upsilon_i^t = \sum_{s\in S_i} z_i^s o_{st} \quad \forall i \in N',\ t \in T \tag{2.3}$$

$$\sum_{d\in D_{im}} h_{im}^d = 1 \quad \forall i \in N',\ m \in M_i \tag{2.4}$$

$$\bar{\upsilon}_{im}^t = \sum_{d\in D_{im}} h_{im}^d e_{dt} \quad \forall i \in N',\ m \in M_i,\ t \in T \tag{2.5}$$

$$\bar{\upsilon}_{im}^t \le \upsilon_i^t \quad \forall i \in N',\ m \in M_i,\ t \in T \tag{2.6}$$

$$\sum_{k\in K} y_{ij}^{kt} \le (\upsilon_i^t + \upsilon_j^t)/2 \quad \forall i,j \in N' (i \ne j),\ t \in T \tag{2.7}$$

$$\sum_{j\in N} y_{ij}^{kt} = \sum_{j\in N} y_{ji}^{kt} \quad \forall i \in N, k \in K, t \in T \tag{2.8}$$

$$\sum_{k\in K}\sum_{i\in N} y_{ij}^{kt} = \begin{cases} \upsilon_j^t & \forall j \in N', t \in T, \\ |K| & j = 0, \forall t \in T \end{cases} \tag{2.9}$$

$$\sum_{i\in N'} q_{im}\bar{\upsilon}_{im}^t \sum_{j\in N} y_{ij}^{kt} \le Q_m \quad \forall m \in M_i, k \in K, t \in T \tag{2.10}$$

$$\sum_{j\in N'} y_{0j}^{kt} \le 1 \quad \forall k \in K, t \in T \tag{2.11}$$

$$\sum_{j:(i,j)\in E, i,j\in\omega} y_{ij}^{kt} \le |\omega| - 1 \quad \forall k \in K, \omega \subseteq N', |\omega| \ge 2 \tag{2.12}$$

$$z_i^s \in \{0,1\} \quad \forall i \in N', s \in S_i \tag{2.13}$$

$$h_{im}^d \in \{0,1\} \quad \forall i \in N', m \in M_i, d \in D_{im} \tag{2.14}$$

$$y_{ij}^{kt} \in \{0,1\} \quad \forall (i,j) \in E, k \in K, t \in T \tag{2.15}$$

$$\upsilon_i^t \in \{0,1\} \quad \forall i \in N', t \in T \tag{2.16}$$

$$\bar{\upsilon}_{im}^t \in \{0,1\} \quad \forall i \in N', m \in M_i, t \in T \tag{2.17}$$

The objective function is to minimize the transportation costs. Constraints (2.2) ensure that a customer-schedule is assigned to each customer. Constraints (2.3) define υ_i^t on the days based on the customer-schedule selected in constraints (2.2). Constraints (2.4) ensure that one commodity-schedule is chosen for each commodity requested by each customer. Constraints (2.5) fix the $\bar{\upsilon}_{im}^t$ variables in the same way that constraints (2.3) fix the υ_i^t variables. Constraints (2.6) ensure that each commodity is delivered on possible visit days for each customer. Constraints (2.7) allow arcs only between customers assigned for visits in the period $t \in T$. Constraints (2.8) are the ordinary flow conservation constraints. Constraints (2.9) certify that nodes are included on routes for periods within their assigned schedule. Constraints (2.10) ensure that compartment capacities are respected. Constraints (2.11) ensure that each vehicle is used at most once a day. Constraints (2.12) are the classical sub-tour elimination constraints, and constraints (2.13-2.17) define the domain of the variables.

2.3 Solution Approach

The PMCVRP is an extension of the CVRP problem and therefore *NP*-hard. Its complexity grows exponentially with the number of customers and time periods. In addition, the multi-compartment vehicles add complexity to the problem.
Our solution approach for solving the PMCVRP is inspired by Alegre et al. (2007) and we iterate between the following three steps:

1) Select a customer-schedule s for each customer $i \in N'$ from the subset S_i.

2) Select a commodity-schedule d for each commodity m ordered by a customer i from the subset D_{im} such that the visit schedule of customer i is satisfied.

3) Design daily routes with the objective of minimizing total cost.

This is integrated in an adaptive large neighborhood search (ALNS) algorithm which is proposed by Røpke and Pisinger (2006) as an extension of the large neighborhood search algorithm. The idea is that in each iteration, part of the current solution is destroyed and then reconstructed in the search for a better solution. The destruction phase is based on removing ρ customers from the current solution and placing them in the unassigned node pool ϕ. In the construction phase, the nodes from ϕ are reinserted into the solution. A number of destroy and repair operators are available and a destroy operator and a repair operator are randomly chosen at each iteration. A weight is assigned to each operator and the selection probability of an operator is related to its weight, which is adjusted dynamically according to its previous success.

The proposed algorithm is initialized with a nearest neighborhood heuristic (NNS) which is described in Section 2.3.1. At each iteration, a new solution χ' is obtained by applying a destroy and a repair operator on the current solution χ. The destroy and repair operators are chosen via a roulette wheel mechanism based on their current weight. The destroy and repair operators are described in Sections 2.3.2 and 2.3.3 and the selection of them is described in Section 2.3.4.

At the end of each iteration, we apply an acceptance rule based on the simulated annealing (SA) to the new solution. If the new solution is accepted, it replaces the current solution. We perform a cooling procedure based on the dynamic repetition schedule of Dayarian et al. (2013). The cooling procedure is performed when the best feasible solution does not improve for σ iterations, with σ equal to the number of routes in the best feasible solution. The dynamic repetition schedule dynamically determines the number of iterations with each temperature. The procedure divides the search into segments that have different numbers of iterations. At the end of each segment, the weights of the repair and destroy operators are updated, and a 2-opt local search is applied to the best solution to further improve the solution. This is described in Section 2.3.5. An overview of the approach is provided in Algorithm 3.

Algorithm 3 Solution Approach

$\chi =$ *initial solution by NNS heuristic*;
te_{init}= initial temperature;
$\chi* = \chi$; $\tau = te_{init}$;
$\sigma =$ *number of routes in* χ;
$NoImp = 0$;
while $\tau \geq te_{final}$ and $NoImp < 3\sigma$ **do**
    $segmentIter = 0$;
    while $segmentIter \leq \sigma$ **do**
        Select a destroy operator;
        Remove ρ customers from the solution;
        Select a repair operator;
        for all $i \in \phi$ **do**
            for all $s \in S_i$ **do**
                for all $m \in M_i$ **do**
                    Assign a random commodity-schedule from subset D_{im} to
                    commodity m;
                end for
                Calculate the insertion cost of customer i with respect to schedule s;
            end for
            Save the information related to the best insertion of the customer i;
        end for
        Insert the customers at the best possible places of χ and obtain χ';
        if χ' satisfies the acceptance criteria **then**
            $\chi \leftarrow \chi'$;
            if χ' is better than $\chi*$ **then**
                $\chi* \leftarrow \chi'$; $\sigma = \sigma(1+\varpi)$, $\varpi \in (0,1)$; $NoImp = 0$;
            end if
        else
            $NoImp = NoImp + 1$;
        end if
        $segmentIter = segmentIter + 1$;
    end while
    Apply a local search on χ and obtain χ';
    if χ' is better than χ, **then**
        $\chi \leftarrow \chi'$;
        if χ' is better than $\chi*$ **then**
            $\chi* \leftarrow \chi'$;
        end if
    end if
    Adjust weights;
    $\sigma =$ *Number of routes*;
    $\tau \leftarrow \tau\kappa$ $(\kappa \in (0,1))$
end while

2.3.1 Initialization

An initial solution is obtained as follows. Based on the visit frequency requirements of each customer, all possible customer-schedules are first determined and a random customer-schedule is assigned to the customer.
Next, based on the customer-schedule assigned to the customer, all possible com- modity-schedules for the commodities requested by the customer are determined and a random commodity-schedule is assigned to each commodity.
Finally, routes are constructed by solving a CVRP for each period of the planning horizon by means of a nearest neighbor heuristic. Each route is started at the depot and repeatedly extends from its current point to the nearest unvisited customer subject to the vehicle compartments' capacities and the selected customer-schedule of the customer, where nearest is measured in terms of travel cost. The route extension continues until there are no more capacity for servicing customers or until all customers assigned to the period have been visited, at which point the vehicle returns to the depot. If there are unvisited customers or unplanned periods in the planning horizon, a new route is initialized.

2.3.2 Destroy Operators

We use four different destroy operators for removing ρ customers from the current solution. $\rho = 0.15 * N'$ determined by tuning. Each customer is visited a known number of times during the planning horizon and as there are routes with the same customers and different visit days, the customers should be removed from all routes in all periods of the planning horizon of the solution and placed in ϕ. The destroy operators are described below.

2.3.2.1 Random Removal

In this heuristic, ρ random customers are removed from all routes in all periods of the planning horizon of the solution and they are placed in ϕ.

2.3.2.2 Worst Removal

This heuristic seeks to obtain a better solution by removing ρ customers with high costs from all routes in all periods of the solution. The cost of each customer is determined by calculating the cost difference of the solution with and without the customer. Let L be an array of all customers, sorted in descending order by cost. Then ρ customers are selected randomly as $L[\vartheta^p|L|]$, where ϑ is a random number between zero and one and p is a parameter that determines the degree of randomization in the heuristic. In the tuning phase, $p = 6$ is selected. The selected customers are removed from all routes in all periods of the planning horizon and are placed in ϕ.

2.3.2.3 Shaw Removal Heuristic

This heuristic is a modified version of the shaw heuristic proposed by Shaw (1998). The idea of this heuristic is to remove customers with similarity from all routes in all periods of the planning horizon of the solution. The similarity of two customers i and j is measured by a relatedness measure $R(i,j)$. The lower value of $R(i,j)$ shows more relatedness of two customers. Our relatedness measure consists of a travel cost term based on distance and a capacity term based on the demand for each commodity. These terms are weighted using the weights η for the travel cost and γ for the commodity-demand of the customers. The relatedness measure is given by

$$R(i,j) = \eta c_{ij} + \gamma \sum_{m \in (M_i \cup M_j)} \mid q_{im} - q_{jm} \mid$$

In each iteration of the heuristic, a random customer $i \in \phi$ is selected and the relatedness of the rest of the customers (in $(N' \setminus \{\phi\})$) compared to customer i is determined. The remaining customers are sorted in an array, L, in increasing order by the relatedness measure. A random customer $j = L[\vartheta^p |L|]$ is selected and removed from all routes in all periods of the planning horizon of the solution and is placed in ϕ, $(\phi = \phi \bigcup j)$. This procedure is repeated ρ times until ρ customers have been removed from the solution and added to ϕ. In the first iteration, $\phi = \emptyset$ and therefore a random customer $i \in N'$ is removed from the solution and placed in ϕ. After computational testing, the values $p = 6$, $\eta = 6$, and $\gamma = 3$ were chosen.

2.3.2.4 Route Removal

This heuristic tries to decrease the number of routes based on the number of available vehicles in each period of the planning horizon. This heuristic identifies the route with the lowest number of customers and the customers of this route are removed from the solution and placed in ϕ. If the number of removed customers is lower than ρ, shaw removal is used to make additional removals until ρ is reached.

2.3.3 Repair Operators

After removing ρ customers by a destroy operator and placing them in ϕ, they are considered for reinsertion. Two different repair operators have been considered for this purpose. They shall restore feasibility by inserting the ρ customers that were removed by the destroy operators.

2.3.3.1 Greedy Heuristic

This heuristic is a modified version of the greedy heuristic described by Røpke and Pisinger (2006). It performs ρ iterations where, in each iteration, a customer with minimum total change of objective value is inserted at the best possible places of the solution.

Due to the periodic nature of the problem, we must assign both a visit schedule to each customer $i \in \phi$, and a delivery schedule to each commodity $m \in M_i$. First, the heuristic calculates the total change

in objective value for each schedule $s \in S_i$ incurred by inserting each customer $i \in \phi$ at the best possible places of all routes in all periods of the solution. It returns a customer-schedule $s \in S_i$ with minimum total change of objective value as a new schedule for the customer i.
Next, a random commodity-schedule from the subset D_{im} is assigned to each commodity $m \in M_i$ requested by the customer i. After assigning the best schedule to each customer $i \in \phi$, the heuristic chooses a customer i with minimum total change of objective value and inserts the selected customer i in the minimum cost position of the best route for each period $t \in T$. In case of a tie the heuristic selects an arbitrary customer. The heuristic removes the selected customer from ϕ and repeats this procedure until there are no more customers available for insertion.

Let x_{ikt} be a variable that indicates the route with k'th lowest insertion cost for customer i in period $t \in T$ of the planning horizon and $\Delta c_{i,x_{i1t},t,s}$ be the change of objective value incurred by inserting the customer i based on schedule $s \in S_i$ at the best position of the route x_{i1t} with the lowest insertion cost in period $t \in T$.

First, we specify $C_{i,s}$ as the total change of objective value (for all periods of the planning horizon) by inserting the customer i in the solution based on visit schedule $s \in S_i$, $C_{i,s} = \sum_{t \in T} \Delta c_{i,x_{i1t},t,s}$. Next, we determine $MC_i = \min_{s \in S_i}\{C_{i,s}\}$ as the minimum total change of objective value for inserting the customer i in the solution, and we assign the best schedule with minimum $C_{i,s}$ to the customer i. Finally, the customer i with minimum MC_i is inserted in the minimum cost positions of the solution, and ϕ is updated, $\phi = \phi \setminus \{i\}$. We repeat this procedure until there are no more customers available for insertion.

2.3.3.2 Regret Heuristic

The regret heuristic is a modified version of the regret heuristic described by Røpke and Pisinger (2006). This heuristic tries to insert the customers of ϕ in the solution by considering the cost difference between the best insertion place of each customer and its best insertion places in alternative routes for each period of the planning horizon. This cost difference is considered as a regret value. Intuitively, the unrouted customers with large regret values should be considered first because the increased cost for not being able to insert them at their best places is higher. Further, customers with small regret value can be inserted later without losing much.

Let $\Delta c_{i,x_{ikt},t,s}$ be the change in objective value incurred by inserting the customer i in the route with the k'th lowest insertion cost based on the schedule $s \in S_i$. We can define the k'th regret value in each period $t \in T$ by $C^*_{i,k,t,s} = \sum_2^k (\Delta c_{i,x_{ikt},t,s} - \Delta c_{i,x_{i1t},t,s})$, which is the cost difference between the best insertion place of each customer and each customer's best insertion places in alternative routes. For example, if $k = 2$, the regret value is the cost difference between the best insertion position and the second best in each period of the planning horizon.

First, we specify $TC^*_{i,k,s}$ as the total regret value of inserting the customer i in the k'th best route for all periods of the planning horizon, $TC^*_{i,k,s} = \sum_{t \in T} C^*_{i,k,t,s}$. There are more than one customer-schedule for each customer i, so next we determine $BTC^*_{i,k} = \max_{s \in S_i}\{TC^*_{i,k,s}\}$ as the maximum regret value of inserting the customer i in the solution and we assign the schedule with maximum regret value to the customer i.

Finally the heuristic chooses to insert the customer i with maximum $BTC^*_{i,k}$, in the positions that lead to the lowest increase in objective value. We update ϕ and repeat this procedure until there are no more customers available for insertion. The regret heuristic with k'th regret value is called the regret-k heuristic. In this research we applied regret-2, regret-3, and regret-4 heuristics.

2.3.4 Selection of Destroy and Repair Operators

We have described the destroy operators (random removal, worst removal, shaw removal, and route removal) and repair operators (greedy heuristic, regret-2, regret-3, and regret-4) in Sections 2.3.2 and 2.3.3. In each iteration of the ALNS, a destroy operator and a repair operator are selected. The procedure for selecting an operator (either destroy or repair) is the same although the selection of a destroy operator and a repair operator is totally independent. The procedure for selecting an operator is as follows:

We associate a weight θ_i with each operator i. Suppose that we have l operators with weights $\theta_i, i \in \{1,2,3,...,l\}$. Initially, the weights of all operators are identical, $\theta_i = 1, i \in \{1,2,3,...,l\}$. Based on $\theta_i, i \in \{1,2,3,...,l\}$, the probability of selection of operator j in each iteration is

$$\theta_j / (\sum_{i=1}^{l} \theta_i) \tag{2.18}$$

The weights of the operators are updated automatically by the adaptive weight adjustment method described by Røpke and Pisinger (2006). The idea is to calculate a score for each operator based on its performance. The entire search is divided into a number of segments, each containing a number of iterations σ. The length of each segment is considered σ at first, but when a new best feasible solution is found, the length of the segment is increased by $\sigma\varpi$ ($\varpi \in (0,1)$). The score of all operators is set to zero at the beginning of each segment. In each iteration, we apply a destroy and a repair operator and the scores of both of them are updated by the same amount. In each iteration, the score of the selected operator is increased by either β_1, β_2, or β_3 with $\beta_1 \geq \beta_2 \geq \beta_3$ as follows. If the new solution is a new global best, the scores of the selected operators are increased by β_1; if the new solution is better than the current one, the scores are increased by β_2; and if the new solution is worse but accepted, the scores are increased by β_3.

After completion of each segment $\hbar$ of the ALNS, the weights of all heuristics are adjusted for the next segment $\hbar + 1$ as follows:

$$\theta_{i,\hbar+1} = \lambda\, \theta_{i,\hbar} + (1-\lambda)\pi_i / \psi_i \tag{2.19}$$

where π_i is the sum of the scores of operator i obtained in the last segment, and ψ_i is the number of times that operator was chosen during the last segment. $\lambda \in [0,1]$ is a parameter that controls the sensitivity of the weights to change in the performance of the selected destroy and repair operators. After testing different values, was selected $\beta_1 = 7.5$, $\beta_2 = 5$, $\beta_3 = 2.5$, and $\lambda = 0.5$.

2.3.5 Acceptance and Stopping Criteria

The criterion for acceptance or rejection of a new solution is adopted from Simulated Annealing. If the new solution χ' is better than the current solution χ, it is accepted. Otherwise, it is accepted by probability

$$\exp[-(f(\chi') - f(\chi))/\tau]$$

where $f(\chi)$ represents the objective value of solution χ, and τ is the temperature parameter. Starting from initial value te_{init}, the temperature is lowered by setting $\tau \leftarrow \tau\kappa$ ($\kappa \in (0,1)$). During the search, the probability of accepting worse solution diminishes as τ decreases. The cooling procedure is performed when the best feasible solution does not improve for σ iterations. The cooling procedure is based on the dynamic repetition schedule of Dayarian et al. (2013). The dynamic repetition schedule dynamically determines the number of iterations in each temperature. This procedure divides the search into a number of segments with different numbers of iterations. The length of each segment is considered σ at first, and when a new best feasible solution is found, the length of the segment is increased by $\sigma\varpi$.

If the best feasible solution does not improve for 3σ iterations or the temperature equals the final temperature te_{final} or the time limit $TL = 600$ seconds is reached, the stopping condition is met and we stop; otherwise, we continue to the next step to improve the current solution. After computational testing, we chose the values $te_{init} = 1500$, $te_{final} = 0$, and $\kappa = 0.995$.

2.4 Computational Experiments

In this section, we present our computational results and analysis, which form the basis of our conclusions. The algorithm is implemented in C++ in Microsoft Visual Studio 2010. All tests were made on a laptop with a Pentium core i5 processor and a clock speed of 1.8 GHz and 8 GB of RAM.

Details about our test sets are presented in Section 2.4.1. We have used the standard benchmark instances of PVRP which are referred to as set 1. Set 1 enables us to compare our results to those presented in existing literature. We have also generated three additional sets of test instances that differ by the number of commodities. They are indexed 2 through 4. In total, we have tested our algorithm on 120 instances. In Section 2.4.2, we analyze the performance of our algorithm as regards its ability to solve PVRP instances, whereas Section 2.4.3 is devoted to the performance as regards the PMCVRP.

2.4.1 Data Instances

Since there are no test problems for the PMCVRP in the literature for evaluating the performance of the proposed algorithm, the standard benchmark instances of the PVRP are used (set 1). This data set contains 30 instances where instances p01-p10 are proposed by Eilon et al. (1974) for the VRP and adapted to the PVRP by Christofides and Beasley (1984). Instance p12 is taken from Russell and Gribbin (1991) and instances p14-p32 are introduced by Chao et al. (1995).

We have also generated a total of 90 new data instances for the PMCVRP and have grouped these into 3 sets. The number of commodities in set 2 is 2. This amount increases to 3 and 4 for sets 3 and 4, respectively. All customer locations in sets 2-4 are based on the locations of the instances in set 1. The vehicles are homogeneous and the number of compartments of each vehicle in each instance is based on the number of commodities.

In each of the three sets, the customers' demand for each commodity in each instance is a random number between the maximum and minimum amount of their demand in the related instance of set 1 and therefore the customers' demand is as follows: The demand for each commodity in instances p01-p10 is random numbers between 0-41, for instance p12 they are random numbers between 0-105, and for the instances p14-p32 they are random numbers between 1-5. In each instance, a random delivery frequency and a compatible list of commodity-schedules are assigned to each commodity ordered by each customer. The three new sets of PMCVRP instances are available at http://www.optimization.dk/PMCVRP.

2.4.2 Results for the PVRP

The PVRP test instances in set 1 have been solved by Christofides and Beasley (1984), Russell and Gribbin (1991), Chao et al. (1995), Hemmelmayr et al. (2009) among others by heuristic algorithms. Baldacci et al. (2011) present an exact method for solving the PVRP. Table 2.1 reports the results for the PVRP data sets in the literature. The best results are marked in bold.

We compare the results obtained by our ALNS algorithm on set 1 for just a single commodity to those obtained by the three most recent algorithms: the exact algorithm proposed by Baldacci et al. (2011), the scatter search (SS) presented by Alegre et al. (2007), and the variable neighborhood search heuristic (VNS) proposed by Hemmelmayr et al. (2009). Table 2.2 provides a comparison with the exact algorithm proposed by Baldacci et al. (2011), and Table 2.3 provides a comparison with the algorithms by Alegre et al. (2007) and Hemmelmayr et al. (2009).

Table 2.2 compares our ALNS results with the best solutions reported by Baldacci et al. (2011) for set 1. The first three columns of this table report the instance name (*Instance*), the number of nodes (*Nodes*), and the number of periods (*Periods*), respectively. Under the *BBMV* and *ALNS* headers, we report the best solutions found by Baldacci et al. (2011) and our ALNS algorithm. The last column presents the gap of the ALNS solution (z_{ALNS}) compared to the best solution of Baldacci et al. (2011) (z_{BBMV}), calculated as $(z_{ALNS} - z_{BBMV})/z_{BBMV}$.

From Table 2.2, it is clear that the performance of the ALNS is better for instances with short time horizons than for instances with long time horizons. The average gap of the ALNS is less than 1.7 percent for instances with at most 4 periods. Our algorithm found instances with long time horizons and a high number of customers more difficult. However, the performance of the algorithm is acceptable for instances with up to 100 nodes and 10 periods with an average gap of 1.8 percent. In total, our solution approach could find the best solutions for 6 instances, and for twenty instances, the maximum gap is less than 5.2 percent. The average gap of the ALNS solutions compared to z_{BBMV} for all PVRP instances is less than 4.6, and the

maximum gap is 18.1.
The moderate performance of our algorithms for instances p29-p32 compared to the best solutions of Baldacci et al. (2011) may be explained by the fact that the number of customers and the number of possible schedules of each node for these instances are higher than for the rest of the instances and the running time of each iteration of the ALNS is higher than for other instances. Therefore, the solution approach could not find a good solution within the time limit. Running these instances with longer duration may help to find a better solution with a lower gap.

Under the *ALP*, *HDH*, and *ALNS* headers of Table 2.3, we present the results of the SS by Alegre et al. (2007), the VNS by Hemmelmayr et al. (2009), and our ALNS algorithm, respectively. Under the *Time* header of this table, we present our computation time in seconds. The last two columns of Table 2.3 show the gap between our results and the results of the other two algorithms (*ALP-ALNS Gap*, and *HDH-ALNS Gap*). The gap is calculated as $(z_{ALNS} - z_{heuristic})/z_{heuristic}$, where $z_{heuristic}$ is the best solution obtained by SS and VNS. A negative gap value indicates that our algorithm outperforms the heuristic for the instance investigated whereas a positive gap value shows that our algorithm was outperformed. For three instances, our algorithm outperformed the VNS and the SS. For the instances p01-p10 and p13-p28, the solution of the ALNS algorithm is close to the VNS and the SS, and the average gap of the ALNS compared to the VNS and the SS for these instances is 1.99 and 2 percent, respectively. This is an acceptable performance when considering that our algorithm is developed for solving the more complicated multi-compartment problem. For the instances with high numbers of customers and high numbers of periods, the performance of our algorithm is worse than the VNS and the SS. Even though the gap value for our algorithm is relatively high for these instances due to their complexity, the performance is still acceptable.

2.4.3 Results for the PMCVRP

Table 2.4 shows computational results and running times for our new data sets. As no paper has addressed this problem before, no comparison is possible.

Columns 2, 4, and 6 of Table 2.4 show the transportation cost for similar instances with different numbers of requested commodities. From these columns we see that the transportation cost is a function of the number of commodities and when the number of commodities increases, transportation cost is increased. In this project, we require each customer to be serviced by at most one vehicle in each period of the planning horizon. The complexity of the problem therefore increases with increasing numbers of commodities as the vehicle may lack capacity for all commodities. Therefore the customer will be serviced by another vehicle, which increases the transportation cost.

Columns 3, 5, and 7 of Table 2.4 present the running times of instances in sets 2-4, respectively. From Column 3, we see that the ALNS is able to stop before the time limit for the instances with at most 100 nodes and 4 periods. The ALNS is able to stop before the time limit for 10 instances of set 3 and 6 instances of set 4. When we compare the running time of the same instances of different sets, we see that the running time for each instance is increased when the number of commodities increases. Due to increased complexity

of the problem when the number of commodities increases, these results are rational. Finally, we note that the running time increases with the number of nodes and with the length of the time horizon, which is also expected.

2.5 Concluding Remarks

In this paper, we have investigated a distribution system in which the customers request periodic pickup or delivery of different commodities over a time horizon and in which the commodities must be kept segregated during transport. Examples could be delivery of groceries or waste collection (pick-up).

We proposed a mathematical formulation for this problem and presented the first heuristic for this class of problems. Our algorithm is based on ALNS.

We have used the standard benchmark instances of PVRP which are referred to as set 1. We have also generated three sets of test instances which differ in number of commodities. They are indexed 2 through 4. In total, we have tested our algorithm on 120 instances.

Computational results for the PVRP data set indicate that for the instances with up to 100 customers and 10 periods, the algorithm can find good solutions within a reasonable time with an average gap of 1.8 percent (compared to the best solution found in the literature). Our algorithm finds instances with long time horizons and high number of customers more difficult to solve, although the average gap for all PVRP instances is less than 4.6 percent.

The results of the new data sets show that the instances with long time horizons, a high number of customers, and many commodities are more challenging for the ALNS algorithm. Because this is the first paper to study the PMCVRP, no direct comparison can be performed for these instances. And for the same reason, there are many options for future research. We have presented a heuristic to obtain feasible solutions, but as the quality of these solutions is unknown, a natural starting point for future research could be aimed at finding an efficient lower bound for this problem. As is the case for other routing problems, finding an exact approach is a natural next step. Therefore, further research could focus on developing algorithms such as Branch-and-Price for this class of problems. Finally, the PMCVRP can be made more applicable by including other factors such as uncertainty of customer demands, time windows with soft penalties, overall transportation costs, and so on in the model.

Table 2.1: Computational results in the literature for the PVRP data sets (set 1), Tan and Beasley (1984) (TB), Christofides and Beasley (1984) (CB), Russell and Gribbin (1991) (RG), Chao et al. (1995) (CGW), Cordeau et al. (1997) (CGL) Alegre et al. (2007) (ALP), Hemmelmayr et al. (2009) (HDH), and Baldacci et al. (2011)(BBMV).

Instance	TB	CB	RG	CGW	CGL	ALP	HDH	BBMV
p01	-	547.4	537.3	**524.6**	**524.6**	531.0	**524.6**	**524.6**
p02	1481.3	1443.1	1355.4	1337.2	1330.1	1324.7	**1332.0**	1322.9
p03	-	546.7	-	**524.6**	**524.6**	537.4	529.0	**524.6**
p04	-	843.9	867.8	860.9	837.9	846.0	847.5	**835.3**
p05	2192.5	2187.3	2141.3	2089.0	2061.4	2043.8	2059.7	**2028.0**
p06	-	938.2	-	881.1	840.3	840.1	884.7	**835.3**
p07	-	839.2	833.6	832.0	829.4	829.7	829.9	**826.1**
p08	2281.8	2151.3	2108.3	2075.1	2054.9	2052.2	2058.4	**2034.2**
p09	-	875.0	-	829.9	829.5	829.7	834.9	**826.1**
p10	1833.7	1674.0	1638.5	1633.2	1630.0	1621.2	1629.8	**1593.5**
p12	-	-	1312.0	1237.4	1239.6	**1231.0**	1258.5	-
p14	-	-	-	**954.8**	**954.8**	**954.8**	**954.8**	**954.8**
p15	-	-	-	**1862.6**	**1862.6**	**1862.6**	**1862.6**	**1862.6**
p16	-	-	-	**2875.2**	**2875.2**	**2875.2**	**2875.2**	**2875.2**
p17	-	-	-	1614.4	**1597.8**	**1597.8**	1601.8	**1597.8**
p18	-	-	-	3217.7	3159.2	3157.0	3147.9	**3136.7**
p19	-	-	-	4846.5	4902.6	4846.5	4851.4	**4834.3**
p20	-	-	-	**8367.4**	**8367.4**	8412.0	**8367.4**	-
p21	-	-	-	2216.1	2184.0	2173.6	2180.3	**2170.6**
p22	-	-	-	4436.4	4307.2	4330.6	4218.5	**4194.0**
p23	-	-	-	6769.0	**6620.5**	6813.5	6644.9	-
p24	-	-	-	3773.0	3704.1	3702.0	3704.6	**3687.5**
p25	-	-	-	3826.0	3781.4	3781.4	3781.4	**3777.2**
p26	-	-	-	3834.0	**3795.3**	**3795.3**	**3795.3**	**3795.3**
p27	-	-	-	23401.6	23017.5	22561.3	22153.3	**21912.9**
p28	-	-	-	23105.1	22569.4	22562.4	22418.5	**22242.5**
p29	-	-	-	24248.2	24012.9	23752.2	22864.2	**22543.8**
p30	-	-	-	80982.1	77179.3	76794.0	75579.2	**74464.3**
p31	-	-	-	80279.1	79382.4	77944.8	77459.1	**76322.0**
p32	-	-	-	83838.7	80909.0	81055.5	79488.0	**78072.9**

Table 2.2: Computational results of the ALNS algorithm with the results of Baldacci et al. (2011) for set 1.

Instance	Nodes	Periods	BBMV	ALNS	BBMV-ALNS GAP
p01	52	2	524.61	**524.61**	0.00
p02	51	5	1322.87	1375.73	4.00
p03	51	5	524.61	**524.61**	0.00
p04	76	2	835.26	875.693	4.84
p05	76	5	2027.99	2187.19	7.85
p06	76	10	835.26	885.24	5.98
p07	101	2	826.14	864.648	4.66
p08	101	5	2034.15	2165.12	6.44
p09	101	8	826.14	845.614	2.36
p10	101	5	1593.45	1695.5	6.40
p14	21	4	954.81	**954.81**	0.00
p15	39	4	1862.63	**1862.63**	0.00
p16	57	4	2875.24	**2875.24**	0.00
p17	41	4	1597.75	**1597.75**	0.00
p18	77	4	3136.69	3187.21	1.61
p19	113	4	4834.34	4846.49	0.25
p21	61	4	2170.61	2196.58	1.20
p22	115	4	4193.95	4447.13	6.04
p24	52	6	3687.46	3718.08	0.83
p25	52	6	3777.15	3793.09	0.42
p26	52	6	3795.32	3803.32	0.21
p27	103	6	21912.85	23447.2	7.00
p28	103	6	22242.51	23374.6	5.09
p29	103	6	22543.76	24949.3	10.67
p30	154	6	74464.26	87797	17.90
p31	154	6	76322.04	85670.5	12.25
p32	154	6	78072.88	92215.9	18.12

Table 2.3: Computational results of the ALNS algorithm with heuristic solutions for set 1.

Instance	ALP	HDH	ALNS	Time	ALP-ALNS Gap	HDH-ALNS Gap
p01	531.02	524.61	**524.61**	86	-1.21	0.00
p02	1324.74	1332.01	1375.73	164	3.85	3.28
p03	537.37	528.97	**524.61**	104	-2.37	-0.82
p04	845.97	847.48	875.693	381	3.51	3.33
p05	2043.75	2059.74	2187.19	302	7.02	6.19
p06	840.1	884.69	885.24	TL	5.37	0.06
p07	829.65	829.92	864.648	528	4.22	4.18
p08	2052.21	2058.36	2165.12	TL	5.50	5.19
p09	829.65	834.92	845.614	TL	1.92	1.28
p10	1621.21	1629.76	1695.5	TL	4.58	4.03
p12	1230.95	1258.46	1333.87	TL	8.36	5.99
p14	954.81	954.81	**954.81**	12	0.00	0.00
p15	1862.63	1862.63	**1862.63**	14	0.00	0.00
p16	2875.24	2875.24	**2875.24**	28	0.00	0.00
p17	1597.75	1601.75	**1597.75**	24	0.00	-0.25
p18	3157	3147.91	3187.21	61	0.96	1.25
p19	4846.49	4851.41	**4846.49**	565	0.00	-0.10
p20	8412.02	8367.4	**8367.4**	467	-0.53	0.00
p21	2173.58	2180.33	2196.58	188	1.06	0.75
p22	4330.59	4218.46	4447.13	TL	2.69	5.42
p23	6813.45	6644.93	6878	TL	0.95	3.51
p24	3702.02	3704.6	3718.08	297	0.43	0.36
p25	3781.38	3781.38	3793.09	208	0.31	0.31
p26	3795.33	3795.32	3803.32	255	0.21	0.21
p27	22561.33	22153.31	23447.2	TL	3.93	5.84
p28	22562.44	22418.52	23374.6	TL	3.60	4.26
p29	23752.15	22864.23	24949.3	TL	5.04	9.12
p30	76793.99	75579.23	87797	TL	14.33	16.17
p31	77944.79	77459.14	85670.5	TL	9.91	10.60
p32	81055.52	79487.97	92215.9	TL	13.77	16.01

Table 2.4: Computational results of the ALNS algorithm for PMCVRP

			Set 2		Set 3		Set 4	
Instance	Nodes	Periods	Cost	Time	Cost	Time	Cost	Time
p01	52	2	650.80	66	652.01	205	693.60	280
p02	51	5	1937.36	74	1983.86	197	2321.00	TL
p03	51	5	605.46	170	601.81	255	675.38	TL
p04	76	2	889.61	271	927.77	441	956.93	575
p05	76	5	2954.10	TL	3208.59	566	3367.62	339
p06	76	10	918.14	TL	989.96	TL	1027.87	TL
p07	101	2	974.63	533	1060.07	TL	1110.95	TL
p08	101	5	2774.55	453	3709.93	TL	3913.81	TL
p09	101	8	965.59	TL	1080.82	TL	1144.92	TL
p10	101	5	2115.61	TL	2503.51	TL	2534.07	TL
p12	163	5	1325.68	TL	1359.38	TL	1378.04	TL
p14	21	4	1300.01	8	1486.15	25	1183.63	36
p15	39	4	2588.81	23	3187.62	101	3423.40	20
p16	57	4	4195.64	82	5107.05	277	5612.33	242
p17	41	4	2012.11	156	2391.30	44	2567.52	251
p18	77	4	4409.39	569	5280.57	TL	5254.07	TL
p19	113	4	7915.34	TL	8295.93	TL	8317.09	TL
p20	184	4	14275.10	TL	16702.10	607	17637.50	TL
p21	61	4	2724.40	385	3000.18	274	3215.25	TL
p22	115	4	5905.03	TL	7057.37	TL	6722.21	TL
p23	168	4	9588.28	TL	11296.20	TL	12095.50	TL
p24	52	6	6924.36	183	8096.52	475	7672.21	TL
p25	52	6	7174.42	403	8307.49	TL	9034.68	495
p26	52	6	6225.52	398	8966.20	413	9151.23	TL
p27	103	6	41965.90	TL	53925.20	TL	57887.70	TL
p28	103	6	40892.30	TL	54444.80	TL	64774.80	TL
p29	103	6	45438.30	TL	54847.70	TL	58793.20	TL
p30	154	6	164643.00	TL	178645.00	TL	208321.00	TL
p31	154	6	162192.00	TL	213454.00	TL	227264.00	TL
p32	154	6	179633.00	TL	196533.00	TL	225364.00	TL

Chapter 3

A Heuristic Branch–and–Price Algorithm for the Multi–Compartment Inventory Routing Problem

History: *This paper was prepared in collaboration with Claudia Archetti and Sanne Wøhlk. The research was conducted from the winter of 2016 to the Summer of 2016. Most of the work has been done during a research visit at Department of Economics and Management, University of Brescia. The paper is expected to be submitted in December 2016. Alternative datasets will be prepared for testing the solution approach before submission.*

A Heuristic Branch–and–Price Algorithm for the Multi–Compartment Inventory Routing Problem

Samira Mirzaei[†], Claudia Archetti[*], Sanne Wøhlk[†],

[†]Department of Economics and Business Economics, Aarhus University, Denmark,
{mirzaei, sanw}@econ.au.dk
[*]Department of Economics and Management, University of Brescia, Brescia, Italy,
claudia.archetti@unibs.it

Abstract

In this paper, we investigate a Multi-Compartment Inventory Routing Problem in a setting where the commodities are kept separated both in the inventories and in the vehicles. Each customer often requires repeated delivery of multiple commodities over a time horizon and may receive them from different vehicles. However, based on the delivery plan of each commodity in each period, the full amount of the commodity must still be delivered by a single vehicle. We propose two mathematical formulations for this problem and we develop a heuristic Branch-and-Price algorithm to find good solutions for this class of problem. Computational experiments show that the proposed heuristic Branch-and-Price algorithm can obtain high quality solutions within a reasonable amount of time for most of the test instances.

3.1 Introduction

Inventory Routing Problems (IRP) are of special interest because they integrate the transportation activities and the inventory management along the supply chain and thus help eliminate the inefficiency caused by solving the underlying routing and inventory management sub-problems separately. IRP has been studied for many years by different researchers, and the interested reader can refer to comprehensive surveys on the topic by Andersson et al. (2010) and Coelho et al. (2013).

Despite the vast body of literature on the IRP, little attention has been paid to supply chains such as petroleum, waste collection, and grocery distribution where customers request delivery of multiple commodities, and these commodities need to be keep separated during transport and storage. We consider such a Multi-Compartment Inventory Routing Problem (MCIRP) and assume that there is dedicated storage for each commodity at the supplier's and customers' locations as well as in the vehicles. Each customer often requires repeated delivery of multiple commodities over a time horizon and may receive them from different vehicles. However, based on the delivery plan of each commodity in each period, the full amount of the commodity must still be delivered by a single vehicle, i.e. we do not allow for split delivery as regards

the single commodity. In line with the definition in Mirzaei and Wøhlk (2016), we refer to this problem as the C-Split MCIRP. We present two mathematical formulations for this problem and we develop a heuristic Branch-and-Price algorithm to find good solutions for this class of problem.

Speranza and Ukovich (1994) and Speranza and Ukovich (1996) study the problem of minimizing the transportation and inventory costs of frequent shipments of several commodities from a single source to a single destination. They consider four situations where the entire quantity of each commodity available at the time of shipment has to be or cannot be shipped and where commodities with different frequencies may or may not share the same vehicle. Speranza and Ukovich (1996) present a mixed integer programming model for the problem and apply a Branch-and-Bound algorithm for solving it. Bertazzi et al. (1997) investigate an extension of this problem, considering one source and several destinations. They present heuristics for solving this problem.

Carter et al. (1996) investigate an inventory allocation-routing problem for grocery distribution. They present a heuristic algorithm which iteratively solves an allocation problem and a vehicle routing problem with time windows.

Sindhuchao et al. (2005) consider a collection system with a central warehouse and a set of suppliers that produce one or more commodities, and assume that the commodities are replenished according to an economic order quantity policy. They present a mathematical model for this problem and obtain a lower bound by using a column generation approach. They solve small instances to optimality with a Branch-and-Price algorithm and present a greedy constructive heuristic and a very large scale neighborhood search algorithm for this problem.

Moin et al. (2011) investigate a distribution network with an assembly plant and many suppliers in which each supplier supplies a distinct commodity and the commodities are unseparated during transport. They present a mathematical model for the problem from which they obtain a lower and upper bound. They also propose a heuristic approach based on a hybrid genetic algorithm for this problem. Mjirda et al. (2012), and Mjirda et al. (2014) investigate the same problem as Moin et al. (2011) and present a two-phase variable neighborhood search (VNS) metaheuristic to solve this problem. In the first phase, a VNS is used for construction of routes without considering the inventory. In the second phase, they improve the initial solution with a variable neighborhood descent and VNS.

Ramkumar et al. (2012) investigate Multi-Commodity IRP in a multi-depot setting and propose a mixed integer linear program for this problem. They can solve instances with at most two vehicles, one commodity, one supplier, three customers, and three periods to optimality.

Coelho and Laporte (2013) investigate a Multi-Commodity IRP in a multi-vehicles setting. They propose a mixed integer linear programming formulation for this problem and present a Branch-and-Cut algorithm for solving it.

Cordeau et al. (2015) study a Multi-Commodity IRP with dedicated storage for each commodity at the customers' locations but unseparated commodities in the vehicle. They present a three-phase decomposition-based heuristic for solving this problem. In the first phase, replenishment plans are specified by using a

Lagrangian-based method. In the second phase, routes are constructed by a simple heuristic. In the last phase, a feedback model is applied for improvement of this solution. Their heuristic approach is efficient for instances with up to 50 customers and 5 commodities.

MCIRP has recently been studied in the practical setting of fuel distribution. Here it is assumed that vehicles are often not equipped with debit meters and the content of a compartment cannot be split between different tanks. Normally, the number of tanks visited by each truck on any given route will not exceed three for this type of problem. Popović et al. (2012) present a mathematical formulation for this problem and develop a VNS heuristic for solving it. Vidović et al. (2014) investigate this problem with assumptions similar to those of Popović et al. (2012). They present a heuristic approach based on a constructive approach and two variable neighborhood descent searches and investigate the impact of the fleet-size cost on the solutions obtained. Coelho and Laporte (2015) describe four categories of this problem. They present two mathematical models and propose a Branch-and-Cut algorithm applicable for all of these categories. They can solve instances with up to 20 customers, three commodities, five compartments, eight vehicles, and five periods.

In all of the Multi-Commodity IRP papers found in the literature, it is assumed that the vehicles have a maximum capacity which is shared among all commodities. In this paper, we study the problem in a setting where the commodities are kept separated in the inventories as well as in the vehicles. This gives us a multi-compartment problem. To distinguish the problem from the Multi-Commodity IRP, we refer to this problem as a Multi-Compartment IRP (MCIRP). The problem consists in deciding, for each time period, the quantity of each commodity to deliver to each customer and in planning the routes of the vehicles in such a way that the sum of inventory and transportation costs is minimized.

The rest of this paper is organized as follows. In Section 3.2, we formally define the MCIRP and present a mathematical model based on flow constraints for this problem. We also propose a set partitioning formulation for the problem in Section 3.3. The heuristic Branch-and-Price algorithm is described in Section 3.4, followed by the results of our computational experiments in Section 3.5. Conclusions and some directions for future research are given in Section 3.6.

Many assumptions of the present paper are aligned with Desaulniers et al. (2015) and Archetti et al. (2014). The data used in this paper is derived from that used in those two papers.

3.2 Problem Definition

In this section, we describe the C-Split Multi-Compartment Inventory Routing Problem in detail and provide a mathematical model for the problem. To assist the reader we define our symmetric problem on a directed complete graph $G(N,E)$ with node set $N = \{0,1,\ldots,n\}$, where node zero represents the supplier and $N' = \{1,\ldots,n\}$ is the set of n customers. Each arc (i,j) in $E = \{(i,j) : i,j \in N\}$ represents the possibility to travel from node $i \in N$ to node $j \in N$ at non-negative travel cost c_{ij}.

Table 3.1: Overview of Notation

Index Sets and Indices	
K	The set of vehicles.
E	The set of undirected arcs, $E = \{(i,j) : i < j, i,j \in N\}$.
N'	The set of customers, $N' = \{1,2,\ldots,n\}$.
N	The set of all locations, $N = N' \cup \{0\}$, where location 0 is the supplier.
M	The set of commodities, $M = \{1,2,\ldots,\lvert M\rvert\}$.
M_i	The set of commodities demanded by customer, $i \in N'$, $M_i \subseteq M$.
	The set of all commodities available at the supplier, $M_0 = M$.
T	The set of time periods, $T = \{0,1,2,\ldots,H\}$.
T'	The set of planning periods, $T' = T \setminus \{0\}$.
Scalars and Parameters	
H	The finite time horizon.
$\hat{I}^0_{0m}$ ($\hat{I}^0_{im}$)	Starting inventory of commodity $m \in M$ at the supplier (in customer $i \in N'$).
Q_m	The capacity of compartment $m \in M$ of each vehicle.
U_{im}	Capacity limit for storage of commodity $m \in M$ at customer $i \in N'$.
c_{ij}	The nonnegative travel cost between locations i and j, $j \in N$.
d^t_{im}	Demand for commodity $m \in M$ at customer $i \in N'$ at period $t \in T'$.
h_{0m} (h_{im})	Unit holding cost for commodity $m \in M$ at the supplier (at customer $i \in N'$).
n	The number of customers.
p^t_m	Fixed production level at the supplier for commodity $m \in M$ in period $t \in T'$.

We consider a finite and discrete time horizon $T = \{0,1,2,\ldots,H\}$, where $t = 0$ represents the starting state and the remaining periods constitute the planning period $T' = T \setminus \{0\}$.

The supplier produces and stores a set of commodities, $M = \{1,2,\ldots,|M|\}$. We use $\hat{I}^0_{0m}$ to indicate the known starting ($t = 0$) inventory level of commodity $m \in M$ at the supplier. The supplier produces a fixed amount of p^t_m units of each commodity $m \in M$ in each time period $t \in T'$ and a unit holding cost h_{0m} is charged for commodity $m \in M$ at the end of every time period $t \in T'$. There is no upper limit on the supplier inventory level.

We use M_i to denote the set of commodities demanded by customer $i \in N'$ over the planning horizon. To ease of notation, we also define $M_0 = M$ for the supplier. Each customer i has a dedicated storage location with limited capacity U_{im} for each commodity $m \in M_i$. We use $\hat{I}^0_{im}$ to denote the known initial ($t = 0$) inventory of commodity $m \in M_i$ at customer $i \in N'$ and let $d^t_{im} \leq U_{im}$ be the known demand of customer i for commodity $m \in M_i$ in period $t \in T'$. Let h_{im} be the unit holding cost of commodity $m \in M_i$ at customer $i \in N'$. Note that in cases where the holding cost is lower at the customer than at the supplier, it would be beneficial to push as much inventory to the customer as possible. However, in this research we do not allow inventory to build up at the end of the planning horizon. Therefore we require that $I^H_{im} = max\{0; I^0_{im} - \sum_{t \in T'} d^t_{im}\}, \forall i \in N', and\ m \in M_i$. In other words, we allow deliveries that are needed at a later time in T, but do not allow pushing goods to a customer that are not needed during the planning

horizon.

The commodities are delivered to the customers by a homogeneous fleet K of vehicles. Each vehicle has $|M|$ compartments and compartment m of the vehicle is dedicated to commodity m and has a known capacity Q_m with $d^t_{im} \leq Q_m, \forall i \in N', m \in M_i, t \in T'$. Each vehicle can perform a single route in every time period $t \in T'$. Any vehicle can deliver any commodity to any customer and n any quantities respecting the capacities of the vehicles and bounds on inventory levels may be delivered as long as they are needed by the customer within the time horizon. We allow a C-Split strategy where a customer may be serviced by multiple vehicles in the same time period, but each commodity may only be delivered by a single vehicle in each time period.

The aim of the problem is to determine the replenishment plan that minimizes total inventory and routing costs in such a way that out-of-stock situations never occur, neither for the customers nor for the supplier. To clarify the sequence of events, Figure 3.1 provides an example with two planning periods, i.e. $T = \{0,1,2\}$, where everything above the line are actions of the supplier, and everything below the line refers to the customers. Table 3.1 provides an overview of the notation.

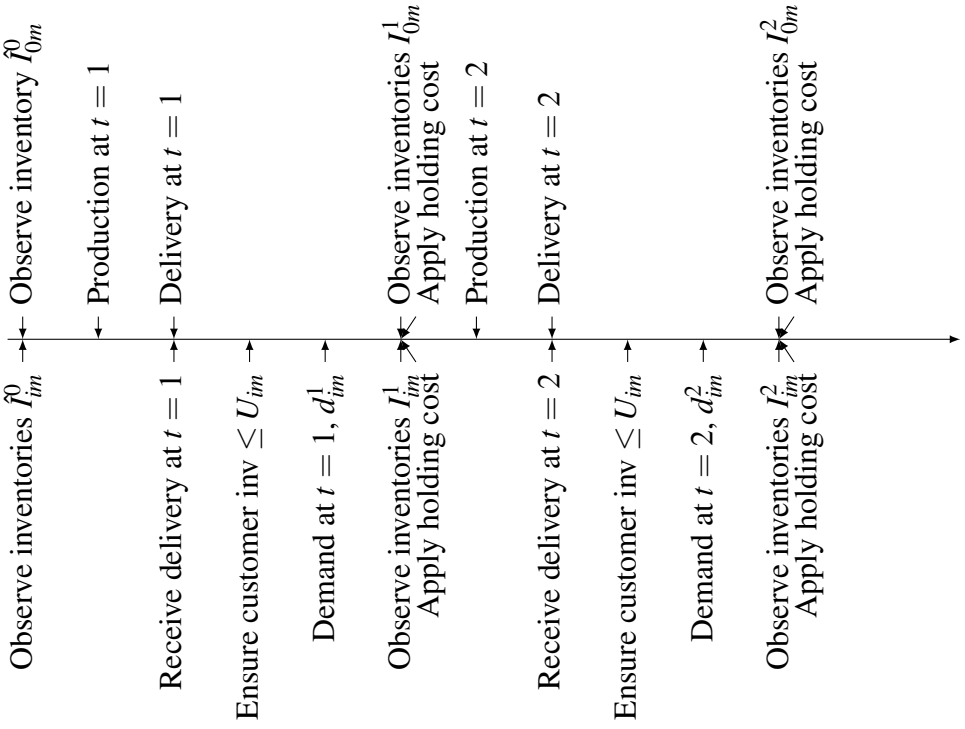

Figure 3.1: Timeline of example with two periods.

In order to formulate the problem mathematically, we define a number of variables. First, to define the routes of the vehicles, we define

$$x^{kt}_{ij} = \begin{array}{l} \text{number of times the arc } (i,j) \in E \text{ is traversed by the vehicle} \\ k \in K \text{ in period } t \in T'. \end{array}$$

Second, to model the deliveries to the customers, we define the following variables

$$y_{im}^{kt} = \begin{cases} 1 & \text{if commodity } m \in M_i \text{ is delivered to customer } i \in N' \\ & \text{by vehicle } k \in K \text{ in period } t \in T' \\ 0 & \text{otherwise.} \end{cases}$$

We define a set of variables to indicate the vehicles that are used to perform deliveries.

$$z^{kt} = \begin{cases} 1 & \text{if vehicle } k \in K \text{ performs a route at time } t \in T' \\ 0 & \text{otherwise.} \end{cases}$$

To simplify notation, we define the delivery quantities q_{im}^{kt} for all commodities and fix $q_{im}^{kt} = 0$ for $i \notin M_i$, i.e. for commodities not requested by the customer. Hence, we define

$$q_{im}^{kt} = \begin{array}{l} \text{quantity of commodity } m \in M \text{ delivered to customer } i \in N' \\ \text{by vehicle } k \in K \text{ in time period } t \in T'. \end{array}$$

Finally, to keep track of the inventory levels, we define

$$I_{im}^{t} = \begin{array}{l} \text{inventory level of commodity } m \in M_i \text{ at customer } i \in N' \\ \text{at the end of time period } t \in T. \end{array}$$

$$I_{0m}^{t} = \begin{array}{l} \text{inventory level of commodity } m \in M \text{ at the supplier} \\ \text{at the end of time period } t \in T. \end{array}$$

Note that the inventory variables are defined for all time periods, $t \in T$, but at the starting time, $(t = 0)$, we have $I_{im}^{0} = \hat{I}_{im}^{0} \, \forall i \in N$.

The C-Split MCIRP can be formulated as follows:

$$\min \quad \sum_{i\in N}\sum_{m\in M_i}\sum_{t\in T'} h_{im} I^t_{im} + \sum_{(i,j)\in E}\sum_{k\in K}\sum_{t\in T'} c_{ij} x^{kt}_{ij}$$

$$\text{st} \quad I^0_{im} = \hat{I}^0_{im} \quad \forall i \in N, m \in M_i \tag{3.1}$$

$$I^t_{0m} = I^{t-1}_{0m} + p^t_m - \sum_{i\in N'}\sum_{k\in K} q^{kt}_{im} \quad \forall m \in M, t \in T' \tag{3.2}$$

$$I^t_{im} = I^{t-1}_{im} + \sum_{k\in K} q^{kt}_{im} - d^t_{im} \quad \forall i \in N', m \in M_i, t \in T' \tag{3.3}$$

$$I^t_{im} \geq 0 \quad \forall i \in N, m \in M_i, t \in T \tag{3.4}$$

$$\sum_{k\in K} q^{kt}_{im} \leq U_{im} - I^{t-1}_{im} \quad \forall i \in N', m \in M_i, t \in T' \tag{3.5}$$

$$I^H_{im} = \max\{0; I^0_{im} - \sum_{t\in T'} d^t_{im}\} \quad \forall i \in N', m \in M_i \tag{3.6}$$

$$q^{kt}_{im} \leq U_{im} y^{kt}_{im} \quad \forall i \in N', m \in M_i, k \in K, t \in T' \tag{3.7}$$

$$\sum_{i\in N'} q^{kt}_{im} \leq Q_m z^{kt} \quad \forall m \in M, k \in K, t \in T \tag{3.8}$$

$$\sum_{k\in K} y^{kt}_{im} \leq 1 \quad \forall i \in N', m \in M_i, t \in T' \tag{3.9}$$

$$\sum_{j\in N} x^{kt}_{ij} = \sum_{j\in N} x^{kt}_{ji} \quad \forall i \in N, m \in M_i, k \in K, t \in T' \tag{3.10}$$

$$\sum_{j\in N'} x^{kt}_{0j} = z^{kt} \quad \forall k \in K, t \in T' \tag{3.11}$$

$$\sum_{j\in N} x^{kt}_{ij} \geq y^{kt}_{im} \quad \forall i \in N, m \in M_i, k \in K, t \in T' \tag{3.12}$$

$$\sum_{(i,j)\in E: i,j\in S} x^{kt}_{ij} \leq |S| - 1 \quad \forall k \in K, t \in T', S \subseteq N', |S| \geq 2 \tag{3.13}$$

$$q^{kt}_{im} \geq 0 \quad \forall m \in M_i, i \in N', k \in K, t \in T' \tag{3.14}$$

$$y^{kt}_{im} \in \{0,1\} \quad \forall i \in N', m \in M_i, k \in K, t \in T' \tag{3.15}$$

$$z^{kt} \in \{0,1\} \quad \forall k \in K, t \in T' \tag{3.16}$$

$$x^{kt}_{ij} \in \{0,1\} \quad \forall (i,j) \in E, k \in K, t \in T' \tag{3.17}$$

In the minimized objective function, the first term is the sum of the inventory holding costs for the supplier and the customers, and the second term is the transportation costs. Constraints (3.1)-(3.6) constitute the inventory constraints. Here, constraints (3.1) fix the starting inventories for both the supplier and the customers whereas (3.2) and (3.3) are inventory balance constraints for the supplier and the customers, respectively. Constraints (3.4) and (3.5) impose lower (equal to zero) and upper bounds on the inventory levels for each commodity at the customers and lower bounds at the supplier. Finally, (3.6) are ensuring that inventory are not building up at the customers in excess of what is necessary in order to satisfy the demand.

Constraints (3.7)-(3.9) define the replenishment plan. Constraints (3.7) ensure that a positive quantity of commodities can only be delivered to a customer i by a vehicle k if the corresponding y^{kt}_{im} variable is one. Constraints (3.8) ensure that compartment capacities are respected and that they are only used if the vehicle

is used, and constraints (3.9) specify that each commodity is brought to a customer at most once in each time period.

The routing of the vehicles is defined by constraints (3.10)-(3.13). Here, (3.10) ensure the continuity of each route and (3.11) ensure that when a vehicle is used (right hand side equals one), then that vehicle also leaves the depot. The purpose of (3.12) is to ensure that if a vehicle services any demand in a node (right hand side equals one), then that vehicle also visits that node. Constraints (3.13) are the classical sub-tour elimination constraints. Finally, (3.14)-(3.17) define the domain of the variables.

3.3 Set Partitioning Model

Desaulniers et al. (2015) present a set partitioning model for the IRP which is used for exact optimization of that problem. In their model, each variable corresponds to a route and an associated Route Delivery Pattern (*RDP*) (defined below). We use the model of Desaulniers et al. (2015) as inspiration for our model for the MCIRP. In our model, we only allow *RDP*s with full sub-deliveries (to be defined below) to be generated, but in the master problem, convex combinations of these can be used. This is similar to the approach used by Desaulniers et al. (2015), where so-called *extreme RDP*s are generated. As we mentioned before, in this research the delivery amount in each period is based on demand of the customers in current period and next periods and the capacity limit of the customers, and even when storing more inventory in the customers location is beneficial and there is space in the customers storage to deliver more than their demand in the following periods, we do not deliver more.

To model the problem, let Ω be the set of all feasible routes. For each route $r \in \Omega$, we define $N_r \subseteq N'$ to be the set of customers visited by the route and $E_r \subset E$ to be the set of arcs used in the route. The transportation cost of the route is thereby given by

$$c_r^{\text{trans}} = \sum_{(i,j)\in E_r} c_{ij}$$

Each route can be executed at any time $t \in T'$, which explains the lack of time index for the set of routes. However, there are multiple options as regards the amount of the commodities to be delivered to each customer and the possible options depend on the time. We therefore associate a set of Route Delivery Patterns (*RDP*) W_r^t with each route $r \in \Omega$ and each period $t \in T'$. Such a *RDP* $w \in W_r^t$ gives information about the amount of each commodity to be delivered to each customer, but also about which parts of this delivery should cover the demand in the following periods.

We formalize this in the following. Consider the delivery of each commodity $i \in M_i$ to a customer $i \in N'$ and assume that the delivered quantity of each commodity is consumed on the basis of the FIFO rule. Given this rule, the remaining quantity of the initial inventory of commodity m at customer i by the end of period s is $\bar{I}_{im}^{0,s} = \max\{0, I_{im}^0 - \sum_{l=1}^{s} d_{im}^l\}$. Therefore, the residual demand (i.e. the demand beyond what is available from the initial inventory) of customer i for commodity m corresponds to

$$\bar{d}^s_{im} = \begin{cases} \max\{0, d^1_{im} - I^0_{im}\} & if\ s = 1, \\ \max\{0, d^s_{im} - \bar{I}^{0,s-1}_{im}\} & if\ s > 1 \end{cases} \qquad \forall i \in N', m \in M_i, s \in T'$$

The quantity delivered at time $t \in T'$ can be seen as a number of sub-deliveries, each satisfying the demand that the customer is facing at time $s \geq t$. We can specify the set of periods related to these sub-deliveries of the commodity $m \in M_i$ to customer $i \in N'$ in period $t \in T'$ as

$$P^+_{imt} = \{s \in \{t, t+1, \ldots, H\} \mid (\bar{d}^s_{im} > 0) \wedge (s = t \vee \sum_{l=t}^{s} d^l_{im} \leq U_{im})\}.$$

Furthermore, we define $P^-_{ims} = \{t \in T' \mid s \in P^+_{imt}\}$ as the set of periods at which a sub-delivery can be made to fulfill the demand of customer $i \in N'$ for commodity $m \in M_i$ in period $s \in T'$.

Each *RDP* $w \in W^t_r$ specifies the quantity $q^s_{imw} \in \{0, \bar{d}^s_{im}\}$ of commodity $m \in M_i$ delivered to the customer $i \in N_r$ in period $t \in T'$ to satisfy the demand of period $s \in P^+_{imt}$. Because we only allow full sub-deliveries in each period, we have $q^s_{imw} \in \{0, \bar{d}^s_{im}\}$. Due to the FIFO property, the only allowed patterns for $w \in W^t_r$ are those with a number of $q^s_{imw} = 0$ (possibly zero), followed by at least one $q^s_{imw} = \bar{d}^s_{im}$, followed by a number of $q^s_{imw} = 0$ (possibly zero).

Given a route $r \in \Omega$, and a *RDP* $w \in W^t_r$, the total amount of commodity $m \in M_i$ delivered to customer $i \in N_r$ is given by

$$q_{imw} = \sum_{s \in P^+_{imt}} q^s_{imw} \tag{3.18}$$

and the total amount of the commodity loaded at the supplier is $q_{mw} = \sum_{i \in N_r} q_{imw}$.

The amount of $m \in M_i$ delivered to $i \in N_r$ at time $t \in T'$ according to $w \in W^t_r$ that will still be in the inventory at the end of period $s \in P^+_{it}$ is given as

$$b^s_{imw} = q_{imw} - \sum_{l=t}^{s} q^l_{imw} \tag{3.19}$$

With this, the total holding cost incurred by *RDP* $w \in W^t_r$ is given as

$$c^{\text{hold}}_w = \sum_{i \in N_r} \sum_{m \in M_i} \sum_{s \in P^+_{imt}} h_{im} b^s_{imw}$$

Therefore, the total cost associated with route $r \in \Omega$ and *RDP* $w \in W^t_r$ is

$$c_{rw} = c^{\text{trans}}_r + c^{\text{hold}}_w = \sum_{(i,j) \in E_r} c_{ij} + \sum_{i \in N_r} \sum_{m \in M_i} \sum_{s \in P^+_{imt}} h_{im} b^s_{imw} \tag{3.20}$$

Note that time t is implicitly given by the *RDP*.

3.3.1 Master Problem

In order to model the problem as a set partitioning problem, we keep all constraints related to the inventories at the supplier and the customers in the master problem, whereas all constraints related to the routes and deliveries are handled in the sub-problem.

For each route $r \in \Omega$, each customer $i \in N'$, and each commodity $m \in M$, we define the parameter a^r_{im} to have the value 1 if route $r \in \Omega$ delivers commodity $m \in M$ to customer $i \in N'$, and zero otherwise. This parameter is used in the model below to ensure that each commodity is only delivered to a customer by one vehicle in each time period.

The model uses two types of variables. Firstly, the I^t_{im} variables defined in Section 3.2 are used to define the inventory level both at the supplier and at the customers. Secondly, for every *RDP* $w \in W^t_r$ of every route $r \in \Omega$ and every time period $t \in T'$, we define continuous route variables with values in $[0,1]$ as

$$\lambda^t_{rw} = \begin{array}{l}\text{The fraction of route } r \in \Omega \text{ that is performed using } \textit{RDP } w \in W^t_r \\ \text{at time } t \in T'\end{array}$$

Then problem can now be formulated as follows.

$$\begin{aligned}
\min \quad & \sum_{m\in M}\sum_{t\in T'} h_{0m} I^t_{0m} + \sum_{r\in\Omega}\sum_{t\in T}\sum_{w\in W^t_r} c_{rw}\lambda^t_{rw} \\
\text{st} \quad & I^0_{0m} = \hat{I}^0_{0m} \quad \forall m \in M && (3.21)\\
& I^t_{0m} = I^{t-1}_{0m} + p^t_m - \sum_{r\in\Omega}\sum_{w\in W^t_r} q^t_{mw}\lambda^t_{rw} \quad \forall m \in M, t \in T' && (3.22)\\
& I^t_{0m} \geq 0 \quad \forall m \in M, t \in T' && (3.23)\\
& \sum_{t\in P^-_{ims}}\sum_{r\in\Omega}\sum_{w\in W^t_r} q^s_{imw}\lambda^t_{rw} = \bar{d}^s_{im} \quad \forall i \in N', m \in M_i, s \in T && (3.24)\\
& \bar{I}^{0,s}_{im} + \sum_{t\in P^-_{ims}}\sum_{r\in\Omega}\sum_{w\in W^t_r} b^s_{imw}\lambda^t_{rw} \leq U_{im} - d^s_{im} \quad \forall i \in N', m \in M_i, s \in T && (3.25)\\
& \sum_{r\in\Omega}\sum_{w\in W^t_r} a^r_{im}\lambda^t_{rw} \leq 1 \quad \forall i \in N', m \in M_i, t \in T && (3.26)\\
& \sum_{r\in\Omega}\sum_{w\in W^t_r} \lambda^t_{rw} \leq |K| \quad \forall t \in T && (3.27)\\
& \lambda^t_{rw} \geq 0 \quad \forall r \in \Omega, t \in T, w \in W^t_r && (3.28)\\
& \sum_{w\in W^t_r} \lambda^t_{rw} \in \{0,1\} \quad \forall r \in \Omega, t \in T && (3.29)
\end{aligned}$$

In the objective function, the first term is the holding cost of the supplier and the second term represents the transportation cost of the routes as well as the holding cost at the customers.

Constraints (3.21)-(3.23) constitute the inventory constraints at the supplier. Here, (3.21) fix the starting inventory, and (3.23) prevent stockouts. (3.22) are the inventory balance constraints, where the last term is the total withdrawal from the inventory at time t and the values of q^t_{mw} are calculated by formula (3.18).

By considering all routes in all periods in P^-_{ims}, constraints (3.24) ensure that the total amount delivered to a customer to cover its demand at time s equals the customer's residual demand at that time. Constraints (3.25) consider the customers' inventories and ensure that in any time period, after the receipt of commodities but before the customers face their demands, the inventories do not exceed the capacity limits. Here, b^s_{imw} in the second term is calculated by formula (3.19) and refers to remaining inventory at the end of the time period, which explains why the demand of that period is subtracted at the right hand side.

Constraints (3.26), together with (3.29), impose that each commodity ordered by a customer is delivered by a single route, and constraints (3.27) ensure that at most $|K|$ vehicles are used in each time period.

Finally, (3.28) are non-negativity constraints for the route-*RDP*-variables and (3.29) ensure that only one route is performed by each vehicle at each time period. By summing over the *RDP*s of the time period, the constraint is imposed on the routes rather than on the *RDP*s. Therefore, this constraint leaves open the option of using convex combinations of the *RDP*s and thereby allows less-than-full *RDP*s in the final solution.

3.3.2 Sub-Problem

As is the case for any routing problem, the sub-problem resulting from the decomposition underlying the reformulation becomes an Elementary Resource Constrained Shortest Path Problem (ESPPRC). However, due to the inventory aspect of the present problem, this problem is combined with a knapsack problem, as is explained in Desaulniers et al. (2015), and we solve it with heuristic labeling algorithms. We now describe the approach we developed for the solution of the sub-problem, which is used in Section 3.4 for heuristic column generation.

Due to the timing aspect in the master problem, a sub-problem is created for each time period $t \in T'$. A feasible solution of the sub-problem for time t consists of a feasible route $r \in \Omega$ and a corresponding feasible *RDP* $w \in W_r^t$. In the master problem, this route $r \in \Omega$ and *RDP* $w \in W_r^t$ at time $t \in T'$ are associated with the variable λ_{rw}^t.

Let π_{mt}^1 be the dual variables of constraints (3.22) and let π_{ims}^2, π_{ims}^3, π_{imt}^4, and π_t^5 be the dual variables of constraints (3.24)-(3.27), respectively. The reduced cost of the variable λ_{rw}^t corresponds to the cost of the route $r \in \Omega$ at time $t \in T'$ and *RDP* $w \in W_r^t$. We will use the term *reduced cost* to describe this cost throughout this section to distinguish it from the costs described in the previous section. We have

$$\begin{aligned}
\bar{c}_{rw}^t \;=\; & c_{rw} + \sum_{m \in M} \pi_{mt}^1 q_{mw}^t - \sum_{i \in N'} \sum_{m \in M_i} \sum_{s \in P_{imt}^+} \pi_{ims}^2 q_{imw}^s \\
& - \sum_{i \in N'} \sum_{m \in M_i} \sum_{s \in P_{imt}^+} \pi_{ims}^3 b_{imw}^s - \sum_{i \in N'} \sum_{m \in M_i} \pi_{imt}^4 - \pi_t^5 \\
& - \sum_{i \in N'} \sum_{m \in M_i} \sum_{j \in N'} \sum_{m' \in M_j} \pi_{im\,jm'}^6 (\alpha_{im\,jm'}^r + \alpha_{jm'\,im}^r)
\end{aligned} \tag{3.30}$$

where the last term comes from constraint (3.34), which is added during the branching, with $\pi_{im\,jm'}^6$ being the dual variable of that constraint and the parameter $\alpha_{im\,jm'}^r$ being defined in Section 3.4.4.

For each time $t \in T'$, the goal of the sub-problem is to identify a negative reduced cost route r and its associated *RDP* w that the route starts and ends in the supplier node and visits a number of customers, and such that the *RDP* provides a feasible delivery plan for the visited customers where each sub-delivery is either full or zero.

The routing part of the sub-problem for period $t \in T'$ can be modeled as an ESPPRC on a directed graph $G^t = (N^t, E^t)$, where N^t and E^t are sets of nodes and arcs, respectively. The node set N^t contains a source node v^S and a sink node v^E which represent the depot at the start and the end of the routes. Furthermore, N^t contains a node v^i for each customer $i \in N'$. The arc set E^t includes all arcs (v^i, v^j) with $i, j \in N', i \neq j$, as well as arcs (v^S, v^i) and (v^i, v^E) $\forall i \in N'$. The costs associated with the arcs of G^t are the reduced cost $\bar{c}_{ij}$ of the routing part and are defined as follows:

$$\bar{c}_{v^i v^j} = \begin{cases} c_{ij} - \pi_t^5 & \text{if } v^i = v^S \\ c_{ij} - \pi_{imt}^4 & \text{otherwise} \end{cases}$$

Let ξ_{im}^s be variables determining the quantity of commodity $m \in M_i$ to be delivered to customer $i \in N'$ at time t to satisfy the demand of period $s \in P_{imt}^+$. Because the holding cost of route r with *RDP* w at time t can be rewritten as

$$c_w^{\text{hold}} = \sum_{i \in N_r} \sum_{m \in M_i} \sum_{s \in P_{imt}^+} \sum_{l=t}^{s-1} h_{im} q_{imw}^l$$

then the reduced cost in (3.30) can be rewritten as

$$\begin{aligned} \bar{c}_{rw}^t \quad = \quad & \sum_{(i,j) \in N', i \neq j} \bar{c}_{v^i v^j} + \sum_{i \in N'} \sum_{m \in M_i} \sum_{s \in P_{imt}^+} \xi_{im}^s (\pi_{mt}^1 - \pi_{ims}^2 + \sum_{l=t}^{s-1} (h_{im} - \pi_{iml}^3)) \\ & - \sum_{i \in N'} \sum_{m \in M_i} \sum_{j \in N'} \sum_{m' \in M_j} \pi_{im\,jm'}^6 (\alpha_{im\,jm'}^r + \alpha_{jm'\,im}^r) \end{aligned} \tag{3.31}$$

where the first term is related to the routing of the vehicles. In the second term, the first part regards the supplier and the deliveries from the supplier, and the last part is related to the inventories at the customers. Finally, the last term comes from the branching. This way of writing the reduced cost is more useful when solving the sub-problem in Section 3.4.

When a solution of the sub-problem is sent to the master problem as a column consisting of a route r and a *RDP* w for time t, the values of q_{imw}^s are set as $q_{imw}^s = \xi_{im}^s$, $\forall i \in N', i \in M_i$, and $s \in P_{imt}^+$.

3.4 Heuristic Branch-and-Price Algorithm

This section describes our heuristic Branch-and-Price algorithm in detail. Firstly in Section 3.4.1, we describe the initialization of the algorithm. In Section 3.4.2, we explain the overall strategy for solving a node in the branching tree. It includes three heuristic labeling algorithms which we describe in Section 3.4.3. We illustrate the branching method in Section 3.4.4, and our node selection strategies in Section 3.4.5.

3.4.1 Initialization

To ensure feasibility for each period $t \in T'$, we create a dummy column that represents a route which starts at the depot, services all customers and delivers all commodities requested by each customer at time t, and returns to the depot. For these routes, we assume that the capacity of each compartment of the vehicle is large enough to hold the demand of all the customers. Each dummy column has a large cost to ensure that it will not become part of the final solution. However, a feasible LP solution can always be obtained by setting the value of the dummy variables to one.

In addition to these dummy columns, we use a nearest neighbor heuristic for initialization of the master problem. At first, the delivery amount of each commodity $m \in M_i$ to each customer $i \in N'$ in each period $t \in T'$ of the planning horizon is set to the residual demand $\bar{d}^t_{im}$. If $\bar{d}^t_{im} = 0$, commodity m is not delivered to the customer i in period t. Next, routes are constructed by solving a CVRP for each period of the planning horizon through applying the nearest neighbor heuristic. Each route is started at the depot and repeatedly extended from its current point to the nearest unvisited customer subject to the capacity constraints of the vehicle compartments and the residual demand of the customer, where closeness is measured in terms of travel cost. The route extension continues until there is no more capacity for servicing customers or until all customers with positive residual demand in that period have been visited, at which point the vehicle returns to the depot. If there are unvisited customers or unplanned periods in the planning horizon, a new route is initialized.

As a result, the number of routes may exceed the number of available vehicles for one or more periods of the planning horizon. In that case, we first identify periods t and t' with higher and lower number of routes compared to the number of available vehicles, respectively. Then route r with the lowest number of customers in period t is determined and the service day of route r is changed from period t to t'. If there is no period with lower number of routes compared to the number of available vehicles, we simply add the routes to the master problem without changing their service day. In this case, instead of using the routes generated by the NNS for period t as basic variable, the algorithm will use the dummy column of period t and the solution is still feasible but with high total cost.

3.4.2 Node Solving Procedure

With the purposes of solving each node of the branching tree, we implement three heuristic labeling algorithms named No-Semi, Friends, and Dominance heuristics. These heuristics are described in Section 3.4.3. We can apply all of these heuristics for solving each node of branching tree, but in order to speed up our solution approach, we execute them as follows:

If a node in the branching tree is explored during Breadth-first search or in case of the root node, all heuristics are used for generating columns. At first, the No-Semi heuristic is used for each period of the planning horizon. If the No-Semi heuristic can produce new columns, these columns are added to the master problem, and the No-Semi heuristic is used again with new dual prices. Otherwise the Friends

heuristic is used for each period of the planning horizon and new columns are added to the master problem. If the Friends heuristic cannot produce any column, the Dominance heuristic is executed. If no heuristic can produce any negative reduced cost column for any period of the planning horizon, the stopping condition is met.

If a node in the branching tree is explored during depth-first search, the algorithm only applies the No-Semi heuristic. This heuristic is executed for each period of the planning horizon. If the No-Semi heuristic finds new columns, they are added to the master problem. If the No-Semi heuristic cannot produce any negative reduced cost column for any period of the planning horizon, the stopping condition is met.

At each iteration, the full set of non-dominated columns with elementary routes is denoted S. If $S \neq \emptyset$, we add up to 600 of these columns to the master problem. The columns are selected among those with the highest negative reduced costs. Initially, the master problem is solved and dual values for the constraints are determined. Then, the heuristic labeling algorithms are executed with these dual values, and new columns are added to the master problem. This procedure is repeated until there is no column with negative reduced cost for any period of the planning horizon.

3.4.3 Heuristic Labeling Algorithm

The three heuristic labeling algorithms have similar structure. Therefore, we describe the general structure of the heuristic labeling algorithms in Section 3.4.3.1, and in Section 3.4.3.2, we elaborate on each of them.

3.4.3.1 General Structure

To solve the sub-problem by means of the heuristic labeling algorithms, we duplicate the node associated with each customer i $|M_i|$ times, and we define a new set N^* to contain $v^{i_m}, \forall i \in N'$, and $\forall m \in M_i$ as well as nodes v^S and v^E to represent the depot. We define a new set $E^* = \{(v^{i_m}, v^{j_{m'}}) : v^{i_m}, v^{j_{m'}} \in N^*\}$ and denote the graph $G^*(N^*, E^*)$. As each of the duplicated nodes denote a customer-commodity pair, we refer to them as c-c nodes. For instance, c-c node v^{i_m} indicates that commodity m is delivered to customer v^i. We associate with each c-c node the demand of the customer for the corresponding commodity, the starting level of the inventory, and the maximum and the minimum levels of the inventory at the customer.

For each c-c node v^{i_m}, we can consider different delivery patterns. We only allow full and zero sub-deliveries for each period $t \in T'$, therefore $\xi^t_{v^{i_m}} \in \{0, \bar{d}^t_{v^{i_m}}\}$. $\xi^t_{v^{i_m}}$ and $\bar{d}^t_{v^{i_m}}$ are defined like ξ^t_{im} and $\bar{d}^t_{im}$ in Section 3.3. We generate all possible delivery patterns for c-c node v^{i_m} based on the description in Section 3.3 and store them in a list $\Gamma^t_{v^{i_m}}$. We represent each delivery pattern for c-c node v^{i_m} by a vector $\xi_{v^{i_m}} = [\xi^t_{v^{i_m}}]$ of size $|T'|$, stating the delivery amount to c-c node v^{i_m} to satisfy the demand of period t, $\forall t \in T'$. A RDP w is associated with label $L_{v^{i_m}}$ which contains $\xi_{v^{i_m}}$ for each visited c-c node. These lists can be created in a preprocessing phase.

In the heuristic labeling algorithms a label represents a partial route which starts at node v^S and extends in G^* while considering dominance rules, plus a RDP which provides a feasible delivery plan for the visited

customers. The algorithm uses a set $\mathscr{L}_{v^{i_m}}$ of labels for each c-c node $v^{i_m} \in N^*$ where each label $L_{v^{i_m}}$ in $\mathscr{L}_{v^{i_m}}$ represents a path starting at node v^S and ending in c-c node v^{i_m} as well as a RDP $w \in W_r^t$.

To denote a partial route r, label $L_{v^{i_m}}$ contains three pieces of information: A 'node' resource vector $cust_{v^{i_m}}$ of size $|N^*|$ where $cust_{v^{i_m}}^{v^{j_{m'}}}$ states the number of visits to c-c node $v^{j_{m'}} \in N^*$ along route r. A capacity resource vector $\sigma_{v^{i_{m'}}}$ of size $|M|$ which represents the capacity consumption of commodity m' along r determined by the delivery patterns. Finally, $cost_{v^{i_m}}$ is associated with label $L_{v^{i_m}}$ to denote the reduced cost of the corresponding route and RDP.

Suppose $L_{v^{i_m}} = (cust_{v^{i_m}}, \sigma_{v^{i_m}}, cost_{v^{i_m}})$ is a label associated with c-c node $v^{i_m} \in N^*$. Suppose $v^{j_{m'}}$ is a c-c node that represents delivery of commodity $m' \in M_j$ to customer $j \in N'$. A label $L_{v^{j_{m'}}} = (cust_{v^{j_{m'}}}, \sigma_{v^{j_{m'}}}, cost_{v^{j_{m'}}})$ is created by extending label $L_{v^{i_m}}$ along arc $(v^{i_m}, v^{j_{m'}})$. Label $L_{v^{i_m}}$ can be extended along arc $(v^{i_m}, v^{j_{m'}})$ as long as there are delivery patterns in the list $\Gamma^t_{v^{j_{m'}}}$. Let $\xi_{v^{j_{m'}}}$ be one of these delivery patterns, then label $L_{v^{j_{m'}}} = (cust_{v^{j_{m'}}}, \sigma_{v^{j_{m'}}}, cost_{v^{j_{m'}}})$ is created by considering this delivery pattern. Let

$$cost_{v^{j_{m'}}} = \begin{cases} cost_{v^{i_m}} + \bar{c}_{v^{i_m} v^{j_{m'}}} - \pi^6_{v^{i_m} v^{j_{m'}}} & \text{if } v^{j_{m'}} = v^E \\ cost_{v^{i_m}} + \bar{c}_{v^{i_m} v^{j_{m'}}} + \tau^{cost}_{\xi_{v^{j_{m'}}}} - \pi^6_{v^{i_m} v^{j_{m'}}} & \text{otherwise} \end{cases}$$

$$\sigma_{v^{j_{m'}}} = \sigma_{v^{i_m}} + \tau^{\sigma_{v^{j_{m'}}}}_{\xi_{v^{j_{m'}}}}, \ \forall m, m' \in M$$

$$cust^{v^{k_\mu}}_{v^{j_{m'}}} = \begin{cases} cust^{v^{k_\mu}}_{v^{i_m}} + 1, \text{ if } v^{j_{m'}} = v^{k_\mu} \\ \qquad \forall v^{k_\mu} \in N^* \\ cust^{v^{k_\mu}}_{v^{i_m}}, \quad \text{otherwise.} \end{cases}$$

Here $\tau^{cost}_{\xi_{v^{j_{m'}}}}$ is the contribution to the reduced cost due to the delivery of commodity m by delivery pattern $\xi_{v^{j_{m'}}}$ and $\tau^{\sigma_{v^{j_{m'}}}}_{\xi_{v^{j_{m'}}}}$ is the total capacity consumption of commodity $m' \in M$ for delivery pattern $\xi_{v^{j_{m'}}}$ and they are calculated as follows:

$$\tau^{cost}_{\xi_{v^{j_{m'}}}} = \sum_{s \in P^+_{jm's}} \xi^s_{v^{j_{m'}}} (\pi^1_{m't} - \pi^2_{v^{j_{m'}} s} + \sum_{l=t}^{s-1} (h_{v^{j_{m'}}} - \pi^3_{v^{j_{m'}} l})) \tag{3.32}$$

$$\tau^{\sigma_{v^{j_{m'}}}}_{\xi_{v^{j_{m'}}}} = \sum_{s \in P^+_{jm's}} \xi^s_{v^{j_{m'}}} \tag{3.33}$$

$h_{v^{j_{m'}}}$, $\pi^2_{v^{j_{m'}} s}$, $\pi^3_{v^{j_{m'}} l}$, and $\pi^6_{v^{i_m} v^{j_{m'}}}$ are defined as $h_{jm'}$, $\pi^2_{jm's}$, $\pi^3_{jm'l}$, and $\pi^6_{imjm'}$ in Sections 3.2, 3.3, and 3.4.4 respectively.

The label $L_{v^{j_{m'}}}$ is feasible if $\sigma_{v^{j_{m'}}} <= Q_{m'}$, and $cust_{v^{j_{m'}}} <= 1$, $\forall v^{k_\mu} \in N^*$.

To avoid enumerating all feasible labels, a dominance rule is applied to discard unpromising labels. A label $L_{v^{i_m}}$ dominates a label $L'_{v^{i_m}}$ in c-c node v^{i_m} if and only if the following conditions are satisfied:

(a)$\sigma_{v^{i_m}} \leq \sigma'_{v^{i_m}}$, $\forall m \in M$;

(b)$(cost_{v^{i_m}} \leq cost'_{v^{i_m}})$;

(c)$cust_{v^{i_m}} \leq cust'_{v^{i_m}}$, $\forall v^{k_\mu} \in N^*$.

With a view to handling the branching rules, T_n, F_n, T_e, and F_e matrices are introduced in Section 3.4.4. Before extending $L_{v^{i_m}}$ to $v^{j_{m'}}$, these matrices are checked to see if the value of T_n for c-c node $v^{j_{m'}}$ equals 1 or the value of T_e for the pair $(v^{i_m}, v^{j_{m'}})$ is 1 in which case label $L_{v^{i_m}}$ cannot be extended to $v^{j_{m'}}$. If the value of F_e for c-c node v^{i_m} is c-c node $v^{j_{m'}}$ or 0, label $L_{v^{i_m}}$ can extend to c-c node $v^{j_{m'}}$.

3.4.3.2 Heuristic Algorithms

When we limit sub-deliveries to zero or full in our heuristic labeling algorithm, we get a reduced number of labels, but when we increase the number of c-c nodes and/or the amount of compartment capacity, we witness an exponentially increasing number of labels and thereby running time. Therefore, exact column generation is not considered. Instead, we use three heuristic labeling algorithms which are constructed as variations of the general structure of the labeling algorithm described in Section 3.4.3.1.

Friends heuristic: This heuristic is based on only extending each label from each c-c node to a predefined number of its closest neighbors. In this heuristic, a subset of c-c nodes $N^*_{v^{i_m}} \in N^*$ is defined for each c-c node $v^{i_m} \in N^*$. $N^*_{v^{i_m}}$ including the η closest neighbors of c-c node v^{i_m}. Each label in c-c node v^{i_m} is only extended to a c-c node $v^{j_{m'}}$ if $v^{j_{m'}} \in N^*_{v^{i_m}}$. After computational testing, the initial value $\eta = 10$ was chosen.

No-Semi heuristic: This heuristic is based on the idea that each customer can only be serviced by a single vehicle in each period of the planning horizon, and the vehicle must deliver the full amount of all commodities ordered for this period by the customer. In line with the definition of Mirzaei and Wøhlk (2016), we refer to this as No-Split delivery. Therefore, a label $L_{v^{i_m}}$ can only extend to c-c node $v^{i_{m'}}$, which is a c-c node of the same customer for commodity m', with $m' \in M_i$, $m' \neq m$, $c_{v^{i_m} v^{i_{m'}}} = 0$, and $m' = m+1$. Furthermore, if node v^{i_m} is a c-c node related to the last commodity $m \in M_i, m = |M_i|$, each label in v^{i_m} can extend to all c-c nodes that are related to the first commodity of different customers. We also consider η closest neighbors for this heuristic.

Dominance heuristic: This is a heuristic labeling algorithm with modified dominance rules that compares labels by considering the remaining capacity of the compartments, the cost of the route, and the number of serviced c-c nodes. In other words, $L_{v^{i_m}}$ can dominate $L'_{v^{i_m}}$ if and only if $\sigma_{v^{i_m}} \leq \sigma'_{v^{i_m}}$, $cost_{v^{i_m}} \leq cost'_{v^{i_m}}$, and the number of c-c nodes in the route of $L_{v^{i_m}}$ is no more than the number of c-c nodes in the route of $L'_{v^{i_m}}$.

We also tested four additional heuristics, but they turned out to be less efficient than the heuristics above and we decided to discard them after tuning. The first heuristic was based on scaling of the capacity of the compartments. The second heuristic was based on extending each label from each node to its neighbors within a predefined distance. The third heuristic was similar to the Dominance heuristic, but with simpler dominance rules. This heuristic compares labels based on the remaining capacity of the compartments and the cost of the routes. The forth heuristic was based on duplicating columns made for period $t \in T'$ by our solution approach, for all periods of the planning horizon.

3.4.4 Branching

In our solution approach we apply two types of branching. The decision variables of the first branch type are based on the flow of commodity m through each c-c node v^{i_m} in each period t ($\sum_{r\in\Omega}\sum_{w\in W_r^t}\alpha_{im}^r\lambda_{rw}^t$). We select a c-c node v^{i_m} with different service periods and create two branches in the branching tree. In the left branch, c-c node v^{i_m} is required to be serviced in period t, and in the right branch, servicing this c-c node in period t is forbidden. To handle this branching rule, we define a binary *tabu matrix* T_n and a binary *fix matrix* F_n, each of size $H \times |N^*|$. If c-c node v^{i_m} should be serviced in period t, we set $F_n(t, v^{i_m}) = 1$, and 0 otherwise. If servicing c-c node v^{i_m} is forbidden in period t, we set $T_n(t, v^{i_m}) = 1$, and 0 otherwise. The procedure for selecting c-c node v^{i_m} and period t for this type of branching is as follows:

1. We select a route with a value closest to 0.5 in the LP-relaxation. Let t be the service day of this route.
2. We identify the first nonzero c-c node of the route and call it v^{i_m}. If $F_n(t, v^{i_m}) = 0$ and there is another route which services c-c node v^{i_m} in a different period, then v^{i_m} is chosen as the c-c node to branch on period t. Otherwise, we consider the next c-c node in the route and repeat this step until a suitable c-c node is found for branching.
3. If there is no suitable c-c node on the selected route, we select the route with a variable closest to 0.5 when excluding already considered routes, and we repeat step 2 based on this route.

The left branch of the first branch type is imposed by fixing the value of constraint (3.26) in the master problem to 1 for customer i, commodity m in period t, and the right branch is imposed by setting this constraint equal to 0. For efficient implementation of this branch type, T_n should be considered in the sub-problem.

The second branching is based on Ryan-Foster branching. The decision variables are based on the flow between c-c nodes v^{i_m} and $v^{j_{m'}}$ in each period $t \in T$, i.e. the flow on arcs $(v^{i_m}, v^{j_{m'}})$ and $(v^{j_{m'}}, v^{i_m})$. The algorithm selects a pair of c-c nodes v^{i_m} and $v^{j_{m'}}$ in period t and makes two branches in the branching tree. In the left branch, the connection of v^{i_m} and $v^{j_{m'}}$ in period t is fixed, and either $v^{j_{m'}}$ must be serviced immediately after v^{i_m}, or v^{i_m} must be serviced immediately after $v^{j_{m'}}$. In the right branch, this connection is forbidden, and $v^{j_{m'}}$ may not be serviced immediately after v^{i_m} in period t, and visa versa. This branching is sufficient for finding an integer solution to the problem.

To handle this branching rule, we define an $H \times |N^*| \times |N^*|$ binary *tabu matrix* T_e and a *fix matrix* F_e with size $H \times 2 \times |N^*|$. If a pair v^{i_m} and $v^{j_{m'}}$ should be serviced in period t, we set $F_e(t, 1, v^{i_m}) = v^{j_{m'}}$ or $F_e(t, 2, v^{i_m}) = v^{j_{m'}}$, and set $F_e(t, 1, v^{j_{m'}}) = v^{i_m}$ or $F_e(t, 2, v^{j_{m'}}) = v^{i_m}$. If the connection between v^{i_m} and $v^{j_{m'}}$ is forbidden, we set $T_e(t, v^{i_m}, v^{j_{m'}}) = 1$ and $T_e(t, v^{j_{m'}}, v^{i_m}) = 1$, and 0 otherwise. The pair v^{i_m} and $v^{j_{m'}}$ is chosen as follows:

1. Route with value closest to 0.5 is selected. Let t be the service day of this route.

2. We consider a pair of c-c nodes v^{i_m} and $v^{j_{m'}}$ of this route, and if at least one of the following conditions are met, we repeat this step with the next pair. Otherwise we go to step 3.

   a) If F_e has two fixed values for c-c node v^{i_m} or c-c node $v^{j_{m'}}$ in period t.

   b) If F_e has one fixed value for c-c node v^{i_m} or $v^{j_{m'}}$ in period t with either $F_e(t,1,v^{i_m}) = v^{j_{m'}}$, or $F_e(t,2,v^{i_m}) = v^{j_{m'}}$, or $F_e(t,1,v^{j_{m'}}) = v^{i_m}$, or $F_e(t,2,v^{j_{m'}}) = v^{i_m}$.

   c) If $T_e(t,v^{i_m},v^{j_{m'}}) = 1$ and $T_e(t,v^{j_{m'}},v^{i_m}) = 1$.

3. If there is another route that services c-c node v^{i_m} or $v^{j_{m'}}$ in period t and if both the predecessor and the successor of $v^{i_m}(v^{j_{m'}})$ are not $v^{j_{m'}}$ (v^{i_m}), then the pair v^{i_m} and $v^{j_{m'}}$ is chosen for branching. Otherwise, we consider the next pair and go to step 2.

4. If there is no pair of c-c nodes in the selected route suitable for branching, we consider the route with the value closest to 0.5 while excluding routes already considered, and go to step 2.

Let $\alpha^r_{imjm'}$ be a parameter that takes the value 1 if route $r \in \Omega$ contains arc (i,j) where m and m' are the commodities which are to be delivered to i and j, and zero otherwise. The second branch type is imposed by adding constraints (3.34) to the master problem and considering F_e, and T_e in the sub-problem:

$$\sum_{r\in\Omega}\sum_{w\in W^t_r}(\alpha^r_{imjm'}+\alpha^r_{jm'im})\lambda^t_{rw} \leq 1 \quad \forall v^{i_m},\ v^{j_{m'}} \in N^*,\ t \in T' \tag{3.34}$$

When a fractional LP-relaxation is obtained, we calculate the value of each candidate variable for both branch types, and we select the one with a fractional value closest to 0.5. If the two resulting fractional values are equal, we choose the first branch type.

3.4.5 Node Selection

Due to the limitations of the standard node selection algorithms, i.e. breadth-first or depth-first search, we explore the branching tree by using a combination of the two which combines the space-efficiency of depth-first search and the completeness of the breadth-first search.

The algorithm starts in a depth-first search fashion by exploring nodes along a left branch until it either reaches a node that has an integer solution or a node that has an LP-relaxation value that is worse than the current best integer solution or until the depth-first search has been applied for b_1 iterations. Next, a breadth-first search explores nodes based on the lowest value of the LP-relaxation of their parents. This is repeated for b_2 iterations. Then the algorithm proceeds with depth-first search and the process is repeated until there are no more nodes for branching or the time limit is reached. However, after 300 iterations, the algorithm shifts to pure Breadth-first search. After computational testing, we chose the values $b_1 = 2|N^*|$ and $b_2 = 0.5|N^*|$.

It should be noted that the value of the LP-relaxation may not be the optimal value because we do not apply exact column generation.

3.5 Computational Experiments

In this section, we present our computational results and analysis that form the basis for our conclusions. The algorithm is implemented in C++ in Microsoft Visual Studio 2010 with the use of IBM ILOG CPLEX 12.6.1 callable library. All tests were made on a laptop with a Pentium core i5 processor and a clock speed of 1.8 GHz and 8 GB of RAM.

Details about our test sets are presented in Section 3.5.1. We have used the IRP benchmark instances created by Archetti et al. (2014) to evaluate the performance of our algorithm. We have also generated a new set of test instances to evaluate our solution approach for the MCIRP. In total, we have tested our algorithm on 210 instances.

Our results are structured as follows. In Section 3.5.2, we analyze the performance of our algorithm as regards its ability to solve the IRP instances, whereas Section 3.5.3 is devoted to the performance as regards the MCIRP.

Tables 3.2, 3.3, 3.4, and 3.5 give the details of our results for the IRP instances and in these tables we report the following: The first column of the tables reports the instance name. Under the *Our Algorithm* header, which contains the information about performance of the heuristic Branch-and-Price algorithm, we report the value of the solution found by the heuristic Branch-and-Price algorithm (*Cost*), the total running time in seconds (*Time*), and the number of solved nodes (*SLN*) and unsolved nodes (*USN*) in the branching tree. Under the *DRL* header, we report the lower bound (*LB*) and upper bound (*UB*) for the IRP dataset presented by Desaulniers et al. (2015). Under the *OptGap* header, we report the optimality gap between the lower bound of Desaulniers et al. (2015) and the best solution of the heuristic Branch-and-Price algorithm. Under the *ABS* and the *Gap* headers, we report the best solutions found by Archetti et al. (2014) and the average gap of the heuristic Branch-and-Price solution compared to their best solution, respectively.

Tables 3.6 and 3.7 give the details of the results for the MCIRP instances and in these tables we report the following: Under the η_{10}, and η_6 headers, we report the results obtained when 10 and 6 neighbors are considered in the No-Semi and the Friends heuristics, respectively. Under each of these headers, there are *Cost*, *Time*, *SLN*, and *USN* sub-headers defined as above and a *Columns* sub-header which reports the number of columns produced by the heuristic Branch-and-Price algorithm until the time limit is reached.

3.5.1 Data Instances

Since there are no test problems for the MCIRP, we use the IRP benchmark instances proposed by Archetti et al. (2014) to evaluate the performance of the proposed algorithm. This dataset is divided into four subsets based on the inventory holding cost (high holding cost(C_H), and low holding cost (C_L)) and the planning horizon (3 and 6 periods) to C_H3, C_H6, C_L3, and C_L6. For example, the planning horizon of the instances in C_H6 is six periods, and the holding cost is high. We only considered the C_L3 and C_L6 subsets.

The number of customers are $5k$ (with $k = 1,2,...,10$ when $H = 3$, and $k = 1,2,...,6$ when $H = 6$) and the capacity of the vehicles is determined by dividing the vehicle capacity in the instances of Archetti

et al. (2007) by the number of desired vehicles (2-5). In each subset, there are 5 instances for each possible combination of the number of customers and the number of vehicles. We used the first two instances of each subset in our tests. In total, we ran 120 instances of this dataset.

We have also generated a total of 90 new data instances for the MCIRP by modifying the instances of the C_L3, and C_L6 subsets by Archetti et al. (2014) as follows:

- Time horizon H: 3, 6;
- Number of customers: $5k$, with $k = 1,2,...,7$ when $H = 3$, and $k = 1,2,...,4$ when $H = 6$;
- Number of commodities: 2.

All customer locations are based on the original dataset and are generated randomly for each coordinate as an integer number in the interval [0,500]. The demand d^t_{im} of customer i for commodity $m \in M_i$ at each discrete time instant of the planning time horizon is constant over time, i.e. $d^t_{im} = d_{im}$. d^t_{im} is determined by dividing the original demand of each customer by two and adding or subtracting a random number between 0 to 2 from this amount. The storage capacity limits, U_{im}, the initial inventory levels, $\hat{I}^0_{im}$ and $\hat{I}^0_{0m}$ and the production rate p^t_m are determined by dividing their original values by two and adjusting by a random number between 0 to 2. The holding costs, h_{im} and h_{0m} are as in the original datasets. Finally, the capacity Q_m of compartment m of the vehicles is determined by dividing the original vehicle capacity by two.

3.5.2 Results for the IRP

We compare the results obtained by the heuristic Branch-and-Price algorithm for just a single commodity to those obtained by the two most recent algorithms: the Branch-Price-and-Cut algorithm proposed by Desaulniers et al. (2015) and the tabu search algorithm presented by Archetti et al. (2014). Tables 3.2, 3.3, 3.4, and 3.5 compare our results with these algorithms. The planning horizon of Tables 3.2 and 3.3 is three periods and the planning horizon of Tables 3.4 and 3.5 is six periods. The time limit (TL) for Tables 3.2 and 3.4 is 1,200 seconds and for Tables 3.3 and 3.5 it is 3,600 seconds.

In this research, we assume that the total capacity of the vehicle compartments exceeds the customers' daily demand for different commodities. However, after running the instances of Archetti et al. (2014), we found some instances where the vehicle capacity was lower than the maximum demand of the customers and our algorithm could not find a feasible solution for them (abs2n5-3, abs1n5-4, abs2n5-4 in Table 3.2, and abs2n5-4 in Table 3.4).

Table 3.2 shows that the algorithm can solve most of the instances with up to 20 nodes. The average optimality gap of the heuristic Branch-and-Price algorithm compared to the *LB* for these instances is 1.14 percent. The average gap of our algorithm compared to the tabu search of Archetti et al. (2007) is 1.05 percent for these instances.

The vehicle capacity is an increasing function of the number of nodes in the IRP instances. For example, the vehicle capacity for the instances with 5 customers and 2 vehicles is 144 while this amount increases to 1,425 for the instance with 30 customers and 2 vehicles. Similar to the usual Branch-and-Price algorithm,

the ability of our algorithm to solve the problem decreases when the number of nodes and the capacity of the vehicles increase. In total, our algorithm can find solutions with reasonable optimality gaps for 8 of 12 instances with 25 and 30 nodes within 1,200 seconds, and the average optimality gap of these instances is 4.98 percent.

We see a similar pattern when we turn our attention to Table 3.3. With an increased number of nodes and capacity of the compartments, the performance quality of our algorithm decreases. Although the heuristic Branch-and-Price algorithm can find solutions with a reasonable optimality gap for 19 of 57 instances and the average optimality gap for these instances is 2.72 percent, the average gap for these instances compared to the results of Archetti et al. (2014) is 1.8 percent. For a given number of customers we also expect to see the best performance for instances with many vehicles compared to fewer vehicles as such instances generally have fewer customers per route. The heuristic is therefore faster and more iterations can be executed within the time limit. Most of the instances in Tables 3.2 and 3.3 show this trend, but not all of them. When the running time is increased from 1200 seconds to 3600 seconds (see Tables 3.2 and 3.3), the algorithm can find a better solution with a lower optimality gap for most of the instances. Due to the difficulty of some of the instances, the algorithm cannot always find a better solution within one hour.

Comparing Tables 3.2 and 3.4, we see that the instances with long time horizons (Table 3.4) are more challenging for the heuristic Branch-and-Price algorithm than those with shorter time horizons. Furthermore, our algorithm only considers zero and full sub deliveries and this policy limits the number of possible delivery patterns. Therefore, the inventory cost of our solution for most of the instances is higher than the inventory cost of the optimal solution. In addition, the capacities of the customers and suppliers in the datasets of Archetti et al. (2014) are higher for the instances with long time horizons than for the instances with the same number of nodes and vehicles and short time horizons. As a result, the inventory cost of our solution is higher for an instance with six periods than for a similar instance with three periods. Hence, the optimality gap of the instances with long time horizons will be higher than in the case of short time horizons too. However, the algorithm can find solutions with a reasonable optimality gap for 11 of 32 instances of Table 3.4 in 1,200 seconds. For these instances, the average optimality gap is 8.13 percent and the average gap compared to *ABS* is 7.47 percent.

Table 3.5 shows that the ability of our algorithm to solve the problems decreases when the number of nodes, the vehicle capacity, and the length of the planning horizons increase. The algorithm can find solutions with a reasonable optimality gap for 10 of 32 instances within 3,600 seconds. For these instances, the average optimality gap is 7.44 percent and the average gap compared to *ABS* is 3.34 percent. As regards the instances with a high number of customers (more than 20 nodes), the algorithm finds the instances with a high number of vehicles with smaller capacity easier to solve, and it can solve more nodes of the branching tree within one hour.

3.5.3 Results for the MCIRP

Tables 3.6 and 3.7 show the computational results and running times for our new MCIRP datasets. As the problem has not been addressed in any previous papers, no comparison is possible.

From the *SLN* and *USN* columns of Table 3.6, it can be seen that the algorithm can solve more nodes in the branching tree when the size of η decreases from 10 to 6. Furthermore, for the instances where solving the root node takes more than 1,200 seconds, the running time decreases when η changes from 10 to 6. When η is changed from 10 to 6, we expect that the ability to produce columns decreases. The number of produced columns of each instance for $\eta = 10$ is higher than $\eta = 6$, which supports our assumption. For 10 instances, the value of the best solution of the heuristic Branch-and-Price algorithm for $\eta = 10$ is lower than for $\eta = 6$, and for 23 instances the value of the best solution of the heuristic Branch-and-Price algorithm is better with $\eta = 6$. The quality of the solutions is unchanged for 19 instances.

We see a similar pattern when we turn our attention to Table 3.7. When η deceases from 10 to 6, the algorithm can produce a lower number of columns for all of the instances, but the algorithm can solve a higher number of the nodes in the branching tree. For 12 instances, our algorithm can find a better solution with $\eta = 10$, and for 7 instances, the solution found by the heuristic Branch-and-Price algorithm for $\eta = 6$ is better than for $\eta = 10$.

3.6 Concluding Remarks

In this paper, we have investigated a distribution system in which the customers request different commodities over a time horizon. A supplier delivers these commodities by a fleet of vehicles while ensuring that the customers always have the needed commodities available. The commodities must be kept separated during transport and there is dedicated storage for each commodity at the supplier and as well as at the customers' locations.

We presented two mathematical formulations for this problem and we developed a heuristic Branch-and-Price algorithm to find good solutions for this class of problem. Our algorithm includes three heuristic pricing algorithms.

We have used the IRP benchmark instances created by Archetti et al. (2014) to evaluate the performance of our algorithm. We have also generated a new set of test instances to evaluate our solution approach for the MCIRP. In total, we have tested our algorithm on 210 instances.

Computational results for the IRP dataset indicate that for the instances with up to 20 customers and 3 periods, the algorithm can find good solutions within a reasonable time with an average optimality gap of 1.14 percent. Similar to the Branch-and-Price algorithm, the capability of our algorithm decreases when the number of nodes, the capacity of vehicles, and the length of the planning horizon increase. But still our algorithm can find solutions with reasonable optimality gaps for 8 of 12 instances with 25-30 nodes and 3 periods within 1,200 seconds, and the average optimality gap of these instances is 4.98 percent. In case of a

higher number of customers (with $H = 3$), our algorithm can find solutions with reasonable optimality gaps for 19 of 57 instances in 3,600 second, and the average optimality gap for these instances is 2.72 percent. We see a similar pattern when we turn our attention to the instances with long planning horizon.

From the results of the new MCIRP datasets, we find the instances with long time horizons, high number of customers, and high vehicle capacity to be more challenging for the heuristic Branch-and-Price algorithm. As this is the first paper to study the MCIRP, no comparison can be performed for these instances.

As this is a new problem, there are multiple directions for future research. Firstly, our algorithm can be extended to include different types of column generations, e.g. based on inclusion of the local search rather than purely using a labeling approach as in the present version. Furthermore, we could consider partial sub deliveries and solve the route node of the branching tree and obtain a lower bound for this problem. Adding different types of cuts to the solution approach can help to decrease the running time of the algorithm and find better solutions. Secondly, the development of more traditional heuristic and metaheuristic algorithms is a natural path for future research. It shall be interesting to see how such algorithms would balance quality and speed compared to the algorithm presented here. Finally, as this is a new problem, the development of exact algorithms is an obvious next step. Due to the long running time of exact labeling, a Branch-and-Price algorithm would benefit from the heuristic labeling presented here such that the exact labeling would only be needed as proof of optimality.

Acknowledgment

This research is partially funded by the Danish Council for Independent Research - Social Sciences. Project "Transportation issues related to waste management".

Table 3.2: Computational results of the heuristic Branch-and-Price algorithm for IRP, $H = 3$, $TL = 1200$.

	Our Algorithm				DRL				
ID	Cost	Time	SLN	USN	LB	UB	OptGap	ABS	Gap
abs1n5-1	1373.89	3	21	0	1373.41	1373.41	0.03	1373.41	0.03
abs2n5-1	1156.37	1	3	0	1155.87	1155.87	0.04	1155.91	0.04
abs1n10-1	2189.01	22	23	0	2186.79	2186.79	0.10	2186.79	0.10
abs2n10-1	2746.06	47	69	0	2744.23	2744.23	0.07	2744.24	0.07
abs1n15-1	2217.41	170	61	0	2203.33	2203.33	0.64	2203.37	0.64
abs2n15-1	2463.69	276	49	0	2461.85	2461.85	0.07	2461.89	0.07
abs1n20-1	2945.58	1222	116	63	2791.96	2791.96	5.50	2792.02	5.50
abs2n20-1	2619.28	933	147	0	2535.04	2535.04	3.32	2535.04	3.32
abs1n25-1	3062.39	1204	43	13	2987.47	2987.47	2.51	2987.53	2.51
abs2n25-1	5416.97	1234	69	66	3340.59	3340.59	62.16	3340.59	62.16
abs1n30-1	5639.01	1245	42	39	3565.60	3565.60	58.15	3565.60	58.15
abs2n30-1	6080.35	1319	25	24	3435.42	3435.42	76.99	3435.42	76.99

Table 3.2: Computational results of the heuristic Branch-and-Price algorithm for IRP, $H = 3$, $TL = 1200$.

	Our Algorithm				DRL				
ID	Cost	Time	SLN	USN	LB	UB	OptGap	ABS	Gap
abs1n5-2	1407.89	2	7	0	1407.59	1407.59	0.02	1407.59	0.02
abs2n5-2	1561.07	1	7	0	1561.07	1561.07	0.00	1561.07	0.00
abs1n10-2	2658.01	89	21	0	2656.21	2656.21	0.07	2656.21	0.07
abs2n10-2	3406.06	233	73	0	3404.52	3404.52	0.05	3404.52	0.05
abs1n15-2	2691.32	688	281	0	2690.08	2690.08	0.05	2690.08	0.05
abs2n15-2	2665.69	717	15	0	2665.57	2665.57	0.00	2665.57	0.00
abs1n20-2	3707.58	1214	214	169	3475.51	3480.38	6.68	3480.38	6.53
abs2n20-2	2782.28	298	87	0	2778.54	2778.54	0.13	2778.54	0.13
abs1n25-2	3432.39	1225	115	86	3357.57	3357.57	2.23	3357.57	2.23
abs2n25-2	6115.97	1545	2	1	3761.30	3805.52	62.60	3791.53	61.31
abs1n30-2	7003.01	1302	2	1	4013.46	4013.46	74.49	4013.46	74.49
abs2n30-2	6748.35	1488	2	1	3882.47	3892.59	73.82	3947.08	70.97
abs1n5-3	1591.65	1	3	0	1578.65	1578.65	0.82	1578.65	0.82
abs2n5-3	-	-	-	-	1791.03	1791.03	-	1791.03	-
abs1n10-3	3188.68	53	25	0	3185.54	3185.54	0.10	3185.54	0.10
abs2n10-3	4337.34	56	37	0	4205.39	4205.39	3.14	4205.39	3.14
abs1n15-3	3076.93	324	407	0	3072.75	3072.75	0.14	3072.79	0.13
abs2n15-3	3311.52	441	167	0	3304.41	3304.41	0.22	3304.41	0.22
abs1n20-3	4150.95	1210	335	310	4022.66	4022.66	3.19	4092.13	1.44
abs2n20-3	3007.28	757	137	0	2998.11	2998.11	0.31	2998.11	0.31
abs1n25-3	3814.86	1224	263	162	3803.92	3803.92	0.29	3811.67	0.08
abs2n25-3	4409.6	1208	98	77	4332.02	4342.38	1.79	4438.17	-0.64
abs1n30-3	4919.17	1209	50	51	4482.40	4482.40	9.74	4668.38	5.37
abs2n30-3	4722.7	1212	47	50	4219.71	4219.71	11.92	4316.07	9.42
abs1n5-4	-	-	-	-	1687.42	1687.42	-	1708.51	-
abs2n5-4	-	-	-	-	1997.96	1997.96	-	1997.96	-
abs1n10-4	3732.26	35	175	0	3652.38	3652.38	2.19	3652.38	2.19
abs2n10-4	4846.16	23	65	0	4615.71	4615.71	4.99	4615.71	4.99
abs1n15-4	3490.49	323	219	0	3487.12	3487.12	0.10	3487.19	0.09
abs2n15-4	3799.06	933	417	0	3797.18	3797.18	0.05	3797.18	0.05
abs1n20-4	4316.2	212	63	0	4279.43	4279.43	0.86	4308.78	0.17
abs2n20-4	3219.84	866	65	0	3214.17	3214.17	0.18	3216.27	0.11
abs1n25-4	4288.27	1225	16	15	3949.39	3949.39	8.58	3949.39	8.58
abs2n25-4	4980.8	1204	194	141	4849.21	4849.21	2.71	4866.58	2.35
abs1n30-4	7849.01	1611	2	1	5076.56	5076.56	54.61	5201.74	50.89
abs2n30-4	7570.35	1252	27	26	4647.61	4671.58	62.89	4748.87	59.41

Table 3.3: Computational results of the heuristic Branch-and-Price algorithm for IRP, $H = 3$, $TL = 3600$.

	Our Algorithm				DRL				
ID	Cost	Time	SLN	USN	LB	UB	OptGap	ABS	Gap
abs1n20-1	2860.38	3678	287	24	2791.96	2791.96	2.45	2792.02	2.45
abs2n20-1	2619.28	933	147	0	2535.04	2535.04	3.32	2535.04	3.32
abs1n25-1	3013.39	1592	83	0	2987.47	2987.47	0.87	2987.53	0.87
abs2n25-1	4494.01	3661	153	144	3340.59	3340.59	34.53	3340.59	34.53
abs1n30-1	5639.01	3633	10	9	3565.60	3565.60	58.15	3565.60	58.15
abs2n30-1	6080.35	3820	35	32	3435.42	3435.42	76.99	3435.42	76.99
abs1n35-1	6613.76	5833	2	1	3354.34	6607.18	96.59	3374.61	95.99
abs2n35-1	5804.7	3663	7	6	3649.72	3661.98	57.69	3735.50	55.39
abs1n40-1	6294.76	3857	45	36	3700.57	6292.38	69.63	3725.84	68.95
abs2n40-1	7639.57	3934	2	1	3836.04	7637.56	94.98	4004.47	90.78
abs1n45-1	7465.33	3734	15	14	-	-	-	3827.63	95.04
abs2n45-1	7779.54	3745	14	13	3877.31	7770.14	99.56	3919.38	98.49
abs1n50-1	7390.74	4400	13	12	4162.27	7378.98	75.57	4272.27	72.99
abs2n50-1	7971.34	3874	2	1	4477.61	4650.66	76.07	4592.98	73.55
abs1n20-2	3679.58	3648	489	306	3475.51	3480.38	5.86	3480.38	5.72
abs2n20-2	2782.28	298	87	0	2778.54	2778.54	0.13	2778.54	0.13
abs1n25-2	3361.39	3428	307	0	3357.57	3357.57	0.11	3357.57	0.11
abs2n25-2	3878.83	3646	314	217	3761.30	3805.52	3.10	3791.53	2.30
abs1n30-2	7003.01	3740	11	10	4013.46	4013.46	74.49	4013.46	74.49
abs2n30-2	6748.35	3711	13	12	3882.47	3892.59	72.61	3947.08	70.97
abs1n35-2	6884.76	3683	2	1	3849.56	3857.31	78.04	3889.07	77.03
abs2n35-2	6532.7	3651	20	19	3996.40	4029.11	61.44	4128.04	58.25
abs1n40-2	7059.76	3628	14	13	4251.82	4265.76	65.20	4306.36	63.94
abs2n40-2	7850.57	4742	2	1	4257.40	7849.13	78.18	4595.91	70.82
abs1n45-2	7589.33	3717	6	5	4248.46	4254.07	78.06	4279.63	77.34
abs2n45-2	8759.54	3847	5	4	4465.27	4480.62	94.86	4527.16	93.49
abs1n50-2	8120.74	3679	5	4	4894.69	8117.49	65.27	4942.69	64.30
abs2n50-2	10437.3	4433	2	1	5058.43	10436.96	101.21	5314.44	96.40
abs1n20-3	4150.95	3639	785	638	4022.66	4022.66	3.14	4092.13	1.44
abs2n20-3	3007.28	757	137	0	2998.11	2998.11	0.31	2998.11	0.31
abs1n25-3	3814.86	3652	612	249	3803.92	3803.92	0.29	3811.67	0.08
abs2n25-3	4409.6	3645	492	295	4332.02	4342.38	1.75	4438.17	-0.64
abs1n30-3	4919.17	3628	176	161	4482.40	4482.40	9.36	4668.38	5.37
abs2n30-3	4442.33	3636	348	279	4219.71	4219.71	5.16	4316.07	2.93
abs1n35-3	6903.76	3683	72	67	4145.67	4145.67	64.57	4271.43	61.63
abs2n35-3	7544.7	3716	14	13	4379.65	4433.24	68.77	4602.56	63.92
abs1n40-3	7322.76	3701	50	49	4251.82	4265.76	64.01	4797.90	52.62

Table 3.3: Computational results of the heuristic Branch-and-Price algorithm for IRP, $H = 3$, $TL = 3600$.

	Our Algorithm				DRL				
ID	Cost	Time	SLN	USN	LB	UB	OptGap	ABS	Gap
abs2n40-3	8379.57	4053	2	1	4258.19	7849.13	83.52	4934.72	69.81
abs1n45-3	8597.33	3617	30	29	4701.69	4706.25	81.91	4756.07	80.77
abs2n45-3	8298.54	3743	12	11	5057.04	5057.04	60.09	5394.01	53.85
abs1n50-3	8615.74	3728	9	8	5465.47	8609.83	55.76	5650.01	52.49
abs2n50-3	10281.3	5006	2	1	-	-	-	6018.02	70.84
abs1n20-4	4316.2	212	63	0	4279.43	4279.43	0.85	4308.78	0.17
abs2n20-4	3219.84	866	65	0	3214.17	3214.17	0.18	3216.27	0.11
abs1n25-4	3951.39	3621	47	20	3949.39	3949.39	0.05	3949.39	0.05
abs2n25-4	4947.93	3639	891	174	4849.21	4849.21	2.03	4866.58	1.67
abs1n30-4	7849.01	3616	25	24	5076.56	5076.56	53.30	5201.74	50.89
abs2n30-4	5084.68	3636	437	428	4647.61	4671.58	9.20	4748.87	7.07
abs1n35-4	4712.19	3642	95	92	4541.69	4541.69	3.62	4711.28	0.02
abs2n35-4	8405.7	3620	33	32	4800.09	4821.13	69.81	5164.84	62.75
abs1n40-4	8614.76	3784	27	26	3700.57	6292.38	90.98	5401.20	59.50
abs2n40-4	8744.57	3633	9	8	3836.04	7637.56	89.39	5491.16	59.25
abs1n45-4	9473.33	3645	5	4	5181.16	5186.87	80.08	5359.67	76.75
abs2n45-4	9161.54	4024	14	13	5698.48	5707.60	55.37	6254.54	46.48
abs1n50-4	10137.7	3699	32	31	-	-	-	6678.38	51.80
abs2n50-4	11677.3	4174	2	1	-	-	-	6940.56	68.25

Table 3.4: Computational results of the heuristic Branch-and-Price algorithm for IRP, $H = 6$, $TL = 1200$.

	Our Algorithm				DRL				
ID	Cost	Time	SLN	USN	LB	UB	OptGap	ABS	Gap
abs1n5-1	3885.41	247	555	0	3736.12	3736.12	4.00	3736.24	3.99
abs2n5-1	3344.53	430	889	0	3148.59	3148.59	6.22	3148.70	6.22
abs1n10-1	6107.85	1203	303	282	5573.34	5624.25	9.59	5617.90	8.72
abs2n10-1	11345.2	1203	143	138	6458.59	6458.59	75.66	6458.63	75.66
abs1n15-1	11651.3	1213	104	103	5788.71	5947.56	101.28	5884.52	98.00
abs2n15-1	12162	1218	70	71	6018.89	6313.10	102.06	6060.86	100.66
abs1n20-1	13111	1218	11	10	7123.44	12363.51	84.05	7300.60	79.59
abs2n20-1	13434.4	1294	6	5	6199.74	12336.27	116.69	6304.28	113.10
abs1n5-2	5218.47	190	337	0	4617.59	4617.59	13.01	4617.59	13.01
abs2n5-2	4697.63	274	815	0	4184.40	4184.40	12.27	4184.40	12.27
abs1n10-2	7187.77	907	49	0	7027.28	7070.22	2.28	7288.48	-1.38
abs2n10-2	8587.12	1204	58	51	8110.00	8299.71	5.88	8290.94	3.57
abs1n15-2	14562.3	1241	8	7	6737.78	14561.56	116.13	6874.43	111.83
abs2n15-2	14146	1283	4	3	6969.90	12538.45	102.96	7189.42	96.76
abs1n20-2	15790	1673	2	1	8528.86	14221.91	85.14	9095.49	73.60
abs2n20-2	15014.4	1553	2	1	6761.18	13617.78	122.07	6995.37	114.63
abs1n5-3	7586.96	7	25	0	5466.63	5466.63	38.79	5479.26	38.47
abs2n5-3	5781.82	56	287	0	5089.23	5089.23	13.61	5089.23	13.61
abs1n10-3	8854.38	1211	101	60	8329.97	8354.44	6.30	8707.31	1.69
abs2n10-3	15031.2	1227	47	48	9802.61	9821.30	53.34	10157.97	47.97
abs1n15-3	14287.3	1239	16	15	7607.28	7836.86	87.81	8220.27	73.81
abs2n15-3	13627	1275	9	8	7945.41	13623.59	71.51	8514.21	60.05
abs1n20-3	16380	1210	31	30	10113.92	16378.84	61.96	10590.22	54.67
abs2n20-3	15601.4	1949	2	1	7390.88	15601.36	111.09	7816.75	99.59
abs1n5-4	7588.7	3	13	0	6406.12	6406.12	18.46	6406.12	18.46
abs2n5-4	-	-	-	-	5972.80	5972.80	-	5972.80	-
abs1n10-4	10487.8	245	147	0	9724.40	14949.45	7.85	10288.34	1.94
abs2n10-4	17370.2	1231	23	22	11545.05	11579.93	50.46	12156.52	42.89
abs1n15-4	15035.3	1221	19	18	8523.78	14303.06	76.39	9099.95	65.22
abs2n15-4	15989	1204	261	250	8901.28	15005.70	79.63	9509.04	68.15
abs1n20-4	19891	1217	35	34	11545.29	17449.71	72.29	12422.29	60.12
abs2n20-4	17098.4	1210	25	24	8103.65	16390.99	111.00	8709.66	96.32

Table 3.5: Computational results of the heuristic Branch-and-Price algorithm for IRP, $H = 6$, $TL = 1200$.

	Our Algorithm				DRL				
ID	Cost	Time	SLN	USN	LB	UB	OptGap	ABS	Gap
abs1n10-1	6107.85	3625	731	662	5573.34	5624.25	9.59	5617.90	8.72
abs2n10-1	11345.20	3610	713	658	6458.59	6458.59	75.66	6458.63	75.66
abs1n15-1	11651.30	3629	351	298	5788.71	5947.56	101.28	5884.52	98.00
abs2n15-1	12162.00	3649	185	182	6018.89	6313.10	102.06	6060.86	100.66
abs1n20-1	13111.00	3666	52	51	7123.44	12363.51	84.05	7300.60	79.59
abs2n20-1	13434.40	3623	18	17	6199.74	12336.27	116.69	6304.28	113.10
abs1n25-1	15580.40	3796	25	24	7300.34	14643.92	113.42	7329.86	112.56
abs2n25-1	15149.40	3776	4	3	8090.53	13631.12	87.25	8242.26	83.80
abs1n10-2	7187.77	907	49	0	7027.28	7070.22	2.28	7288.48	-1.38
abs2n10-2	8587.12	3603	185	142	8110.00	8299.71	5.88	8290.94	3.57
abs1n15-2	14562.30	3615	47	50	6737.78	14561.56	116.13	6874.43	111.83
abs2n15-2	14146.00	3651	33	32	6969.90	12538.45	102.96	7189.42	96.76
abs1n20-2	15790.00	3624	21	20	8528.86	14221.91	85.14	9095.49	73.60
abs2n20-2	15014.40	3622	153	152	6761.18	13617.78	122.07	6995.37	114.63
abs1n25-2	18820.40	4459	2	1	8118.38	16524.44	131.82	8478.07	121.99
abs2n25-2	16573.40	3824	9	8	9343.83	15568.39	77.37	9762.24	69.77
abs1n10-3	8854.38	3610	263	110	8329.97	8354.44	6.30	8707.31	1.69
abs2n10-3	10328.10	3603	1 425	534	9802.61	9821.30	5.36	10157.97	1.67
abs1n15-3	14287.30	3658	54	59	7607.28	7836.86	87.81	8220.27	73.81
abs2n15-3	13627.00	3681	34	37	7945.41	13623.59	71.51	8514.21	60.05
abs1n20-3	16380.00	3627	261	248	10113.92	16378.84	61.96	10590.22	54.67
abs2n20-3	8411.89	3629	297	282	7390.88	15601.36	13.81	7816.75	7.61
abs1n25-3	18823.40	3629	36	35	9027.20	18587.36	108.52	9673.94	94.58
abs2n25-3	18299.40	4153	2	1	10722.71	18297.88	70.66	11458.21	59.71
abs1n10-4	10487.80	245	147	0	9724.40	14949.45	7.85	10288.34	1.94
abs2n10-4	12841.10	286	97	0	11545.05	11579.93	11.23	12156.52	5.63
abs1n15-4	9213.97	3620	77	74	8523.78	14303.06	8.10	9099.95	1.25
abs2n15-4	15989.00	3726	30	29	8901.28	15005.70	79.63	9509.04	68.15
abs1n20-4	19891.00	3618	324	321	11545.29	17449.71	72.29	12422.29	60.12
abs2n20-4	8943.98	3615	235	230	8103.65	16390.99	10.37	8709.66	2.69
abs1n25-4	18006.90	3625	57	58	9985.15	17600.84	80.34	11074.58	62.60
abs2n25-4	20758.40	4055	2	1	12185.99	20385.49	70.35	13289.94	56.20

Table 3.6: Computational results of the heuristic Branch-and-Price algorithm for MCIRP, $H = 3$, $TL = 1200$.

	$\eta = 10$					$\eta = 6$				
ID	Cost	Time	SLN	USN	Columns	Cost	Time	SLN	USN	Columns
abs1n5-1	**1376.54**	9	27	0	915	**1376.54**	5	13	0	591
abs2n5-1	**1156.37**	3	3	0	484	**1156.37**	1	0	0	351
abs1n10-1	**2264.26**	1221	267	48	29817	2281.01	79	47	0	6460
abs2n10-1	**2746.06**	284	75	0	14622	**2746.06**	1263	24	13	7823
abs1n15-1	2540.09	1220	76	63	32705	**2448.41**	1219	171	8	28750
abs2n15-1	**2541.69**	1317	34	9	34646	2867.69	1243	72	49	22413
abs1n20-1	4691.27	1321	14	13	40470	**4580.26**	1312	55	54	35389
abs2n20-1	**3030.82**	1238	25	28	35159	4012.42	1954	25	26	16679
abs1n25-1	5521.76	1446	2	1	29861	**4692.27**	1233	26	21	21203
abs2n25-1	**5995.97**	1973	2	1	28841	**5995.97**	1229	6	5	15030
abs1n30-1	**6062.01**	2398	2	1	31569	**6062.01**	1042	31	0	15706
abs2n30-1	**6519.35**	2485	2	1	30066	**6519.35**	1887	2	1	27086
abs1n35-1	**6633.76**	4010	2	1	37243	**6633.76**	1587	2	1	15414
abs2n35-1	**5990.7**	17768	2	1	59638	**5990.7**	5604	2	1	39617
abs1n5-2	**1407.89**	13	0	0	206	1517.65	1	0	0	189
abs2n5-2	**1561.07**	11	69	0	342	**1561.07**	3	3	0	246
abs1n10-2	2688.01	1207	36	23	6526	**2658.01**	641	19	0	4123
abs2n10-2	**3406.06**	806	83	0	7293	3567.37	1209	89	30	4590
abs1n15-2	**2712.17**	344	63	0	14228	2779.08	1052	239	0	11769
abs2n15-2	**2673.69**	1214	23	0	21831	2808.69	131	0	0	5652
abs1n20-2	5623.27	1219	22	21	26196	**4796.38**	1238	61	62	37776
abs2n20-2	4623.99	1207	12	11	16257	**3116.84**	1915	7	6	8462
abs1n25-2	6014.76	1267	18	17	29028	**4723.1**	1214	52	51	24817
abs2n25-2	6495.97	1230	8	7	23725	**5434.15**	1703	41	42	27923
abs1n30-2	7003.01	1243	11	10	26484	**7002.76**	1207	25	28	16760
abs2n30-2	**7161.35**	6115	2	1	29793	**7161.35**	3659	2	1	17052
abs1n35-2	**6719.76**	1561	2	1	24839	**6719.76**	1243	40	39	20242
abs2n35-2	**7057.7**	6580	2	1	40611	**7057.7**	6261	2	1	26177
abs1n5-3	**1591.65**	3	3	0	171	**1591.65**	3	5	0	116
abs2n5-3	**1644.17**	6	0	0	113	**1644.17**	1	0	0	103
abs1n10-3	**3224.71**	1162	85	0	2883	3284.62	220	53	0	3039
abs2n10-3	4367.52	881	21	0	2318	**4337.34**	323	29	0	1942
abs1n15-3	3138.69	1211	419	244	17803	**3130.17**	1223	236	107	12159
abs2n15-3	**3570.02**	1212	77	62	18747	4197.83	1419	23	24	5729
abs1n20-3	6057.27	1225	114	111	32532	**4207.64**	1213	74	59	28305
abs2n20-3	**3975.71**	1479	41	42	20208	5133.99	3346	2	1	7254
abs1n25-3	6998.76	1247	2	1	13496	**4844.15**	1209	33	36	17091

Table 3.6: Computational results of the heuristic Branch-and-Price algorithm for MCIRP, $H = 3$, $TL = 1200$.

	$\eta = 10$					$\eta = 6$				
ID	Cost	Time	SLN	USN	Columns	Cost	Time	SLN	USN	Columns
abs2n25-3	7180.97	1369	2	1	12322	**5496.62**	1207	67	68	15157
abs1n30-3	6468.01	1253	3	2	14967	**6314.57**	1211	77	70	19656
abs2n30-3	**6960.35**	1400	2	1	16200	**6960.35**	2295	2	1	14682
abs1n35-3	7056.76	1873	2	1	21214	**6717.64**	1217	30	29	19852
abs2n35-3	**7642.7**	9403	2	1	31039	**7642.7**	3373	2	1	16580
abs1n5-4	**1592.3**	1	0	0	87	1592.95	2	5	0	95
abs2n5-4	**1644.17**	1	0	0	113	**1644.17**	1	0	0	103
abs1n10-4	3768.13	1208	52	45	1713	**3732.26**	897	129	0	1302
abs2n10-4	**4846.16**	189	173	0	2014	4909.16	1216	121	28	1838
abs1n15-4	3709.61	1206	16	17	7019	**3593.73**	1205	209	146	8467
abs2n15-4	5428.59	1232	3	2	5030	**3991.69**	1210	221	110	5838
abs1n20-4	**4611.58**	1213	107	84	26208	6628.27	1372	3	2	5213
abs2n20-4	**3338.9**	1204	70	59	12187	6036.99	1283	3	2	5355
abs1n25-4	7448.76	1220	14	13	13371	**4241.14**	1230	70	25	22963
abs2n25-4	8767.97	3170	2	1	11994	**6414.67**	1256	21	22	12174
abs1n30-4	8453.01	1229	11	10	17962	**7267.1**	1210	38	43	15949
abs2n30-4	**7899.35**	1923	2	1	15708	**7899.35**	1570	2	1	10148
abs1n35-4	9146.76	3983	2	1	19669	**5835.24**	1348	22	17	22865
abs2n35-4	**8485.7**	16092	2	1	25213	**8485.7**	3275	2	1	17241

Table 3.7: Computational results of the heuristic Branch-and-Price algorithm for MCIRP, $H = 6$, $TL = 3600$.

	$\eta = 10$					$\eta = 6$				
ID	Cost	Time	SLN	USN	Columns	Cost	Time	SLN	USN	Columns
abs1n5-1	**4023.99**	201	191	0	2371	4027.56	57	61	0	1335
abs2n5-1	**3500.73**	529	317	0	2997	3547.73	1203	147	70	1970
abs1n10-1	**6503.58**	1216	56	59	25866	10813.6	1220	109	112	19219
abs2n10-1	**11345.2**	1231	49	46	21733	**11345.2**	1216	35	34	12983
abs1n15-1	**12059.3**	1686	2	1	30208	**12059.3**	1213	3	2	19371
abs2n15-1	**12141**	4772	2	1	35894	12162	1408	2	1	16660
abs1n20-1	**13735**	4004	2	1	44721	**13735**	3196	2	1	38190
abs2n20-1	**13226.4**	3370	2	1	47167	**13226.4**	3945	2	1	30399
abs1n5-2	**5222.55**	7	5	0	500	5223.85	7	0	0	415
abs2n5-2	**4852.71**	13	15	0	508	4855.41	60	29	0	509
abs1n10-2	**7598.79**	1211	136	129	14761	13768.6	1219	2	1	6160
abs2n10-2	**15692.2**	1207	45	46	14390	**15692.2**	1661	2	1	6906
abs1n15-2	**14562.3**	2703	2	1	21644	**14562.3**	1631	2	1	14704
abs2n15-2	15038	3329	2	1	22218	**14286.6**	1726	2	1	13712
abs1n20-2	16022	4293	2	1	32754	**14454.6**	4335	2	1	22793
abs2n20-2	**15822.4**	5792	2	1	42789	**15822.4**	3648	2	1	21536
abs1n5-3	**7852.5**	3	3	0	224	7853.36	2	0	0	165
abs2n5-3	5798.03	2	0	0	267	**5796.41**	11	11	0	257
abs1n10-3	**9127.52**	1205	127	104	8267	16763.6	1208	29	28	3308
abs2n10-3	17379.2	1221	141	140	10201	**17235.5**	1377	4	3	4740
abs1n15-3	**15339.3**	1273	2	1	13937	**15339.3**	1216	37	36	9998
abs2n15-3	13627	3409	2	1	18091	**13283.1**	1212	25	24	10066
abs1n20-3	**16396**	4860	2	1	23220	**16396**	1474	2	1	15435
abs2n20-3	**15601.4**	5060	2	1	27841	**15601.4**	1598	2	1	16039
abs1n5-4	7594.93	2	0	0	140	**7593.42**	3	3	0	137
abs2n5-4	**6001.72**	2	0	0	184	6002.22	2	0	0	176
abs1n10-4	**11245.7**	1356	31	30	3187	11478.5	1215	32	33	2283
abs2n10-4	**13105.6**	1205	39	40	4933	15507.4	1226	12	11	2286
abs1n15-4	**15035.3**	1281	2	1	9360	**15035.3**	1210	49	48	7141
abs2n15-4	**16354**	2904	2	1	12609	**16354**	1212	32	31	7303
abs1n20-4	**21343**	4362	2	1	18712	**21343**	1210	8	7	13083
abs2n20-4	17098.4	5916	2	1	19861	**17045.2**	1227	2	1	13750

Appendix

Tuning of the Adaptive Large Neighborhood Search Heuristic Presented in Chapter 2

In the following, we shortly describe the programming and tuning process of the Adaptive Large Neighborhood Search Heuristic for the PMCVRP.
As the first step, we implemented a random removal heuristic for removing nodes from the current solution and a greedy heuristic for reinserting the removed nodes in the solution. For tuning of ρ, we used an initial value of $\rho = 10$ and ran the algorithm for all PVRP instances. The average gap compared to the best known solution was 10.21 percent and the maximum gap was 41.87 percent. The performance of the random removal and of the greedy heuristics for some of the instances was acceptable, but compared to the best known solutions for most of the instances the gap was high. We selected the instances V-P4, V-P5, V-P8, V-P10, V-P12 to be used for further tuning because the gaps compared to the best known solution for these instances were high, and they differed with respect to the number of customers and periods. We changed the value of ρ and ran the algorithm for the selected instances. Finally, we selected $\rho = 0.15 * N'$ which gave the smallest average gap among the tested values. We also observed that for some of the instances, the number of routes was higher than the number of available vehicles and we implemented a route removal heuristic.

In the next step, we applied shaw and worst removal. The shaw removal was controlled by parameters p, η, γ and worst removal was controlled by parameter p that determines the degree of randomization in the heuristic. At first, the parameter values were set near to the values for similar parameters suggested in Røpke and Pisinger (2006) ($p = 3, \eta = 9, \gamma = 2$). Then the algorithm was executed for each of these heuristics separately. We repeated this procedure with different values of the parameters, and finally the values of the parameters for shaw and worst removal were determined based on the minimum average gap for all of the tuning instances ($p = 6$, $\eta = 6$, and $\gamma = 3$).

In the third step, we added the regret heuristics and we wrote a piece of code for selecting destroy and repair operators. In this step, we had to tune $\beta_1, \beta_2, \beta_3$, and λ. Again, we started with the suggested values of Røpke and Pisinger (2006) for these parameters and ran the algorithm with these values. We changed

the value of these parameters based on the performance of the algorithm, and we finally selected the values $\beta_1 = 7.5$, $\beta_2 = 5$, $\beta_3 = 2.5$, and $\lambda = 0.5$.

For acceptance or rejection of the new solutions, we used the acceptance rules of the simulated annealing. In this step, we needed to tune te_{init}, te_{final}, and κ. Initially we used $te_{init} = 1000$, $te_{final} = 0$, and $\kappa = 0.8$. After running the selected instances with different values of these parameters, we saw that with a decrease of te_{init} and a decrease of κ, the running time of the instances and the quality of the solutions were decreased and visa versa. Finally, the values of the parameters $te_{init} = 1500$, $te_{final} = 0$, and $\kappa = 0.995$ were selected as they provided a good balance between solution quality and computation time. The final version of the algorithm provided an average gap of 4.6 percent.

Tuning of the Heuristic Branch–and–Price Algorithm for the Multi–Compartment Inventory Routing Problem in Chapter 3

In the following, we shortly describe the programming and tuning process of the Heuristic Branch–and–Price Algorithm for the MCIRP presented in chapter 3.
As the first step, we implemented a labeling algorithm based on Desaulniers et al. (2015). The difference between our labeling algorithm and the one by Desaulniers et al. (2015) is related to our delivery policy, as in this research, we only considered full sub deliveries. We ran the algorithm for the instances of Archetti et al. (2014) and we observed that an increase in the number of nodes causes a significant increase in the running time of the algorithm. For instance, our labeling at this point could solve an instance with 20 customers and 3 vehicles in 5367.32 seconds.

In order to increase the algorithm speed, we added a heuristic based on modified dominance rules. This heuristic compared labels based on the remaining capacity of the compartments and the cost of the routes. We observed that this heuristic could only create a limited number of solutions and was not really efficient for improving the speed of the algorithm. Therefore we implemented the dominance heuristic which could generate a higher number of solutions and decreased the running time of the algorithm.

In the next step, we added 2 heuristics based on scaling of the capacity of the compartments and scaling of the distance in order to dominate more solutions. The running time for the root node for an instance with 25 customers, 3 periods, and 2 vehicles decreased to 606.344 seconds. In addition, 3767 columns were generated before the stopping condition was met.

In the third step, we implemented the friends heuristic. We deactivated the heuristics based on scaling of the capacity of the compartments and scaling of the distance and we ran the algorithm with the friends heuristic. The running time of the root node for an instance with 25 customers, 3 periods, and 2 vehicles decreased to 107.394 seconds and 7621 columns were generated before the stopping condition was met. As a result, we decided to discard the heuristics based on scaling of capacity and distance. We added the branching part to our algorithm and ran the instances of the Archetti et al. (2014).

In the last step, we ran the algorithm using a new dataset. We observed that for the new dataset, the algorithm was slow. Therefore, we added a heuristic named No-Semi heuristic, which enhanced the speed of the algorithm.

Bibliography

Achuthan, N. R., L. Caccetta, and S. P. Hill (2003). An improved branch-and-cut algorithm for the capacitated vehicle routing problem. *Transportation Science 37*(2), 153–169.

Alegre, J., M. Laguna, and J. Pacheco (2007). Optimizing the periodic pick-up of raw materials for a manufacturer of auto parts. *European Journal of Operational Research 179*(3), 736–746.

Andersson, H., A. Hoff, M. Christiansen, G. Hasle, and A. Løkketangen (2010). Industrial aspects and literature survey: Combined inventory management and routing. *Computers & Operations Research 37*(9), 1515–1536.

Angelelli, E. and M. G. Speranza (2002). The periodic vehicle routing problem with intermediate facilities. *European Journal of Operational Research 137*(2), 233–247.

Archetti, C., L. Bertazzi, G. Laporte, and M. G. Speranza (2007). A branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transportation Science 41*(3), 382–391.

Archetti, C., N. Bianchessi, S. Irnich, and M. G. Speranza (2014). Formulations for an inventory routing problem. *International Transactions in Operational Research 21*(3), 353–374.

Archetti, C., N. Bianchessi, and M. G. Speranza (2015). A branch-price-and-cut algorithm for the commodity constrained split delivery vehicle routing problem. *Computers & Operations Research 64*, 1–10.

Archetti, C., N. Boland, and M. G. Speranza (2014). A matheuristic for the multi-vehicle inventory routing problem. Technical report.

Archetti, C., A. M. Campbell, and M. G. Speranza (2014). Multicommodity vs. single-commodity routing. *Transportation Science*.

Avella, P., M. Boccia, and A. Sforza (2004). Solving a fuel delivery problem by heuristic and exact approaches. *European Journal of Operational Research 152*(1), 170–179.

Baldacci, R., E. Bartolini, A. Mingozzi, and A. Valletta (2011). An exact algorithm for the period routing problem. *Operations research 59*(1), 228–241.

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

Ben Abdelaziz, F., C. Roucairol, and C. Bacha (2002). Deliveries of liquid fuels to sndp gas stations using vehicles with multiple compartments. In *Systems, Man and Cybernetics, 2002 IEEE International Conference on*, Volume 1, pp. 478–483. IEEE.

Bertazzi, L., M. G. Speranza, and W. Ukovich (1997). Minimization of logistic costs with given frequencies. *Transportation Research Part B: Methodological 31*(4), 327–340.

Blakeley, F., B. Argüello, B. Cao, W. Hall, and J. Knolmajer (2003). Optimizing periodic maintenance operations for schindler elevator corporation. *Interfaces 33*(1), 67–79.

Boland, N., J. Dethridge, and I. Dumitrescu (2006). Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Operations Research Letters 34*(1), 58–68.

Brown, G. G., C. J. Ellis, G. W. Graves, and D. Ronen (1987). Real-time, wide area dispatch of mobil tank trucks. *Interfaces 17*(1), 107–120.

Brown, G. G. and G. W. Graves (1981). Real-time dispatch of petroleum tank trucks. *Management Science 27*(1), 19–32.

Carter, M. W., J. M. Farvolden, G. Laporte, and J. Xu (1996). Solving an integrated logistics problem arising in grocery distribution. *INFOR: Information Systems and Operational Research 34*(4), 290–306.

Chajakis, E. D. and M. Guignard (2003). Scheduling deliveries in vehicles with multiple compartments. *Journal of Global Optimization 26*, 43–78.

Chao, I., B. L. Golden, E. Wasil, et al. (1995). An improved heuristic for the period vehicle routing problem. *Networks 26*(1), 25–44.

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

Coelho, L. C., J.-F. Cordeau, and G. Laporte (2013). Thirty years of inventory routing. *Transportation Science 48*(1), 1–19.

Coelho, L. C. and G. Laporte (2013). A branch-and-cut algorithm for the multi-product multi-vehicle inventory-routing problem. *International Journal of Production Research 51*(23-24), 7156–7169.

Coelho, L. C. and G. Laporte (2015). Classification, models and exact algorithms for multi-compartment delivery problems. *European Journal of Operational Research 242*(3), 854–864.

Cordeau, J.-F., M. Gendreau, and G. Laporte (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks 30*(2), 105–119.

Cordeau, J.-F., D. Laganà, R. Musmanno, and F. Vocaturo (2015). A decomposition-based heuristic for the multiple-product inventory-routing problem. *Computers & Operations Research 55*, 153–166.

Cornillier, F., F. F. Boctor, G. Laporte, and J. Renaud (2007). An exact algorithm for the petrol station replenishment problem. *Journal of the Operational Research Society 59*(5), 607–615.

Cornillier, F., F. F. Boctor, G. Laporte, and J. Renaud (2008). A heuristic for the multi-period petrol station replenishment problem. *European Journal of Operational Research 191*(2), 295–305.

Dantzig, G.B. and P. Wolfe (1960). Decomposition principle for linear programs. *Operations Research 8*(1),

101–111.

Dantzig, G.B. and P. Wolfe (1961). The decomposition algorithm for linear programs. *Econometrica*, *29* (4), 767–778.

Dayarian, I., T. G. Crainic, M. Gendreau, and W. Rei (2013). An adaptive large neighborhood search heuristic for a multi-period vehicle routing problem. Technical report, Technical Report CIRRELT-2013-60, Interuniversity Research Center on Enterprise Networks, Logistics and Transportation (CIRRELT).

Derigs, U., J. Gottlieb, J. Kalkoff, M. Piesche, F. Rothlauf, and U. Vogel (2011). Vehicle routing with compartments: applications, modelling and heuristics. *OR spectrum 33*(4), 885–914.

Desaulniers, G., J. Desrosiers, and M. Solomon (Eds.) (2005) Column generation. *Springer*.

Desaulniers, G., J. G. Rakke, and L. C. Coelho (2015). A branch-price-and-cut algorithm for the inventory-routing problem. *Transportation Science 50*(3), 1060 – 1076.

Drummond, L. M., L. S. Ochi, and D. S. Vianna (2001). An asynchronous parallel metaheuristic for the period vehicle routing problem. *Future generation computer systems 17*(4), 379–386.

Eilon, S., C. Watson-Gandy, N. Christofides, and R. de Neufville (1974). Distribution management-mathematical modelling and practical analysis. *Systems, Man and Cybernetics, IEEE Transactions on* (6), 589–589.

El Fallahi, A., C. Prins, and R. W. Calvo (2008). A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Computers & Operations Research 35*(5), 1725–1741.

Francis, P. M., K. R. Smilowitz, and M. Tzur (2008). The period vehicle routing problem and its extensions. In *The vehicle routing problem: latest advances and new challenges*, pp. 73–102. Springer.

Fukasawa, R., H. Longo, J. Lysgaard, M. P. de Aragão, M. Reis, E. Uchoa, and R. F. Werneck (2006). Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical programming 106*(3), 491–511.

Gaudioso, M. and G. Paletta (1992). A heuristic for the periodic vehicle routing problem. *Transportation Science 26*(2), 86–92.

Hadjiconstantinou, E. and R. Baldacci (1998). A multi-depot period vehicle routing problem arising in the utilities sector. *Journal of the Operational Research Society*, 1239–1248.

Hamzadayi, A., S. Topaloglu, and S. Y. Kose (2013). Nested simulated annealing approach to periodic routing problem of a retail distribution system. *Computers & Operations Research 40*(12), 2893–2905.

Hemmelmayr, V. C., K. F. Doerner, and R. F. Hartl (2009). A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research 195*(3), 791–802.

Henke, T., M. G. Speranza, and G. Wäscher (2015). The multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research 246*(3), 730–743.

Irnich, S. and G. Desaulniers (2005) Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. Solomon (Eds.): *Column generation*. Springer. 33 - 65.

Lahyani, R., L. C. Coelho, M. Khemakhem, G. Laporte, and F. Semet (2015). A multi-compartment vehicle routing problem arising in the collection of olive oil in tunisia. *Omega 51*, 1–10.

Laporte, G., M. Gendreau, J. Y. Potvin, and F. Semet (2000). Classical and modern heuristics for the vehicle routing problem. *International transactions in operational research 7*(4-5), 285–300.

Lysgaard, J., A. N. Letchford, and R. W. Eglese (2004). A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming 100*(2), 423–445.

Mirzaei, S. and S. Wøhlk (2016). A branch-and-price algorithm for two multi-compartment vehicle routing problems. *EURO Journal on Transportation and Logistics*. Advance online publication. doi: 10.1007/s13676-016-0096-x.

Mjirda, A., B. Jarboui, R. Macedo, and S. Hanafi (2012). A variable neighborhood search for the multi-product inventory routing problem. *Electronic Notes in Discrete Mathematics 39*, 91–98.

Mjirda, A., B. Jarboui, R. Macedo, S. Hanafi, and N. Mladenović (2014). A two phase variable neighborhood search for the multi-product inventory routing problem. *Computers & Operations Research 52*, 291–299.

Moin, N., S. Salhi, and N. Aziz (2011). An efficient hybrid genetic algorithm for the multi-product multi-period inventory routing problem. *International Journal of Production Economics 133*(1), 334–343.

Muyldermans, L. and G. Pang (2010). On the benefits of co-collection: Experiments with a multi-compartment vehicle routing algorithm. *European Journal of Operational Research 206*(1), 93–103.

Norouzi, N., M. Sadegh-Amalnick, and M. Alinaghiyan (2015). Evaluating of the particle swarm optimization in a periodic vehicle routing problem. *Measurement 62*, 162–169.

Pecin, D., A. Pessoa, M. Poggi, and E. Uchoa (2014). Improved branch-cut-and-price for capacitated vehicle routing. In *International Conference on Integer Programming and Combinatorial Optimization*, pp. 393–403. Springer.

Popović, D., M. Vidović, and G. Radivojević (2012). Variable neighborhood search heuristic for the inventory routing problem in fuel delivery. *Expert Systems with Applications 39*(18), 13390–13398.

Rahimi-Vahed, A., T. G. Crainic, M. Gendreau, and W. Rei (2015). Fleet-sizing for multi-depot and periodic vehicle routing problems using a modular heuristic algorithm. *Computers & Operations Research 53*, 9–23.

Ralphs, T. K., L. Kopman, W. R. Pulleyblank, and L. E. Trotter (2003). On the capacitated vehicle routing problem. *Mathematical programming 94*(2-3), 343–359.

Ramkumar, N., P. Subramanian, T. Narendran, and K. Ganesh (2012). Mixed integer linear programming model for multi-commodity multi-depot inventory routing problem. *OPSEARCH 49*(4), 413–429.

Reed, M., A. Yiannakou, and R. Evering (2014). An ant colony algorithm for the multi-compartment vehicle routing problem. *Applied Soft Computing 15*, 169–176.

Repoussis, P. P., C. D. Tarantilis, and G. Ioannou (2007). A hybrid metaheuristic for a real life vehicle routing problem. In *Numerical Methods and Applications*, pp. 247–254. Springer.

Røpke, S. and D. Pisinger (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science 40*(4), 455–472.

Russell, R. and W. Igo (1979). An assignment routing problem. *Networks 9*(1), 1–17.

Russell, R. A. and D. Gribbin (1991). A multiphase approach to the period routing problem. *Networks 21*(7), 747–765.

Ryan, D. M. and B. A. Foster (1981). *An integer programming approach to scheduling.* In A. Wren (Eds.): *Computer scheduling of public transport urban passenger vehicle and crew scheduling*. North-Holland, Amsterdam. 269-280.

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *Principles and Practice of Constraint Programming-P98* , pp. 417–431. Springer.

Sindhuchao, S., H. E. Romeijn, E. Akçali, and R. Boondiskulchok (2005). An integrated inventory-routing system for multi-item joint replenishment with limited vehicle capacity. *Journal of Global Optimization 32*(1), 93–118.

Speranza, M. G. and W. Ukovich (1994). Minimizing transportation and inventory costs for several products on a single link. *Operations Research 42*(5), 879–894.

Speranza, M. G. and W. Ukovich (1996). An algorithm for optimal shipments with given frequencies. *Naval Research Logistics (NRL) 43*(5), 655–671.

Tan, C. and J. Beasley (1984). A heuristic algorithm for the period vehicle routing problem. *Omega 12*(5), 497–504.

Toth, P. and D. Vigo (2014). *Vehicle routing: Problems, methods, and applications.* MOS-SIAM Series on optimization.

Tseng, Y. Y., W. L. Yue, and M. A. Taylor (2005). The role of transportation in logistics chain. In *Proceedings of the Eastern Asia Society for Transportation Studies*, Volume 5, pp. 1657–1672.

Vanderbeck, F. (2000). *On Dantzig-Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm.* Operations Research *48*(1), 111-128.

Van der Bruggen, L., R. Gruson, and M. Salomon (1995). Reconsidering the distribution structure of gasoline products for a large oil company. *European Journal of Operational Research 81*(3), 460–473.

Vidović, M., D. Popović, and B. Ratković (2014). Mixed integer and heuristics model for the inventory routing problem in fuel delivery. *International Journal of Production Economics 147*, 593–604.

Wang, Q., Q. Ji, and C. H. Chiu (2014). Optimal routing for heterogeneous fixed fleets of multicompartment vehicles. *Mathematical Problems in Engineering 2014*.

DEPARTMENT OF ECONOMICS AND BUSINESS ECONOMICS

AARHUS UNIVERSITY
SCHOOL OF BUSINESS AND SOCIAL SCIENCES
www.econ.au.dk

PhD dissertations since 1 July 2011

2011-4 Anders Bredahl Kock: Forecasting and Oracle Efficient Econometrics

2011-5 Christian Bach: The Game of Risk

2011-6 Stefan Holst Bache: Quantile Regression: Three Econometric Studies

2011:12 Bisheng Du: Essays on Advance Demand Information, Prioritization and Real Options in Inventory Management

2011:13 Christian Gormsen Schmidt: Exploring the Barriers to Globalization

2011:16 Dewi Fitriasari: Analyses of Social and Environmental Reporting as a Practice of Accountability to Stakeholders

2011:22 Sanne Hiller: Essays on International Trade and Migration: Firm Behavior, Networks and Barriers to Trade

2012-1 Johannes Tang Kristensen: From Determinants of Low Birthweight to Factor-Based Macroeconomic Forecasting

2012-2 Karina Hjortshøj Kjeldsen: Routing and Scheduling in Liner Shipping

2012-3 Soheil Abginehchi: Essays on Inventory Control in Presence of Multiple Sourcing

2012-4 Zhenjiang Qin: Essays on Heterogeneous Beliefs, Public Information, and Asset Pricing

2012-5 Lasse Frisgaard Gunnersen: Income Redistribution Policies

2012-6 Miriam Wüst: Essays on early investments in child health

2012-7 Yukai Yang: Modelling Nonlinear Vector Economic Time Series

2012-8 Lene Kjærsgaard: Empirical Essays of Active Labor Market Policy on Employment

2012-9 Henrik Nørholm: Structured Retail Products and Return Predictability

2012-10 Signe Frederiksen: Empirical Essays on Placements in Outside Home Care

2012-11 Mateusz P. Dziubinski: Essays on Financial Econometrics and Derivatives Pricing

2012-12 Jens Riis Andersen: Option Games under Incomplete Information

2012-13 Margit Malmmose: The Role of Management Accounting in New Public Management Reforms: Implications in a Socio-Political Health Care Context

2012-14 Laurent Callot: Large Panels and High-dimensional VAR

2012-15 Christian Rix-Nielsen: Strategic Investment

2013-1 Kenneth Lykke Sørensen: Essays on Wage Determination

2013-2 Tue Rauff Lind Christensen: Network Design Problems with Piecewise Linear Cost Functions

2013-3 Dominyka Sakalauskaite: A Challenge for Experts: Auditors, Forensic Specialists and the Detection of Fraud

2013-4 Rune Bysted: Essays on Innovative Work Behavior

2013-5 Mikkel Nørlem Hermansen: Longer Human Lifespan and the Retirement Decision

2013-6 Jannie H.G. Kristoffersen: Empirical Essays on Economics of Education

2013-7 Mark Strøm Kristoffersen: Essays on Economic Policies over the Business Cycle

2013-8 Philipp Meinen: Essays on Firms in International Trade

2013-9 Cédric Gorinas: Essays on Marginalization and Integration of Immigrants and Young Criminals – A Labour Economics Perspective

2013-10 Ina Charlotte Jäkel: Product Quality, Trade Policy, and Voter Preferences: Essays on International Trade

2013-11 Anna Gerstrøm: World Disruption - How Bankers Reconstruct the Financial Crisis: Essays on Interpretation

2013-12 Paola Andrea Barrientos Quiroga: Essays on Development Economics

2013-13 Peter Bodnar: Essays on Warehouse Operations

2013-14 Rune Vammen Lesner: Essays on Determinants of Inequality

2013-15 Peter Arendorf Bache: Firms and International Trade

2013-16 Anders Laugesen: On Complementarities, Heterogeneous Firms, and International Trade

2013-17	Anders Bruun Jonassen: Regression Discontinuity Analyses of the Disincentive Effects of Increasing Social Assistance
2014-1	David Sloth Pedersen: A Journey into the Dark Arts of Quantitative Finance
2014-2	Martin Schultz-Nielsen: Optimal Corporate Investments and Capital Structure
2014-3	Lukas Bach: Routing and Scheduling Problems - Optimization using Exact and Heuristic Methods
2014-4	Tanja Groth: Regulatory impacts in relation to a renewable fuel CHP technology: A financial and socioeconomic analysis
2014-5	Niels Strange Hansen: Forecasting Based on Unobserved Variables
2014-6	Ritwik Banerjee: Economics of Misbehavior
2014-7	Christina Annette Gravert: Giving and Taking – Essays in Experimental Economics
2014-8	Astrid Hanghøj: Papers in purchasing and supply management: A capability-based perspective
2014-9	Nima Nonejad: Essays in Applied Bayesian Particle and Markov Chain Monte Carlo Techniques in Time Series Econometrics
2014-10	Tine L. Mundbjerg Eriksen: Essays on Bullying: an Economist's Perspective
2014-11	Sashka Dimova: Essays on Job Search Assistance
2014-12	Rasmus Tangsgaard Varneskov: Econometric Analysis of Volatility in Financial Additive Noise Models
2015-1	Anne Floor Brix: Estimation of Continuous Time Models Driven by Lévy Processes
2015-2	Kasper Vinther Olesen: Realizing Conditional Distributions and Coherence Across Financial Asset Classes
2015-3	Manuel Sebastian Lukas: Estimation and Model Specification for Econometric Forecasting
2015-4	Sofie Theilade Nyland Brodersen: Essays on Job Search Assistance and Labor Market Outcomes
2015-5	Jesper Nydam Wulff: Empirical Research in Foreign Market Entry Mode

2015-6 Sanni Nørgaard Breining: The Sibling Relationship Dynamics and Spillovers

2015-7 Marie Herly: Empirical Studies of Earnings Quality

2015-8 Stine Ludvig Bech: The Relationship between Caseworkers and Unemployed Workers

2015-9 Kaleb Girma Abreha: Empirical Essays on Heterogeneous Firms and International Trade

2015-10 Jeanne Andersen: Modelling and Optimisation of Renewable Energy Systems

2015-11 Rasmus Landersø: Essays in the Economics of Crime

2015-12 Juan Carlos Parra-Alvarez: Solution Methods and Inference in Continuous-Time Dynamic Equilibrium Economies (with Applications in Asset Pricing and Income Fluctuation Models)

2015-13 Sakshi Girdhar: The Internationalization of Big Accounting Firms and the Implications on their Practices and Structures: An Institutional Analysis

2015-14 Wenjing Wang: Corporate Innovation, R&D Personnel and External Knowledge Utilization

2015-15 Lene Gilje Justesen: Empirical Banking

2015-16 Jonas Maibom: Structural and Empirical Analysis of the Labour Market

2015-17 Sylvanus Kwaku Afesorgbor: Essays on International Economics and Development

2015-18 Orimar Sauri: Lévy Semistationary Models with Applications in Energy Markets

2015-19 Kristine Vasiljeva: Essays on Immigration in a Generous Welfare State

2015-20 Jonas Nygaard Eriksen: Business Cycles and Expected Returns

2015-21 Simon Juul Hviid: Dynamic Models of the Housing Market

2016-1 Silvia Migali: Essays on International Migration: Institutions, Skill Recognition, and the Welfare State

2016-2 Lorenzo Boldrini: Essays on Forecasting with Linear State-Space Systems

2016-3 Palle Sørensen: Financial Frictions, Price Rigidities, and the Business Cycle

2016-4 Camilla Pisani: Volatility and Correlation in Financial Markets: Theoretical Developments and Numerical Analysis

2016-5 Anders Kronborg: Methods and Applications to DSGE Models

2016-6 Morten Visby Krægpøth: Empirical Studies in Economics of Education

2016-7 Anne Odile Peschel: Essays on Implicit Information Processing at the Point of Sale: Evidence from Experiments and Scanner Data Analysis

2016-8 Girum Dagnachew Abate: Essays in Spatial Econometrics

2016-9 Kai Rehwald: Essays in Public Policy Evaluation

2016-10 Reza Pourmoayed: Optimization Methods in a Stochastic Production Environment

2016-11 Sune Lauth Gadegaard: Discrete Location Problems – Theory, Algorithms, and Extensions to Multiple Objectives

2016-12 Lisbeth Palmhøj Nielsen: Empirical Essays on Child Achievement, Maternal Employment, Parental Leave, and Geographic Mobility

2016-13 Louise Voldby Beuchert-Pedersen: School Resources and Student Achievement: Evidence From Social and Natural Experiments

2016-14 Mette Trier Damgaard: Essays in Applied Behavioral Economics

2016-15 Andrea Barletta: Consistent Modeling and Efficient Pricing of Volatility Derivatives

2016-16 Thorvardur Tjörvi Ólafsson: Macrofinancial Linkages and Crises in Small Open Economies

2016-17 Carlos Vladimir Rodríguez Caballero: On Factor Analysis with Long-Range Dependence

2016-18 J. Eduardo Vera-Valdés: Essays in Long Memory

2016-19 Magnus Sander: Returns, Dividends, and Optimal Portfolios

2016-20 Ioana Daniela Neamtu: Wind Power Effects and Price Elasticity of Demand for the Nordic Electricity Markets

2016-21 Anne Brink Nandrup: Determinants of Student Achievement and Education Choice

2016-22 Jakob Guldbæk Mikkelsen: Time-Varying Loadings in Factor Models: Theory and Applications

2016-23 Dan Nguyen: Formidability and Human Behavior: An Interdisciplinary Approach

2016-24 Martin Petri Bagger: Attention and Decision-Making: Separating Top-Down from Bottom-Up Components

2016-25 Samira Mirzaei: Optimization Algorithms for Multi-Commodity Routing and Inventory Routing Problems

ISBN: 9788793195547