**HEC Montréal**
Affiliée à l'Université de Montréal

# Models and Solution Algorithms for Production Routing Problems

par
Yossiri Adulyasak
Service de l'enseignement de la gestion des opérations et de la logistique

Thèse présentée à la Faculté des études supérieures et postdoctorales
en vue de l'obtention du grade de Philosophiæ Doctor (Ph.D.) en administration

Décembre 2012

**HEC Montréal**

Affiliée à l'Université de Montréal

Cette thèse intitulée:

**Models and Solution Algorithms for Production Routing Problems**

présentée par:

Yossiri Adulyasak

a été évaluée par un jury composé des personnes suivantes:

Jacques Roy (HEC Montréal)
président-rapporteur

Jean-François Cordeau (HEC Montréal)
codirecteur de recherche

Raf Jans (HEC Montréal)
codirecteur de recherche

Vedat Verter (McGill University)
membre du jury

Jonathan F. Bard (The University of Texas at Austin)
examinateur externe

Gilbert Laporte (HEC Montréal)
représentant du directeur

# Résumé

Un système de planification intégrée de la chaîne d'approvisionnement est un outil permettant d'optimiser conjointement plusieurs décisions de planification de manière à réaliser des économies liées à la coordination de différentes activités réalisées séquentiellement dans la chaîne. Au cours des dernières années, plusieurs entreprises ont mis en place de tels systèmes et ont réalisé des économies de plusieurs millions de dollars. La clé du succès repose dans le développement de systèmes pouvant non seulement produire des solutions de coût minimal, mais aussi être utilisés de manière rapide et efficace. Dans cette thèse, nous présentons des modèles et des algorithmes pour un problème combiné de production et de tournées de véhicules (PPT) se posant dans les chaînes d'approvisionnement comportant un site de production et plusieurs détaillants. Ce problème de planification opérationnelle est une généralisation du problème de gestion des stocks et tournées (PST) qui incorpore de plus les décisions liées à la production.

Bien que le PPT et le PST aient reçu beaucoup d'attention au cours de la dernière décennie, peu d'études ont proposé des algorithmes de résolution exacts et la plupart des études ont supposé la présence d'un seul véhicule. Nous traitons ici le cas à plusieurs véhicules et nous présentons des formulations avec et sans indice de véhicule pour résoudre le PPT et le PST sous deux politiques de gestion des stocks: celle du niveau maximal (NM) et celle du niveau cible (NC). Les formulations avec indice de véhicule sont renforcées en utilisant des contraintes brisant la symétrie tandis que les formulations sans indice de véhicule sont renforcées par l'ajout d'inégalités valides. Des algorithmes de séparation et coupes sont proposés pour résoudre les différentes formulations. Nous rapportons des résultats détaillés pour comparer les différentes formulations et nous analysons aussi les gains liés à l'utilisation du calcul parallèle.

Afin de résoudre de plus grandes instances, nous présentons aussi une heuristique efficace basée sur la recherche adaptative à grand voisinage (RAGV). L'idée centrale

de la RAGV est de détruire et de réparer une solution de manière répétitive afin de l'améliorer. Dans notre heuristique, les variables binaires associées aux décisions de production sont traitées par une technique d'énumération inspirée du branchement local et les décisions de distribution sont gérées par les opérateurs de la RAGV. Les décisions continues sont pour leur part obtenues en résolvant un problème de flot dans un réseau. Cette heuristique est adaptée pour résoudre de manière unifiée le PPT et le PST avec les politiques NM et NC. Elle fournit aussi des solutions de départ pour les algorithmes exacts. Des tests étendus ont été réalisés sur de grandes instances provenant de la littérature pour évaluer la performance de l'heuristique.

Puisque la demande future est souvent inconnue en pratique, nous traitons finalement le PPT avec demande incertaine. Le problème est modélisé comme un processus de décision en deux étapes où la première étape consiste à décider de la production et des tournées avant que la demande soit connue avec certitude, alors que la deuxième étape consiste à décider des quantités produites et livrées une fois que la demande est connue. Bien que les algorithmes de branchement et coupes développés précédemment puissent être adaptés pour traiter ce problème, il devient difficile à résoudre étant donné sa taille qui augmente rapidement avec le nombre de scénarios. Nous proposons plutôt deux reformulations de Benders qui sont incorporées à une méthode d'énumération pour résoudre le problème. Ces approches sont comparées à une extension de l'algorithme de branchement et coupes que nous avons proposé pour le problème déterministe. Nous discutons également des possibilités de réoptimisation offertes par la décomposition de Benders et qui peuvent être particulièrement utiles dans des environnements stochastiques.

**Mots clés:** système de planification intégrée de la chaîne d'approvisionnement; problème de production et tournées; problème de gestion des stocks et tournées; algorithmes de séparation et coupes; recherche adaptative à grand voisinage; demande incertaine; problème stochastique en deux étapes; décomposition de Benders.

# Abstract

An integrated supply chain operational planning system is a tool that is used to jointly optimize several planning decisions thereby capturing the additional benefits of coordination between sequential activities in the chain. In recent years, many companies have set up integrated planning systems and achieved multi-million cost savings. The key to success is an application that is not only able to produce solutions with minimal costs, but that can also be used in an effective and timely manner. In this thesis, we present solution algorithms for an integrated operational planning problem arising in a typical supply chain network with a production facility and multiple retailers, called the production routing problem (PRP). This problem is a generalization of the inventory routing problem (IRP) obtained by incorporating production lot-sizing decisions.

Although the PRP and the IRP have received much attention in the past decade, only a few studies have introduced exact algorithms to solve the problems and most studies assume a single vehicle due to the complexity of considering multiple vehicles. In our study, we address the multi-vehicle aspect and introduce multi-vehicle PRP and IRP formulations, with and without a vehicle index, to solve the problems under both the maximum level (ML) and order-up-to level (OU) inventory replenishment policies. The vehicle index formulations are further improved by using symmetry breaking constraints, while the non-vehicle index formulations are strengthened by several cuts. Branch-and-cut algorithms are proposed to solve the different formulations. We report extensive computational experiments to compare the two formulation schemes and further explore the performance of the best formulation in the context of parallel computing.

To handle larger instances, we also introduce an efficient heuristic based on the adaptive large neighborhood search (ALNS) framework. The basic idea of ALNS is to repeatedly destroy and repair the solution in the hope of achieving an improvement.

In our heuristic, binary variables representing setup and routing decisions are handled by an enumeration scheme inspired from local branching and by upper-level ALNS operators, respectively, while continuous variables indicating production, inventory and delivery quantity decisions are set by solving a network flow subproblem. This heuristic is adapted for the PRP and IRP with both the ML and OU policies and is used to determine initial solutions for the exact solution algorithms. Extensive computational experiments have been performed on PRP-ML benchmark instances from the literature to evaluate the performance of the algorithm on large instances.

As demand is often uncertain in practice, we finally address the PRP under demand uncertainty. The problem is modeled as a two-stage decision process, where the first stage consists of making setup and routing decisions before the realization of demand, and the second stage involves quantity decisions made when the demand becomes known. Although the previously developed branch-and-cut algorithms can be adapted to solve this problem, they may become inefficient when the number of possible scenarios increases. We propose two different Benders reformulation schemes which are implemented within a branch-and-cut framework to solve the problem. These approaches are also compared to an extension of the branch-and-cut algorithm that we proposed for the deterministic problem. We further discuss the reoptimization capabilities of the Benders approach which can be particularly useful in stochastic environments.

**Keyword:** Integrated supply chain planning; production routing; inventory routing; branch-and-cut; adaptive large neighborhood search; demand uncertainty; two-stage stochastic problem; Benders decomposition.

# Contents

# List of Tables

# List of Figures

# List of Appendices

# Acknowledgements

I would like to express my sincere gratitude to my supervisors, Professors Jean-François Cordeau and Raf Jans, for their support, guidance and constant understanding. They have been wise and kind mentors through my doctoral program and dissertation. Without their support, I could not imagine to have achieved so much. It has been a great honor and pleasure to work with them. Further, I would like to thank Professor Vedat Verter, the member of my dissertation committee, for his advice and suggestions at various stages during my Ph.D, and Professor Jonathan F. Bard, the external examiner, for his valuable comments that greatly helped improve my thesis.

I am grateful to my two supervisors, and to HEC Montréal, the GERAD and the Canada Research Chair in Logistics and Transportation for their financial support which provided me the greatest opportunity to pursue my Ph.D. studies.

I wish to thank the staff as well as my friends at the university, GERAD and CIRRELT for their wonderful assistance in all matters. Also, I would like to thank the RQCHP for providing high-performance computing facilities for my experiments. My Ph.D. life would have been much more difficult without them.

My sincere thanks go also to my former advisor, Dr. Manoj Lohatepanont, who introduced me to the fields of Operations Research and Logistics and has always been my great mentor.

Many many thanks to my friends in Thailand for their support and all Thai friends in Montréal for making my life enjoyable and eventful.

Most importantly, I would like to express my greatest gratitude to my family for their support and love which are invaluable to me.

# Chapter 1
# Introduction

In a typical supply chain which consists of sequential activities of production, storage and distribution, each individual process is often planned and optimized using predetermined decisions from its previous activity. For example, a production planner makes production lot-sizing decisions in order to minimize production and inventory costs at the production facility. The planned lot-sizing decisions are then used as inputs in subsequent steps of distribution planning. Since the decisions are limited by the plan of the former process, the benefits of coordination in the planning process have been left behind.

It has been over five decades since two classical problems in logistics, namely lot-sizing and vehicle routing, were introduced by Wagner and Whitin (1958), and by Dantzig and Ramser (1959), respectively. The lot-sizing problem (LSP) consists of making production lot-sizing and inventory decisions over a given planning horizon, while the vehicle routing problem (VRP) concerns the design of vehicle routes to make deliveries to customers in each period. Each of these problems has been the object of countless studies, but most of these focus on a single problem and very few address the integration of the two problems. As is well known in supply chain planning, however, focusing on cost minimization in one area of the supply chain often leads to higher costs in other areas. There is thus a strong incentive to integrate decision-making in closely related problems. Several success stories related to integrated inventory and distribution planning have been reported in the literature. For example, the Kellogg company implemented an integrated production and distribution planning system yielding \$35-40 million in potential savings (Brown et al., 2001), while Frito-Lay put into place an integrated production, inventory, distribution and routing system which led to a 10% decrease in logistics costs (Çetinkaya et al., 2009).

The focus of this study is on an integrated planning problem that involves production, inventory, distribution and routing decisions, called the production routing problem (PRP), which is an extension of the well-known inventory routing problem (IRP). The PRP is actually a combination of the LSP and VRP and becomes the IRP when the production decisions are dropped. Chandra and Fisher (1994) showed that double-digit percentage savings can be achieved by solving the PRP using even a simple heuristic procedure compared to sequentially solving the two separate planning problems. This brought more attention to the PRP in recent years. Due to the complexity of the problem, however, the majority of the research has focused on heuristic procedures.

As the benefits of the integrated problem depend on the quality and performance of the solution algorithm, this dissertation aims to provide efficient tools for the PRP under various restrictions, namely production, inventory and vehicle capacity. We first focus on developing exact algorithms for the PRP, thus filling an important gap in the literature. Although some exact algorithms have been proposed before, most of them assume a single vehicle and the multi-vehicle aspect, which is very important in practice, is often neglected. We look specifically at exact algorithms for the PRP with multiple vehicles. The approaches we develop can also be used to solve the IRP.

To handle large instances, we also introduce a heuristic based on the adaptive large neighborhood search (ALNS) framework introduced by Ropke and Pisinger (2006). The basic idea of the ALNS is to repeatedly destroy and repair a part of the current solution to obtain an improved solution using search operators. These operators are probabilistically selected based on empirical scores. In our procedure, the binary variables are handled by an enumeration technique and by the upper-level search operators of the ALNS and the continuous variables are set by solving a network flow problem. This procedure is called optimization-based adaptive large neighborhood search (Op-ALNS). The Op-ALNS heuristically solves the first part using several move operators, fixes the binary variables, and determines the optimal values of the remaining continuous variables in the second part by solving a minimum cost flow problem.

As the PRP is optimized over multiple periods, uncertainty of demand is a common issue in real-world operations and assuming deterministic demand can possibly lead to costly, sub-optimal solutions. To address this issue, we introduce the stochastic PRP (SPRP) with demand uncertainty in a two-stage decision process and develop an exact solution algorithm based on the Benders decomposition framework. We further discuss the reoptimization capability of the Benders algorithm which is particularly useful in stochastic environments.

The work presented in the dissertation has yielded three scientific articles as follows.

1. Adulyasak, Y., Cordeau, J.-F., Jans, R. Optimization-Based Adaptive Large Neighborhood Search for the Production Routing Problem. *Transportation Science* (Article in Advance), 2012.

2. Adulyasak, Y., Cordeau, J.-F., Jans, R. Formulations and Branch-and-Cut Algorithms for Multi-Vehicle Production and Inventory Routing Problems. GERAD Tech Rep. G-2012-14. 40 pages. Submitted to *INFORMS Journal on Computing* in April 2012 (Revision submitted in October 2012).

3. Adulyasak, Y., Cordeau, J.-F., Jans, R. Benders Decomposition for Production Routing under Demand Uncertainty. GERAD Tech Rep. G-2012-57. 35 pages. Submitted to *Operations Research* in October 2012.

At the beginning of the corresponding chapters, we provide details on the sections that were part of the articles.

The remainder of the thesis is organized as follows. Chapter 2 first provides a review of the related literature. Formulations and exact algorithms for the PRP, including extensive computational experiments are then presented in Chapter 3. Chapter 4 presents the Op-ALNS heuristic and the computational results. This is followed by the PRP under demand uncertainty in Chapter 5. Finally, Chapter 6 concludes the dissertation and discusses possible directions for future work.

# Chapter 2
# Literature Review

The broad categories of integrated production-distribution problems arising in operational supply chain planning are shown in Figure 2.1. We consider problems with a discrete time finite horizon and dynamic demand. The first problem is the integrated version of lot-sizing and distribution planning with direct shipments (see, e.g., Federgruen and Tzur (1999); Jin and Muriel (2009)). This problem determines production lot sizes over a discrete planning horizon in order to minimize total production, setup, inventory and direct shipment costs. This problem can also be seen as a two-level lot-sizing problem or a one-warehouse multi-retailer problem. The second integrated problem is the Inventory Routing Problem (IRP) which incorporates the routing aspect but disregards production lot-sizing decisions (see Andersson et al. (2010)). In the IRP, the production quantities are typically assumed to be known in advance. The problem is thus to determine the amounts to deliver to the customers and distribution plans including routing decisions over a planning horizon. The integrated production, inventory, distribution and routing problem is presented on the right. The problem is known under many different names: the integrated production and distribution problem (Chandra and Fisher, 1994; Fumero and Vercellis, 1999; Boudia and Prins, 2009; Armentano et al., 2011), the integrated production, inventory, distribution and routing problem (Bard and Nananukul, 2009a, 2010; Lei et al., 2006), or the production routing problem (Ruokokoski et al., 2010). We prefer to use the name "Production Routing Problem" (PRP) as it clearly highlights the analogy with the IRP. The PRP is actually a generalization of the IRP obtained by considering production decisions (i.e., production quantity and setup decisions). The PRP reduces to the integrated lot-sizing with direct shipment problem by dropping the routing aspect, and to the IRP by disregarding the production aspect.

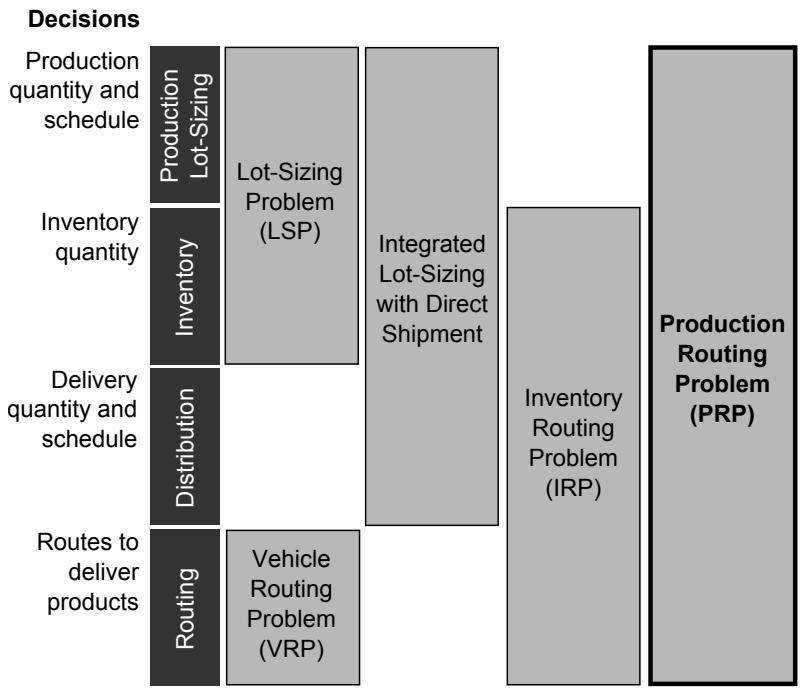



Figure 2.1: Supply Chain Planning Models

The main focus of this research is on the PRP which is the most general problem. In this chapter, we will briefly review the recent literature on the three categories of integrated production-distribution problems mentioned in Section 2.1. The subsequent sections will focus on the recent developments of the PRP. Section 2.2 presents a review of the PRP formulation schemes, followed by approaches to compute lower bounds in Section 2.3, exact solution algorithms in Section 2.4, heuristics algorithms in Section 2.5 and metaheuristics in Section 2.6. Finally, Section 2.7 discusses the PRP with demand uncertainty.

## 2.1 Integrated Production-Distribution Problems

We focus on the problems with a discrete time finite horizon which are closely related to our research. This section presents a brief overview of the three categories.

### 2.1.1 Integrated Lot-Sizing with Direct Shipment

We first provide a general overview of the integrated production and distribution planning without routing decisions. For consistency, our main focus here is the problem that incorporates the production aspect, e.g., production setup cost and/or setup time, and distribution decisions where the cost of delivery is customer specific and incorporates fixed costs (e.g., direct shipment vehicle dispatching cost) and/or unit costs of delivery. In each period, production can be made and the products are directly transported from the manufacturing plant to the customers. The products can be stored at the plant or at the customers incurring inventory holding costs in order to satisfy demands. The production, setup, inventory and shipment costs are minimized over the planning horizon. The network representation of this problem is shown in Figure 2.2.

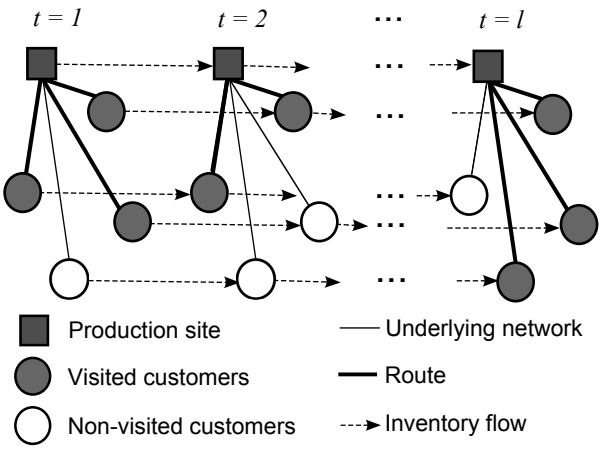



Figure 2.2: Integrated Lot-Sizing with Direct Shipments

The integrated production and direct shipment distribution planning was studied by some researchers. The problem incorporates the production setup cost and mostly considers the distribution cost as a fixed cost or a complex cost function. In the case of uncapacitated production and uncapacitated vehicles, this problem is also known as the one-warehouse multi-retailer problem (OWMR). Federgruen and Tzur (1999) considered the multi-item OWMR and developed a time-partitioning heuristic to solve the problem. In this heuristic, the planning horizon is partitioned into smaller

intervals and each small interval is solved by a Lagrangian relaxation based branch-and-bound approach. Solyalı and Süral (2012) proposed a new strong formulation based on the combined transportation and shortest path model to solve the OWMR with a single product. The simple truckload distribution problem for one product and one customer was studied by Alp et al. (2003). They considered a fixed transportation cost per container and employed a dynamic programming approach to solve the problem. Li et al. (2004) focused on a piecewise linear transportation cost function where the supplier has the option to deliver by direct delivery with truckload (TL) or less-than-truckload (LTL) transportation. They developed a dynamic programming approach to solve the one product and one customer problem. Jaruphongsa et al. (2007) proposed other dynamic programming algorithms to solve the problem with TL and LTL cost structures. The multi-item problem with one customer was considered by Lee et al. (2005). They developed a heuristic algorithm by using the property that a feasible flow is an extreme flow if it does not contain a loop in the network without arc capacities. A problem with a special cost structure, where the supplier can get a discount from transportation capacity reservation, was studied by van Norden and van de Velde (2005). A more general piece-wise linear transportation cost function was addressed by Rizk et al. (2006). They decomposed the integrated problem into uncapacitated lot-sizing and time-independent subproblems and applied a Lagrangian relaxation technique to obtain lower bounds. In the more general case of multiple customers, Chand et al. (2007) developed a dynamic programming algorithm to solve the problem in which backlogging is allowed. Jaruphongsa and Lee (2008) considered the problem with split delivery under time window restrictions and employed dynamic programming algorithms to solve the problem. A special problem of lot-sizing with truckload shipment where transshipments between the customers are allowed was considered by Herer and Tzur (2001).

There is a link between the lot-sizing with truckload cost structure and the classical lot-sizing with batch size where the batch quantity is smaller than the maximum production quantity in one period. The truck capacity can be viewed as the fixed batch quantity limit and the cost of dispatching one truck can also be considered as the production cost of one batch. There is also a link between the lot-sizing problem

with transshipments and the lot-sizing problem with production substitution where a product can be used to substitute for the demand of another product (Hsu et al., 2005). The cost of transshipment between customer locations can be viewed as the cost of production substitution.

### 2.1.2 Inventory Routing Problem (IRP)

When the routing aspect is included and the production aspect is disregarded, the problem is transformed into the integrated inventory distribution and routing problem. It is generally known as the inventory routing problem (IRP) (Andersson et al., 2010), which was extensively studied in the past decade and applied to land transportation and maritime logistics where the inventory plays an important role. Additionally, this problem arises in contexts where a vendor managed inventory (VMI) system is implemented. In a VMI system, a supplier or a vendor monitors its customers' inventory levels and makes the decision to replenish the products to its customers. This system transfers the inventory and ordering tasks from customers to the vendor in order to reduce the workload, lower costs and achieve a better efficiency through supply chain collaboration.

The network of the IRP is shown in Figure 2.3. The starting point is a warehouse where there is no production decision and a vehicle can visit more than one customer by traveling along its route. As a more generic version of the VRP, which consists of the decisions on delivery quantities and routes to serve customers, the decisions in the IRP also include the timing to serve the customers' demands. This makes the problem much more difficult than the classical VRP due to the complex periodic routing and inventory decisions. The IRP is obviously NP-hard since it contains the VRP as a special case (Coelho et al., 2012c).

The IRP first appeared in a gas delivery study by Bell et al. (1983). The problem was solved using a Lagrangian relaxation method and was decomposed by time period and by vehicle. Christiansen (1999) introduced an IRP application in a maritime context, called the inventory pickup and delivery problem, and applied a Dantzig-Wolfe decomposition and column generation approach to solve the problem. Carter

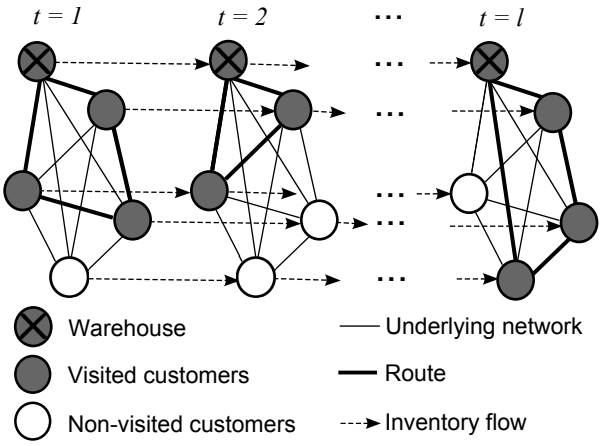



Figure 2.3: Inventory Routing Problem

et al. (1996) and Campbell and Savelsbergh (2004) proposed efficient heuristic procedures by decomposing the IRP into an allocation problem (AP) and a vehicle routing problem (VRP). Since the IRP is a complicated combinatorial problem, several metaheuristics, e.g., tabu search (Rusdiansyah and Tsao, 2005), genetic algorithm (Abdelmaguid and Dessouky, 2006), greedy randomized adaptive search procedure (GRASP) (Savelsbergh and Song, 2007), hybrid heuristic with combined tabu search and MIPs (Archetti et al., 2011), and adaptive large neighborhood search (ALNS) (Coelho et al., 2012a,c), have been proposed. Gaur and Fisher (2004) discussed a periodic IRP where the demand pattern is repeated and developed a heuristic to solve the problem. As mentioned in Andersson et al. (2010), few exact algorithms have been proposed to solve the IRP due to its complexity. Notable exceptions include a branch-and-cut procedures by Archetti et al. (2007) and Solyalı and Süral (2011) to solve the IRP with a single capacitated vehicle. Archetti et al. (2007) introduced several valid inequalities to solve the problem under three different inventory replenishment policies. In the first policy, called order-up-to level (OU), a visited customer receives exactly the amount which brings its inventory up to a predefined target stock level (TSL). The second policy, called maximum level (ML), allows delivery quantities to be any positive value but the inventory at each customer cannot exceed its maximum stock level. The third policy is similar to the ML policy but there is no

maximum stock level imposed at the customers. Solyalı and Süral (2011) strengthened the formulation of the IRP with the OU policy of Archetti et al. (2007) by using a shortest path network reformulation. Savelsbergh and Song (2008) considered the IRP with continuous moves and developed a branch-and-cut algorithm to solve the problem where the network consists of multiple plants and multiple customers.

### 2.1.3 Production Routing Problem (PRP)

The two integrated problems discussed in the previous sections each disregard one important aspect of the supply chain operational planning process, i.e., the integrated lot-sizing problem with direct shipment does not incorporate routing decisions, while the IRP disregards the production part. We present here recent developments on the integrated version of the lot-sizing, inventory, distribution and routing problem, which is referred to as the production routing problem (PRP) (Ruokokoski et al., 2010).

Figure 2.4 illustrates the structure of the PRP. The supply chain network consists of a production plant and multiple retailers, which we can consider as customers of the plant. Both the plant and the retailers have their own storage areas (e.g., warehouses) to keep finished products. The demand at each retailer has to be satisfied in every period of the planning horizon. In each period, the plant must decide whether or not to make the product and determine the corresponding lot size. If production does take place, this process incurs a fixed setup cost as well as unit production costs. In addition, the lot size cannot exceed the production capacity. In each period, deliveries are made from the plant to the retailers by a limited number of capacitated vehicles and routing costs are incurred. If products are stored at the plant or at the retailers, unit inventory holding costs are also incurred. Similar to the IRP, the problem is obviously NP-hard since it contains the VRP as a special case (Boudia et al., 2007; Archetti et al., 2011).

The PRP has also received more attention in recent years. The benefits of coordination in the PRP were first discussed by Chandra (1993) and Chandra and Fisher (1994). They showed that 3-20% cost savings can be achieved by solving the PRP compared to sequentially solving the separate problems. Similar to the IRP, most

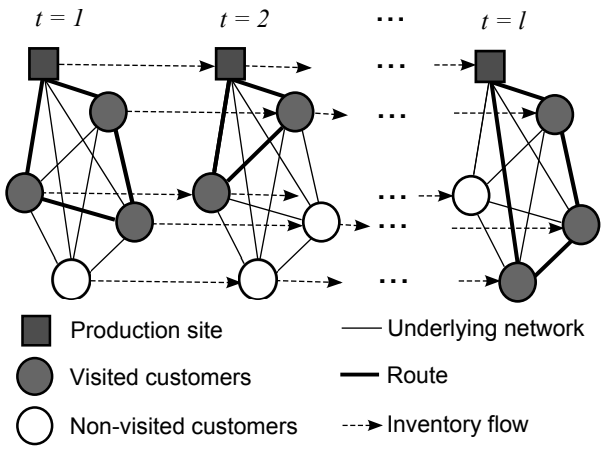



Figure 2.4: Production Routing Problem

of the previous studies employed heuristic procedures to solve the problem. Several metaheuristics, such as GRASP (Boudia et al., 2007), memetic algorithm (Boudia and Prins, 2009), and tabu search (Bard and Nananukul, 2009a; Armentano et al., 2011) have been employed to solve the PRP. Archetti et al. (2011) discussed the PRP under the ML and OU policies and developed an integer linear programming (ILP) heuristic to solve the problem. Bard and Nananukul (2009b, 2010) introduced a heuristic based on a branch-and-price framework to solve the PRP.

Due to its complexity, few studies have introduced exact algorithms or methods to compute strong lower bound for the PRP. Fumero and Vercellis (1999) and Solyalı and Süral (2009) developed a Lagrangian relaxation approach to obtain lower bounds based on the multi-commodity flow formulation. Ruokokoski et al. (2010) and Archetti et al. (2011) employed a branch-and-cut approach similar to that of Archetti et al. (2007) to solve the PRP. Ruokokoski et al. (2010) explored different lot-sizing reformulations for the PRP with uncapacitated production and a single uncapacitated vehicle, while Archetti et al. (2011) focused on the PRP with uncapacitated production and a single capacitated vehicle, and introduced several valid inequalities to solve the problem.

## 2.2 Production Routing Problem Formulations

We consider a single product PRP network that consists of a production plant and multiple customers. Since the PRP is an integrated version of the LSP and VRP, several formulation schemes can be used to model the problem. Recent reviews of the LSP and VRP can be found in Jans and Degraeve (2008) and Laporte (2009), respectively.

### 2.2.1 A Basic PRP Formulation

We present a general case where the PRP network is defined on a complete directed graph $G = (N, A)$ where $N$ represents the set of the plant and the customers indexed by $i \in \{0, ..., n\}$ and $A = \{(i, j) : i, j \in N, i \neq j\}$ is the set of arcs. The plant is represented by node 0 and we further define the set of customers $N_c = N \setminus \{0\}$ and the set of time periods $T = \{1, ..., l\}$. We denote by $u$ the unit production cost, $f$ the fixed production setup cost, $h_i$ the unit inventory holding cost at node $i$, and $c_{ij}$ the transportation cost from node $i$ to $j$. Let $C$ and $Q$ be the production and vehicle capacity, respectively. Let also $m$ be the maximum number of vehicles that can be dispatched in each period. The demand of customer $i$ in period $t$ is represented by $d_{it}$ and the maximum inventory level at node $i$ is represented by $L_i$. We further let $M_t = \min\left\{C, \sum_{j=t}^{l} \sum_{i \in N_c} d_{ij}\right\}$ and $\widetilde{M}_{it} = \min\left\{L_i, Q, \sum_{j=t}^{l} d_{ij}\right\}$.

The decision variables are defined as follows. Let $p_t$ be the production quantity in period $t$ and $y_t$ equal to one if there is production at the plant in period $t$, 0 otherwise. The inventory level at node $i$ at the end of period $t$ is represented by the variable $I_{it}$. We also use the parameter $I_{i0}$ to represent the initial inventory level available at node $i$ at the start of the planning horizon (period 0). In the distribution part, we define $q_{it}$ to be the quantity delivered to customer $i$ in period $t$ and $w_{it}$ to be the load of a vehicle before making a delivery to customer $i$ in period $t$. Let $z_{it}$ be equal to one if node $i$ is visited in period $t$, 0 otherwise and let $x_{ijt}$ be an arc variable, equal to one if a vehicle travels from node $i$ to $j$ in period $t$, 0 otherwise.

We first present a model based on the basic LSP and VRP formulations. It is also the most compact one in terms of the number of variables and constraints. The PRP

is formulated with variables that control the amounts delivered by a homogenous fleet of vehicles. A basic formulation based on that of Bard and Nananukul (2009b, 2010) is as follows.

$$\min \sum_{t \in T} \left( up_t + fy_t + \sum_{i \in N} h_i I_{it} + \sum_{(i,j) \in A} c_{ij} x_{ijt} \right) \tag{2.1}$$

s.t.

$$I_{0,t-1} + p_t = \sum_{i \in N_c} q_{it} + I_{0t} \qquad \forall t \in T \tag{2.2}$$

$$I_{i,t-1} + q_{it} = d_{it} + I_{it} \qquad \forall i \in N_c, \forall t \in T \tag{2.3}$$

$$p_t \leq M_t y_t \qquad \forall t \in T \tag{2.4}$$

$$I_{0t} \leq L_0 \qquad \forall t \in T \tag{2.5}$$

$$I_{i,t-1} + q_{it} \leq L_i \qquad \forall i \in N_c, \forall t \in T \tag{2.6}$$

$$q_{it} \leq \widetilde{M}_{it} z_{it} \qquad \forall i \in N_c, \forall t \in T \tag{2.7}$$

$$\sum_{j \in N} x_{ijt} = z_{it} \qquad \forall i \in N_c, \forall t \in T \tag{2.8}$$

$$\sum_{j \in N} x_{jit} = \sum_{j \in N} x_{ijt} \qquad \forall i \in N_c, \forall t \in T \tag{2.9}$$

$$\sum_{j \in N_c} x_{0jt} \leq m \qquad \forall t \in T \tag{2.10}$$

$$w_{it} - w_{jt} \geq q_{it} - \widetilde{M}_{it}(1 - x_{ijt}) \qquad \forall (i,j) \in A, \forall t \in T \tag{2.11}$$

$$0 \leq w_{it} \leq Q z_{it} \qquad \forall i \in N_c, \forall t \in T \tag{2.12}$$

$$p_t, I_{it}, q_{it} \geq 0 \qquad \forall i \in N, \forall t \in T \tag{2.13}$$

$$y_t, z_{it}, x_{ijt} \in \{0,1\} \qquad \forall i,j \in N, \forall t \in T. \tag{2.14}$$

The objective function (2.1) minimizes the total production, setup, inventory and routing costs. Constraints (2.2)-(2.6) represent the lot-sizing part of the problem. Constraints (2.2) and (2.3) are the inventory flow balance at the plant and customers, respectively. Constraints (2.4) are the setup forcing and production capacity constraints. The constraints force the setup variable to be one if production takes place in a given period and limit the production quantity to the minimum value

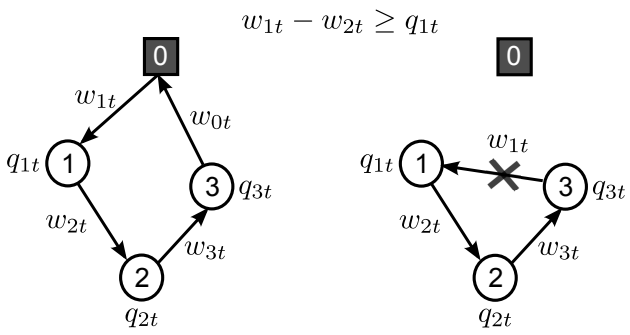


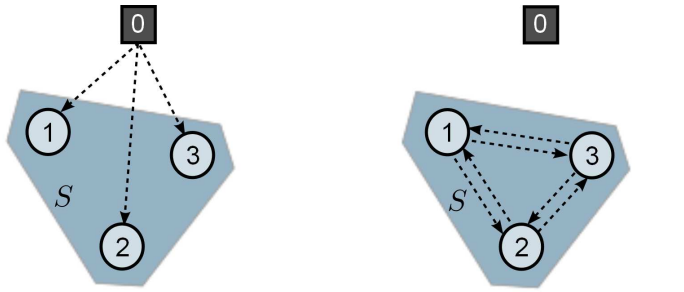

Figure 2.5: Vehicle Restriction and Subtour Elimination Constraints (2.11).

between the production capacity and the total demand in the remaining periods. Constraints (2.5) and (2.6) limit the maximum inventory at the plant and customers, respectively. The remaining sets of constraints, i.e., (2.7)-(2.12), are the vehicle loading and routing restrictions. Constraints (2.7) allow a positive delivery quantity only if customer $i$ is visited in period $t$ and each customer can be visited by at most one vehicle (2.8). Constraints (2.9) are the vehicle flow conservation. Constraints (2.10) limit the number of trucks that can be used to the number of available trucks. Constraints (2.11) are the vehicle loading restrictions and subtour elimination constraints in the form of the Miller-Tucker-Zemlin inequalities (Miller et al., 1960). These constraints do not allow taking an arc that generates a subtour as shown in Figure 2.5; the arc $(3, 1, t)$ cannot be taken because $w_{1t} - w_{2t} \geq q_{1t}$ is not valid. Constraints (2.12) are the vehicle capacity constraints.

In Bard and Nananukul (2009b, 2010), the subtour elimination constraints (2.11) and vehicle capacity (2.12) constraints are used. However, this subtour elimination constraint set can lead to a weak formulation in the routing part (Toth and Vigo, 2001). Constraints (2.11) and (2.12) can be replaced by other subtour elimination constraints, i.e.,

- fractional capacity constraints (FCCs) (Letchford and Salazar-González, 2006) as presented in Chandra and Fisher (1994):

$$\sum_{i \notin S} \sum_{j \in S} x_{ijt} \geq \sum_{i \in S} q_{it}/Q \quad \forall S \subseteq N_c : |S| \geq 1, \forall t \in T \tag{2.15}$$




$x_{01t} + x_{02t} + x_{03t} \geq 1$ $\quad$ $x_{12t} + x_{21t} + x_{13t} + x_{31t} + x_{23t} + x_{32t} \leq 2$

Figure 2.6: FCCs (left) and GFSECs (right) Illustration

- generalized fractional subtour elimination constraints (GFSECs):

$$\sum_{i \in S} \sum_{j \in S} x_{ijt} \leq |S| - \sum_{i \in S} q_{it}/Q \quad \forall S \subseteq N_c : |S| \geq 1, \forall t \in T. \tag{2.16}$$

Figure 2.6 depicts how these constraints eliminate subtours. Suppose that $(q_{1t} + q_{2t} + q_{3t})/Q = 1$. The FCC forces one arc from outside the set $S$ to connect to a node inside $S$, while the GFSEC does not allow more than two arcs in the set $S$. Both of the constraints can eliminate subtours in the set $S$.

These subtour elimination constraints are similar to the subtour elimination constraints for the VRP. However, unlike in the VRP where the delivery quantity to each customer is known a priori, the value $(\sum_{i \in S} q_{it})/Q$ cannot be rounded up in the PRP because the delivery quantity $q_{it}$ is a decision variable and this would result in a non-linear formulation. Additionally, by adding these constraints to the formulation, the problem becomes much larger due to an exponential number of subsets. A branch-and-cut procedure is generally used to solve the problem effectively. These constraints are initially removed and added iteratively during the branch-and-cut process. Note that in the case where an undirected graph is assumed, the formulations can be converted by using the method presented by Toth and Vigo (2001).

### 2.2.2 LSP and VRP Formulation Schemes

For the LSP, a review of basic formulation and reformulation schemes was presented by Pochet and Wolsey (2006). In short, the basic LSP formulation is a weak formulation that gives poor quality lower bounds. Many reformulation schemes can be used to strengthen the formulation. The major LSP reformulation schemes considered for the PRP include the shortest path (Eppen and Martin, 1987) and facility location (Krarup and Bilde, 1977) reformulations. In the uncapacitated LSP, these two formulations have the integrality property, i.e., feasible mixed-integer solutions are obtained by solving the LP relaxation.

The formulation schemes for the VRP follow the classification presented by Toth and Vigo (2001). The different formulations are typically used to solve the problems with different characteristics. For example, the basic formulation is a compact formulation that is suitable for a homogeneous fleet and a limited number of vehicles. To handle a heterogeneous fleet (e.g., various fleet size, consumption, speed, or costs), one should employ a formulation in which a vehicle index can be incorporated, e.g., a multi-commodity flow formulation. In some VRP applications in which the vehicle routes are predefined or have few possibilities, e.g., a truck can only serve the customers in the same cluster, or in maritime applications where the vessel schedule option is predefined, it is more appropriate to use a set-partitioning formulation. This formulation is also generally used as a master problem in the column generation process. Another issue in VRP formulations is the set of subtour elimination constraints. Applying different subtour elimination constraints can result in different lower bounds (Letchford and Salazar-González, 2006). Table 2.I provides the summary for PRP formulations that appeared in previous studies.

Some formulations were used in the PRP to deal with specific issues, e.g., the heterogeneous fleet by Lei et al. (2006) and the column generation approach by Bard and Nananukul (2009b, 2010). The efficiency of different LSP reformulations in PRP was studied by Ruokokoski et al. (2010).

Table 2.I: Production Routing Problem Formulation Schemes

| **LSP →**<br>**VRP ↓** | **Basic LSP** | **Facility Location** | **Shortest Path** |
|---|---|---|---|
| **Basic VRP** | Chandra and Fisher (1994)<br>Bard and Nananukul (2010, 2009b)<br>Ruokokoski et al. (2010) | Ruokokoski et al. (2010) | Ruokokoski et al. (2010) |
| **Vehicle Index** | Archetti et al. (2011) | Boudia et al. (2007, 2008)†<br>Boudia and Prins (2009)† | - |
| **Multi-Commodity Flow** | Fumero and Vercellis (1999)<br>Lei et al. (2006)<br>Solyalı and Süral (2009)<br>Armentano et al. (2011) | - | - |
| **Set Partitioning** | Bard and Nananukul (2009b, 2010) | - | - |

†Facility location reformulation in the distribution echelon only

#### 2.2.2.1 Comparison Between LSP Formulations

Ruokokoski et al. (2010) investigated the quality of lower bounds by using different LSP reformulation schemes compared to the basic LSP formulation on the PRP with uncapacitated production and a single uncapacitated vehicle. The computational results show that the LP relaxation values of the shortest path and facility location reformulations when the subtour elimination constraints are dropped are much improved compared to the basic formulation. The LP bound obtained by the shortest path reformulation is greater than or equal to the LP bound by the facility location reformulation which follows from the proof in Solyalı and Süral (2012), but the difference between the bounds is very small. The facility location reformulation provides better computational performance when adding the valid inequalities for the routing problem (including the subtour elimination constraints) and solving the integer problem using a branch-and-cut process.

#### 2.2.2.2 Comparison Between VRP Formulations

The VRP reformulation scheme is generally used to deal with different routing problem characteristics. Since there is no research that focuses on the different routing formulations for the PRP, we briefly review the VRP literature. An overview of

the alternative formulations for the VRP can be found in Toth and Vigo (2001). Letchford and Salazar-González (2006) discussed the lower bound quality provided by non-vehicle index and vehicle index VRP formulations. With certain families of valid inequalities (e.g., FCCs), the vehicle index VRP formulation gives the same bound as the non-vehicle index in the VRP. In practice, the non-vehicle index formulation is preferable because it contains fewer variables. The vehicle index formulation is necessary when the vehicles are not identical or when the specific cuts for the vehicle index formulation are used. The multi-commodity flow formulation provides an advantage to eliminate subtours through flow conservation constraints, which do not grow exponentially with the number of customers. The LP relaxation lower bound of the multi-commodity flow formulation is equal to the LP relaxation of the non-vehicle index VRP with FCCs. For the set partitioning formulation, when only elementary (feasible) routes are permitted, the LP relaxation of the formulation is also at least as strong as the LP relaxation of the multi-commodity flow formulation. However, the number of variables in this formulation grows exponentially with the size of the problem. One must trade-off lower bound quality with computing time due to the problem size.

## 2.3 Approaches to Compute Lower bounds

Since the PRP is a very complicated combinatorial problem and contains a large number of binary variables, the quality of the lower bound of the basic PRP formulation obtained by solving the LP relaxation is generally very poor. Nananukul (2008) solved the LP relaxation of the basic formulation in Section 2.2.1 with small instances with 3-8 periods and 5-20 customers and reported gaps of 30%-260% compared to the optimal solutions. The LP relaxation of this formulation is not practical in providing relaxed solutions in exact algorithms such as branch-and-bound or measuring the quality of other solution approaches. Hence, alternative relaxation methods have to be developed to obtain better lower bounds.

### 2.3.1 Lagrangian Relaxation

Lagrangian relaxation (see Fisher (1981)) is an approach to obtain lower bounds by dualizing constraints with Lagrangian multipliers and decomposing the problem into subproblems which are more easily solvable. A Lagrangian relaxation for a variant of the PRP, where unit transportation costs are assumed, was proposed by Fumero and Vercellis (1999). They decomposed the basic LSP and the multi-commodity flow VRP reformulation into subproblems by dualizing the plant inventory constraints and the vehicle capacity constraints. The problem is then transformed into four subproblems, i.e., production (PROD), inventory (INV), distribution (DIS), and routing (ROU) subproblems. The first two subproblems can be solved by inspection and the DIS subproblem can be solved by an LP solver. The lower bound of the ROU subproblem is calculated by the minimum cost network flow problem. Instances with up to 8 periods, 12 customers and 10 products were tested and the algorithm could obtain solutions with an average optimality gap of 5.5%.

A similar Lagrangian relaxation approach was used by Solyalı and Süral (2009) to solve the PRP with the order-up-to level (OU) policy. However, the lower bounds obtained by this approach were weak. On the instances with 8 customers and 5 periods, the lower bound produced by the Lagrangian relaxation has an average deviation of 33.16% from the optimal value. They also tested the performance of the formulation based on the multi-commodity fixed charge network flow problem using the same instances and the longest computing time was approximately 20 hours to obtain the optimal solution.

### 2.3.2 Column Generation

In a column generation procedure, a basic formulation is decomposed into a restricted master problem (RMP) and subproblems. The original variables are replaced by a convex combination of extreme points of the subproblems which are generated and added iteratively by solving the subproblems. More details about recent general column generation approaches can be found in Lübbecke and Desrosiers (2005).

Bard and Nananukul (2010) proposed a RMP and subproblem formulations for the PRP and developed a branch-and-price procedure. Let $R(t)$ be the sets of delivery plans in period $t$ where a delivery plan, indexed by $r$, is characterized by the delivery quantity to each customer and routing decisions. The binary variable $\theta_{rt}$ is equal to one if the delivery plan $r$ for period $t$ is selected. The parameter $c_{rt}$ is the total cost of using delivery plan $r$ in period $t$, and $\mu_{rt}^i$ is the amount delivered to customer $i$ with delivery plan $r$ in period $t$. The RMP is formulated as follows.

$$\min \sum_{t \in T} \left( up_t + fy_t + \sum_{i \in N} h_i I_{it} + \sum_{r \in R(t)} c_{rt} \theta_{rt} \right) \tag{2.17}$$

s.t. (2.4), (2.5)-(2.6), (2.13) and

$$I_{0,t-1} + p_t = \sum_{i \in N_c} \sum_{r \in R(t)} \mu_{rt}^i \theta_{rt} + I_{0t} \quad \forall t \in T \tag{2.18}$$

$$I_{i,t-1} + \sum_{r \in R(t)} \mu_{rt}^i \theta_{rt} = d_{it} + I_{it} \quad \forall i \in N_c, \forall t \in T \tag{2.19}$$

$$\sum_{r \in R(t)} \theta_{rt} \leq 1 \quad \forall t \in T \tag{2.20}$$

$$\theta_{rt} \in \{0, 1\} \quad \forall t \in T, \forall r \in R(t). \tag{2.21}$$

The objective function (2.17) and constraints (2.18)-(2.19) are equivalent to (2.1) and (2.2)-(2.3), respectively. At most one delivery plan can be selected in each period (2.20). In the column generation process, the integrality of the variables $\theta_{rt}$ is relaxed to obtain the lower bounds of the RMP.

The subproblem is the delivery schedule generator. Denote by $\alpha_t^1, \alpha_{it}^2$ and $\alpha_t^3$ the dual variables of the RMP associated with the constraints (2.18), (2.19) and (2.20), respectively. The subproblem is decomposed into a VRP subproblem in each time period. The subproblem for time period $t$, referred to as $SVRP_t$, is as follows.

$$\min \sum_{(i,j) \in A} c_{ij} x_{ijt} - \sum_{i \in N_c} (\alpha_t^1 + \alpha_{it}^2) q_{it} + \alpha_t^3 \tag{2.22}$$

s.t. (2.13)-(2.14) and

$$q_{it} \leq \min\{Q, \sum_{j=t}^{l} d_{ij}\} z_{it} \quad \forall i \in N_c \tag{2.23}$$

$$\sum_{j \in N} x_{ijt} = z_{it} \quad \forall i \in N_c \tag{2.24}$$

$$\sum_{j \in N} x_{jit} = \sum_{j \in N} x_{ijt} \quad \forall i \in N_c \tag{2.25}$$

$$\sum_{j \in N_c} x_{0jt} \leq m \tag{2.26}$$

$$w_{it} - w_{jt} \geq q_{it} - \widetilde{M}_{it}(1 - x_{ijt}) \quad \forall (i,j) \in A \tag{2.27}$$

$$0 \leq w_{it} \leq Q \quad \forall i \in N_c. \tag{2.28}$$

Constraints (2.23)-(2.28) are equivalent to constraints (2.7)-(2.12).

Since $SVRP_t$ employs the VRP structure, it is very time consuming to solve the problem to optimality. Therefore, Bard and Nananukul (2009b, 2010) developed a heuristic to handle this subproblem.

## 2.4 Exact Solution Algorithms

Exact solution algorithms for PRP are very scant. To the best of our knowledge, only two algorithms were proposed to solve the PRP: the branch-and-cut of Archetti et al. (2011) and the branch-and-cut with strong reformulation of Ruokokoski et al. (2010).

### 2.4.1 Branch-and-Cut Algorithm by Archetti et al. (2011)

Archetti et al. (2011) studied the PRP with uncapacitated production and a single capacitated vehicle. The formulation is similar to the basic LSP formulation but the subtour elimination constraints are in the generalized form of the traveling salesman problem (TSP) that is used by Archetti et al. (2007) for the IRP and Gendreau et al. (1998) for the selective TSP as follows:

$$\sum_{i \in S} \sum_{j \in S} x_{ijt} \leq \sum_{i \in S} z_{it} - z_{et} \qquad \forall S \subseteq N_c : |S| \geq 2, \forall e \in S, \forall t \in T. \tag{2.29}$$

These constraints are used in place of constraints (2.11)-(2.12) in the basic formulation. The authors also add the following inequalities as presented in Archetti et al. (2007) for the inventory routing problem:

$$I_{i,t-s-1} \geq \sum_{j=0}^{s} d_{i,t-j} \left(1 - \sum_{j=0}^{s} z_{i,t-j}\right) \qquad \forall i \in N_c, \forall t \in T, s = 0, 1..., t-1 \tag{2.30}$$

$$z_{it} \leq z_{0t} \qquad \forall i \in N_c, \forall t \in T \tag{2.31}$$

$$x_{ijt} \leq z_{it} \text{ and } x_{ijt} \leq z_{jt} \qquad \forall (i,j) \in A, \forall t \in T. \tag{2.32}$$

The inequalities (2.30) can be interpreted as follows: if there is no shipment delivered during the time interval $[t-s, t]$, the inventory level in period $t-s-1$ must be sufficient to satisfy the demand in this interval. These constraints are similar to the so-called $(l, s, WW)$ inequalities (Pochet and Wolsey, 2006) imposed at each customer separately. By adding them to the PRP, they could strengthen the lot-sizing part of the customer replenishment and provide better lower bounds. The inequalities (2.31) and (2.32) are logical constraints. Constraints (2.31) impose that a customer cannot be visited if the vehicle is not dispatched, while constraints (2.32) impose that the arc $(i, j)$ can be greater than zero only if nodes $i$ and $j$ are visited.

Archetti et al. (2011) also presented other sets of inequalities for the PRP with uncapacitated production as follows:

$$I_{t-1} \leq \sum_{i \in N_c} \sum_{j=t}^{l} d_{ij}(1 - y_t) \quad \forall t \in T \tag{2.33}$$

$$p_t \geq \frac{f}{h_0 j}(y_{t-j} + y_t - 1) \quad \forall 2 \leq t \leq l, \forall 1 \leq j \leq t-1. \tag{2.34}$$

Constraints (2.33) are valid for the uncapacitated production because it is not optimal to produce when the plant has a positive inventory level. Constraints (2.34) enforce a minimum production quantity if production takes place. These constraints together with the other two constraints that are used to prevent a stockout (see Archetti et al. (2011)) are all added a priori to the formulation.

In the branch-and-cut process, the subtour elimination constraints (2.29) are removed and only the violated cuts are added iteratively during the branching process. The performance of the algorithm is tested on generated test instances with 14 customers and 6 periods. Most of the instances are solved to optimality within a few seconds.

### 2.4.2 Branch-and-Cut Algorithm by Ruokokoski et al. (2010)

The uncapacitated lot-sizing problem and a single uncapacitated truck is considered by Ruokokoski et al. (2010). They use the stronger LSP reformulations, e.g., facility location and shortest path reformulations, to solve the PRP. The formulation is similar to that of Archetti et al. (2011) but the subtour elimination constraints (2.29) are replaced with

$$\sum_{i \notin S} \sum_{j \in S} x_{ijt} + \sum_{i \in S} \sum_{j \notin S} x_{ijt} \geq 2 z_{et} \qquad \forall S \subseteq N_c : |S| \geq 2, \forall e \in S, \forall t \in T. \tag{2.35}$$

The authors used a heuristic and an exact separation procedure based on a minimum $s-t$ cut problem to detect other violated sets. They also adapted the generalized comb and 2-matching inequalities presented by Fischetti et al. (1997) which were developed for a generalized traveling salesman problem (GTSP) to the PRP. The results show that when all valid inequalities are used, the facility location LSP reformulation provides the best LP relaxation. This algorithm can solve problems with 80 customers and 8 periods within about 30 minutes.

## 2.5 Heuristic Algorithms

### 2.5.1 Decomposition-Based Heuristics

This section presents a review of the heuristic methods that were developed to solve the PRP by decomposing the problem into production and distribution planning subproblems. The initial solution is obtained by sequentially solving each problem and a heuristic procedure is called to improve the solutions. This approach was first

introduced by Chandra (1993) and Chandra and Fisher (1994) to solve the multi-product PRP. The integrated problem is decoupled into the capacitated lot-sizing problem and the distribution scheduling problem. The lot-sizing problem is solved to optimality and a distribution schedule for each period is produced by applying a simple heuristic together with a 3-opt procedure (Lin, 1965). The result is further improved by allowing production shifting across periods if the total cost is reduced. This heuristic algorithm provides approximately 6% cost savings compared to the uncoordinated approach with no improvement heuristic procedure in the small test instances.

Boudia et al. (2008) developed an improved decomposition based approach by first determining production lot sizes as large as possible to cover some future periods. The distribution plan in each period is constructed by the savings algorithm (Clarke and Wright, 1964). The algorithm finds opportunities to reduce production costs by adopting the Wagner-Whitin algorithm (Wagner and Whitin, 1958) for the LSP. Then, a local search procedure based on 3-opt moves, insertion, and swap heuristics are called to improve the solution. The algorithm is tested on the large instances generated by Boudia et al. (2005) with 50-200 customers and 20 periods. It provides 10%-15% cost savings compared to the two-phase decoupled heuristic, called H1, which basically provides a solution from the production plan identified by the Wagner-Whitin method, and the delivery plans generated by a 3-opt procedure.

Instead of focusing only on the lot-sizing part in the first phase of the algorithm, Lei et al. (2006) incorporated the distribution part and developed a heuristic algorithm to find a production-distribution plan. The delivery plan in each period is later determined by partitioning demands in each period and constructing shortest path networks to solve the route plans sequentially for each truck. They consider the problem with a heterogeneous fleet of vehicles. In test instances with 8-12 customers, 5 vehicles, and 3-4 time periods, this algorithm provides solutions within a second with a 2% average gap compared to CPLEX solutions in two hours.

Archetti et al. (2011) decomposed the PRP into the uncapacitated lot-sizing and inventory-routing subproblems and developed a heuristic to solve the decomposed problems. The algorithm starts by fixing production quantities equal to the demand in

each period and solving the IRP by a heuristic procedure. In this process, each retailer is selected sequentially and a search tree is solved to determine the time periods and vehicles used to serve that retailer. After that, the uncapacitated lot-sizing subproblem is solved to further explore whether the production plan can be improved by shifting some production quantity to reduce the production and inventory costs. A heuristic procedure is applied to the current solution to obtain further improvements by removing two retailers, and then a problem is formulated to find the minimum insertion cost of these retailers. If the total cost is reduced, the uncapacitated LSP subproblem is solved again and the process is repeated until there is no improvement. The authors evaluated the performance of this algorithm by comparing it to the best solutions found by the exact branch-and-cut solution procedure as described in Section 2.4.1 on instances with 14 customers, 6 periods and one vehicle. This heuristic provides solutions within 1% of optimality in a few seconds.

### 2.5.2 Branch-and-Price Heuristic

Bard and Nananukul (2009b, 2010) presented a heuristic based on the branch-and-price framework using the RMP and subproblems as described in Section 2.3.2. The branch-and-price scheme is a decomposition based procedure which involves a branching process. At each branching node, starting from the initial solution, variables in the RMP are fixed and column generation is performed to add variables to the RMP and solve it again until an optimal solution is found. Then, the branching process continues until an optimal solution to the original problem is obtained. The readers are referred to Barnhart et al. (1998) for more details on branch-and-price.

The branching process starts with the production setup variables ($y_t$) until all these variables have integer values. Subsequently, the variables $\theta_{rt}$ are considered. Branching on $\theta_{rt}$ directly, however, results in an unbalanced branching tree. When the variable $\theta_{rt}$ is set to one, the delivery plan $r$ is used and all corresponding $q_{it}$ and $x_{ijt}$ variables are fixed. But when $\theta_{rt} = 0$, it is very difficult to manage the variables to only exclude the delivery plan $r$. Therefore, it is more appropriate to branch on the $x_{ijt}$ variables. This branching scheme is similar to the branch-on-edge approach presented in Bramel and Simchi-Levi (2001). The depth-first-search strategy is used to quickly

find the incumbent solution. To improve the branch-and-price procedure, several features are included in the process. First, an initial solution is generated by the tabu search heuristic presented by Bard and Nananukul (2009a). Second, during the column generation process, instead of solving the subproblems to optimality, they are solved by the separation based heuristic algorithms of Bard and Nananukul (2009b). Third, the branching scheme is modified to branch on groups of variables. And fourth, a rounding heuristic procedure is used. With these modifications, the performance of the branch-and-price process is substantially improved. The experiments on instances with up to 50 customers and 8 periods showed that this branch-and-price heuristic provides better solution quality compared to those obtained by CPLEX (solution costs are improved by 12.2% on average) within one hour of computing time.

## 2.6 Metaheuristics

Some decomposition based heuristic approaches require relatively short runtimes but provide poor solution quality because the initial solutions are created based on decoupling procedures which do not take into account the benefits of the production and distribution coordination. To overcome this obstacle, metaheuristics can be applied to further explore the solution space and provide better solutions. In this section, we present several metaheuristics that have been developed to solve the PRP.

### 2.6.1 Greedy Randomized Adaptive Search Procedure (GRASP)

We refer to Feo and Resende (1995) for an overview of GRASP. Basically, the procedure consists of two main phases, i.e., construction and local search. In the construction phase, an initial solution is provided via an adaptive randomized greedy algorithm. Then the local search phase is applied to improve the solution.

A GRASP for the PRP was developed by Boudia et al. (2007). In the construction phase, an initial solution is generated by sequentially developing a production and delivery plan. Starting from the first period onwards, the production plan is preliminarily determined by producing a sufficient amount to cover the demand in

the period without excess production which creates inventory at the plant. Then, delivery routes in the period are constructed by an insertion algorithm. The demands with the cheapest insertion costs are stored and one of them is randomly drawn to be inserted in each iteration. After all demands are satisfied, the algorithm finds the customer demands in the earliest future period which can be satisfied by the residual production capacity, the vehicle capacity, and the customers' storage. The insertion process is again performed to improve the solution and the saving heuristic is called to find a better routing solution. All routing plans are fixed and the production plan improvement algorithm is applied to shift production quantities to combine with a production plan in earlier periods if the cost is lower, i.e., the incurred storage cost at the plant is less than the setup cost. In the local search phase, the routing plan of each period is improved by using a 3-opt procedure, inserting, and swapping. Moves across periods are also considered if the cost can be reduced. Boudia et al. (2007) also developed a path relinking procedure (see Glover (1996)) as a post-processor. In this process, solutions obtained during the GRASP are ranked according to their total cost and a limited number of solutions are stored in a pool of elite solutions. Then, any two solutions in the pool are chosen to create a new solution by transferring some delivery quantities in one of these solutions to another period according to the delivery quantities in the other solution to reduce the differences between these two solutions. This process could slightly improve the solutions obtained by GRASP.

### 2.6.2 Memetic Algorithm (MA)

Informally speaking, a MA is an improved genetic algorithm (GA). The basic idea of the genetic algorithm is to generate new solutions from a population of initial solutions which are represented by chromosomes (or bitstrings) using natural evolution, i.e., crossover or mutation, to create new offsprings. In a memetic algorithm, a local search procedure is additionally applied to improve both the initial population and offsprings of the genetic algorithm. This approach was first introduced by Moscato (1999). Boudia and Prins (2009) applied this approach to the PRP with an additional function to control the population with a threshold policy. The new solutions found are accepted only if they improve the current solution more than a threshold value.

This approach is called memetic algorithm with population management (MA|PM) (Sörensen and Sevaux, 2006).

In the study of Boudia and Prins (2009), solutions are represented by a tuple of trips, shipping quantities, and production plans. An initial population is created through a simple heuristic procedure that preliminarily sets a production plan in each period equal to the total demand, then a savings heuristic is used to generate the delivery plan and the production plan is later adjusted by the Wagner-Whitin algorithm. Offsprings of the population are generated by crossing over the randomly selected parents. The local search procedure, similar to the algorithm in GRASP by Boudia et al. (2007), is then called to improve the offspring solutions. Those offsprings are included in the population only if they reduce the cost more than the constant diversity threshold. The algorithm terminates after the maximum number of iterations is reached.

### 2.6.3 Tabu Search

The concept of tabu search was introduced by Glover (1989). In this procedure, a solution is randomly examined and a move is performed to the best neighbor of the current solution. In order to avoid cycling and to get out of local optima, all solutions that were examined and passed the acceptance criteria are stored in a tabu list and these solutions are forbidden from the search procedure. The tabu search approach is known as one of the most efficient solution methods for the VRP (Gendreau et al., 2001).

Bard and Nananukul (2009a) proposed a tabu search with a special feature, the so-called reactive tabu search (RTS), where the size of the tabu list is dynamic. They create an initial solution by solving the integrated lot-sizing and distribution problem which is a modified PRP obtained by dropping the routing constraints (2.8)-(2.12), removing variables $x_{ijt}$, and assuming that the delivery cost is equal to the round trip transportation cost. Then, a subsequent routing decision is made by applying a capacitated vehicle routing problem (CVRP) subroutine based on a tabu search proposed by Carlton and Barnes (1996). They developed a neighborhood search procedure to improve the incumbent solution by swapping the delivery quantities of

two customers in two delivery periods. An insertion heuristic is applied if the swapped customer is not assigned to the new delivery plan. Another move, called *transfer*, is also performed to transfer the whole delivery quantity of a customer in period $t$ so that it can be combined with another delivery to this customer in some previous period $t' < t$ without violating capacity constraints. Then, the customer is removed from the route in period $t$ and thus transportation costs can be reduced.

A different tabu search procedure was developed by Armentano et al. (2011) who used path relinking. The tabu search is used to move some shipment quantity from customer $i$ in period $t$ to another period $t' \neq t$ without violating inventory capacity. If the customer is not visited in period $t'$, this customer is inserted into that period according to the cheapest insertion rule. The shipment quantity and the production plan are adjusted, and the new total cost is calculated accordingly. Every customer-period combination is considered and the one that minimizes the total cost is chosen. A path relinking procedure is used to diversify the search. In this procedure, the current tabu search solution is set as an initiating solution and another solution, called the guiding solution, is selected from a pool of elite solutions. The path relinking procedure tries to reduce the differences in delivery quantities of the two solutions by transferring the different amounts to another period. When all moves are finished, the guiding solution becomes the initiating solution and the new solution becomes the guiding solution. The process is repeated until both solutions have the same value.

### 2.6.4 Computational Evaluation of the PRP Metaheuristics

A comparison of the computational performance of the metaheuristics described in this section is reported in Tables 2.II and 2.III. The computational experiments were conducted with the test instances from Boudia et al. (2005). The instances consist of the sets of problems with 50, 100, 200 customers and 20 time periods and there are 30 instances per set. All test evaluations were performed on workstations with comparable CPU performances. The results show that the GRASP of Boudia et al. (2007) can obtain solutions within a relatively short runtime, but the best solutions are provided by the tabu search algorithm with path relinking procedure (TSPR) developed by Armentano et al. (2011).

Table 2.II: Average Results for Different Metaheuristics

| Prob set | $N_c$ | $l$ | H1[1] | GRASP[2] | MA\| PM[3] | RTS[4] | TSPR[5] |
|---|---|---|---|---|---|---|---|
| B1 | 50 | 20 | 511579 | 443264 | 393263 | 369662 | 361704 |
| B2 | 100 | 20 | 963649 | 791839 | 714627 | 712294 | 685898 |
| B3 | 200 | 20 | 1312612 | 1070026 | 1001634 | 1034923 | 951638 |

[1]Boudia et al. (2008)
[2]Boudia et al. (2007)
[3]Boudia and Prins (2009)
[4]Bard and Nananukul (2009a)
[5]Armentano et al. (2011)

Table 2.III: Average Computational Time (in Seconds) for Different Metaheuristics

| Prob set | $N_c$ | $l$ | H1[1] | GRASP[2] | MA\| PM[3] | RTS[4] | TSPR[5] |
|---|---|---|---|---|---|---|---|
| B1 | 50 | 20 | 0.1 | 87 | 172.7 | 451.8 | 317.0 |
| B2 | 100 | 20 | 0.5 | 415.9 | 1108.1 | 1133.8 | 1147.6 |
| B3 | 200 | 20 | 2.1 | 1801.8 | 4098.5 | 3060.2 | 3926.4 |

[1,2,3]executed on 2.30 GHz PC
[4]executed on 2.53 GHz PC
[5]executed on 2.80 GHz PC

## 2.7 PRP with Demand Uncertainty

In this section, we provide a review of the literature concerning demand uncertainty. The difficulty of handling demand uncertainty in the PRP and IRP lies in the fact that it makes the problem intractable (Hvattum and Løkketangen, 2009; Solyalı et al., 2012). To the best of our knowledge, no studies have specifically addressed the PRP with demand uncertainty. There are, however, a few studies that have discussed this issue for the stochastic IRP (SIRP) as shown in Table 2.IV.

Table 2.IV: Summary of the Literature on the Problem with Demand Uncertainty

| Author (s) | Time horizon | Decision process | Delivery | Approach |
|---|---|---|---|---|
| Federgruen and Zipkin (1984) | Single period | Single stage | Route | Decomposition-based heuristic |
| Jaillet et al. (2002) | Discrete finite | Markov | Direct | Fixed interval policy analysis |
| Kleywegt et al. (2002a) | Discrete infinite | Markov | Direct | Approximate dynamic programming |
| Kleywegt et al. (2004) | Discrete infinite | Markov | Route | Approximate dynamic programming |
| Adelman (2004) | Discrete infinite | Markov | Route | Approximate dynamic programming |
| Hvattum et al. (2009) | Discrete infinite | Markov | Route | Scenario-tree-based heuristic |
| Hvattum and Løkketangen (2009) | Discrete infinite | Markov | Route | Scenario-tree-based heuristic |
| Bertazzi et al. (2011) | Discrete finite | Markov | Route | Approximate dynamic programming |
| Solyalı et al. (2012) | Discrete finite | Two-stage | Route | Branch-and-cut |
| Coelho et al. (2012b) | Discrete finite | Markov | Route | Decomposition-based heuristic |

Federgruen and Zipkin (1984) considered the SIRP with random demands but in a single period planning horizon. The uncertainty is incorporated in the non-linear objective function. The problem is decomposed into an inventory allocation (IA) and a number of traveling salesman problems (TSPs), one for each vehicle. The first subproblem is solved by an exact algorithm and the dual solutions are used to evaluate whether the total cost can be reduced by switching two customers between two different routes. The authors also showed that this approach could provide 6-7% savings compared to the solution obtained by solving a deterministic vehicle routing problem (VRP). A different SIRP involving long term planning horizons was addressed by Jaillet et al. (2002). In this study, a repeated distribution pattern is used and the delivery cost for each customer is fixed. If a stockout occurs, an extra (non-scheduled) delivery is required and a fixed penalty cost is applied. The problem is solved in a rolling horizon framework by using approximations of the direct shipment delivery costs.

The IRP with a discrete time infinite horizon was addressed in several studies. Kleywegt et al. (2002a) considered the SIRP with direct deliveries and the problem is represented by a Markov decision process. Since the state space is too large to compute, the authors employed approximate dynamic programming techniques to solve the problem. Kleywegt et al. (2004) extended the previous study to a more general case where a vehicle can deliver to multiple customers in the same route and developed approaches to handle the problem when a vehicle can visit up to three customers in a route. Adelman (2004) focused on the same problem and proposed a

price-directed approach where the future costs of current actions are approximated using optimal dual prices. This approach can be used to solve the problem with an unbounded number of customers per route. Hvattum et al. (2009) and Hvattum and Løkketangen (2009) used heuristics based on finite scenario trees to solve the same problem.

For the IRP with a discrete finite planning horizon, Bertazzi et al. (2011) addressed the stochastic problem with the order-up-to level (OU) policy and a penalty cost is incurred when a stockout occurs. They employed a heuristic rollout algorithm using an approximate cost-to-go. This approximate cost is formulated as a MIP and is solved by a branch-and-cut algorithm. Solyalı et al. (2012) addressed the single vehicle IRP with demand uncertainty in a discrete finite planning horizon, but the distribution of demand is unknown and backlogging is allowed. This problem is called the robust inventory routing problem (RIRP). They presented robust formulations and used a branch-and-cut algorithm similar to that of Archetti et al. (2007). Finally, Coelho et al. (2012b) considered a dynamic and stochastic variant of the IRP in which demands are gradually revealed over time. They proposed a heuristic to solve the problem and evaluated the value of demand forecasts and transshipments between customers.

# Chapter 3
# Exact Algorithms for Production Routing Problems

This chapter is based on the following article.

- Adulyasak, Y., Cordeau, J.-F., Jans, R. Formulations and Branch-and-Cut Algorithms for Multi-Vehicle Production and Inventory Routing Problems. GERAD Tech Rep. G-2012-14. 40 pages. Submitted to *INFORMS Journal on Computing* in April 2012 (Revision submitted in October 2012).

The purpose of this chapter is to introduce new exact algorithms for the PRP, which is a generalization of the IRP. Both are difficult problems arising in the planning of integrated supply chains. These problems are solved in an attempt to jointly optimize production, inventory, distribution and routing decisions. Specifically, we emphasize the multi-vehicle aspect, which is often neglected due to its complexity. Although the main focus of the chapter is on the PRP, the approaches that we introduce in this chapter can be adapted to solve the IRP with multiple vehicles.

## 3.1 Introduction

As mentioned in Chapter 2, few exact algorithms have been proposed to solve the IRP due to its complexity. To represent instance sizes, we use the notation $ac/b\text{p}/c\text{v}$ where $a$, $b$ and $c$ are the number of customers, periods and vehicles, respectively. Archetti et al. (2007) developed a branch-and-cut approach for the IRP with a single vehicle and analyzed three different replenishment policies for the customers. In the first policy, called order-up-to level (OU), a visited customer receives exactly the amount which brings its inventory up to a predefined target stock level (TSL). The second policy, called maximum level (ML), allows delivery quantities to be any positive value

but the inventory at each customer cannot exceed its maximum stock level. The third policy is similar to the ML policy but there is no maximum stock level imposed at the customers. Archetti et al. (2007) used different inequalities to strengthen the formulation for each policy and could solve instances up to 45c/3p/1v and 30c/6p/1v to optimality within two hours for the IRP with the OU and ML policy, respectively. Solyalı and Süral (2011) proposed a stronger formulation for the single-vehicle IRP-OU using a shortest-path network representation of the OU policy at each customer and used a similar branch-and-cut approach as Archetti et al. (2007). They could solve instances up to 60c/3p/1v and 15c/12p/1v to optimality within four hours.

Variants of the IRP have been proposed as well. Christiansen (1999) introduced an IRP application in a maritime context, called the inventory pickup and delivery problem, and applied a Dantzig-Wolfe decomposition and column generation approach to solve the problem. Savelsbergh and Song (2008) considered the IRP with continuous move where a product is distributed from a set of plants to a set of customers by multiple vehicles. In this study, minimum delivery quantities are imposed and inventory costs are disregarded. The authors proposed a multi-commodity flow formulation with a vehicle index and developed a branch-and-cut approach to solve the problem.

Few studies have introduced exact algorithms or even methods to compute strong lower bounds for the PRP. Fumero and Vercellis (1999) developed a Lagrangian relaxation approach to obtain lower bounds and heuristic solutions for a variant of the PRP where unit transportation costs are assumed and the routing decisions can be determined by solving a minimum cost flow problem. Instances with up to 12 customers, 8 periods and 10 products were tested and the algorithm could obtain solutions with an average optimality gap of 5.5%. A similar Lagrangian relaxation approach was used by Solyalı and Süral (2009) to solve the PRP-OU. However, the lower bounds obtained by this approach were weak. On the instances with 8c/5p/1v, the lower bound produced by the Lagrangian relaxation has an average deviation of 33.16% from the optimal value. They also tested the performance of the formulation based on the multi-commodity fixed charge network flow problem using the same instances and the longest computing time was approximately 20 hours to obtain the optimal solution. Ruokokoski et al. (2010) explored the performance of different lot-sizing

reformulation schemes for the PRP-ML with uncapacitated production and a single uncapacitated vehicle, and further employed a branch-and-cut approach similar to that of Archetti et al. (2007) to solve the problem. Bard and Nananukul (2010) introduced a branch-and-price procedure for the PRP-ML with multiple vehicles. Because their subtour elimination constraints are in the form of the Miller-Tucker-Zemlin inequalities (Miller et al., 1960), they obtained rather weak lower bounds, and only the instances up to 10c/2p/5v (Nananukul, 2008) were solved to optimality within 30 minutes. The emphasis of the study was instead on a heuristic procedure using the branch-and-price framework. Archetti et al. (2011) adapted the branch-and-cut approach of Archetti et al. (2007) for the PRP-ML with uncapacitated production and a single vehicle. Several valid inequalities were also used to strengthen the formulation. However, computational testing was only performed on 14c/6p/1v instances and not all instances were solved to optimality within two hours.

Table 3.I presents a summary of the exact algorithms for the PRP and IRP in the literature. We classify the problems along three dimensions: IRP versus PRP, the replenishment policy (ML versus OU) and the number of vehicles (single versus multiple). The size of the problems that can be solved to optimality and the computing time limit (in hours) are shown in brackets. This table clearly shows an important gap in the existing literature. The only exact algorithm for the multiple vehicle PRP is that of Bard and Nananukul (2010) which only solved relatively small instances to optimality compared to the results on the single vehicle case.

Table 3.I: Summary of Exact Algorithms for the Deterministic PRP and IRP with Single Product, Single Plant and Multiple Customers

| **Problem** | **Maximum Level (ML)** | | **Order-Up-To Level (OU)** | |
|---|---|---|---|---|
| | **Single vehicle** | **Multiple vehicles** | **Single vehicle** | **Multiple vehicles** |
| **IRP** | Archetti et al. (2007) $[45c/3p/1v-2h]$ , $[30c/6p/1v-2h]$ | - | Archetti et al. (2007) $[45c/3p/1v-2h]$, $[30c/6p/1v-2h]$ Solyalı and Süral (2011) $[60c/3p/1v-4h]$, $[15c/12p/1v-4h]$ | - |
| **PRP** | Ruokokoski et al. (2010) † Archetti et al. (2011) $[14c/6p/1v-2h]$‡ | Bard and Nananukul (2010) $[10c/2p/5v-0.5h]$ | - | - |

† tests were performed only on the uncapacitated single vehicle case

‡ some instances were not solved to optimality

In this chapter, we consider a single product and a production-distribution network that consists of a production plant and multiple customers which have their own storage area. At the beginning of the planning horizon, the production plant and the customers may have initial inventory. In each period, each customer must have sufficient inventory to satisfy its demand. In the case of the PRP, the plant must decide whether or not to produce the product and the quantity to be produced. If production takes place, fixed setup and unit production costs are incurred. The produced quantities can be transported by a limited number of capacitated vehicles to the customers and routing costs are paid. The product can also be stored at the plant or at the customers and unit inventory holding costs are incurred. We consider the cases where the customer replenishment is controlled by the ML and OU policies. The hypotheses we adopt are generally in line with Chandra and Fisher (1994), Fumero and Vercellis (1999) and Archetti et al. (2011) for the ML policy and Solyalı and Süral (2009) for the OU policy. There is a slight difference on the imposed maximum inventory in this study compared to Archetti et al. (2011). To be well aligned with the concept of the OU policy, we set the delivery quantity to each customer equal to the difference between its current stock level and its TSL before demand consumption. In contrast, the TSL is imposed after demand consumption (which typically is not known in advance in practice) in Archetti et al. (2011). This also applies to the ML policy where the maximum inventory level is imposed before demand consumption. It should also be noted that the replenishment practice in our PRP and the literature stated above is slightly different from the IRP presented in Archetti et al. (2007) and Solyalı and Süral (2011). In the latter studies, the delivery to the customers must take place before the distribution facility is replenished in each period, while in our PRP, the quantity produced in period $t$ can be delivered to customers to satisfy their demand in the same period. These two practices, however, can be converted into each other as we show in the Appendix. Since the PRP is a generalization of the IRP, we prefer to use the name PRP in the remainder of this chapter to represent both the IRP and PRP unless stated otherwise. Note that the name MVPRP is used to represent the PRP with the multi-vehicle (MV) aspect.

The main contributions of this chapter are threefold. First, we present strong formulations and exact algorithms for both the IRP and PRP with multiple vehicles, thereby filling several important gaps in the literature as shown in Table 3.I. Several formulations are presented and branch-and-cut algorithms are proposed to solve the problems under both the OU and ML policy. Second, we propose several valid inequalities and symmetry breaking constraints to strengthen the formulations, and test the effect of these inequalities. Third, we provide extensive computational results of the new formulations and further explore the performance of the algorithm on a multi-core machine.

The rest of the chapter is organized as follows. Section 3.2 presents different formulations of the MVPRP. Section 3.3 describes the valid inequalities that are applied to the formulations. The details of the branch-and-cut approaches are discussed in Section 3.4 and the details of the heuristic algorithm to calculate upper bounds are presented in Section 3.5. This is followed by the discussion of computational experiments in Section 3.6, and by conclusions.

## 3.2 MVPRP Formulations

This section presents the main notation and the mathematical formulations of the MVPRP with ML and OU policy.

### 3.2.1 Notation

The PRP can be defined on a complete undirected graph $G = (N, E)$ with the following notation.

Sets:

$T$ set of time periods, indexed by $t \in \{1, \ldots, l\}$, and $T' = T \cup \{l+1\}$;

$N$ set of plant and customers, indexed by $i \in \{0, \ldots, n\}$, where the plant is represented by node 0 and $N_c = N \setminus \{0\}$ is the subset of $n$ customers;

$E$ set of edges, $E = \{(i, j) : i, j \in N, i < j\}$;

$K$ set of identical vehicles, indexed by $k \in \{1, \ldots, m\}$;

$E(S)$ set of edges $(i, j) \in E$ such that $i, j \in S$, where $S \subseteq N$ is a given set of nodes;

$\delta(S)$ set of edges incident to a node set $S$, $\delta(S) = \{(i, j) \in E : i \in S, j \notin S$ or $i \notin S, j \in S\}$ (for simplicity, we also write $\delta(i)$ for $\delta(\{i\})$ to represent set of edges incident to node $i$).

Variables:

$p_t$ production quantity in period $t$;

$I_{it}$ inventory at node $i$ at the end of period $t$;

$y_t$ equal to 1 if there is production at the plant in period $t$, 0 otherwise;

$z_{ikt}$ equal to 1 if node $i$ is visited by vehicle $k$ in period $t$, 0 otherwise;

$x_{ijkt}$ if vehicle $k$ travels directly between node $i$ and node $j$ in period $t$, 0 otherwise;

$q_{ikt}$ quantity delivered to customer $i$ with vehicle $k$ in period $t$;

Parameters:

$u$ unit production cost;

$f$ fixed production setup cost;

$h_i$ unit inventory holding cost at node $i$;

$c_{ij}$ transportation cost between nodes $i$ and $j$;

$d_{it}$ demand at customer $i$ in period $t$;

$C$ production capacity;

$Q$ vehicle capacity;

$L_i$ maximum or target inventory level at node $i$;

$I_{i0}$ initial inventory available at node $i$

### 3.2.2 Multi-Vehicle Formulations for the ML Policy

In this section, we introduce two formulations for the MVPRP-ML: one with and one without a vehicle index.

#### 3.2.2.1 Formulation with a Vehicle Index for the ML Policy

To formulate the MVPRP-ML with a vehicle index, we extend the single-vehicle PRP formulation used by Archetti et al. (2011), as follows.

$$\min \sum_{t \in T} \left( up_t + fy_t + \sum_{i \in N} h_i I_{it} + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} x_{ijkt} \right) \tag{3.1}$$

s.t.

$$I_{0,t-1} + p_t = \sum_{i \in N_c} \sum_{k \in K} q_{ikt} + I_{0t} \qquad \forall t \in T \tag{3.2}$$

$$I_{i,t-1} + \sum_{k \in K} q_{ikt} = d_{it} + I_{it} \qquad \forall i \in N_c, \forall t \in T \tag{3.3}$$

$$p_t \leq \min \left\{ C, \sum_{i \in N_c} \sum_{j=t}^{l} d_{ij} \right\} y_t \qquad \forall t \in T \tag{3.4}$$

$$I_{0t} \leq L_0 \qquad \forall t \in T \tag{3.5}$$

$$I_{i,t-1} + \sum_{k \in K} q_{ikt} \leq L_i \qquad \forall i \in N_c, \forall t \in T \tag{3.6}$$

$$\sum_{i \in N_c} q_{ikt} \leq Q z_{0kt} \qquad \forall k \in K, \forall t \in T \tag{3.7}$$

$$\sum_{k \in K} z_{ikt} \leq 1 \qquad \forall i \in N_c, \forall t \in T \tag{3.8}$$

$$q_{ikt} \leq \min \left\{ L_i, Q, \sum_{j=t}^{l} d_{ij} \right\} z_{ikt} \qquad \forall i \in N_c, \forall k \in K, \forall t \in T \tag{3.9}$$

$$\sum_{(j,j') \in \delta(i)} x_{jj'kt} = 2 z_{ikt} \qquad \forall i \in N, \forall k \in K, \forall t \in T \tag{3.10}$$

$$\sum_{(i,j) \in E(S)} x_{ijkt} \leq \sum_{i \in S} z_{ikt} - z_{ekt} \qquad \forall S \subseteq N_c : |S| \geq 2, \forall e \in S, \forall k \in K, \forall t \in T \tag{3.11}$$

$$p_t, I_{it}, q_{ikt} \geq 0 \qquad \forall i \in N, \forall k \in K, \forall t \in T \tag{3.12}$$

$$y_t, z_{ikt}, \in \{0, 1\} \qquad \forall i \in N, \forall k \in K, \forall t \in T \tag{3.13}$$

$$x_{ijkt} \in \{0,1\} \qquad \forall (i,j) \in E : i \neq 0, \forall k \in K, \forall t \in T \tag{3.14}$$

$$x_{0jkt} \in \{0,1,2\} \qquad \forall j \in N_c, \forall k \in K, \forall t \in T. \tag{3.15}$$

The objective function (3.1) minimizes the total production, setup, inventory and routing costs. Constraints (3.2) and (3.3) ensure the inventory flow balance at the plant and at the customers, respectively. Constraints (3.4) are the setup forcing and production capacity constraints at the plant: they force the setup variable to be one if production takes place and limit the production quantity to the minimum of the production capacity and the total demand in the remaining periods. The inventory quantity at the production facility at the end of each period is limited by constraints (3.5) and the inventory quantities at the customers after delivery cannot exceed their inventory capacities (3.6). The total quantity loaded in each vehicle can be at most the vehicle capacity as specified by (3.7). Constraints (3.8) allow each customer to be visited at most once in each period. Constraints (3.9) allow a positive delivery quantity from vehicle $k$ to node $i$ in period $t$ only if this node is visited by the vehicle in period $t$. Since in the ML policy it is never optimal to carry inventory at the end of the planning horizon, the delivery quantity to a customer is limited by the minimum value between the inventory capacity at the customer, the vehicle capacity and the total demand of the customer in the remaining periods. Constraints (3.10) are the degree constraints. They require the number of edges incident to node $i$ to be 2 if it is visited. Constraints (3.11) eliminate for each vehicle subtours that do not go through the depot.

Archetti et al. (2007, 2011) also strengthen the formulation using several valid inequalities. We present here the inequalities that are valid for the PRP with capacitated production. Note that we extend the original inequalities for the multi-vehicle case. Denote by $t'$ and $t''$, the earliest period when the plant must produce and the earliest period when at least one customer must be replenished to prevent a stock-out, respectively, i.e., $t' = \text{argmin}_{1 \leq t \leq l} \left\{ \sum_{i \in N_c} \max \left\{ 0, \sum_{j=1}^{t} d_{ij} - I_{i0} \right\} - I_{00} > 0 \right\}$, and $t'' = \min_{i \in N_c} t''_i$, where $t''_i = \text{argmin}_{1 \leq t \leq l} \left\{ \sum_{j=1}^{t} d_{ij} - I_{i0} > 0 \right\}$. Let also $\kappa$ be the

minimum shipping quantity in $t''$, i.e., $\kappa = \sum_{i \in N_c} \max\left\{0, \sum_{j=1}^{t''} d_{ij} - I_{i0}\right\}$. First, two inequalities are used to prevent stockouts:

$$\sum_{t=1}^{t'} y_t \geq 1 \tag{3.16}$$

$$\sum_{k \in K} \sum_{t=1}^{t''} z_{0kt} \geq \left\lceil \frac{\kappa}{Q} \right\rceil. \tag{3.17}$$

Second, the following inequalities are imposed to strengthen customer replenishments:

$$I_{i,t-s-1} \geq \left(\sum_{j=0}^{s} d_{i,t-j}\right)\left(1 - \sum_{k \in K} \sum_{j=0}^{s} z_{ik,t-j}\right) \quad \forall i \in N_c, \forall t \in T, s = 0, 1..., t-1. \tag{3.18}$$

Finally, the following inequalities are imposed for the routing part:

$$z_{ikt} \leq z_{0kt} \qquad \forall i \in N_c, \forall k \in K, \forall t \in T \tag{3.19}$$

$$x_{ijkt} \leq z_{ikt} \text{ and } x_{ijkt} \leq z_{jkt} \qquad \forall (i,j) \in E(N_c), \forall k \in K, \forall t \in T. \tag{3.20}$$

The formulation (3.1)-(3.20) will be referred to as $F(ML)|k$.

#### 3.2.2.2 Formulation without a Vehicle Index for the ML Policy

The previous formulation has the drawback that the number of variables grows in proportion to the number of vehicles. Alternatively, one can express the routing constraints with variables that do not comprise a vehicle index. The formulation is written using the variables $q, z$ and $x$ with the same notation as in the previous section but the vehicle index $k$ is dropped. The only exception is the variable $z_{0t}$ which is changed to be an integer variable representing the number of vehicles leaving the plant in period $t$. The formulation without vehicle index can be stated as follows.

$$\min \sum_{t \in T} \left( up_t + f y_t + \sum_{i \in N} h_i I_{it} + \sum_{(i,j) \in E} c_{ij} x_{ijt} \right) \tag{3.21}$$

s.t. (3.4)-(3.5) and

$$I_{0,t-1} + p_t = \sum_{i \in N_c} q_{it} + I_{0t} \qquad \forall t \in T \tag{3.22}$$

$$I_{i,t-1} + q_{it} = d_{it} + I_{it} \qquad \forall i \in N_c, \forall t \in T \tag{3.23}$$

$$I_{i,t-1} + q_{it} \leq L_i \qquad \forall i \in N_c, \forall t \in T \tag{3.24}$$

$$q_{it} \leq \min \left\{ L_i, Q, \sum_{j=t}^{l} d_{ij} \right\} z_{it} \qquad \forall i \in N_c, \forall t \in T \tag{3.25}$$

$$\sum_{(j,j') \in \delta(i)} x_{jj't} = 2z_{it} \qquad \forall i \in N, \forall t \in T \tag{3.26}$$

$$z_{0t} \leq m \qquad \forall t \in T \tag{3.27}$$

$$Q \sum_{(i,j) \in E(S)} x_{ijt} \leq \sum_{i \in S} (Q z_{it} - q_{it}) \qquad \forall S \subseteq N_c : |S| \geq 2, \forall t \in T \tag{3.28}$$

$$p_t, I_{it}, q_{it} \geq 0 \qquad \forall i \in N, \forall t \in T \tag{3.29}$$

$$y_t, z_{it}, \in \{0, 1\} \qquad \forall i \in N_c, \forall t \in T \tag{3.30}$$

$$z_{0t} \in \mathbb{Z}^+ \qquad \forall t \in T \tag{3.31}$$

$$x_{ijt} \in \{0, 1\} \qquad \forall (i,j) \in E : i \neq 0, \forall t \in T \tag{3.32}$$

$$x_{0jt} \in \{0, 1, 2\} \qquad \forall j \in N_c, \forall t \in T. \tag{3.33}$$

Constraints (3.22)-(3.26) are equivalent to (3.2)-(3.3), (3.6) and (3.9)-(3.10), respectively. Constraints (3.27) limit the number of vehicles leaving the production facility to the number of available vehicles in each period. Constraints (3.28) are the subtour elimination and vehicle capacity constraints. If one divides the inequalities by $Q$, these constraints have a form similar to the generalized fractional subtour elimination constraints (GFSECs) for the VRP (Toth and Vigo, 2001). Unlike GFSECs in the VRP, however, we cannot round up the value of the term $q_{it}/Q$ because it contains the $q_{it}$ variable and therefore they do not provide strong LP relaxation bounds. We prefer to use the form (3.28) since preliminary tests have indicated that the original form of GFSECs is numerically unstable due to the fractional right hand side.

We can also rewrite inequalities (3.17)-(3.20) for the non-vehicle index formulation as follows:

$$\sum_{j=1}^{t''} z_{0j} \geq \left\lceil \frac{\kappa}{Q} \right\rceil \tag{3.34}$$

$$I_{i,t-s-1} \geq \left( \sum_{j=0}^{s} d_{i,t-j} \right) \left( 1 - \sum_{j=0}^{s} z_{i,t-j} \right) \quad \forall i \in N_c, \forall t \in T, s = 0, 1..., t-1 \tag{3.35}$$

$$z_{it} \leq z_{0t} \quad \forall i \in N_c, \forall t \in T \tag{3.36}$$

$$x_{ijt} \leq z_{it} \text{ and } x_{ijt} \leq z_{jt} \quad \forall (i,j) \in E(N_c), \forall t \in T. \tag{3.37}$$

The non-vehicle index formulation together with the inequalities (3.34)-(3.37) in this section and (3.16) will be referred to as $F(ML)|nk$.

We also remark here on reformulation schemes for the PRP-ML. We have tested the facility location reformulation, called four-index facility location (FIFL), proposed by Ruokokoski et al. (2010). The preliminary results show that, in our case where production, inventory and vehicle capacities are imposed, the facility location reformulation is slightly inferior in terms of computing times to the basic formulation with the inequalities used in Archetti et al. (2007, 2011). The main reasons are, first, that the inequalities (3.18) already substantially strengthen the formulation, and second, that the FIFL formulation has a much larger number of variables compared to the basic formulation.

### 3.2.3 Multi-Vehicle Formulations for the OU Policy

This section presents two formulations for the MVPRP-OU: one with and one without a vehicle index.

#### 3.2.3.1 Formulation with a Vehicle Index for the OU Policy

In the OU policy, when a customer is visited, the inventory before demand consumption must be replenished to reach its TSL. Archetti et al. (2007, 2011) added constraints to the formulation $F(ML)|k$ to solve the single vehicle IRP with the OU policy. However, it has been shown by Solyalı and Süral (2011) that a stronger version

of the IRP-OU can be obtained by using a shortest-path network representation for the customer inventory replenishment part. This reformulation scheme exploits the characteristic of the OU policy that the delivery quantity for a customer $i$ visited in period $t$ is equal to the total demand consumption in the interval between $t-1$ and the previous visit in period $v < t$. In our preliminary test, we have also observed that this reformulation is far superior to the formulations of Archetti et al. (2007, 2011) for multiple vehicles. As a consequence, we adopt the reformulation presented in Solyalı and Süral (2011) and extend it using a vehicle index. We define $d_{i0} = d_{i,l+1} = 0$ and use the following additional notation:

$\lambda_{ikvt}$ binary variable, equal to 1 if node $i$ is visited by vehicle $k$ in period $t$ and the previous visit is in period $v$, 0 otherwise;

$g_{ivt}$ total delivery quantity when customer $i$ is visited in period $t$ and the previous visit is in period $v$;

$e_{ivt}$ total inventory holding cost when customer $i$ is visited in period $t$ and the previous visit is in period $v$;

$\mu(i,t)$ the latest period after period $t$ when customer $i$ can be replenished next without having a stockout, i.e., $\mu(i,t) = \text{argmax}_{t<v\leq l+1}\{g_{itv} \leq L_i\}$;

$\pi(i,t)$ the earliest period before period $t$ when customer $i$ can be replenished without having a stockout, i.e., $\pi(i,t) = \text{argmin}_{0\leq v<t}\{g_{ivt} \leq L_i\}$.

The parameters $g_{itv}$ and $e_{itv}$ can be calculated as follows:

$$g_{ivt} = \begin{cases} \sum_{j=1}^{t-1} d_{ij} + (L_i - I_{i0}) & \text{if } v = 0 \\ \sum_{j=v}^{t-1} d_{ij} & \text{if } 0 < v < t \leq l \\ 0 & \text{if } t = l+1 \end{cases}$$

$$e_{ivt} = \begin{cases} h_i \left( \sum_{j=1}^{t-1} (I_{i0} - \sum_{l=1}^{j} d_{il}) \right) & \text{if } v = 0 \\ h_i \left( \sum_{j=v}^{t-1} (L_i - \sum_{l=v}^{j} d_{il}) \right) & \text{if } 0 < v < t \leq l+1 \end{cases}$$

Preprocessing can be used to eliminate variables associated with infeasible delivery quantities $g_{ivt} > Q$ and $g_{ivt} > L_i$. The strong formulation, referred to as $F(OU)|k$, is as follows.

$$\min \sum_{t\in T}\left(up_t + fy_t + h_0 I_{0t} + \sum_{(i,j)\in E}\sum_{k\in K} c_{ij}x_{ijkt}\right) + \sum_{i\in N_c}\sum_{k\in K}\sum_{t=T'}\sum_{v=\pi(i,t)}^{t-1} e_{ivt}\lambda_{ikvt} \quad (3.38)$$

s.t. (3.5), (3.8), (3.10)-(3.15) and

$$I_{0,t-1} + p_t = \sum_{i\in N_c}\sum_{k\in K}\sum_{v=\pi(i,t)}^{t-1} g_{ivt}\lambda_{ikvt} + I_{0t} \qquad \forall t \in T \quad (3.39)$$

$$p_t \le Cy_t \qquad \forall t \in T \quad (3.40)$$

$$\sum_{i\in N_c}\sum_{v=\pi(i,t)}^{t-1} g_{ivt}\lambda_{ikvt} \le Qz_{0kt} \qquad \forall k \in K, \forall t \in T \quad (3.41)$$

$$\sum_{v=\pi(i,t)}^{t-1} \lambda_{ikvt} = z_{ikt} \qquad \forall i \in N_c, \forall k \in K, \forall t \in T \quad (3.42)$$

$$\sum_{k\in K}\sum_{t=1}^{\mu(i,0)} \lambda_{ik0t} = 1 \qquad \forall i \in N_c \quad (3.43)$$

$$\sum_{k\in K}\sum_{v=\pi(i,t)}^{t-1} \lambda_{ikvt} - \sum_{k\in K}\sum_{v=t+1}^{\mu(i,t)} \lambda_{iktv} = 0 \qquad \forall i \in N_c, \forall t \in T \quad (3.44)$$

$$\sum_{k\in K}\sum_{t=\pi(i,l+1)}^{l} \lambda_{ikt,l+1} = 1 \qquad \forall i \in N_c \quad (3.45)$$

$$\lambda_{ikvt} \in \{0,1\} \qquad \forall i \in N_c, \forall k \in K, \forall v,t \in T. \quad (3.46)$$

The objective function (3.38) and constraints (3.39)-(3.41) are equivalent to (3.1), (3.2), (3.4) and (3.7), respectively. Constraints (3.42) provide the link between the $\lambda_{ikvt}$ and $z_{ikt}$ variables. Constraints (3.43)-(3.45) represent the shortest-path network of the OU policy at each customer.

As in the formulation presented by Solyalı and Süral (2011), the inequalities (3.19)-(3.20) are also added to strengthen the routing part of the formulation. We further add (3.16)-(3.17) to reinforce the production part.

#### 3.2.3.2 Formulation without a Vehicle Index for the OU Policy

The non-vehicle index formulation for the OU policy can be written using the same notation as the formulation $F(ML)|nk$, using the variable $\lambda$ as in the previous section, but without the vehicle index $k$. The formulation, referred to as $F(OU)|nk$, is as follows.

$$\min \sum_{t\in T}\left(up_t + fy_t + h_0 I_{0t} + \sum_{(i,j)\in E} c_{ij}x_{ijt}\right) + \sum_{i\in N_c}\sum_{t\in T'}\sum_{v=\pi(i,t)}^{t-1} e_{ivt}\lambda_{ivt} \tag{3.47}$$

s.t. (3.5), (3.40) , (3.26)-(3.27), (3.29)-(3.33), and

$$I_{0,t-1} + p_t = \sum_{i\in N_c}\sum_{v=\pi(i,t)}^{t-1} g_{ivt}\lambda_{ivt} + I_{0t} \qquad \forall t \in T \tag{3.48}$$

$$\sum_{v=\pi(i,t)}^{t-1} \lambda_{ivt} = z_{it} \qquad \forall i \in N_c, \forall t \in T \tag{3.49}$$

$$\sum_{t=1}^{\mu(i,0)} \lambda_{i0t} = 1 \qquad \forall i \in N_c \tag{3.50}$$

$$\sum_{v=\pi(i,t)}^{t-1} \lambda_{ivt} - \sum_{v=t+1}^{\mu(i,t)} \lambda_{itv} = 0 \qquad \forall i \in N_c, \forall t \in T \tag{3.51}$$

$$\sum_{t=\pi(i,l+1)}^{l} \lambda_{it,l+1} = 1 \qquad \forall i \in N_c \tag{3.52}$$

$$Q\sum_{(i,j)\in E(S)} x_{ijt} \leq \sum_{i\in S}\left(Qz_{it} - \sum_{v=\pi(i,t)}^{t-1} g_{ivt}\lambda_{ivt}\right) \qquad \forall S \subseteq N_c : |S| \geq 2, \forall t \in T \tag{3.53}$$

$$\lambda_{ivt} \in \{0,1\} \qquad \forall i \in N_c, \forall v,t \in T. \tag{3.54}$$

Constraints (3.48)-(3.52) are equivalent to (3.39) and (3.42)-(3.45), respectively. Constraints (3.53) are equivalent to (3.28). Note that inequalities (3.16), (3.34) and (3.36)-(3.37) are also added a priori in our implementation in order to make a fair comparison with the other formulations.

### 3.2.4 Formulations for the MVIRP

All formulations above can be easily modified to solve the MVIRP by disregarding the production part (i.e., production setup and quantity decisions). First, the production setup variable $y_t$ is set to one, i.e., $y_t = \bar{y}_t = 1, \forall t \in T$. Second, denote by $B_t$ the production quantity made available in each period, constraints (3.4) and (3.40) are replaced with $p_t = B_t \bar{y}_t, \forall t \in T$, and parameters $\min\left\{L_i, Q, \sum_{j=t}^{l} d_{ij}\right\}$ in constraints (3.9) and (3.25) are replaced with $\min\{L_i, Q\}$. The rest of the formulations remains unchanged.

## 3.3 Valid Inequalities

In this section, we introduce two groups of new valid inequalities: one for the vehicle index and the other for the non-vehicle index formulation.

### 3.3.1 Valid Symmetry Breaking Inequalities for the Vehicle Index Formulations

In each period $t$, there are two main symmetry issues which stem from the presence of identical vehicles. Denote by $\overline{m}_t$ the number of dispatched vehicles in period $t$. First, in vehicle dispatching, there are $\binom{m}{\overline{m}_t}$ possible options to select $\overline{m}_t$ vehicles from the fleet. Second, among the selected vehicles, there are still $\overline{m}_t!$ options to swap the routes that are assigned to each dispatched vehicle. These two types of symmetry are present in each period, and hence there can be $\left[\binom{m}{\overline{m}_1}\overline{m}_1!\right]\left[\binom{m}{\overline{m}_2}\overline{m}_2!\right]...\left[\binom{m}{\overline{m}_l}\overline{m}_l!\right]$ equivalent solutions. For example, for an instance with three periods and three vehicles, if two vehicles are used in each period, there are $\left[\binom{3}{2}2!\right]^3 = 216$ equivalent solutions that can be obtained by re-indexing the vehicles. Such symmetry issues typically slow down the branch-and-bound process due to the duplications in the search process (Sherali and Smith, 2001).

To break the first type of symmetry, we can use the following symmetry breaking constraints (SBCs) to allow vehicle $k+1$ to be dispatched only if vehicle $k$ is also dispatched:

$$\text{(SBC0)} \quad z_{0kt} \geq z_{0,k+1,t} \qquad \forall 1 \leq k \leq m-1, \forall t \in T.$$

To address the second symmetry issue, we can use different sets of symmetry breaking constraints. These sets cannot be imposed together but each of them can be used in conjunction with SBC0. The first set breaks the symmetry of the routes by ordering them according to their total route costs:

$$\text{(SBC1)} \quad \sum_{(i,j)\in E} c_{ij} x_{ijkt} \geq \sum_{(i,j)\in E} c_{ij} x_{ij,k+1,t} \qquad \forall 1 \leq k \leq m-1, \forall t \in T.$$

Alternatively, one can impose that the vehicles be ordered according to their total delivery quantity:

$$\text{(SBC2)} \sum_{i\in N_c} q_{ikt} \geq \sum_{i\in N_c} q_{i,k+1,t} \quad \forall 1 \leq k \leq m-1, \forall t \in T \quad \text{for } F(ML)|k$$

$$\text{or} \sum_{i\in N_c} \sum_{v=\pi(i,t)}^{t-1} g_{ivt} \lambda_{ikvt} \geq \sum_{i\in N_c} \sum_{v=\pi(i,t)}^{t-1} g_{ivt} \lambda_{i,k+1,vt} \qquad \forall 1 \leq k \leq m-1, \forall t \in T$$

$$\text{for } F(OU)|k.$$

We also use the lexicographic ordering constraints (Sherali and Smith, 2001; Degraeve et al., 2002; Jans, 2009) to assign a unique number to each possible set of customers for a route and we order the vehicles according to their assigned number. The lexicographic ordering constraints can be imposed first with respect to customer one only, next with respect to customers one and two, and so on:

$$\text{(SBC3)} \quad \sum_{i=1}^{j} 2^{(j-i)} z_{ikt} \geq \sum_{i=1}^{j} 2^{(j-i)} z_{i,k+1,t} \qquad \forall j \in N_c, \forall 1 \leq k \leq m-1, \forall t \in T.$$

We can also use only the final constraint of SBC3 including all the customers to impose a unique ordering in each period:

$$\text{(SBC4)} \quad \sum_{i=1}^{n} 2^{(n-i)} z_{ikt} \geq \sum_{i=1}^{n} 2^{(n-i)} z_{i,k+1,t} \qquad \forall 1 \leq k \leq m-1, \forall t \in T.$$

### 3.3.2 Valid Inequalities for the Non-Vehicle Index Formulations

To strengthen the non-vehicle index formulations, we add the following inequalities a priori.

$$Qz_{0t} \geq \sum_{i \in N_c} q_{it} \qquad \forall t \in T \quad \text{for } F(ML)|nk \tag{3.55}$$

$$\text{or} \quad Qz_{0t} \geq \sum_{i \in N_c} \sum_{v=\pi(i,t)}^{t-1} g_{ivt} \lambda_{ivt} \qquad \forall t \in T \quad \text{for } F(OU)|nk. \tag{3.56}$$

Constraints (3.55) and (3.56) require the number of vehicles leaving the production facility to be sufficient to carry delivery quantities to all customers in each period.

Since GFSECs (3.28) and (3.53) are generally weak, we further strengthen the non-vehicle index formulations by adding the following subtour elimination constraints (SECs):

$$\sum_{(i,j) \in E(S)} x_{ijt} \leq \sum_{i \in S} z_{it} - z_{et} \quad \forall S \subseteq N_c : |S| \geq 2, e \in S, \forall t \in T. \tag{3.57}$$

These cuts are used to prevent subtours in each period, but they do not take into account the vehicle capacity. Therefore, they have to be used together with GFSECs (3.28) or (3.53) to generate feasible multi-vehicle routes.

To take into account the periodic routing decisions of the MVPRP, we also add another set of constraints to the formulation, called multi-period generalized fractional

subtour elimination constraints (MGFSECs), as follows.

$$Q\sum_{t\in R}\sum_{(i,j)\in E(S)} x_{ijt} \leq \sum_{t\in R}\sum_{i\in S}(Qz_{it}-q_{it}) \;\; \forall S\subseteq N_c : |S|\geq 2, \forall R\subseteq T \tag{3.58}$$

$$\text{or} \quad Q\sum_{t\in R}\sum_{(i,j)\in E(S)} x_{ijt} \leq \sum_{t\in R}\sum_{i\in S}\left(Qz_{it}-\sum_{v=\pi(i,t)}^{t-1} g_{ivt}\lambda_{ivt}\right) \;\; \forall S\subseteq N_c : |S|\geq 2, \forall R\subseteq T. \tag{3.59}$$

Similar to the GFSECs (3.28) and (3.53), constraints (3.58) (for $F(ML)|nk$) and (3.59) (for $F(OU)|nk$) prevent subtours and ensure that the number of vehicles is sufficient to carry the delivery quantity to the set of customers $S$ during the time period set $R$. These constraints are an aggregated version of the GFSECs ((3.28) and (3.53)) and equivalent to GFSECs when $|R| = 1$.

Denote by $\rho(S,r)$ the minimum number of vehicles that must be dispatched to carry the demands in customer set $S$ during periods 1 to $r$, calculated as $\rho(S,r) = \left\lceil \sum_{i\in S}\left(\sum_{t=1}^{r} d_{it} - I_{i0}\right)^{+}/Q \right\rceil$. The following inequalities are also imposed:

$$\sum_{t=1}^{r}\sum_{(i,j)\in\delta(S)} x_{ijt} \geq 2\rho(S,r) \quad \forall S\subseteq N_c : |S|\geq 2, r\in T. \tag{3.60}$$

These constraints ensure that the total number of vehicles entering and leaving the set of customers $S$ from period 1 to period $r$ must be sufficient to carry the demands during these periods. Figure 3.1 illustrates constraints (3.60) when the minimum number of vehicles required to satisfy the total demand to customers 1, 2 and 3 in periods 1 and 2 is equal to two. Therefore, the inequality $x_{011} + x_{021} + x_{031} + x_{012} + x_{022} + x_{032} \geq 4$ is imposed.

## 3.4 Branch-and-Cut Approaches

Since all the formulations contain an exponentially large number of subtour elimination constraints, a natural way to solve the problems is to use a branch-and-cut technique. In this process, the subtour elimination constraints, i.e., constraints (3.11) for the $F(ML)|k$ and $F(OU)|k$, and GFSECs for the $F(ML)|nk$ and $F(OU)|nk$, are

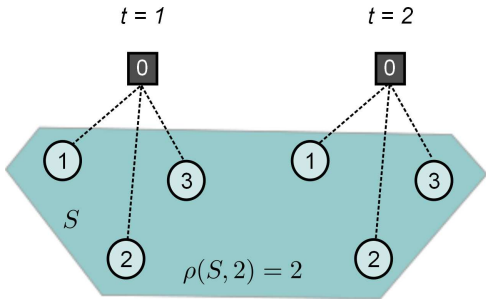



Figure 3.1: Illustration of Constraints (3.60)

dropped from the formulations and are added iteratively when they are violated at each node of the branch-and-bound tree. In this section, we provide the details of our branch-and-cut approaches for both types of formulations. For the variable selection, we first branch on the $y$, $z$, and $x$ variables, respectively. The branching priority is given first to the $y$ variables since the setups generally incur large fixed costs. The next priority is given to the $z$ variables because the $x$ variables are more likely to be integer because of the connectivity constraints (3.10) and (3.26). Also, the $\lambda$ variables can be easily set by inspection when all the $z$ variables are fixed. Among the variables with the same priority, we use the default settings in CPLEX 12.3 to select a specific variable to branch on. The remaining parameters are set to their default values.

### 3.4.1 Branch-and-Cut for the Vehicle Index Formulations

To solve the vehicle index formulations $F(ML)|k$ and $F(OU)|k$, we use an exact separation algorithm that solves a minimum $s-t$ cut problem to detect violated subtour elimination constraints for each vehicle in each period. This is valid since vehicle tours are distinguished by the vehicle index. Denote by $\bar{z}_{it}$ and $\bar{x}_{ijt}$ and the current values of variables $z_{it}$ and $x_{ijt}$. At each node of the branch-and-bound tree, if a subtour on a set of nodes $S$ is found for vehicle $k$ in period $t$, we add the inequalities (3.11) with $e = \text{argmax}_{i \in S}\{\bar{z}_{ikt}\}$ to the formulation. We have implemented the separation algorithm described in Ruokokoski et al. (2010) and use the minimum $s-t$ cut algorithm of the Concorde callable library (Applegate et al., 2011). To find a violated SEC, a graph $G^*_{kt} = (N^*_{kt}, E^*_{kt})$ associated with a fractional solution is constructed for a vehicle $k$ in period $t$ where $N^*_{kt} = \{i \in N | \bar{z}_{ikt} > 0\}$ and $E^*_{kt} =$

$\{(i,j) \in E | \bar{x}_{ijkt} > 0\}$. Then, the minimum $s-t$ cut is solved for each customer $i \in N^*_{kt} \setminus \{0\}$ by setting the plant node as the source node and the selected customer $i$ as the sink node. A violated SEC is identified if the value of the minimum cut is less than $2\bar{z}_{ikt}$. This separation algorithm is more time consuming than the heuristic used by Archetti et al. (2007, 2011) and Solyalı and Süral (2011), but it has proven to be efficient in our branch-and-cut algorithm.

We have also tested different options for adding the subtour elimination constraints to the formulation. The first option is to use constraints (3.11) and when a subtour is detected, add the cut only for the specific vehicle for which it was violated. The second option is to add this cut for *all* vehicles in the same period instead of adding it for the specific vehicle only. The third option is to use an aggregated version of constraints (3.11), which is also equivalent to (3.57):

$$\sum_{k \in K} \sum_{(i,j) \in E(S)} x_{ijkt} \leq \sum_{k \in K} \sum_{i \in S} z_{ikt} - \sum_{k \in K} z_{ekt} \qquad \forall S \subseteq N_c : |S| \geq 2, \forall e \in S, \forall t \in T. \tag{3.61}$$

The results indicate that the first option was the best strategy, while the second option was slightly worse and the third option was far worse than the other two.

### 3.4.2 Branch-and-Cut for the Non-Vehicle Index Formulation

The three different subtour eliminations constraints, GFSECs ((3.28) and (3.53)), SECs (3.57) and MGFSECs ((3.58) and (3.59)), are used for the formulations without vehicle index. To detect SECs, we use the same separation algorithm as for the vehicle index formulations described above to find and generate the cuts for each period $t$. For the GFSECs, we use the four separation algorithms described by Lysgaard et al. (2004) for the CVRP. One of these heuristics is in fact an exact separation procedure when all $x_{ijt}$ variables are integer. Denote by $\bar{z}_{it}$, $\bar{x}_{ijt}$, $\bar{q}_{it}$ and $\bar{\lambda}_{ivt}$ the current values of variables $z_{it}$, $x_{ijt}$, $q_{it}$ and $\lambda_{ivt}$ in the branch-and-bound tree. The CVRP solution for the separation routine in period $t$ can be constructed as follows. First, we consider only the nodes for which $\bar{z}_{it} > 0$. Then, the weight of edge $(i,j)$ is set to $\bar{x}_{ijt}$ and the delivery quantity for customer $i$ is set to $\bar{q}_{it}$ for $F(ML)|nk$ or to $\sum_{v=\pi(i,t)}^{t-1} g_{ivt}\bar{\lambda}_{ivt}$ for

$F(OU)|nk$. Note that the maximum number of subsets produced by the separation algorithms for each period $t$ is limited to $n$ per call. Then, the violated GFSECs are added to the formulation.

For the MGFSECs, we developed a greedy heuristic separation algorithm. Denote by $\bar{z}_{it}$ the current solution values of variables $z_{it}$. For each subset $R \subseteq T$, we consider the set of customers with $\sum_{t \in R} \bar{z}_{it} > 0$ and calculate the value $s_i = \sum_{t \in R} \bar{q}_{it} / \sum_{t \in R} \lceil \bar{z}_{it} \rceil$ (or $s_i = \sum_{t \in R} \sum_{v=\pi(i,t)}^{t-1} g_{ivt} \bar{\lambda}_{ivt} / \sum_{t \in R} \lceil \bar{z}_{it} \rceil$ for $F(OU)|nk$), which represents the average delivery quantity per visit to customer $i$ during the period set $R$. Customers are ranked in descending order of the value of $s_i$ and stored in an ordered list. Then, an empty set of customers $S$ and an empty set of violated sets $\xi(S)$ are created. The separation algorithm starts by adding the first customer in the ordered list to $S$ and checks whether the MGFSEC of the set $S$ is violated. The next customer is then added to $S$ and the algorithm checks for the MGFSEC again, and so on. The violated MGFSECs are stored in $\xi(S)$. If a violated MGFSEC is found or all the customers in the ordered list have been added to $S$, the set is emptied and the first customer in the ordered list is removed. The algorithm then starts again by adding the new first customer in the ordered list to $S$. This process is repeated until the ordered list is empty or $n$ violated cuts are found. In our implementation, we consider the subset $R$ of all two and three consecutive periods in $T$.

The separation algorithm above can also be used to generate the cuts (3.60). We first set $R = \{t : 1 \leq t \leq r\}, \forall r \in T$, or the set from time period 1 to $r \in T$, and use the demand $d_{it}$ in place of the delivery quantity $\bar{q}_{it}$ in the calculation of $s_i$. The algorithm is then set to detect the cuts (3.60) instead of MGFSECs.

Because it is very time consuming to solve all separation problems at every node of the branch-and-bound tree, we use the following cut generation strategy for the non-vehicle index formulations. At the root node, all separation algorithms are called to generate the GFSECs, SECs, MGFSECs and (3.60). At each further node of the tree, only the GFSECs and SECs are considered in the following sequence: (1) the separation algorithm for the SECs is called, (2) if there is no violated SECs, the separation algorithms for the GFSECs are called.

The process described above can significantly improve the performance of the algorithm since using GFSECs alone is inefficient due to the fractional coefficients. Using the SECs can efficiently eliminate subtours (even though the vehicle capacity can still be violated), while adding GFSECs and MGFSECs can eliminate the routes with exceeded vehicle capacity. The computational results in Section 3.6.2.2 show a significant performance improvement by using the three cuts together compared to using GFSECs alone.

## 3.5 Heuristics for Setting Initial Upper Bound

We develop a heuristic to compute upper bounds used in the branch-and-cut algorithms. This heuristic is based on the adaptive large neighborhood search (ALNS) framework proposed by Ropke and Pisinger (2006) for the VRP. The basic idea of the ALNS is to repeatedly destroy and repair a solution using several heuristic operators to seek for improvement. These operators are probabilistically selected based on empirical scores related to their success in terms of finding improved solutions. We use specialized operators to handle the binary variables, and the remaining continuous variables are set by solving a network flow model embedded into these operators. We call this procedure an optimization-based adaptive large neighborhood search (Op-ALNS). The details of the heuristic procedure and its adaptations for the variants of the PRP are presented in Chapter 4.

## 3.6 Computational Experiments

The branch-and-cut algorithms were coded in C# on MonoDevelop 2.2 using CPLEX 12.3 under Scientific Linux 6.1. The experiments were performed on a workstation with an Intel Xeon 2.67GHz processor and 24GB of RAM. A multiple core processor where each core has the same specifications was also used for the parallel computing experiments in Section 3.6.4. The Op-ALNS heuristic was coded in C# using Microsoft Visual Studio 2008 and executed on a workstation with a 2.10 GHz CPU and 2 GB of RAM under Windows XP.

In all tables, we report the average CPU times in seconds and the average number of nodes in the columns *CPU* and *Nodes*, respectively. Column *%LB* shows the

final lower bound as a percentage of the best upper bound found by all approaches. Boldface letters are used to indicate the best results. If all instances of a given problem size are solved to optimality, we put boldface letters on the smallest total computing time, otherwise boldface letters are put on the best average percentage of lower bounds.

### 3.6.1 Details of the Instances

We created two test sets, i.e., MVPRP and MVIRP, from the instances available in the literature. For the MVPRP instances, there are two published datasets that are used in several studies, i.e., Boudia et al. (2005) and Archetti et al. (2011), but both datasets were designed for heuristics and the instances are too large for our exact algorithms. We thus generated smaller instances for our computational experiments. Because the Archetti et al. (2011) dataset takes into account many different aspects, e.g., inventory costs at customers, initial inventory at customers, and varying transportation and production costs, while the Boudia et al. (2005) instances have zero inventory cost at the customers and the problem sizes are generally too large, we decided to use a subset of the Archetti et al. (2011) dataset to create our own test set. The generated MVPRP dataset consists of instances with $n = 10$ to 50, to 40 and to 30 customers for time horizons with $l = 3, 6$ and 9 periods, respectively. The number of vehicles is set to $m = 2$ or 3 for the instances with $n \leq 25$ and to $m = 3$ or 4 for the instances with $25 < n \leq 50$. For the MVIRP instances, we adapted the IRP instances for the single vehicle case presented in Archetti et al. (2007). Our test set consists of the instances with 5 to 50 and to 25 customers for time horizons with $l = 3$ and 6 periods, respectively. The number of vehicles is set using the same method as for the MVPRP instances. There are 336 MVPRP instances and 600 MVIRP instances for both the ML and OU policy. The details of the instances are provided in the Appendix.

### 3.6.2 Effect of Valid Inequalities

This section presents the analysis of the inequalities introduced in Section 3.3. These experiments were conducted on the instances with $n \leq 15$ and the computing time limit was set to one hour. The detailed results are provided in the Appendix.

#### 3.6.2.1 Effect of Vehicle Symmetry Breaking Constraints on the Vehicle Index Formulations

We analyze the effect of symmetry breaking constraints SBC0 alone and SBC0 together with one of the other constraints SBC1-SBC4 for the formulations $F(ML)|k$ and $F(OU)|k$. The average results on the MVPRP and MVIRP instances are shown in Table 3.II. In our tests, we used the default settings of CPLEX which allow the solver to detect and generate its own symmetry breaking constraints.

Table 3.II: Average Results with Different SBCs on the MVPRP and MVIRP Instances

| Problem | # | None | | | SBC0 | | | SBC0+1 | | | SBC0+2 | | | SBC0+3 | | | SBC0+4 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | %LB | CPU | Nodes | %LB | CPU | Nodes | %LB | CPU | Nodes | %LB | CPU | Nodes | %LB | CPU | Nodes | %LB | CPU | Nodes |
| MVPRP-ML | | | | | | | | | | | | | | | | | | | |
| Optimal | 40 | 99.9 | 659.9 | 6128 | 100.0 | 417.1 | 3574 | 100.0 | 216.2 | 1374 | 100.0 | 332.2 | 2664 | 100.0 | 161.6 | 1473 | 100.0 | **147.2** | 1367 |
| Not optimal | 8 | 97.8 | 3173.2 | 11151 | 97.9 | 2987.9 | 9386 | **98.1** | 2818.9 | 7307 | 97.9 | 2966.8 | 8425 | **98.1** | 2794.0 | 8121 | **98.1** | 2792.7 | 8564 |
| Total | 48 | 99.5 | 1078.8 | 6965 | 99.6 | 845.6 | 4543 | **99.7** | 650.0 | 2363 | 99.6 | 771.3 | 3624 | **99.7** | 600.4 | 2581 | **99.7** | 588.1 | 2567 |
| MVPRP-OU | | | | | | | | | | | | | | | | | | | |
| Optimal | 44 | 100.0 | 244.3 | 2124 | 100.0 | 240.8 | 1802 | 100.0 | 134.5 | 903 | 100.0 | 232.6 | 1662 | 100.0 | **118.0** | 786 | 100.0 | 193.5 | 1208 |
| Not optimal | 4 | 97.6 | 3600.0 | 8421 | 97.5 | 3600.0 | 8978 | 97.7 | 3600.0 | 7974 | 97.3 | 3600.0 | 7991 | **98.2** | 3564.7 | 9041 | 97.8 | 3600.0 | 7893 |
| Total | 48 | 99.8 | 524.0 | 2648 | 99.8 | 520.7 | 2400 | 99.8 | 423.3 | 1492 | 99.7 | 513.2 | 2189 | **99.9** | 405.2 | 1474 | 99.8 | 477.4 | 1765 |
| MVIRP-ML | | | | | | | | | | | | | | | | | | | |
| Optimal | 100 | 99.6 | 807.4 | 82297 | 99.9 | 607.9 | 66896 | 100.0 | 173.2 | 7751 | 99.9 | 520.4 | 55501 | 100.0 | **88.4** | 4568 | 100.0 | 98.9 | 4954 |
| Not optimal | 20 | 89.3 | 3600.0 | 36402 | 90.3 | 3600.0 | 36170 | 92.0 | 3600.0 | 34414 | 91.1 | 3600.0 | 37335 | **95.6** | 3261.5 | 41581 | 94.9 | 3438.3 | 45567 |
| Total | 120 | 97.9 | 1272.8 | 74648 | 98.3 | 1106.6 | 61775 | 98.7 | 744.3 | 12195 | 98.4 | 1033.7 | 52474 | **99.3** | 617.3 | 10737 | 99.1 | 655.5 | 11723 |
| MVIRP-OU | | | | | | | | | | | | | | | | | | | |
| Optimal | 100 | 99.9 | 369.6 | 5518 | 99.9 | 288.6 | 3803 | 100.0 | 142.1 | 1977 | 100.0 | 269.6 | 3733 | 100.0 | **98.9** | 1460 | 100.0 | 104.6 | 1518 |
| Not optimal | 20 | 88.8 | 3600.0 | 29863 | 90.2 | 3600.0 | 31349 | 90.7 | 3600.0 | 26404 | 89.4 | 3600.0 | 29394 | **95.4** | 2981.2 | 23282 | 95.3 | 3025.4 | 24768 |
| Total | 120 | 98.1 | 908.0 | 9575 | 98.3 | 840.5 | 8394 | 98.4 | 718.4 | 6048 | 98.2 | 824.7 | 8009 | **99.2** | 579.3 | 5097 | **99.2** | 591.4 | 5393 |

Table 3.III provides a summary of the time reduction factors for each approach, calculated as the average computing time spent to solve an instance size without using any of our SBCs, divided by the average computing time of using each cut strategy. A time factor equal to 2 means the algorithm spent on average only half the time by using the SBC strategy compared to using no additional SBCs.

The results clearly show the benefits of using the SBCs compared to relying only on those generated by CPLEX. The original formulation without any additional SBC

Table 3.III: Summary of the Time Reduction Factors with Different SBCs on Instances Solved to Optimality

| Problem | Value | SBC0 | SBC0+1 | SBC0+2 | SBC0+3 | SBC0+4 |
|---|---|---|---|---|---|---|
| MVPRP-ML | Min | 1.16 | 1.62 | 1.29 | **1.65** | 1.48 |
| | Max | 4.42 | 6.45 | 5.76 | **11.65** | 9.26 |
| | Avg | 2.11 | 3.25 | 2.38 | **4.53** | 4.15 |
| MVPRP-OU | Min | 0.93 | 0.91 | 0.76 | 0.98 | **0.99** |
| | Max | 1.47 | **6.13** | 2.14 | 5.20 | 5.31 |
| | Avg | 1.14 | 1.72 | 1.15 | **2.03** | 1.94 |
| MVIRP-ML | Min | 0.94 | 1.24 | 1.12 | 1.39 | **1.59** |
| | Max | 3.87 | 16.55 | 3.49 | **42.50** | 40.64 |
| | Avg | 1.87 | 4.89 | 1.99 | **9.70** | 9.09 |
| MVIRP-OU | Min | 0.80 | 0.62 | 0.43 | **1.50** | 1.36 |
| | Max | 2.15 | 3.35 | 1.89 | **7.92** | 6.51 |
| | Avg | 1.19 | 1.49 | 1.08 | **3.13** | 3.02 |

provides the worst results and adding SBC0 could generally improve the computing times and reduce the number of nodes in the branch-and-bound tree. The combination of SBC0 together with one of the other SBCs could further speed up the solution process, except for SBC2 where some results are worse than using the CPLEX cuts alone. The cut strategies SBC0+SBC3 and SBC0+SBC4 provide good results, but SBC0+SBC3 is slightly better overall. The average time factor reductions obtained by using SBC0+SBC3 within the maximum computing time limit of one hour are 4.53, 2.03, 9.70 and 3.13 for the MVPRP-ML, MVPRP-OU, MVIRP-ML and MVIRP-OU instances, respectively. By adding SBC0+SBC3 to the $F(ML)|k$ and $F(OU)|k$ formulations, the algorithm could also solve 36 instances that could not be solved to optimality within one hour using the formulations without these inequalities. We thus consider the formulations $F(ML)|k$ and $F(OU)|k$ with SBC0+SBC3 in the remaining computational experiments.

#### 3.6.2.2 Effect of Valid Inequalities for the Non-Vehicle Index Formulations

In this section, we analyze the effect of the valid inequalities that we implemented for the formulations $F(ML)|nk$ and $F(OU)|nk$. First, we evaluate the effects of the valid inequalities on the lower bounds at the root node of the branch-and-bound tree. To avoid misinterpretation due to the impact of the CPLEX cuts, we conducted the experiments without these cuts. The average lower bounds are shown in Table

3.IV. The numbers presented are equal to the average lower bounds at the root node compared to the optimal solutions or the best upper bounds if the instances were not solved to optimality. Each column shows the results of using each cut presented in Section 3.3.2, where the columns *None* and *All* present the results without using any additional cuts (i.e., only GFSECs (3.28) and (3.53) are applied) and with all cuts together, respectively.

Table 3.IV: Effects of the Valid Inequalities for the Non-Vehicle Index Formulations on Average Lower Bounds at the Root Node for the MVPRP and MVIRP Instances

| Problem | Maximum Level (ML) | | | | | | Order-Up-To Level (OU) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | None | (3.55) | (3.57) | (3.58) | (3.60) | All | None | (3.56) | (3.57) | (3.59) | (3.60) | All |
| MVPRP | 93.3 | 93.3 | 94.7 | 93.5 | 93.7 | **95.2** | 94.0 | 94.6 | 94.5 | 94.4 | 94.1 | **95.2** |
| MVIRP | 83.5 | 83.6 | 85.4 | 84.8 | 87.4 | **89.0** | 85.7 | 86.1 | 88.3 | 87.1 | 87.4 | **89.7** |

The results show that adding all the cuts together generally provides the best lower bounds and it has more effect on larger instances. We can also observe that the inequalities have more effect on the lower bound at the root node for the MVIRP than for the MVPRP. Note that the average CPU times at the root node without any additional inequalities for the MVPRP and MVIRP are 0.4 and 0.2 seconds, respectively, and with all the inequalities are 1.3 and 0.5 seconds, respectively.

In Table 3.V, we report the average results of using all these cuts with our exact algorithms. The results of the branch-and-cut without and with the additional valid inequalities from Section 3.3.2 are shown in columns $F(ML)|nk$, $F(OU)|nk$ and in columns $F(ML)|nk^+$, $F(OU)|nk^+$, respectively.

Table 3.V: Effects of the Valid Inequalities for the Non-Vehicle Index Formulations on the Branch-and-Cut Algorithm for the MVPRP and MVIRP Instances

| Problem | Maximum Level (ML) | | | | | | | Order-Up-To Level (OU) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | # | $F(ML)\|nk$ | | | $F(ML)\|nk^+$ | | | # | $F(OU)\|nk$ | | | $F(OU)\|nk^+$ | | |
| | | %LB | CPU | Nodes | %LB | CPU | Nodes | | %LB | CPU | Nodes | %LB | CPU | Nodes |
| MVPRP | | | | | | | | | | | | | | |
| Optimal | 32 | 100.0 | 397.3 | 28639 | 100.0 | **161.3** | 10337 | 40 | 100.0 | 109.7 | 7325 | 100.0 | **69.6** | 5105 |
| Not optimal | 16 | 98.9 | 3600.0 | 83703 | **99.3** | 2954.0 | 47366 | 8 | 98.8 | 3530.5 | 65816 | **98.9** | 3255.2 | 43935 |
| Total | 48 | 99.6 | 1464.9 | 46994 | **99.8** | 1092.2 | 22680 | 48 | **99.8** | 679.8 | 17073 | **99.8** | 600.6 | 11576 |
| MVIRP | | | | | | | | | | | | | | |
| Optimal | 80 | 99.9 | 215.8 | 36379 | 100.0 | **130.9** | 23356 | 80 | 100.0 | 66.1 | 5573 | 100.0 | **14.0** | 1215 |
| Not optimal | 40 | 95.4 | 3039.4 | 154296 | **96.5** | 3077.9 | 124501 | 40 | 95.0 | 3105.8 | 155052 | **96.5** | 3014.3 | 113077 |
| Total | 120 | 98.4 | 1157.0 | 75685 | **98.8** | 1113.2 | 57071 | 120 | 98.3 | 1079.4 | 55399 | **98.8** | 1014.1 | 38502 |

We see that applying all the inequalities of Section 3.3.2 could provide significant improvements in the branch-and-cut procedure for both formulations $F(ML)|nk$ and $F(OU)|nk$ in terms of lower bounds (both at the root node and final lower bounds), computing times and the number of nodes in the branch-and-bound tree. We thus used the formulations $F(ML)|nk^+$ and $F(OU)|nk^+$ in the further computational experiments.

### 3.6.3 Comparison of the Vehicle Index and Non-Vehicle Index Formulations

In this section, we compare the performance of the vehicle index formulations $F(ML)|k$ and $F(OU)|k$, and the non-vehicle index formulations $F(ML)|nk$ and $F(OU)|nk$ using the full test sets. The Op-ALNS procedure as described in Section 3.5 is used to calculate the initial upper bounds and these bounds are given to CPLEX before the branch-and-cut algorithm starts. The computing time limit for the branch-and-cut approaches was set to two hours in these experiments. The results are provided in Tables 3.VI and 3.VII. The column $h_i$ in Table 3.VII indicates the group of the MVIRP instances, i.e., low (L) or high (H) inventory costs. We report the quality of the initial upper bounds using the Op-ALNS procedure next to the results of the exact algorithms and the column *%Diff* indicates the percentage difference of the total costs obtained by the Op-ALNS from the optimal objective value or the best upper bound found by the vehicle index and non-vehicle index formulations if the instances were not solved to optimality.

Table 3.VI: Average Results on MVPRP Instances

| $n$ | $l$ | $m$ | MVPRP-ML | | | | | | MVPRP-OU | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $F(ML)\|k$ | | $F(ML)\|nk$ | | Op-ALNS | | $F(OU)\|k$ | | $F(OU)\|nk$ | | Op-ALNS | |
| | | | %LB | CPU | %LB | CPU | %DIFF | CPU | %LB | CPU | %LB | CPU | %DIFF | CPU |
| 10 | 3 | 2 | 100.0 | 0.2 | 100.0 | **0.1** | 0.4 | 4.6 | 100.0 | 0.4 | 100.0 | **0.1** | 0.0 | 4.2 |
| 10 | 3 | 3 | 100.0 | 0.6 | 100.0 | **0.3** | 1.1 | 4.3 | 100.0 | 0.6 | 100.0 | **0.2** | 0.0 | 4.4 |
| 15 | 3 | 2 | 100.0 | **2.5** | 100.0 | 8.6 | 0.9 | 6.6 | 100.0 | **17.6** | 100.0 | 54.2 | 2.0 | 5.8 |
| 15 | 3 | 3 | 100.0 | **26.6** | 100.0 | 35.4 | 1.0 | 6.6 | 100.0 | **17.6** | 100.0 | 234.4 | 1.0 | 6.7 |
| 20 | 3 | 2 | 100.0 | **2.6** | 100.0 | 4.0 | 1.0 | 9.8 | 100.0 | **59.6** | 100.0 | 91.5 | 1.5 | 7.4 |
| 20 | 3 | 3 | 100.0 | **51.1** | 100.0 | 70.9 | 0.8 | 10.0 | 100.0 | **692.5** | 100.0 | 1193.7 | 1.3 | 8.4 |
| 25 | 3 | 2 | 100.0 | **3.9** | 100.0 | 9.5 | 1.5 | 12.4 | 100.0 | **2854.8** | $99.5^{(4)}$ | 7200.0 | 1.9 | 11.1 |
| 25 | 3 | 3 | 100.0 | **85.8** | 100.0 | 589.5 | 1.4 | 14.3 | $98.4^{(4)}$ | 7200.0 | $\mathbf{98.8}^{(4)}$ | 7200.0 | 2.5 | 14.3 |
| 30 | 3 | 3 | 100.0 | **194.2** | $99.9^{(1)}$ | 1954.5 | 0.9 | 23.7 | $97.8^{(4)}$ | 7200.0 | $\mathbf{98.0}^{(4)}$ | 7200.0 | 1.7 | 19.9 |
| 30 | 3 | 4 | 100.0 | **2027.8** | $99.8^{(2)}$ | 4463.3 | 1.7 | 28.1 | $96.3^{(4)}$ | 7200.0 | $\mathbf{97.1}^{(4)}$ | 7200.0 | 1.5 | 23.3 |
| 35 | 3 | 3 | 100.0 | **1221.2** | $99.5^{(2)}$ | 4279.9 | 2.4 | 36.7 | $95.6^{(4)}$ | 7200.0 | $\mathbf{96.0}^{(4)}$ | 7200.0 | 0.0 | 28.8 |
| 35 | 3 | 4 | $\mathbf{99.0}^{(3)}$ | 6515.4 | $\mathbf{99.0}^{(4)}$ | 7200.0 | 2.1 | 43.0 | $94.6^{(4)}$ | 7200.0 | $\mathbf{95.4}^{(4)}$ | 7200.0 | 0.5 | 37.7 |
| 40 | 3 | 3 | $\mathbf{99.7}^{(1)}$ | 4097.6 | $99.4^{(4)}$ | 7200.0 | 1.3 | 51.7 | $97.6^{(4)}$ | 7200.0 | $\mathbf{97.8}^{(4)}$ | 7200.0 | 0.6 | 32.0 |
| 40 | 3 | 4 | $98.3^{(3)}$ | 5970.7 | $\mathbf{98.9}^{(4)}$ | 7200.0 | 0.8 | 52.4 | $95.3^{(4)}$ | 7200.0 | $\mathbf{95.8}^{(4)}$ | 7200.0 | 0.0 | 42.9 |
| 45 | 3 | 3 | $99.1^{(3)}$ | 5647.0 | $\mathbf{99.4}^{(4)}$ | 7200.0 | 0.6 | 67.5 | $96.7^{(4)}$ | 7200.0 | $\mathbf{97.0}^{(4)}$ | 7200.0 | 0.2 | 45.9 |
| 45 | 3 | 4 | $97.2^{(4)}$ | 7200.0 | $\mathbf{97.8}^{(4)}$ | 7200.0 | 0.5 | 72.1 | $95.0^{(4)}$ | 7200.0 | $\mathbf{95.7}^{(4)}$ | 7200.0 | 0.0 | 56.7 |
| 50 | 3 | 3 | $\mathbf{99.3}^{(2)}$ | 5243.6 | $\mathbf{99.3}^{(4)}$ | 7200.0 | 1.2 | 90.4 | $96.2^{(4)}$ | 7200.0 | $\mathbf{96.5}^{(4)}$ | 7200.0 | 0.0 | 59.2 |
| 50 | 3 | 4 | $98.3^{(3)}$ | 7058.4 | $\mathbf{98.5}^{(4)}$ | 7200.0 | 0.4 | 85.1 | $94.5^{(4)}$ | 7200.0 | $\mathbf{95.1}^{(4)}$ | 7200.0 | 0.0 | 65.9 |
| 10 | 6 | 2 | 100.0 | 1.9 | 100.0 | **0.6** | 0.6 | 7.3 | 100.0 | 0.5 | 100.0 | **0.3** | 0.1 | 9.5 |
| 10 | 6 | 3 | 100.0 | **12.5** | 100.0 | 15.1 | 0.4 | 8.4 | 100.0 | 1.3 | 100.0 | **0.2** | 0.1 | 14.0 |
| 15 | 6 | 2 | 100.0 | **97.5** | $99.8^{(1)}$ | 1940.9 | 1.0 | 13.8 | 100.0 | **8.6** | 100.0 | 35.3 | 0.4 | 14.1 |
| 15 | 6 | 3 | 100.0 | **1105.8** | $99.5^{(4)}$ | 7200.0 | 1.6 | 14.0 | 100.0 | **73.8** | 100.0 | 146.8 | 0.7 | 17.6 |
| 20 | 6 | 2 | 100.0 | **84.9** | $99.8^{(1)}$ | 2501.2 | 1.3 | 22.1 | 100.0 | **23.7** | 100.0 | 281.6 | 0.4 | 22.4 |
| 20 | 6 | 3 | 100.0 | **806.5** | $99.5^{(2)}$ | 3608.5 | 1.5 | 20.6 | 100.0 | **269.7** | 100.0 | 297.2 | 0.5 | 32.7 |
| 25 | 6 | 2 | 100.0 | **170.6** | $99.9^{(2)}$ | 3663.4 | 1.8 | 28.4 | 100.0 | **328.8** | 100.0 | 1552.4 | 1.2 | 30.1 |
| 25 | 6 | 3 | $\mathbf{99.9}^{(1)}$ | 2811.0 | $99.4^{(2)}$ | 4150.6 | 1.7 | 34.2 | $99.3^{(4)}$ | 7200.0 | $\mathbf{99.4}^{(4)}$ | 7200.0 | 1.6 | 48.4 |
| 30 | 6 | 3 | $\mathbf{99.4}^{(2)}$ | 4347.1 | $99.3^{(4)}$ | 7200.0 | 1.4 | 52.3 | 100.0 | **1634.9** | $99.7^{(2)}$ | 4002.2 | 0.5 | 58.7 |
| 30 | 6 | 4 | $97.9^{(4)}$ | 7200.0 | $\mathbf{98.8}^{(4)}$ | 7200.0 | 0.4 | 59.3 | $98.9^{(4)}$ | 7200.0 | $\mathbf{99.2}^{(4)}$ | 7200.0 | 1.1 | 90.3 |
| 10 | 9 | 2 | 100.0 | **20.5** | 100.0 | 30.2 | 1.8 | 13.8 | 100.0 | 15.3 | 100.0 | **4.5** | 1.4 | 18.4 |
| 10 | 9 | 3 | 100.0 | **405.2** | 100.0 | 1095.9 | 1.8 | 12.9 | 100.0 | 75.3 | 100.0 | **35.5** | 1.3 | 26.7 |
| 15 | 9 | 2 | $\mathbf{99.6}^{(2)}$ | 3797.7 | $99.4^{(4)}$ | 7200.0 | 1.6 | 24.1 | 100.0 | **887.1** | $99.4^{(2)}$ | 5143.6 | 1.2 | 28.2 |
| 15 | 9 | 3 | $97.4^{(4)}$ | 7200.0 | $\mathbf{98.7}^{(4)}$ | 7200.0 | 1.9 | 25.7 | $98.6^{(2)}$ | 5612.0 | $\mathbf{98.8}^{(4)}$ | 7200.0 | 1.5 | 55.7 |
| 20 | 9 | 2 | 100.0 | **1480.0** | $99.6^{(2)}$ | 5859.9 | 1.0 | 36.8 | 100.0 | **590.9** | $99.7^{(2)}$ | 4081.3 | 0.8 | 42.6 |
| 20 | 9 | 3 | $98.9^{(2)}$ | 4546.3 | $\mathbf{99.2}^{(4)}$ | 7200.0 | 1.0 | 39.2 | $99.2^{(3)}$ | 6430.5 | $\mathbf{99.3}^{(3)}$ | 5737.2 | 0.4 | 70.4 |
| Optimal | | | 100.0 | **371.5** | 99.9 | 1777.7 | 1.2 | 16.0 | 100.0 | **397.5** | 99.9 | 1281.8 | 0.9 | 19.1 |
| Not optimal | | | 98.8 | 5510.4 | **99.0** | 6965.4 | 1.1 | 53.6 | 96.9 | 7042.8 | **97.3** | 7102.5 | 0.8 | 46.1 |
| Total | | | 99.5 | 2336.4 | **99.6** | 3761.2 | 1.2 | 30.4 | 98.6 | 3329.3 | **98.8** | 3849.8 | 0.8 | 31.0 |

$^{(-)}$ indicates the number of instances (out of 4) that were not solved to optimality

Table 3.VII: Average Results on MVIRP Instances.

| $n$ | $l$ | $m$ | $h_i$ | MVIRP-ML | | | | | | MVIRP-OU | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $F(ML)\|k$ | | $F(ML)\|nk$ | | Op-ALNS | | $F(OU)\|k$ | | $F(OU)\|nk$ | | Op-ALNS | |
| | | | | %LB | CPU | %LB | CPU | %DIFF | CPU | %LB | CPU | %LB | CPU | %DIFF | CPU |
| 5 | 3 | 2 | L | 100.0 | **0.1** | 100.0 | **0.1** | 1.3 | 3.5 | 100.0 | **0.0** | 100.0 | 0.1 | 0.5 | 3.4 |
| 5 | 3 | 3 | L | 100.0 | 0.3 | 100.0 | **0.2** | 0.5 | 3.7 | 100.0 | **0.0** | 100.0 | 0.1 | 0.9 | 3.8 |
| 10 | 3 | 2 | L | 100.0 | **1.3** | 100.0 | 3.3 | 3.0 | 5.5 | 100.0 | **2.0** | 100.0 | 3.0 | 2.8 | 6.6 |
| 10 | 3 | 3 | L | 100.0 | **8.3** | 100.0 | 19.2 | 4.1 | 6.0 | 100.0 | 4.8 | 100.0 | **4.4** | 1.1 | 7.4 |
| 15 | 3 | 2 | L | 100.0 | **4.4** | 100.0 | 26.7 | 2.2 | 7.9 | 100.0 | **5.4** | 100.0 | 26.7 | 6.1 | 8.1 |
| 15 | 3 | 3 | L | 100.0 | **38.5** | 100.0 | 1701.0 | 7.0 | 10.3 | 100.0 | **19.1** | 100.0 | 83.3 | 9.2 | 13.7 |
| 20 | 3 | 2 | L | 100.0 | **31.2** | 100.0 | 1054.5 | 6.1 | 10.9 | 100.0 | **67.1** | 99.8$^{(1)}$ | 1791.9 | 6.3 | 10.6 |
| 20 | 3 | 3 | L | 100.0 | **386.8** | 96.9$^{(3)}$ | 4540.2 | 5.3 | 11.6 | 100.0 | **509.0** | 98.4$^{(2)}$ | 4400.1 | 9.9 | 14.1 |
| 25 | 3 | 2 | L | 100.0 | **71.0** | 99.1$^{(1)}$ | 1837.9 | 5.4 | 18.0 | 100.0 | **127.8** | 100.0 | 503.6 | 10.6 | 4.6 |
| 25 | 3 | 3 | L | 100.0 | **978.3** | 96.2$^{(5)}$ | 7200.0 | 6.1 | 17.2 | 100.0 | **1035.3** | 98.7$^{(3)}$ | 5764.7 | 8.8 | 22.9 |
| 30 | 3 | 3 | L | 100.0 | **1962.3** | 94.5$^{(4)}$ | 6091.3 | 7.2 | 24.8 | 100.0 | **3054.1** | 97.1$^{(4)}$ | 6593.6 | 8.8 | 29.0 |
| 30 | 3 | 4 | L | **93.4**$^{(4)}$ | 6203.8 | 92.8$^{(5)}$ | 7200.0 | 5.9 | 27.3 | 91.8$^{(4)}$ | 6586.0 | **94.4**$^{(5)}$ | 7200.0 | 8.9 | 37.4 |
| 35 | 3 | 3 | L | 100.0 | **4124.9** | 94.1$^{(5)}$ | 7200.0 | 6.5 | 32.7 | **96.4**$^{(3)}$ | 5262.4 | 96.1$^{(5)}$ | 7200.0 | 9.9 | 41.3 |
| 35 | 3 | 4 | L | 90.4$^{(5)}$ | 7200.0 | **91.5**$^{(5)}$ | 7200.0 | 5.2 | 36.1 | 86.9$^{(5)}$ | 7200.0 | **91.5**$^{(5)}$ | 7200.0 | 6.9 | 51.6 |
| 40 | 3 | 3 | L | **97.3**$^{(3)}$ | 5699.9 | 94.1$^{(5)}$ | 7200.0 | 7.4 | 47.0 | **97.1**$^{(4)}$ | 7078.1 | 95.9$^{(4)}$ | 7200.0 | 8.5 | 47.7 |
| 40 | 3 | 4 | L | 86.7$^{(5)}$ | 7200.0 | **90.3**$^{(5)}$ | 7200.0 | 1.9 | 49.9 | 83.9$^{(5)}$ | 7200.0 | **89.5**$^{(5)}$ | 7200.0 | 5.4 | 62.0 |
| 5 | 6 | 2 | L | 100.0 | **4.0** | 100.0 | 15.6 | 3.6 | 5.7 | 100.0 | **1.5** | 100.0 | 4.6 | 2.8 | 5.9 |
| 5 | 6 | 3 | L | 100.0 | **140.9** | 100.0 | 273.3 | 3.3 | 6.8 | 100.0 | **0.2** | 100.0 | 3.6 | 1.4 | 7.6 |
| 10 | 6 | 2 | L | 100.0 | **168.2** | 98.5$^{(2)}$ | 3017.7 | 4.2 | 9.7 | 100.0 | **219.6** | 99.3$^{(1)}$ | 2295.7 | 8.2 | 9.7 |
| 10 | 6 | 3 | L | **96.7**$^{(2)}$ | 4877.0 | 96.4$^{(4)}$ | 6083.8 | 3.8 | 11.4 | **99.1**$^{(1)}$ | 3704.9 | 97.4$^{(2)}$ | 5449.0 | 5.3 | 12.0 |
| 15 | 6 | 2 | L | 100.0 | **950.5** | 96.3$^{(4)}$ | 7200.0 | 3.9 | 18.0 | 100.0 | **736.1** | 96.4$^{(4)}$ | 7200.0 | 10.1 | 16.5 |
| 15 | 6 | 3 | L | **96.1**$^{(5)}$ | 7200.0 | 92.9$^{(5)}$ | 7200.0 | 4.1 | 18.8 | **93.5**$^{(5)}$ | 7200.0 | 92.3$^{(5)}$ | 7200.0 | 8.7 | 20.6 |
| 5 | 3 | 2 | H | 100.0 | **0.1** | 100.0 | **0.1** | 0.9 | 3.2 | 100.0 | **0.1** | 100.0 | **0.1** | 0.0 | 3.3 |
| 5 | 3 | 3 | H | 100.0 | 0.3 | 100.0 | **0.1** | 0.5 | 3.7 | 100.0 | **0.0** | 100.0 | 0.1 | 0.0 | 4.0 |
| 10 | 3 | 2 | H | 100.0 | **1.5** | 100.0 | 6.1 | 2.1 | 5.3 | 100.0 | **2.1** | 100.0 | 3.0 | 0.9 | 6.2 |
| 10 | 3 | 3 | H | 100.0 | **9.8** | 100.0 | 29.7 | 2.6 | 6.0 | 100.0 | **4.9** | 100.0 | 5.7 | 2.2 | 7.1 |
| 15 | 3 | 2 | H | 100.0 | **4.4** | 100.0 | 42.0 | 2.0 | 7.8 | 100.0 | **6.3** | 100.0 | 37.5 | 4.0 | 9.4 |
| 15 | 3 | 3 | H | 100.0 | **33.6** | 100.0 | 980.9 | 4.0 | 8.5 | 100.0 | **16.5** | 100.0 | 39.1 | 3.8 | 13.9 |
| 20 | 3 | 2 | H | 100.0 | **30.5** | 100.0 | 1586.6 | 2.4 | 11.4 | 100.0 | **63.0** | 100.0 | 1766.1 | 3.6 | 11.9 |
| 20 | 3 | 3 | H | 100.0 | **373.7** | 98.6$^{(3)}$ | 4746.9 | 4.6 | 11.3 | 100.0 | **570.5** | 99.3$^{(1)}$ | 4099.1 | 6.1 | 15.9 |
| 25 | 3 | 2 | H | 100.0 | **56.6** | 99.7$^{(1)}$ | 1800.1 | 2.8 | 16.8 | 100.0 | **122.5** | 100.0 | 1620.8 | 4.3 | 5.4 |
| 25 | 3 | 3 | H | 100.0 | **896.5** | 98.7$^{(5)}$ | 7200.0 | 4.3 | 18.2 | 100.0 | **1121.3** | 99.4$^{(3)}$ | 5350.0 | 5.0 | 22.0 |
| 30 | 3 | 3 | H | 100.0 | **1565.4** | 98.4$^{(4)}$ | 6042.0 | 3.4 | 25.9 | 100.0 | **3052.5** | 98.6$^{(4)}$ | 6826.9 | 6.5 | 28.2 |
| 30 | 3 | 4 | H | **98.0**$^{(3)}$ | 5967.8 | 97.4$^{(5)}$ | 7200.0 | 2.7 | 29.0 | 96.7$^{(4)}$ | 6182.1 | **97.9**$^{(5)}$ | 7200.0 | 4.5 | 34.8 |
| 35 | 3 | 3 | H | 100.0 | **2624.5** | 98.1$^{(5)}$ | 7200.0 | 4.4 | 35.3 | **99.0**$^{(3)}$ | 5498.3 | 98.6$^{(5)}$ | 7200.0 | 5.3 | 41.1 |
| 35 | 3 | 4 | H | 96.7$^{(5)}$ | 7200.0 | **97.0**$^{(5)}$ | 7200.0 | 2.6 | 36.1 | 94.4$^{(5)}$ | 7200.0 | **96.0**$^{(5)}$ | 7200.0 | 3.8 | 50.7 |
| 40 | 3 | 3 | H | **99.0**$^{(3)}$ | 5400.6 | 98.2$^{(5)}$ | 7200.0 | 4.5 | 46.0 | **99.2**$^{(3)}$ | 6700.1 | 98.7$^{(4)}$ | 6567.8 | 5.5 | 48.9 |
| 40 | 3 | 4 | H | 95.0$^{(5)}$ | 7200.0 | **96.6**$^{(5)}$ | 7200.0 | 0.8 | 51.5 | 93.7$^{(5)}$ | 7200.0 | **95.8**$^{(5)}$ | 7200.0 | 2.8 | 61.5 |
| 5 | 6 | 2 | H | 100.0 | **3.2** | 100.0 | 15.6 | 2.2 | 5.7 | 100.0 | **1.4** | 100.0 | 4.4 | 2.5 | 5.7 |
| 5 | 6 | 3 | H | 100.0 | **81.1** | 100.0 | 182.0 | 1.8 | 6.6 | 100.0 | **0.2** | 100.0 | 4.0 | 1.4 | 7.5 |
| 10 | 6 | 2 | H | 100.0 | **113.5** | 99.1$^{(2)}$ | 3013.5 | 3.3 | 9.8 | 100.0 | **272.4** | 99.8$^{(1)}$ | 2428.4 | 6.1 | 9.1 |
| 10 | 6 | 3 | H | **97.9**$^{(2)}$ | 4669.7 | 97.7$^{(3)}$ | 5668.3 | 2.5 | 11.4 | **99.3**$^{(1)}$ | 3561.8 | 97.9$^{(3)}$ | 5298.3 | 6.1 | 11.6 |
| 15 | 6 | 2 | H | 100.0 | **717.2** | 98.3$^{(4)}$ | 7200.0 | 1.8 | 17.6 | 100.0 | **763.1** | 98.0$^{(5)}$ | 7200.0 | 5.8 | 15.7 |
| 15 | 6 | 3 | H | **98.1**$^{(5)}$ | 7200.0 | 96.2$^{(5)}$ | 7200.0 | 2.8 | 17.3 | **96.4**$^{(5)}$ | 7200.0 | 95.6$^{(5)}$ | 7200.0 | 4.1 | 19.7 |
| Optimal | | | | 100.0 | **480.7** | 99.0 | 2507.1 | 3.5 | 12.0 | 100.0 | **392.6** | 99.5 | 1935.4 | 4.7 | 11.0 |
| Not Optimal | | | | **95.5** | 6334.9 | 95.1 | 6979.3 | 3.7 | 31.8 | 94.8 | 6269.5 | **95.5** | 6893.9 | 6.1 | 38.6 |
| Total | | | | **98.8** | 2077.3 | 97.9 | 3726.8 | 3.6 | 17.4 | **98.3** | 2262.6 | **98.3** | 3513.1 | 5.1 | 19.8 |

$^{(-)}$ indicates the number of instances (out of 5) that were not solved to optimality

The results in Tables 3.VI and 3.VII clearly indicate the performance of the two formulation schemes. The vehicle index formulations $F(ML)|k$ and $F(OU)|k$ outperformed the non-vehicle index formulations in finding the optimal solution even though the former requires many more variables. A possible reason is that the GFSECs, which are fractional, must be used for the non-vehicle index formulations, whereas the SECs, in which the right hand sides are integer, are used for the vehicle index formulations. The non-vehicle index formulations $F(ML)|nk$ and $F(OU)|nk$, however, could generally provide better lower bounds on the instances that were not solved to optimality within two hours except for the MVIRP-ML instances. The average %LB of the non-vehicle index formulations after two hours is slightly better than for the vehicle index formulations on the MVPRP instances, while the lower bounds produced by the vehicle index formulations instances are better on the MVIRP-ML and as good as the non-vehicle index formulation on the MVIRP-OU. The computing times on the MVIRP instances increase in a greater ratio when the number of periods increases compared to the MVPRP instances. The largest instance sizes that can be solved to optimality are $35c/3p/3v$ for MVPRP-ML and MVIRP-ML, $30c/3p/3v$ for MVIRP-OU and $25c/3p/2v$ for the MVPRP-OU.

The Op-ALNS could generally provide high quality solutions on the MVPRP instances, where the deviations from the optimal solutions or best upper bounds are 1.2% and 0.8% for the MVPRP-ML and MVPRP-OU, respectively. The Op-ALNS is not competitive for the smallest instances but it could provide a solution for most instances within one minute. The results on the MVIRP are not as good as for the MVPRP, but the Op-ALNS could still provide good quality solutions within a few seconds. Since the Op-ALNS was originally developed for the PRP, where it has to take the production and setup costs into account, the operators mainly attempt to find a solution where production and distribution costs are minimized. In the IRP, however, because a known production quantity is made available in each period, the algorithm has to be modified to put more emphasis on the inventory and distribution part. We also found that by expanding the neighborhood (i.e., the number of customer-period combinations) by 15-30%, the average solution quality can be further improved by

approximately 1-2% but with double the average computing time. However, the current setting is preferable since it has been shown in Chapter 4 that the Op-ALNS procedure could handle small to very large PRP instances effectively. It is worth noting that the Op-ALNS is much faster than the MVIRP heuristic presented by Coelho et al. (2012a), which had an average computing time of 2,000 seconds for the instances with three periods and of 8,000 seconds for the instances with six periods. We also remark that, on the instances with $n \leq 15$ that were solved to optimality, setting the initial upper bounds in CPLEX leads to an average improvement of 11.8% and 6.2% in computing time on the MVPRP and MVIRP instances, respectively.

### 3.6.4 Performance of the Branch-and-Cut Algorithm on Multi-Core Processors

Nowadays, most computers have multiple core processors and modern solvers like CPLEX have the capability to perform parallel optimization. In this section, we report the results of experiments performed on an eight-core processor. We first show the comparison of the average results by using single and multiple core processors on the instances that were solved to optimality by the single core machine in the previous section in Table 3.VIII. In the results for the eight-core processor, columns *CPU* and *WC* show the average total aggregate CPU time of all cores and average wallclock time, respectively. Because we did not run other tasks when running the algorithm, the wallclock time is a good approximation of the maximum CPU time spent on all cores. We also calculate the ratio of the aggregate CPU time and wallclock time (CPU/WC) in column *C/W* to show the benefit of parallel computing on multi-core processors. Since the main purpose of the experiments is to explore the largest instances that could be solved to optimality using multi-core processors, the formulations with a vehicle index are used as they are better at finding optimal solutions. The maximum computing time was set to 12 hours of wallclock time.

Table 3.VIII: Average Results Using Single and Multi-Core Processors on MVPRP and MVIRP Instances Solved to Optimality

| Problem | 1 Core | | 8 Cores | | | | %Change | |
|---|---|---|---|---|---|---|---|---|
| | CPU | Nodes | CPU | WC | C/W | Nodes | CPU | Nodes |
| MVPRP-ML | 371.5 | 1078 | **353.2** | 216.1 | 1.5 | 1171 | -4.9 | 8.6 |
| MVPRP-OU | **397.5** | 2159 | 488.8 | 324.4 | 1.3 | 2564 | 23.0 | 18.8 |
| MVIRP-ML | **480.7** | 3795 | 548.1 | 302.9 | 1.6 | 3924 | 14.0 | 3.4 |
| MVIRP-OU | **392.6** | 1476 | 446.4 | 212.9 | 1.7 | 1714 | 13.7 | 16.1 |

We can see that using multiple cores could reduce the elapsed time spent to solve the problems, but the aggregate CPU times and number of nodes generally increase compared to the results on the single core machine. The detailed results of using multi-core processors on all instances can be found in the Appendix. By using parallel computing, the algorithm could solve the instances up to $50c/3p/3v$ for the MVPRP-ML, $35c/6p/3v$ for the MVPRP-OU, and $45c/3p/3v$ for the MVIRP-ML and MVIRP-OU, and the average computing time was 2.1, 0.8, 4.5 and 5.6 hours, respectively. The algorithm, however, still could not solve the MVPRP-OU instances with 3 periods larger than $25c/3p/2v$. These instances appear to be more difficult to solve compared to the instances with 6 periods because they have lower initial inventory levels at customers as described in the Appendix. Therefore, larger quantities must be delivered to customers during initial periods in the planning horizon to satisfy the OU policy. The multi-core processors have more impact on the MVIRP instances where the average CPU/WC ratio is 2.2 compared to 1.6 for the MVPRP, and the maximum CPU/WC ratio of 4.1 could be achieved. Table 3.IX shows the summary of the largest instances that were solved to optimality by using single and multiple core processors. We also report the largest instance sizes for which the algorithm could obtain a solution within 2% of optimality for the results produced by a single core machine. The letter in parentheses indicates whether the solution was produced by the vehicle index formulations ($k$) or the non-vehicle index formulations ($nk$) (only for the single core as only the vehicle index formulations were solved on multiple cores). Note that for the instances solved to optimality, we report the instance size where the smaller instances were all solved to optimality, e.g., the instance size $20c/9p/2v$ on the MVPRP-ML solved by the single core processor is not considered as the largest since the instance size $15c/9p/2v$ is not solved to optimality.

Table 3.IX: Summary of the Largest Instances Solved to Optimality Using Single and Multiple Core Processors

| **Problem** | **Maximum Level (ML)** | | | **Order-Up-To Level (OU)** | | |
|---|---|---|---|---|---|---|
| | **1 Core** ($2h$) | | **8 Cores** ($12h$) | **1 Core** ($2h$) | | **8 Cores** ($12h$) |
| | Optimal | $\leq 2\%$ | Optimal | Optimal | $\leq 2\%$ | Optimal |
| MVIRP | $35c/3p/3v$ $(k)$ | -same- | $45c/3p/3v$ | $30c/3p/3v$ $(k)$ | -same- | $45c/3p/3v$ |
| | $15c/6p/2v$ $(k)$ | -same- | $25c/6p/2v$ | $15c/6p/2v$ $(k)$ | -same- | $15c/6p/3v$ |
| MVPRP | $35c/3p/3v$ $(k)$ | $50c/3p/4v$ $(nk)$ | $50c/3p/3v$ | $25c/3p/2v$ $(k,nk)$ | $30c/3p/3v$ $(nk)$ | $25c/3p/2v$ |
| | $25c/6p/2v$ $(k)$ | $30c/6p/4v$ $(nk)$ | $30c/6p/3v$ | $30c/6p/3v$ $(k,nk)$ | $30c/6p/4v$ $(k,nk)$ | $35c/6p/3v$ |
| | $10c/9p/3v$ $(k,nk)$ | $20c/9p/3v$ $(k,nk)$ | $20c/9p/2v$ | $20c/9p/2v$ $(k)$ | $20c/9p/3v$ $(k,nk)$ | $25c/9p/2v$ |

### 3.6.5 Results on Single-Vehicle Instances

We also tested the performance of the algorithms using the two formulation schemes on the single-vehicle instances and the results are presented in the Appendix. We observe that the single vehicle case is much easier to solve compared to the multiple vehicle case. Most instances were solved to optimality within a few seconds and the algorithms spent less than two minutes on average. We also performed the experiments on the PRP instances with $14c/6p/1v$ and uncapacitated production used in Archetti et al. (2011). All instances were now solved to optimality in a few seconds.

### 3.6.6 Results When Allowing Multiple Visits

In addition to these tests, we performed experiments on the cases where 2 and 3 visits per customer per period are allowed. The results are shown in the Appendix A.2.4. By allowing up to 2 and 3 visits per customer per period, the total costs on all the MVPRP instances remain the same compared to the case where a single visit is allowed. There are very few solutions where multiple visits are used (about 0.25% on average) and their total cost is equal to the case of a single visit because the problem has multiple optimal solutions. These multiple optimal solutions are found in the instances with zero inventory holding costs at customers because some deliveries can be made in advance and stored at customers without additional costs. In these MVPRP instances, allowing multiple visits is not beneficial because it generally incurs a higher routing cost. However, one can possibly take advantage of this feature in the case where customers have demands higher than the vehicle capacity. In this case, more than one vehicle can be used to deliver sufficient quantities to these customers instead of using carry-over inventories to satisfy the demands. For the MVIRP, the

total costs slightly decreased (0.6% on average) when allowing a maximum of 2 or 3 visits while the number of multiple visits is approximately 5-6% of the total number of visits. We could obtain slight improvements in the total costs in the MVIRP because the production quantities at the plant, which are fixed, can be delivered to some customers with lower inventory holding costs. Note that the average CPU time also increased by approximately 26% and 38% when 2 or 3 visits are allowed, respectively.

## 3.7 Conclusion

We have studied the multi-vehicle variant of the production routing problem (PRP) and the inventory routing problem (IRP). Two strong formulations, one with a vehicle index and the other one without a vehicle index, were introduced. We proposed several valid inequalities including symmetry breaking constraints to strengthen the formulations and developed branch-and-cut approaches to solve the problems. We also adapted the previously developed Op-ALNS heuristic for the PRP-ML to the PRP-OU and IRP (both ML and OU) to determine the initial upper bounds for our branch-and-cut algorithms. The Op-ALNS could generally provide high quality solutions, especially for the PRP instances, within a few seconds. The results show that the vehicle index formulations are superior in finding optimal solutions, while the non-vehicle index formulations could generally provide better lower bounds on larger instances that were not solved to optimality within two hours. Finally, we could solve the instances with up to 35 customers, 3 periods and 3 vehicles to optimality for the IRP and PRP with the ML policy, and 30 (resp. 25) customers, 3 periods and 3 vehicles for the IRP (resp. PRP) with the OU policy within two hours. By using multi-core processors, we could further solve the instances with up to 45 and 50 customers, 3 periods and 3 vehicles for the IRP (both ML and OU) and PRP with the ML policy, respectively, and the instances with 35 customers, 6 periods and 3 vehicles are also solved for the PRP with the OU policy. We have thus filled important gaps in the literature concerning the exact solution of the IRP and PRP with multiple vehicles.

# Chapter 4
# Heuristic Algorithms for Production Routing Problems

In this chapter, we present heuristics to compute solutions and upper bounds which are also used in the branch-and-cut algorithms of the previous chapter. The heuristic for the PRP-ML presented in Sections 4.1-4.3 and the computational results in Section 4.5 are part of the article 1 below, and the adaptations of the heuristic to the PRP-OU and the IRP (both ML and OU) in Section 4.4 are part of the article 2.

1. Adulyasak, Y., Cordeau, J.-F., Jans, R. Optimization-Based Adaptive Large Neighborhood Search for the Production Routing Problem. *Transportation Science* (Article in Advance), 2012.

2. Adulyasak, Y., Cordeau, J.-F., Jans, R. Formulations and Branch-and-Cut Algorithms for Multi-Vehicle Production and Inventory Routing Problems. GERAD Tech Rep. G-2012-14. 40 pages. Submitted to *INFORMS Journal on Computing* in April 2012 (Revision submitted in October 2012).

## 4.1 Introduction

In this chapter, we introduce a novel heuristic for the PRP. This heuristic is based on the *adaptive large neighborhood search* (ALNS) framework first proposed by Ropke and Pisinger (2006) for vehicle routing problems. ALNS itself extends the ideas of the large neighborhood search introduced by Shaw (1997). ALNS repeatedly destroys a part of the current solution and reconstructs it in the hope of achieving an improvement. To this end, several heuristics are used to search the solution neighborhood and they are randomly selected based on past success. ALNS has been very successful in vehicle routing (see Ropke and Pisinger (2006), Pisinger and Ropke (2007), Azi et al.

(2010), Laporte et al. (2010), Ribeiro and Laporte (2012)). However, applying ALNS in the context of the PRP is more difficult because the problem involves quantity decisions as well as complicating constraints. To overcome this difficulty, the binary variables indicating the customers to visit and the vehicle routes are handled by the operators of the ALNS and a network flow model is used to set the corresponding optimal continuous variables. Similar ideas were recently put forward by Coelho et al. (2012c) to address the IRP with transshipment. In our approach, we use an enumeration scheme adapted from local branching techniques (Fischetti and Lodi, 2003) to generate initial solutions with different setup configurations. These solutions are further explored in the ALNS part of the heuristic. Because of the complexity of the problem, we introduce new types of operators, called *selection* and *transformation*, that are specifically adapted to the multi-period structure of the PRP, in which the number of deliveries to each customer is part of the decisions. We tested this new heuristic on two different sets of benchmark instances. The results show that the proposed heuristic is very efficient and generally outperforms all other heuristics for the PRP reported in the literature.

The rest of this chapter is organized as follows. Section 4.2 presents a formulation of the PRP-ML that is used in the heuristic procedure. The ALNS heuristic is then described in detail in Section 4.3. This is followed by the adaptations of the heuristic to the PRP-OU and the IRP (both ML and OU) in Section 4.4 and the computational results in Section 4.5.

## 4.2 Mathematical Formulation

In this section, we first introduce the mathematical formulation that is used throughout this chapter. Since the formulation used in the heuristic algorithm is different from the previous section and the shipment quantity variable with and without vehicle index are required, we introduce the variable $r_{it}$ to represent the shipment quantity to customer $i$ in period $t$ (instead of the variable $q_{it}$) and retain variable $q_{ikt}$. We further define $M_t = \min\{C, \sum_{j=t}^{l} \sum_{i \in N_c} d_{ij}\}$ and $\widetilde{M}_{it} = \min\{Q, L_i, \sum_{j=t}^{l} d_{ij}\}$. The PRP is defined on a complete graph $G = (N, A)$ where $A = \{(i, j) : i, j \in N, i \neq j\}$

and we assume throughout that $c_{ij} = c_{ji}, \forall (i,j) \in A$. The remaining notation follow the previous chapter. The formulation used in this section is as follows.

$$\min \sum_{t \in T} \left( up_t + fy_t + \sum_{i \in N} h_i I_{it} + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} x_{ijkt} \right) \tag{4.1}$$

s.t.

$$I_{0,t-1} + p_t = \sum_{i \in N_c} r_{it} + I_{0t} \quad \forall t \in T \tag{4.2}$$

$$I_{i,t-1} + r_{it} = d_{it} + I_{it} \quad \forall i \in N_c, \forall t \in T \tag{4.3}$$

$$p_t \leq M_t y_t \quad \forall t \in T \tag{4.4}$$

$$I_{0t} \leq L_0 \quad \forall t \in T \tag{4.5}$$

$$I_{i,t-1} + r_{it} \leq L_i \quad \forall i \in N_c, \forall t \in T \tag{4.6}$$

$$r_{it} = \sum_{k \in K} q_{ikt} \quad \forall i \in N_c, \forall t \in T \tag{4.7}$$

$$q_{ikt} \leq \widetilde{M}_{it} \sum_{j \in N} x_{ijkt} \quad \forall k \in K, \forall i \in N_c, \forall t \in T \tag{4.8}$$

$$\sum_{i \in N_c} q_{ikt} \leq Q \quad \forall k \in K, \forall t \in T \tag{4.9}$$

$$\sum_{k \in K} \sum_{j \in N} x_{ijkt} = z_{it} \quad \forall i \in N_c, \forall t \in T \tag{4.10}$$

$$\sum_{j \in N} x_{jikt} = \sum_{j \in N} x_{ijkt} \quad \forall k \in K, \forall i \in N_c, \forall t \in T \tag{4.11}$$

$$\sum_{j \in N_c} x_{0jkt} \leq 1 \quad \forall k \in K, \forall t \in T \tag{4.12}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ijkt} \leq |S| - 1 \quad \forall S \subseteq N_c : |S| \geq 2, \forall k \in K, \forall t \in T \tag{4.13}$$

$$p_t, I_{it}, r_{it}, q_{ikt} \geq 0 \quad \forall i \in N, \forall k \in K, \forall t \in T \tag{4.14}$$

$$y_t, z_{it}, x_{ijkt} \in \{0,1\} \quad \forall i,j \in N, \forall k \in K, \forall t \in T. \tag{4.15}$$

The objective function (4.1) minimizes the total production, setup, inventory and routing costs. Constraints (4.2) and (4.3) are the inventory flow balance at the plant

and at the customers, respectively. Constraints (4.4) are the setup forcing and production capacity constraints: they force the setup variable to be one if production takes place and limit the production quantity to the minimum of the production capacity and the total demand in the remaining periods. Constraints (4.5) and (4.6) impose inventory limits at the plant and customer level, respectively. Constraints (4.7) ensure that the total quantity shipped to each customer in each period is equal to total delivery quantity to the customer from all vehicles. Constraints (4.8) allow a positive delivery quantity from vehicle $k$ to node $i$ in period $t$ only if this node is visited by the vehicle in period $t$. The total quantity loaded in each truck can be at most the truck capacity as specified by (4.9). Constraints (4.10) enforce $z_{it} = 1$ if there is a truck leaving from customer $i$. These constraints together with constraints (4.8) prevent split deliveries. Constraints (4.11) are the vehicle flow conservation. Each vehicle can leave from the depot at most once per period (4.12). Constraints (4.13) eliminate subtours for each vehicle.

This basic PRP formulation contains a large number of binary and continuous variables and cannot be solved efficiently using general optimization software. In preliminary testing, we observed that the problem with just 10 customers, 10 time periods and one vehicle (although with different subtour elimination constraints similar to the Miller-Tucker-Zemlin constraints (Miller et al., 1960)) has more than 1,000 binary variables and takes more than an hour to solve with CPLEX. This formulation is thus intractable for large size instances.

## 4.3 Adaptive Large Neighborhood Search (ALNS) Heuristic

We first describe the structure of the heuristic for the PRP-ML in this section and explain how this heuristic is adapted for the PRP-OU and the IRP (for both the ML and OU) in Section 4.4.

### 4.3.1 General Structure of the Heuristic

We propose a decomposition-based heuristic to solve the problem. The basic idea is to reduce the complexity of the problem by decomposing it into several subproblems which are easier to solve. The setup variables at the plant are handled in the initialization phase, where we construct a pool of several solutions with different setup schedules. Those solutions are then improved in the next phase by applying ALNS. The binary $z_{it}$ and $x_{ijkt}$ variables are handled by selection and transformation operators, while the optimal value of the remaining continuous variables can be determined using a minimum cost network flow algorithm. Since a decomposed part of the problem, i.e., the continuous variables, is handled by an exact optimization algorithm and the rest of the problem is handled by the ALNS operators, we refer to our algorithm as an optimization-based adaptive large neighborhood search (Op-ALNS). An overview of the heuristic procedure is provided in Algorithm 4.1.

In the **initialization phase**, we generate a small set of initial solutions, represented by $s^i$, with different production setup configurations. The purpose of generating several initial solutions is to explore different parts of the search space and thus to avoid being trapped in local optima. More importantly, it is also difficult and time consuming to change the production setups during the improvement process since it generally affects the solution to a great extent, i.e., when a production setup decision is changed, the delivery and routing decisions associated with that setup have to be adjusted, and thus affect the other decisions. Additionally, since the production setups generally incur large fixed costs, they form an important part of the objective function value. To generate initial solutions in this step, we sequentially solve the decomposed problems of the PRP, namely the production-distribution and the routing problems, to determine a preliminary production and distribution plan. The solution is stored and the algorithm starts constructing another initial solution with a different production setup configuration using the *setup move* procedure which employs an enumeration scheme adapted from the local branching technique (Fischetti and Lodi, 2003). This process is repeated until the maximum number of initial solutions $n_s^{max}$ is reached.

**Algorithm 4.1** Op-ALNS

$s^{best} \leftarrow \phi$ and $S \leftarrow \phi$
**1. Initialization Phase:**
**for** $i = 1 \rightarrow n_s^{max}$ **do**
    Generate a new initial solution $s^i$ and add it to the initial solution set $S$
    Apply *setup move* procedure
**end for**

**2. Improvement Phase:**
**2A.** Restricted false start
$\psi_{ALNS} \leftarrow 10$
**for all** solution $s^i \in S$ **do**
    $s^i$, *candidates*, *iterations* $\leftarrow$ *ALNS*$(s^i, \psi_{ALNS})$
    **if** $f(s^i) < f(s^{best})$ **then**
        $s^{best} \leftarrow s^i$
    **end if**
**end for**
Sort all solutions in $S$ in descending order of total cost
**2B.** Full restart
*total iterations* $\leftarrow 0$ and *total node candidates* $\leftarrow 0$
$\psi_{ALNS} \leftarrow \infty$
**while** *total iterations* $\leq \psi_{max}$ **and** *total node candidates* $\leq n_q^{max}$ **do**
    $s^i$, *candidates*, *iterations* $\leftarrow$ *ALNS*$(s^i, \psi_{ALNS})$
    **if** $f(s^i) < f(s^{best})$ **then**
        $s^{best} \leftarrow s^i$
    **end if**
    *total iterations* $\leftarrow$ *total iterations* + *iterations*
    *total node candidates* $\leftarrow$ *total node candidates* + *candidates*
    $i \leftarrow i + 1$
**end while**

In the **improvement phase**, the algorithm tries to improve the initial solutions from the previous phase using ALNS and network flows. This process is represented by *ALNS*$(s^i, \psi_{ALNS})$ where $\psi_{ALNS}$ is the maximum number of iterations in the process and $s^i$ is the initial solution which is currently explored. The details of this process are provided in Algorithm 4.2 in Section 4.3.3. At each iteration, a selection operator

is applied to select some customer-period combinations and a transformation operator is used to modify the solution by removing and reinserting some or all of these combinations. Each customer-period combination is referred to as a node candidate. In the transformation procedure, once a move is performed, all the binary variables are fixed and the resulting minimum cost flow problem is solved to determine the optimal flow cost. This process returns the improved solution ($s^i$), the number of iterations used in the process (*iterations*), and the number of combinations that are examined in the process (*candidates*).

The improvement phase consists of a restricted false start and a full restart. Since many initial solutions obtained by the initialization process might be infeasible or their total cost may be very high, all the initial solutions are improved in the restricted false start (process 2A) where a limited number of iterations are performed on each of the generated initial solutions. The improved initial solutions are then sorted in descending order of total cost at the end of this process. Next, some improved initial solutions will be examined in the full restart process (process 2B). The process starts by selecting an initial solution from the top of the initial solution pool. The $ALNS(s^i, \psi_{ALNS})$ continues until the stopping condition is reached. The best solution is stored in $s^{best}$ and the total iterations and total node candidates are updated. Subsequently, the next initial solution is selected to be examined. The heuristic procedure terminates when it reaches the maximum number of iterations $\psi_{max}$ or the maximum number of explored node candidates $n_q^{max}$.

### 4.3.2 Initialization Phase

Initial solutions are generated by sequentially solving two decomposed problems, namely the production-distribution (PD) and the routing (R) subproblems. The first subproblem is used to determine a production, inventory and distribution plan, and the second subproblem determines the routes to serve the customers according to the distribution plan obtained from the first subproblem.

#### 4.3.2.1 Production-Distribution (PD) Subproblem

The PD subproblem contains the lot-sizing, inventory and customer shipment decisions. To formulate the problem, we define $\sigma_i$ as the approximate cost of visiting customer $i$, which we calculate as

$$\sigma_i = \min\{2c_{0i}, \min_{j,k \in N, j \neq k}(c_{ij} + c_{ik})\}.$$

It is the minimum value between the cost of making a round trip from the production facility and the cost to the nearest two neighbors of customer $i$. We solve the PD problem with this cost in an attempt to approximately take transportation costs into account, without explicitly modeling the routing decisions. A similar approach was used by Bard and Nananukul (2009a) to generate initial solutions for their tabu search algorithm. The production-distribution problem can be formulated as follows:

$$\min \sum_{t \in T} \left( up_t + fy_t + \sum_{i \in N} h_i I_{it} + \sum_{i \in N_c} \sigma_i z_{it} \right) \tag{4.16}$$

s.t., (4.2)-(4.6),(4.14)-(4.15), and

$$r_{it} \leq \widetilde{M}_{it} z_{it} \quad \forall i \in N_c, \forall t \in T \tag{4.17}$$

$$\sum_{i \in N_c} r_{it} \leq mQ \qquad \forall t \in T. \tag{4.18}$$

The objective function (4.16) minimizes the total production setup, unit production, inventory and approximate transportation costs. Constraints (4.17) allow a positive delivery quantity to node $i$ only if this node is visited. Constraints (4.18) ensure that the total quantity shipped in each time period does not exceed the total capacity of all vehicles.

Although this formulation has a less complicated structure than the PRP formulation because all routing constraints are removed, it is still difficult to solve because of its size and the number of binary variables it contains. However, it is not necessary to solve the problem to optimality because the inventory, distribution and routing

decisions will be largely modified in the improvement phase. We thus employ a *fix-and-optimize* scheme to solve the problem in two steps. In the first step, we fix all customer visit variables ($z_{it}$) to one and solve the model to obtain the production setup decisions ($y_t$). In the second step, all the production setup decisions are fixed according to the solution obtained in the first step and we solve the resulting model to determine the remaining variables, i.e., production quantity ($p_t$), inventory quantity ($I_{it}$), shipment quantity ($r_{it}$) and shipment visits ($z_{it}$). To avoid long computing times, the solver is stopped when a solution within 1% of optimality is found, or the number of nodes explored during the branch-and-bound process exceeds 5000. If no feasible solution is found, the solver continues until a feasible solution is obtained. In our experiments, a feasible solution could usually be obtained within a few seconds for most instances and within a few minutes for the largest instances with 200 customers and 20 periods. Note that these settings do not significantly affect the solution quality and overall performance of the algorithm.

#### 4.3.2.2 Routing (R) Subproblem

The solution from the PD subproblem can be interpreted as a set of delivery demands to the customers in each period. In the subsequent step, we generate a routing plan for the planned deliveries. These deliveries are made with a fleet of capacitated vehicles and each vehicle can visit more than one customer. Since the demands in each period are independent of each other, vehicle routes can be constructed independently for each period. The routing solution obtained in this step might not be feasible because constraints (4.18) do not guarantee that the total quantity to be delivered in a given day can fit in the available vehicles when split deliveries are not allowed. Hence, the number of vehicles might be exceeded. We allow this type of infeasibility by using a dummy vehicle with unlimited capacity to carry those unsatisfied demands, with a unit penalty cost $\theta = 10 \max_{i \in N}(h_i)$. The unit penalty is used because a larger delivery quantity is more difficult to relocate. The penalty is ten times the maximum inventory cost to ensure that it is large enough for the algorithm to finish with a feasible solution as the total cost can be reduced by moving the exceeded delivery quantity to another period (which can incur smaller inventory costs than the

penalty), while being low enough to allow some infeasible solutions during the search. The decision variable $\delta_{it}$ is equal to one if customer $i$ is served by the dummy vehicle in period $t$.

Before solving this subproblem, we fix $r_{it}$ and $z_{it}$ according to the result from the solved PD subproblem. The routing subproblem in each period $t$ can be formulated as a capacitated vehicle routing problem (CVRP):

$$\min \sum_{i \in N} \left( \sum_{j \in N} c_{ij} \sum_{k \in K} x_{ijkt} + \theta \bar{r}_{it} \delta_{it} \right), \tag{4.19}$$

subject to the routing constraints of the problem in each time period $t$ corresponding to (4.8)-(4.15) and

$$\sum_{k \in K} q_{ikt} + \bar{r}_{it} \delta_{it} = \bar{r}_{it} \quad \forall i \in N_c. \tag{4.20}$$

Constraints (4.20) are equivalent to (4.7) but allow the dummy vehicle to serve customers. Since the CVRP itself is a difficult problem, we use the Clarke and Wright (1964) savings heuristic to quickly obtain a solution. If the number of routes exceeds the number of available vehicles, the routes are sorted in decreasing order of total delivery quantity and the available vehicles are assigned to the routes from the top of the list. The customers that belong to the unassigned routes are then assigned to the dummy vehicle.

#### 4.3.2.3 Setup Move Procedure

Once an initial solution has been generated, it is stored in the initial solution pool and another initial solution with different production setups will be generated. To this end, we iteratively add inequalities inspired from local branching (Fischetti and Lodi, 2003) that force the model to produce a different solution to the PD subproblem. Let $s$ be the solution index, where $s = 1$ designates the first initial solution without any local branching inequality, and $\overline{y}_t^s$ the value of the production setup variable $y_t$ in solution $s$. The following local branching inequalities are added to the PD subproblem

to generate solution $\bar{s}$:

$$\sum_{y_t|\overline{y}_t^s=1} (1 - y_t) + \sum_{y_t|\overline{y}_t^s=0} y_t \geq 1 \qquad s = 1, ..., \bar{s} - 1. \tag{4.21}$$

This inequality forces at least *one* of the production setup variables in each previous solution to be changed. In our implementation, the maximum number of initial solutions in the pool is set to 10 to avoid excessive computing times.

### 4.3.3 Improvement Phase

The improvement procedure plays an important role to refine the initial solutions generated in the initialization step. Since the PRP contains both binary and continuous variables, developing a heuristic that handles both types of variables is difficult. For example, when a solution is modified by removing a customer from a route and inserting it in a different period, one has to identify the new delivery quantity for the customer, which may also affect the production, inventory, and other delivery quantity decisions. In addition, unlike in the VRP where the removed nodes have to be reinserted to obtain a feasible solution, it is not always necessary in the PRP to reinsert the removed nodes because the demands can be satisfied from available inventory. Furthermore, the removed nodes can be inserted in multiple periods. To address these issues, we use several move operators to handle the binary $z_{it}$ and $x_{ijkt}$ variables, and the remaining continuous variables are set by solving a network flow model.

Unlike the original ALNS framework which employs *destroy* and *repair* schemes, we developed two different sets of operators to solve the PRP. The first set of operators, called *selection*, is used to create an ordered list of node candidates. A node candidate is defined as a combination of a customer $i$ and time period $t$. These candidates are not removed from the solution in this step. Instead, an operator from the second set, called *transformation*, is applied to examine the node candidates according to the operator rules. The transformation operator will remove and reinsert one or many node candidates in one or many time periods. The operator is repeated until all node candidates in the list have been examined.

During the transformation process, the operator performs two main steps to handle the two different parts of the problem separately. In the first step, the binary decisions $(z_{it}, x_{ijkt})$ are modified (by removing and reinserting nodes) according to the operator rules. Note that the production setup variables $y_t$ remain fixed according to the initial solution which is currently considered. Then, those binary variables are fixed and the continuous variables $(p_t, I_{it}, r_{it}, q_{ikt})$ are adjusted by solving the minimum cost flow problem (MCF). Since the transformation operator is repeated until all node candidates in the list are examined, many new solutions can be generated at each iteration and the solutions are accepted according to a simulated annealing criterion.

#### 4.3.3.1 ALNS Framework

Our ALNS heuristic employs several simple heuristics to select the node candidates and transform the solutions. The different operators are probabilistically selected according to weights calculated based on the success of each operator in terms of improving solutions. We now describe the main elements of our ALNS implementation by following the framework presented by Ropke and Pisinger (2006).

1. *Large neighborhood:* We define a node candidate $v$ as a pair $(i, t)$ where $i$ is a customer and $t$ is a time period. Given the current solution $s$, we select $n_q$ node candidates to be put in the ordered list. One of the selection operators is chosen to generate the ordered list of node candidates, and one of the transformation operators is used to remove and reinsert some of these candidates. We use two neighborhood definitions, i.e., a small and an extended search space, to select the node candidates. The small search space randomly picks $n_q$ within the range $[1, \lambda_{ub}\kappa]$, where $0 \leq \lambda_{ub} \leq 1$ and $\kappa = \sum_{i \in N_c} \sum_{t \in T} z^s_{it}$, i.e., total number of visits in the current solution $s$. If the algorithm cannot find an improved feasible solution for $\omega$ iterations, we expand the range of the search space to $[\lambda'_{lb}\kappa, \lambda'_{ub}\kappa]$, where $0 \leq \lambda_{ub} \leq \lambda'_{lb} \leq \lambda'_{ub} \leq 1$. In our implementation, we use $\lambda_{ub} = \lambda'_{lb} = 0.1$ and $\lambda'_{ub} = 0.4$, which means the number of node candidates varies between 1 node and 10% of the total number of visits for the small search space, and between 10% and 40% of the total number of visits for the large search space. The large search space is used when an improved feasible solution is not found for $\omega = 25$ iterations.

2. *Adaptive search engine:* Like in many other ALNS implementations, the choice of the selection and transformation operators to be performed at each iteration is controlled by a roulette-wheel mechanism. Denote by $w_i$ the weight of operator $i$, $H_s$ and $H_t$, the set of selection and transformation operators, respectively. One selection and one transformation operator are selected independently at each iteration. An operator $i$ is selected with probability $p_i = w_i / \sum_{j \in H_k} w_j$, where $H_k = H_s$ for the selection operators and $H_k = H_t$ for the transformation operators.
3. *Adaptive weight adjustment:* The weight of an operator $i$ is calculated based on two factors: (i) the average percentage of calls that the operator $i$ can improve the current solution and (ii) the average time per iteration that the operator requires. All weights are set to 10 for the first 20 iterations. They are then recalculated every 5 iterations. After the first 10 iterations, we also check if there is any operator that is not used and select that operator to be performed to ensure that all operations are called before recalculating the weights. Denote by $\tau_i$ and $\eta_i$ the number of times that the operator $i$ is performed and is able to improve the current solution, respectively, and by $T_{i/\underline{i}}$ the relative processing time factor, calculated as the average time per iteration for operator $i$, divided by $\underline{i}$, the lowest average computing time among all operators in the same operator set. Let $\alpha$ be the weight adjustment parameter. The weights are calculated as follows:

$$w_i = \frac{10^{\mu}}{T_{i/\underline{i}}}, \text{ where } \mu = \left(1 + \alpha \frac{\eta_i}{\tau_i}\right).$$

   The weights are calculated based on the relative frequency with which the operator can improve the solution. The parameter $\alpha$ defines the importance given to the operators that can improve the solution, i.e., high $\alpha$ values encourage the algorithm to concentrate on the operators that can frequently improve the solution, thus promoting intensification but reducing diversification. The denominator $T_{i/\underline{i}}$ represents the time efficiency of the algorithm as some operators are more time consuming than others. We use $\alpha = 3$ in our study.
4. *Acceptance and stopping criteria:* After a new solution is obtained by solving the MCF, we apply a simulated annealing (SA) criterion to decide whether this

solution should be accepted. The algorithm always accepts an improved solution and it also accepts a worse solution with probability $\rho = \exp\left(\frac{-100}{T}\left(\frac{f(s')-f(s)}{f(s)}\right)\right)$, where $f(s)$ and $f(s')$ are the total cost of the current solution ($s$) and the new solution ($s'$), respectively. The temperature $T > 0$ is calculated by the formula $T = T_n = cT_{n-1}$ at iteration $n$. We initially set $T_0 = 0.3$ and $c = 0.995$. Note that several new solutions can be found during one transformation process. The current ALNS call stops when (i) the number of iterations for one initial solution exceeds the maximum number of iterations ($\psi_{ALNS}$) (this stopping criterion is only used in the restricted false start where $\psi_{ALNS} = 10$), (ii) the current solution has not improved for $\psi_A = 100$ iterations, or (iii) the algorithm cannot find a better solution than the best overall solution from all previous ALNS calls for $\psi_B = 300$ iterations. Then, the next initial solution from the pool is selected, the weights and temperature are reset, and the ALNS is called again to improve the selected solution. This process is repeated until a global termination condition is met, i.e., (i) it reaches the maximum number of iterations $\psi_{max} = 1000$ or (ii) the cumulative number of examined node candidates is larger than or equal to 15,000 candidates. The latter condition is used to prevent too long computing times for large instances.

The details of the ALNS process are provided in Algorithm 4.2.

#### 4.3.3.2 Details of the Selection and Transformation Operators

We use several simple heuristics to explore the neighborhood and ensure both intensification and diversification. In our ALNS heuristic, the selection operators are employed to create the list of node candidates and the transformation operators are used to transform the solution by examining the node candidates in the generated ordered list and produce new solutions. The details of the operators are provided below.

**Selection operators.** The selection operators select a number of node candidates from the current solution and put them into an ordered list. In addition, if the current solution has infeasible routes that require a dummy vehicle, we use the method presented in Section 4.3.2.2 to assign customers to the dummy vehicle. The customers

**Algorithm 4.2** *ALNS*$(s^i, \psi_{ALNS})$

$s \leftarrow s^i$
$s^{feas} \leftarrow \phi$
*iterations* $\leftarrow 0$
*candidates* $\leftarrow 0$
Initialize operator weights and temperature $T$
**while** *iterations* $\leq \psi_{ALNS}$ **and** *iterations* without improvement $\leq \psi_A$ **and** *iterations* without new best solution $\leq \psi_B$ **do**
    Select one *selection* and one *transformation* operator probabilistically
    Determine $n_q$, apply the selected *selection* to $s$ to generate the ordered candidate list and update $n_q$
    **while** the number of node candidates in the list $> 0$ **do**
        Select and transform the node candidate(s) from the list according to the *transformation* operator rule
        Fix the binary decisions, solve the MCF and apply the *trimming* process to obtain a new solution $s'$
        **if** $f(s')$ is feasible **and** $f(s') < f(s^{feas})$ **then**
            $s^{feas} \leftarrow s'$
        **end if**
        **if** $f(s') < f(s)$ **or** $f(s')$ is accepted by the SA criterion **then**
            $s \leftarrow s'$
        **end if**
        Remove the examined node candidates from the list according to the selected *transformation* operator
    **end while**
    *candidates* $\leftarrow$ *candidates* $+ n_q$
    $T \leftarrow cT$
    **if** *iterations* $\geq 20$ and *iterations* is a multiple of 5 **then**
        Update operator weights
    **end if**
    *iterations* $\leftarrow$ *iterations* $+ 1$
**end while**
**return** $s^{feas}$, *candidates*, *iterations*

that are served by the dummy vehicle are always added to the bottom of the list and the number of node candidates $n_q$ is updated.

We propose seven selection operators which can be separated into two different types. The first type contains the operators that only select the node candidates

while the second type are the operators that also slightly modify the solution before selecting the node candidates. The purpose of the second type of operators is to alter the solution before applying the transformation step, so as to provide further diversification. The details of the selection operators are as follows.

1. *Selection by ranking:* Five operators are used to select the $n_q$ node candidates where $n_q$ is randomly chosen as described Section 4.3.3.1. For each visited customer $j$ in period $t$, an index $s_{jt}$ is calculated. Each operator uses a different method to calculate $s_{jt}$. Those candidates are then ranked according to this index and those that come first are selected to be inserted in the node candidate list. The details of the operators are as follows.
   (a) *Random selection (S1):* There is no ranking index in this operator and the visited customers are randomly selected. The main purpose of this operator is to avoid that the algorithm repeatedly chooses similar node candidates during the search.
   (b) *Unit savings greedy selection (S2):* The visited customers are selected according to the relative unit travel cost savings by shortcutting the path. For a node $j$ with predecessor $i$ and successor $k$ in period $t$, the index $s_{jt}$ is calculated as follows.

$$s_{jt} = \frac{c_{ij} + c_{jk} - c_{ik}}{\max\{\epsilon, d_{jt} - I_{j,t-1}\}}, \text{ where } \epsilon = 0.001.$$

   The denominator represents the degree to which a delivery is needed to satisfy the demand of customer $j$ and the value cannot be less than $\epsilon$. Node candidates are ranked in descending order of this index. The purpose of this operator is to find potential node candidates that can be easily removed without incurring a stockout and can quickly improve the solution.
   (c) *Radius selection (S3)*: This operator is inspired by the *Shaw removal* scheme (Shaw, 1997, Ropke and Pisinger, 2006) and aims to remove nodes that are similar in some respect. In the first step, we select the visited customer $j'$ in period $t'$ that will reduce the routing cost the most when removed, i.e., $(j', t') = \text{argmax}_{j \in N_c, t \in T | z_{jt}=1}\{c_{ij} + c_{jk} - c_{ik}, \forall i, k \in N \backslash \{j'\}\}$, to be the reference

node and put it in the node candidate list. We calculate the index as the travel cost from the reference node $j'$ to each customer in period $t'$ including the customers that are *not* visited, i.e., $s_{it'} = c_{ij'}, \forall i \in N_c \backslash \{j'\}$ and rank them in ascending order. Note that the node candidates in other periods $t \neq t'$ are not considered in this operator. The purpose of this operator is to select the nodes that are close to the reference node $j'$ to be inserted together.

(d) *Maximum inventory utilization selection (S4)*: We rank the visited customers by the inventory capacity utilizations, i.e., $s_{jt} = I_{jt}/L_j$ in descending order. The purpose of this operator is to select the nodes that can reduce the tightness of the inventory constraints in the current solution.

(e) *Minimum delivery quantity selection (S5)*: The visited customers are ranked by the delivered quantities to each customer in each period, i.e., $s_{jt} = r_{jt}$ in ascending order. The purpose of this operator is to get the node candidates that can be removed without reinserting or can be reinserted easily.

2. *Selection with modification:* These two operators are used to slightly modify the solution and select some visited customers to be added to the ordered node candidate list. Note that the number of node candidates $n_q$ is not determined a priori in these operators and it is updated after the ordered list is constructed. These operators differ from the transformation operators because they are mainly used to alter some part of the solution to diversify the search, while the transformation operators will remove and reinsert the nodes according to the provided list only. The modified solutions produced by these selection operators, however, may be infeasible or have a much higher cost than the original solution.

(a) *Random route duplication (S6):* This operator randomly chooses a route in any period between $[1, l-1]$ and duplicates it into the next period. If some customers in the route are already visited in the next period, they will first be removed from their current route before duplication. Next, the algorithm will produce the list by selecting all the nodes that belong to the original route to be the node candidates and order them according to their current sequence in the selected route. The purpose of this operator is to increase the flexibility of the routing part before calling the transformation operator.

(b) *Random period rerouting (S7)*: One period is randomly chosen and the routes in that period are reconstructed. More precisely, we randomly draw one period from those that use the maximum number of vehicles, and reconstruct the routes by the savings algorithm. Then, all nodes that belong to the routes which have a vehicle capacity utilization less than 75% in that period are added to the node candidate list and ordered according to their current sequence in the selected route. This process can generate a new routing solution since the delivery quantities are frequently changed during the ALNS process and the savings algorithm will generate different routes. The purpose of this operator is to quickly modify the routing part of the solution and try to refine it again during the transformation process.

**Transformation operators.** We use seven transformation operators to examine the node candidates and generate new solutions by removing and reinserting a subset of the node candidates. The notation $[\alpha, \beta, \gamma]$ is used in order to describe these operators. The term $\alpha$ represents the number of *node candidates* that are inserted together at once, the term $\beta$ is the number of *time periods* in which the node candidates are inserted, and the term $\gamma$ is the maximum number of *times* that the selected transformation process has to be repeated to examine all candidates in the list. For example, $[1, 2, n_q]$ represents a transformation operator that inserts one node candidate into two time periods at the same time and the operator is performed $n_q$ times since only one candidate is examined at once.

The examined node candidates are inserted in the routes according to the cheapest insertion rule. After the insertion is done, all binary variables are fixed and the MCF is solved to calculate the flow cost. The transformation operators also use a process called *trimming* to remove the nodes that are assigned to a route but not visited (i.e., customer $i$ with $z_{it} = 1$, but $r_{it} = 0$) and the total cost is recalculated. The trimming process is called after the new solution is obtained by solving MCF.

We denote by $i_v$ and $t_v$ the customer and the period of the selected node candidate $v$, respectively. The transformation schemes can be classified into three main groups as follows.

1. *Single-node-multiple-periods transformation:* The operators in this type will remove and insert *one* node candidate at a time in different time periods. It starts by selecting a node candidate $v$ from the top of the list. Then a set of potential time period bundles for node $v$, denoted by $\Theta_v$ is constructed. Each member $T_v \in \Theta_v$ represents a bundle of time periods in which the node will be inserted. The operators will remove the node and reinsert it in each period of $T_v$. Once the node has been inserted, the total cost is computed by solving the MCF and calling the trimming process. The new solution is accepted according to the SA criterion. This step is repeated until all time period bundles in the set $\Theta_v$ have been considered. Then, the next node candidate $v+1$ is selected to be examined and the process is started again. We use $\phi$ to represent the null time period, i.e., the node is not reinserted in any time period. The details of the second step for each operator are as follows.
   (a) *One-node, one-period transformation (T1)* $[1, 1, 4n_q]$: It removes and inserts the selected candidate $v$ into each time period bundle in the set $\Theta_v = \{\phi, \{t_v - 1\}, \{t_v\}, \{t_v+1\}\}$, i.e., the node candidate is not reinserted ($\phi$) or it is reinserted into one of the time periods $t_v - 1, t_v, t_v + 1$. The node candidate is inserted in the route of a vehicle $k$ with $\sum_{i \in N_c} q_{ikt} < Q$.
   (b) *Two-adjacent-period transformation (T2)* $[1, 2, 3n_q]$: This operator removes and inserts the node candidate into each time period bundle in the set $\Theta_v = \{\phi, \{t_v-1, t_v\}, \{t_v, t_v+1\}\}$, i.e., none, two consecutive periods $t_v-1$ and $t_v$, and two consecutive periods $t_v$ and $t_{v+1}$. To avoid transforming a feasible solution into an infeasible one, the candidate is inserted in the route of a vehicle $k$ with $\sum_{i \in N_c} q_{ikt} + r_{i_v t} < Q$ if $t = t_v$, and $\sum_{i \in N_c} q_{ikt} < Q$ if $t \neq t_v$.
   (c) *Three-adjacent-period transformation (T3)* $[1, 3, 2n_q]$: Instead of inserting into two adjacent periods, this operator removes and inserts the node candidate into each time period bundle in the set $\Theta_v = \{\phi, \{t_v - 1, t_v, t_v + 1\}\}$, i.e., none, and three consecutive periods $t_v - 1, t_v, t_v + 1$ together at once. Similar to the *Two-adjacent-period transformation (T2)* operator, the node candidate is inserted in the route of a vehicle $k$ with $\sum_{i \in N_c} q_{ikt} + r_{i_v t} < Q$ if $t = t_v$, and $\sum_{i \in N_c} q_{ikt} < Q$ if $t \neq t_v$.

2. *Multiple-nodes-single-period transformation:* These operators remove and insert multiple node candidates into one time period at a time. Given a selected node candidate $v$, $n'_q \leq n_q$ node candidates (i.e., from node $v$ to node $v + n'_q - 1$ in the node candidate list) are selected to be inserted together in period $t_v$. We make a list of all the distinct customers that belong to the selected node candidates. The customers in this list are removed from period $t_v$ if they are present. The nodes are inserted in any route $k$ with $\sum_{i \in N_c} q_{ikt_v} < Q$ and they are not necessarily inserted in the same route. Then, the MCF and trimming processes are called respectively to evaluate the cost, and the solution is accepted according to the SA criterion. After that, the algorithm is repeated again to select the $n'_q$ candidates starting from $v + 1$. The process is repeated until all nodes in the list are examined. The details of the operators are as follows.
   (a) *Two-node transformation (T4)* $[2, 1, n_q - 1]$: This operator removes and reinserts $n'_q = 2$ consecutive node candidates in the list together. If the number of node candidates $n_q$ is smaller than 2, the operator *T1* is called instead.
   (b) *Three-node transformation (T5)* $[3, 1, n_q - 2]$: This operator removes and reinserts $n'_q = 3$ consecutive node candidates in the list together. If the number of node candidates $n_q$ is smaller than 3, the operator *T4* is called instead.
3. *Multiple-nodes-multiple-periods transformation:* The operators in this category explore larger neighborhoods compared to the two above categories. These operators are started by listing all the distinct customers and time periods that are present in the list of the node candidates in sets. Let $N_q^C$ be the set of all the distinct customers in the list of node candidates and $T_q$ be the set of all the distinct time periods in the list of node candidates. The details of the operators are as follows.
   (a) *All-node-greedy transformation (T6)* $[n_q, |T_q|, 1]$: It simply removes all node candidates from the current solution and each of them is reinserted in the same period which it was removed from in the hope of finding a better assignment of customers to routes (but possibly in a different vehicle route). Then, the MCF and trimming processes are called. The operator is only performed once per call.

(b) *All-node, one-period transformation (T7)* $[|N_q^C|, 1, |T_q|]$: The earliest period in the set $T_q$ is selected and all customers in the set $N_q^C$ are removed and sequentially reinserted into that period according to the order in the original node candidate list produced by the selection operator. Then, the MCF and trimming processes are called to assess the cost and the solution is accepted according to the SA criterion. Afterwards, the next period in $T_q$ is selected and the process is repeated until all periods in the set $T_q$ are examined.

#### 4.3.3.3 Minimum Cost Flow Subproblem (MCF)

Every time the solution is changed during a transformation process, the MCF is called to find the optimal values of the continuous variables that minimize the total unit production and inventory costs. We denote by $\bar{y}_t$, $\bar{z}_{it}$ and $\bar{x}_{ijkt}$ the fixed production setup, node visit and arc variables in a given solution, and by $k'$ the index of the dummy vehicle. The MCF formulation is as follows:

$$\min \sum_{t \in T} (up_t + \sum_{i \in N} h_i I_{it} + \theta q_{ik't}) \tag{4.22}$$

subject to (4.2)-(4.6), (4.8)-(4.9), (4.14) but replace the binary variables with $\bar{y}_t$, $\bar{z}_{it}$ and $\bar{x}_{ijkt}$, and

$$\sum_{k \in K} q_{ikt} + q_{ik't} = r_{it} \quad \forall i \in N_c. \tag{4.23}$$

The objective function (4.22) minimizes the total unit production and inventory holding costs plus the unit penalty costs of using the dummy vehicle. Constraints (4.23) are equivalent to (4.7) but allow the dummy vehicle to serve customers.

#### 4.3.3.4 Incumbent Solution Improvement

At each iteration, if a new overall best solution is found after a transformation process is finished, we try to further improve the solution by running the traveling salesman problem (TSP) subroutine on each route individually. In this study, the TSP tours are reconstructed by the *GENIUS* procedure (Gendreau et al., 1992) and improved

by the *3-opt* procedure (Lin, 1965). If a better solution is found, it is stored as the new incumbent solution.

## 4.4 Adaptations of the Op-ALNS for the PRP-ML to Other Variants

We explain how this algorithm is adapted for the MVPRP-OU and the MVIRP (both ML and OU) in Sections 4.4.1 and 4.4.2, respectively. Note that only the initialization phase of the original Op-ALNS is modified in these adaptations. The computational results of the Op-ALNS for the PRP and IRP with ML and OU policies can be found in Section 3.6.3.

### 4.4.1 Adaptation of the Op-ALNS for the PRP-OU

Unlike the ML policy where the selection and transformation operators could handle infeasible routes effectively by repeatedly reallocating delivery quantities, it is much more difficult to remove and reinsert node candidates from infeasible routes in the OU policy since the delivery quantity is defined by the difference between the inventory level and the TSL. Thus, it is easier to start from initial solutions with feasible routes in the initialization process to ensure that feasible solutions can be obtained at the end of the process. We simply solve the formulation with a vehicle index to take into account the vehicle capacity for each vehicle separately. The first subproblem in the initialization phase for the PRP-OU is as follows:

$$\min \sum_{t\in T}(up_t + fy_t + h_0 I_{0t} + \sum_{i\in N_c}\sum_{k\in K}\sigma_i z_{ikt}) + \sum_{t\in T'}\sum_{i\in N_c}\sum_{k\in K}\sum_{v=1}^{l+1} e_{itv}\lambda_{iktv} \qquad (4.24)$$

subject to (3.4), (3.8), (3.12)-(3.13), (3.39)-(3.46).

The best version of the SBCs from Section 3.3.1 is also added to improve the performance of the formulation. Note that the delivery quantity variable $q_{ikt}$ is calculated as $q_{ikt} = \sum_{v=1}^{l+1} g_{itv}\bar{\lambda}_{iktv}$. This problem is solved in an attempt to approximately take transportation costs into account, without explicitly modeling the routing decisions. Then, the routes for the vehicles are determined by solving the traveling

salesman problem (TSP) for each vehicle individually. We construct the TSP tours using the *GENIUS* procedure (Gendreau et al., 1992) and improve them by the 3-opt procedure (Lin, 1965). Additionally, to solve the MCF for the OU policy, we set the inventory level of a visited customer $i$ in period $t$ as $I_{it} = L_i - d_{it}$. The remaining of the Op-ALNS algorithm for the OU policy is the same as for the ML policy.

### 4.4.2 Adaptation of the Op-ALNS for the IRP

To solve the IRP, the modification in Section 3.2.4 must be applied first. Since production setups are irrelevant for the IRP, we generate initial solutions with different customer visit decisions. Denote by $\bar{z}^s_{it}$ the value of the customer visit variable $z_{it}$ in solution $s$. The original local branching inequality (4.21) is replaced with the following inequality to generate an initial solution $\bar{s}$:

$$\sum_{z_{it}|\bar{z}^s_{it}=1} (1 - z_{it}) + \sum_{z_{it}|\bar{z}^s_{it}=0} z_{it} \geq \left\lceil 0.25 \sum_{i \in N_c} \sum_{t \in T} \bar{z}^s_{it} \right\rceil \quad s = 1, ..., \bar{s} - 1 \tag{4.25}$$

The inequality (4.25) forces at least 25% of the total customer visits over the horizon to change. The maximum number of initial solutions in the pool is set to 10 for the IRP and the other parts of the algorithm remain the same.

## 4.5 Computational Experiments

The performance of the Op-ALNS and its adaptations are presented in the previous chapter in Section 3.6.3. In this section, we evaluate the efficiency and robustness of our Op-ALNS algorithm on large PRP instances presented in literature. We perform experiments using two benchmark test sets. The details of the benchmarks are as follows.

- The sets **A1**, **A2** and **A3** in Archetti et al. (2011) consist of instances with 6 time periods and 14, 50 and 100 customers with constant demand, no production capacity and no plant inventory capacity, but with initial inventory at the plant and customers. These test sets contain small to medium size problems. There are 96

instance types and five instances per type, for a total of $96 \times 5 = 480$ instances per set. Those instances are grouped into four classes with different parameter settings. With respect to the distribution part, there are two major groups of instances, i.e., the instance set A1 for the PRP with a single capacitated vehicle and sets A2 and A3 with an unlimited number of capacitated vehicles. In our experiments, we set the number of vehicles in sets A2 and A3 to a sufficiently large number.

- The sets **B1**, **B2** and **B3** in Boudia et al. (2005) consist of instances with 50, 100 and 200 customers and 20 time periods. There are 30 instances per set. The size of these instances is larger than those of the Archetti et al. and they are considered as large to very large size instances. In addition, unlike in the first benchmark, constraints on the production capacity, plant inventory capacity and maximum number of trucks are imposed. Customer demand is dynamic and there is no inventory holding cost at the customers. According to the description given in Boudia et al. (2007), production in period $t$ becomes only available in period $t+1$, but no inventory costs are incurred before the production becomes available. As a consequence, the demand in period 1 must be satisfied from the initial inventory at the plant. In the experiments of Boudia et al. (2007) and Boudia and Prins (2009), the initial inventory at the plant is set equal to the total customer demand in the first period and no inventory holding cost is incurred for the initial inventory between period 0 and 1 (Prins, 2012). To solve these instance sets, we simply set the variable $y_1 = 0$, which also enforces $p_1 = 0$, i.e., the production made available in period 1 is equal to zero and thus the customer demand in period 1 is satisfied by the initial inventory at the plant.

In these two benchmarks, the maximum inventory at customers are imposed after demand comsumption, while constraints (4.6) in our formulation are imposed before demand comsumption. Therefore, we simply set $L_i = \bar{L}_i + d_{it}$ where $\bar{L}_i$ is the original value in the instances and this does not have any effect on the computational results. The overview of these instances is presented in Tables 4.I and 4.II.

Our algorithm was coded in C# using Microsoft Visual Studio 2008 under Windows XP. The experiments have been executed on a workstation with a 2.10 GHz

Table 4.I: Overview of the Benchmark Instances

| **Benchmark** | Archetti et al. | | | Boudia et al. | | |
|---|---|---|---|---|---|---|
| **Instance set** | **A1** | **A2** | **A3** | **B1** | **B2** | **B3** |
| No. of instances | 480 | 480 | 480 | 30 | 30 | 30 |
| No. of periods | 6 | 6 | 6 | 20 | 20 | 20 |
| No. of customers | 14 | 50 | 100 | 50 | 100 | 200 |
| No. of trucks | 1 | $\infty$ | $\infty$ | 5 | 9 | 13 |
| Demand | C | C | C | V | V | V |
| Production capacity | $\infty$ | $\infty$ | $\infty$ | C | C | C |
| Plant inventory capacity | $\infty$ | $\infty$ | $\infty$ | C | C | C |
| Customer inventory capacity | C | C | C | C | C | C |
| Initial inventory at plant | 0 | 0 | 0 | V | V | V |
| Initial inventory at customers | V | V | V | 0 | 0 | 0 |
| Vehicle capacity | C | C | C | C | C | C |

V - Varying, C - Constant, $\infty$- Unlimited

Table 4.II: Descriptions of the Archetti et al. Instance Classes

| **Class** | **Type** | **Descriptions** |
|---|---|---|
| Class I | 1-24 | Standard instances |
| Class II | 25-48 | High production unit cost, (Class I) $u\times10$ |
| Class III | 49-72 | Large transportation costs, (Class I) $Coordinates\times5$ |
| Class IV | 73-96 | No customer inventory costs |

Duo CPU and 2 GB of RAM, which is comparable to the workstations used in previous research. We used CPLEX 12.2 to solve the PD subproblems and the *network optimizer* module in CPLEX to solve the MCF. We use a single run on each instance since the results are relatively consistent between runs as shown in Section 4.5.3. In order to demonstrate the efficiency of our optimization-based ALNS approach, we report the results obtained after the false start process, and during the full restart process within 100, 500 and 1000 iterations in the columns Op-ALNS with <FS>, <I-100>, <I-500> and <I-1000>, respectively.

### 4.5.1 Computational Results on the Archetti et al. Benchmark

We compare our approach to the heuristic $\mathcal{H}$ presented in Archetti et al. (2011). According to the results of Archetti et al., the average optimality gap for the heuristic $\mathcal{H}$ on the set A1 is 2.25%, 0.30%, 3.65% and 1.00% for the classes I, II, III and

IV, respectively, and the overall gap is 0.91%. This is an excellent benchmark to demonstrate the performance of our algorithm.

The summary of the results are provided in Table 4.III. We use boldface symbols if the total cost obtained by Op-ALNS is lower than the solution in the literature, and the symbol * to indicate the best solution. The number of new best solutions for each problem type can be found in column *New*. The Op-ALNS is able to produce better average results for classes I, II and IV of the instance set A1 (which uses a single vehicle and has 14 customers) and for all four classes of the sets A2 and A3 (which use multiple vehicles and have 50 and 100 customers, respectively). Our algorithm could provide better average results for 11 out of the 12 different class-set combinations and there are 930 out of 1440 (65%) new best solutions found. For the multiple vehicle test sets A2 and A3, Op-ALNS already provides on average a better solution in the false start phase. The average cost details of the best solutions provided by Op-ALNS after 1000 iterations are presented in Table 4.IV.

Table 4.III: Summary of the Average Total Costs Obtained by Different Heuristics on the Archetti et al. Instances

| Prob set | $N_c$ | $l$ | Class | $Ins$ | $\mathcal{H}$ | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | New |
|---|---|---|---|---|---|---|---|---|---|---|
| A1 | 14 | 6 | Class I | 120 | 52729 | **52706** | **52464** | **52344** | **52332*** | 63 |
| | | | Class II | 120 | 371325 | 371682 | **371237** | **371184** | **371184*** | 57 |
| | | | Class III | 120 | 96532* | 104492 | 102884 | 101652 | 100979 | 24 |
| | | | Class IV | 120 | 202734 | 202924 | 202770 | **202718** | **202717*** | 67 |
| | | | Total | 480 | 180830* | 182951 (1.2%) | 182339 (0.8%) | 181975 (0.6%) | 181803 (0.5%) | 211 |
| A2 | 50 | 6 | Class I | 120 | 167879 | **166978** | **166398** | **166124** | **166109*** | 89 |
| | | | Class II | 120 | 1282069 | **1279412** | **1278541** | **1278245** | **1278237*** | 95 |
| | | | Class III | 120 | 224944 | 228997 | 226605 | **224903** | **224467*** | 62 |
| | | | Class IV | 120 | 695540 | **692484** | **692135** | **692034** | **692027*** | 120 |
| | | | Total | 480 | 592608 | **591968** (-0.1%) | **590920** (-0.3%) | **590327** (-0.4%) | **590210*** (-0.4%) | 366 |
| A3 | 100 | 6 | Class I | 120 | 307072 | **304069** | **303496** | **303178** | **303045*** | 100 |
| | | | Class II | 120 | 2369957 | **2369857** | **2368206** | **2367385** | **2367299*** | 86 |
| | | | Class III | 120 | 411237 | 417226 | 413610 | **411106** | **410451*** | 66 |
| | | | Class IV | 120 | 1281771 | **1278241** | **1277784** | **1277602** | **1277560*** | 101 |
| | | | Total | 480 | 1092509 | **1092348** (-0.0%) | **1090774** (-0.2%) | **1089818** (-0.2%) | **1089589*** (-0.3%) | 353 |

* indicates the best solution

(-) indicates the % difference of the solution obtained by Op-ALNS compared to $\mathcal{H}$

Table 4.IV: Average Values of the Op-ALNS Best Solutions on the Archetti et al. Instances

| Prob set | Class | Total | C.Inv | P.Inv | Trans | Prod | D.Qty | N.Visits | N.Vehs | N.Setups |
|---|---|---|---|---|---|---|---|---|---|---|
| A1 | Class I | 52332 | 7640 | 2765 | 5812 | 36115 | 543.6 | 20.6 | 1.0 | 1.2 |
| | Class II | 371184 | 7679 | 3387 | 6137 | 353980 | 543.6 | 21.2 | 1.0 | 1.0 |
| | Class III | 100979 | 8014 | 2435 | 53040 | 37490 | 543.6 | 18.3 | 1.0 | 1.4 |
| | Class IV | 202717 | 0 | 2529 | 5499 | 194689 | 543.6 | 18.7 | 1.0 | 1.0 |
| | Total | 181803 | 5833 | 2779 | 17622 | 155568 | 543.6 | 19.7 | 1.0 | 1.2 |
| A2 | Class I | 166109 | 20517 | 1922 | 15579 | 128091 | 2126.2 | 88.1 | 5.5 | 2.1 |
| | Class II | 1278237 | 26825 | 11035 | 15967 | 1224410 | 2126.2 | 90.4 | 5.3 | 1.0 |
| | Class III | 224467 | 23246 | 1574 | 70840 | 128808 | 2126.2 | 74.8 | 5.5 | 2.2 |
| | Class IV | 692027 | 0 | 3018 | 14092 | 674917 | 2126.2 | 78.7 | 7.1 | 1.3 |
| | Total | 590210 | 17647 | 4387 | 29119 | 539056 | 2126.2 | 83.0 | 5.9 | 1.7 |
| A3 | Class I | 303045 | 37228 | 1791 | 28295 | 235731 | 4019.2 | 203.6 | 8.4 | 2.9 |
| | Class II | 2367299 | 50949 | 22687 | 28103 | 2265560 | 4019.2 | 188.8 | 8.8 | 1.0 |
| | Class III | 410451 | 43614 | 2258 | 129973 | 234606 | 4019.2 | 158.2 | 9.3 | 2.6 |
| | Class IV | 1277560 | 0 | 3550 | 25202 | 1248808 | 4019.2 | 159.1 | 12.5 | 1.5 |
| | Total | 1089589 | 32947 | 7572 | 52893 | 996176 | 4019.2 | 177.4 | 9.8 | 2.0 |

Costs
Total - Average total cost
C.Inv - Average inventory cost at customers
P.Inv - Average inventory cost at plant
Trans - Average transportation cost
Prod - Average total production cost

Values
D.Qty - Average delivery quantity
N.Visits - Average number of customer visits
N.Vehs - Average number of vehicles used
N.Setups - Average number of production setups

Table 4.V provides a comparison of computing times for the $\mathcal{H}$ heuristic and the Op-ALNS. The column N.Cuts is the average number of local branching inequalities applied to obtain the setup schedule in the best solution. The column N.Nodes shows the average number of node candidates that are examined, and the column N.MCF is the average number of times that the MCF is called in the entire process. The solutions obtained after the false start process are very good while this phase only takes a relatively short computing time. It is worth noting that the computational times of our Op-ALNS algorithm are less sensitive to the size of the instances compared to the $\mathcal{H}$ procedure. The Op-ALNS is outperformed by the $\mathcal{H}$ heuristic only on the small size instance set A1. The computing times of Op-ALNS with 100 iterations in the full restart process are comparable to the computing times of $\mathcal{H}$ for the instance set A2, while Op-ALNS can reach more than 500 iterations for the larger size instance set A3 within a similar amount of computing time compared to $\mathcal{H}$. Finally, it should be

noted that our algorithm is more general than $\mathcal{H}$, since $\mathcal{H}$ cannot handle production capacities, which make the problem more complicated.

Table 4.V: Summary of the Average Computational Times in Seconds Obtained by Different Heuristics on the Archetti et al. Instances

| Prob set | Class | $\mathcal{H}^1$ | Op-ALNS$^2$ <FS> | Op-ALNS$^2$ <I-100> | Op-ALNS$^2$ <I-500> | Op-ALNS$^2$ <I-1000> | N.Cuts | N.Nodes | N.MCF |
|---|---|---|---|---|---|---|---|---|---|
| A1 | Class I | - | 1.7 | 2.4 | 5.2 | 9.2 | 0.6 | 3245 | 5478 |
| | Class II | - | 1.4 | 2.0 | 4.8 | 8.9 | 0.1 | 3304 | 5564 |
| | Class III | - | 1.7 | 2.2 | 5.2 | 9.2 | 2.6 | 2948 | 4902 |
| | Class IV | - | 1.5 | 2.2 | 4.8 | 8.7 | 0.5 | 3051 | 5211 |
| | Total | - | 1.6 | 2.2 | 5.0 | 9.0 | 0.9 | 3137 | 5289 |
| A2 | Class I | 11.3 | 6.2 | 10.8 | 28.6 | 50.2 | 0.2 | 5877 | 12721 |
| | Class II | 12.4 | 5.8 | 10.9 | 28.5 | 49.5 | 0.0 | 5891 | 12068 |
| | Class III | 9.5 | 5.7 | 9.6 | 24.4 | 42.7 | 1.7 | 4890 | 10274 |
| | Class IV | 10.9 | 5.5 | 9.7 | 25.7 | 44.1 | 0.8 | 5264 | 10396 |
| | Total | 11.0 | 5.8 | 10.3 | 26.8 | 46.6 | 0.7 | 5481 | 11365 |
| A3 | Class I | 188.0 | 24.4 | 47.5 | 136.1 | 249.1 | 1.9 | 9922 | 23862 |
| | Class II | 216.7 | 21.2 | 42.8 | 124.7 | 221.1 | 0.0 | 9289 | 21597 |
| | Class III | 167.8 | 20.5 | 38.2 | 106.7 | 190.8 | 3.9 | 7631 | 18348 |
| | Class IV | 181.1 | 19.8 | 37.4 | 108.4 | 189.0 | 0.5 | 8043 | 17697 |
| | Total | 188.4 | 21.5 | 41.5 | 119.0 | 212.5 | 1.6 | 8722 | 20376 |

- the computational times of the $\mathcal{H}$ procedure for the instance A1 are negligible

[1] executed on 2.40 GHz PC

[2] executed on 2.10 GHz Duo CPU PC

### 4.5.2 Computational Results on the Boudia et al. Benchmark

We next compare our results for the Boudia et al. benchmark with several other metaheuristic approaches, i.e., greedy randomized adaptive search procedure - GRASP (Boudia et al., 2007), memetic algorithm with population management - MA|PM (Boudia and Prins, 2009), reactive tabu search - RTS (Bard and Nananukul, 2009a), and tabu search with path relinking - TSPR (Armentano et al., 2011). The previous best solutions are provided by the TSPR procedure developed by Armentano et al.

Table 4.VI provides the summary of the average total costs of each heuristic and Table 4.VIII provides the summary of the computational times. The Op-ALNS presented in this paper outperforms all other algorithms from the literature by providing better average solutions. The Op-ALNS can find the solutions with lower average total costs compared to the previous best known solutions even with just the false

start process where the algorithm spends relatively short computing times compared to the other heuristics. The algorithm can further improve the solution quality when we allow 100, 500 and 1000 iterations in the full restart improvement phase. We found 89 new best solutions or 99% of the total number of instances. For the larger size sets B2 and B3, we found better solutions for every instance. Note that for the sets B2 and B3, the algorithm stops after the maximum number of node candidates (15,000 nodes) has been reached. It is worth noting that the relative improvement of Op-ALNS compared to the solutions of Armentano et al. is higher for the larger instances. The average cost details of the best solutions provided by Op-ALNS after the algorithm terminates are given in Table 4.VII.

Table 4.VI: Summary of the Average Total Costs Obtained by Different Heuristics on the Boudia et al. Instances

| Prob set | $N_c$ | $l$ | *Ins* | GRASP | MA\|PM | RTS | TSPR | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | New |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| B1 | 50 | 20 | 30 | 443264 | 393263 | 369662 | 361704 | **351448** (-2.8%) | **348982** (-3.5%) | **347260** (-3.9%) | **346878*** (-4.1%) | 29 |
| B2 | 100 | 20 | 30 | 791839 | 714627 | 712294 | 685898 | **641092** (-6.5%) | **638976** (-6.8%) | **637037** (-7.1%) | **636962*** (-7.1%) | 30 |
| B3 | 200 | 20 | 30 | 1070026 | 1001634 | 1034923 | 951638 | **881653** (-7.3%) | **878511** (-7.6%) | **876761*** (-7.8%) | - - | 30 |

* indicates the best solution

(-) indicates the % difference of the solution obtained by Op-ALNS compared to TSPR

Table 4.VII: Average Values of the Op-ALNS Best Solutions on the Boudia et al. Instances

| Prob set | Total | C.Inv | P.Inv | Trans | Prod | D.Qty | N.Visits | N.Vehs | N.Setups |
|---|---|---|---|---|---|---|---|---|---|
| B1 | 346878 | 0 | 36206 | 109006 | 201667 | 206198.4 | 298.9 | 5.0 | 4.0 |
| B2 | 636962 | 0 | 27924 | 207705 | 401333 | 454168.3 | 708.4 | 9.0 | 5.7 |
| B3 | 876761 | 0 | 27532 | 249229 | 600000 | 828279.0 | 1090.9 | 13.0 | 5.0 |

### 4.5.3 Evaluation of the Op-ALNS Configurations

In this section, we report additional experiments to evaluate the performance of the Op-ALNS using different configurations. To avoid excessive computing times, we selected subsets of the instances, i.e., the first instance in each class from the instance type 1 of the sets A1, A2 and A3, and the first instance from the sets B1, B2 and

Table 4.VIII: Summary of the Average Computational Times in Seconds Obtained by Different Heuristics on the Boudia et al. Instances

| Prob set | GRASP[1] | MA\|PM [1] | RTS[2] | TSPR[3] | Op-ALNS[4] <FS> | Op-ALNS[4] <I-100> | Op-ALNS[4] <I-500> | Op-ALNS[4] <I-1000> | N.Cuts | N.Nodes | N.MCF |
|---|---|---|---|---|---|---|---|---|---|---|---|
| B1 | 93.5 | 172.7 | 330.6 | 317.0 | 121.4 | 157.5 | 298.4 | 481.3 | 2.5 | 11980 | 31180 |
| B2 | 415.9 | 1108.1 | 975.6 | 1147.7 | 402.0 | 611.9 | 1404.6 | 1569.9 | 2.8 | 15022 | 39732 |
| B3 | 1893.8 | 4098.5 | 2492.3 | 3926.4 | 1563.0 | 2846.3 | 5794.2 | - | 3.0 | 15036 | 39491 |

[1] executed on 2.30 GHz PC
[2] executed on 2.53 GHz PC
[3] executed on 2.80 GHz PC
[4] executed on 2.10 GHz Duo CPU PC

B3. This gives a total of 15 instances. First, the Op-ALNS is performed 10 times using the default configuration where all operators and the TSP subroutine are used. The average total costs and computing times are calculated and used as the point of reference. Then, several different configurations, where one operator type or the TSP subroutine is sequentially removed, are tested and compared with this reference. The results are shown in Table 4.IX. The first group of rows shows the average % cost deviation ($\%Dev$), calculated as $\left(f(s_c) - f(\hat{s})\right)/f(\hat{s})$, where $f(s_c)$ and $f(\hat{s})$ are the objective function values obtained by configuration $c$ and by the default setting, respectively. The second group of rows shows the ratio of the average computing time using the configuration $x$ ($T_x$) over the default configuration ($T_0$). A ratio equal to 2 means that the configuration $x$ spends on average double the computing time compared to the default configuration. The columns 2-6 show the results of the Op-ALNS when each operator type of the algorithm is removed, and column 7 is the results when the TSP subroutine is removed.

Since the heuristic contains random elements, i.e., the selection of operators, the simulated annealing acceptance mechanism and the random selection operator (*S1*), different runs can lead to different solutions and different costs. The impact of this is evaluated by performing 10 runs on each instance and calculating the percent coefficient of variation (%CV), which is the ratio of the standard deviation to the mean of the objective function values obtained in 10 runs using the default configuration. This information is reported in the last column. Note that we consider the results obtained in the full restart process within 1000 iterations.

Table 4.IX: Performance Evaluation of the Op-ALNS Using Different Op-ALNS Configurations

| | Prob set | S.Rank S1-S5 | S.Mod S6-S7 | T.Periods T1-T3 | T.Nodes T4-T5 | T.Large T6-T7 | TSP | M.Runs %*CV* |
|---|---|---|---|---|---|---|---|---|
| %*Dev* | A1 | 0.33 | 0.69 | 1.54 | 1.26 | 0.53 | 0.29 | 0.11 |
| | A2 | 0.79 | 0.04 | 0.18 | 0.14 | 0.00 | 0.03 | 0.12 |
| | A3 | 0.44 | 0.04 | 0.02 | 0.05 | 0.10 | 0.07 | 0.08 |
| | B1 | 1.53 | 0.74 | -0.10 | 0.07 | 0.05 | 0.20 | 0.09 |
| | B2 | 0.07 | 0.04 | 0.28 | 0.32 | 0.00 | -0.03 | 0.12 |
| | B3 | 0.25 | 0.06 | 0.09 | -0.01 | -0.03 | 0.11 | 0.14 |
| $T_x/T_0$ | A1 | 1.58 | 0.64 | 0.60 | 0.73 | 1.57 | 0.97 | 11.71 |
| | A2 | 1.85 | 0.78 | 0.74 | 0.70 | 1.47 | 0.96 | 6.91 |
| | A3 | 1.18 | 0.92 | 0.76 | 0.72 | 1.24 | 0.99 | 4.74 |
| | B1 | 0.72 | 1.14 | 0.88 | 0.66 | 1.07 | 0.95 | 6.11 |
| | B2 | 0.67 | 0.98 | 0.72 | 0.56 | 1.36 | 1.01 | 3.91 |
| | B3 | 0.78 | 0.97 | 0.89 | 0.50 | 1.22 | 0.97 | 3.70 |

By removing each operator type, the results are generally worse than the default configuration where all operator types are used. The impact of removing each operator type varies due to different features of the problem sets. The *selection by ranking (S1-S5)* and *single-node-multiple-periods transformation (T1-T3)* operators have the most significant impact on the solution quality when the operator type is removed. The *multiple-nodes-multiple-periods transformation (T6-T7)* operator, however, is mostly effective on the set A1. Additionally, the TSP subroutine could provide additional improvements on the instance set A1 and B1 but the effect is negligible on the other sets. The results on the multiple runs in the final column show that the algorithm has no significant impact from the randomness in the Op-ALNS because the %CV of the total costs are very small, i.e., ranging between 0.08-0.14%. The variation of the set B3 is slightly larger than the others since the algorithm terminates after the maximum number of node candidates reached while it could perform less than 500 iterations in the full restart process

The number of node candidates in the *selection by ranking (S1-S5)* operator type is proportional to the total number of visits, while the number of node candidates in the *selection with modification (S6-S7)* depends upon the number of customers in the selected route(s). Consequently, the average computing times on the sets A1-A3 significantly increase when removing the *selection by ranking* operator type as the number of nodes in the *selection with modification* is generally larger, and the

results are opposite when solving the sets B1-B3 where the number of node candidates using the *selection by ranking* is smaller than the other type. The computing times also decrease when either the *single-node-multiple-periods transformation (T1-T3)* or the *multiple-nodes-single-period transformation (T4-T5)* transformation operator type is removed since the algorithm could put more emphasis on the operators that are less time comsuming, but the solution quality is worse compared to the default configuration. Removing the *multiple-nodes-multiple-periods transformation (T6-T7)* operator, however, result in an increase in computing time because this operator type is less time comsuming compared to the other two types. We also observe that the computing time of the TSP subroutine is negligible so that we use the TSP subroutine to ensure that the algorithm is robust. The %CV of the computing times, however, are significantly larger compared to the %CV of the total costs.

### 4.5.4 Analysis of the Selection and Transformation Operators

This section provides the performance analysis of the selection and transformation operators used in the Op-ALNS. The summary of the performance of the operators for the instance sets A1, A2 and A3 are provided in Tables 4.X and 4.XI, and for the instance sets B1, B2 and B3 in Tables 4.XII and 4.XIII. We consider the number of calls in both the false start and full restart process. With respect to the percentage of calls that each selection operator can improve a solution, the *Random selection (S1)* and the *Unit savings greedy selection (S2)* operators are the best for the instances with multiple vehicles, while the *Random Route Duplication (S6)* is the best for the single vehicle instances (A1). In the single vehicle case, the Op-ALNS cannot take advantage of some operators that are mainly used to improve vehicle routes, i.e., the operators *S7* and *T6* that can generally modify several routes at the same time. For the transformation operators, the *Two-adjacent-period transformation (T2)* generally perform well in most instances except the instance sets A1 and B1 where the *Three-node transformation (T5)* and the *Two-node transformation (T4)* are the best, respectively.

Table 4.XIV shows the performance of each operator type. The selection operators in the *Selection by ranking* group are better in finding a new solution. The main

Table 4.X: Performance of Selection Operators for the Archetti et al. Instances

| Prob | Class | Average number of calls | | | | | | | Average % of calls to improve a solution | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| set | | *S1* | *S2* | *S3* | *S4* | *S5* | *S6* | *S7* | *S1* | *S2* | *S3* | *S4* | *S5* | *S6* | *S7* |
| A1 | Class I | 171 | 191 | 182 | 152 | 163 | 78 | 163 | 4% | 5% | 3% | 3% | 2% | 7% | 3% |
| | Class II | 168 | 193 | 180 | 152 | 169 | 79 | 160 | 4% | 6% | 3% | 4% | 2% | 7% | 4% |
| | Class III | 176 | 194 | 185 | 158 | 158 | 69 | 160 | 4% | 6% | 3% | 3% | 2% | 6% | 3% |
| | Class IV | 159 | 180 | 174 | 159 | 169 | 81 | 177 | 3% | 5% | 1% | 2% | 2% | 6% | 1% |
| | Average | 169 | **190** | 180 | 155 | 165 | 77 | 165 | 4% | 5% | 3% | 3% | 2% | **7%** | 3% |
| A2 | Class I | 224 | 188 | 149 | 138 | 156 | 132 | 113 | 12% | 11% | 7% | 9% | 7% | 5% | 6% |
| | Class II | 234 | 187 | 153 | 131 | 155 | 127 | 113 | 14% | 13% | 9% | 10% | 9% | 6% | 8% |
| | Class III | 242 | 182 | 146 | 133 | 164 | 114 | 120 | 14% | 12% | 8% | 9% | 9% | 3% | 8% |
| | Class IV | 188 | 207 | 147 | 129 | 160 | 153 | 116 | 10% | 11% | 6% | 8% | 6% | 7% | 5% |
| | Average | **222** | 191 | 149 | 133 | 159 | 132 | 116 | **12%** | **12%** | 7% | 9% | 8% | 5% | 7% |
| A3 | Class I | 248 | 180 | 149 | 115 | 150 | 144 | 114 | 16% | 13% | 8% | 12% | 10% | 7% | 5% |
| | Class II | 243 | 182 | 163 | 111 | 155 | 137 | 110 | 18% | 15% | 11% | 14% | 12% | 9% | 7% |
| | Class III | 266 | 185 | 150 | 101 | 171 | 108 | 119 | 19% | 15% | 11% | 14% | 13% | 6% | 9% |
| | Class IV | 203 | 189 | 156 | 106 | 155 | 164 | 126 | 14% | 12% | 7% | 11% | 9% | 9% | 5% |
| | Average | **240** | 184 | 155 | 108 | 158 | 138 | 117 | **17%** | 14% | 9% | 13% | 11% | 8% | 6% |

Table 4.XI: Performance of Transformation Operators for the Archetti et al. Instances

| Prob | Class | Average number of calls | | | | | | | Average % of calls to improve a solution | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| set | | *T1* | *T2* | *T3* | *T4* | *T5* | *T6* | *T7* | *T1* | *T2* | *T3* | *T4* | *T5* | *T6* | *T7* |
| A1 | Class I | 161 | 88 | 144 | 260 | 55 | 290 | 102 | 4% | 7% | 7% | 3% | 11% | 1% | 3% |
| | Class II | 162 | 91 | 150 | 252 | 58 | 285 | 101 | 5% | 8% | 8% | 3% | 11% | 1% | 3% |
| | Class III | 146 | 83 | 128 | 284 | 46 | 308 | 106 | 6% | 8% | 8% | 2% | 11% | 1% | 3% |
| | Class IV | 157 | 93 | 142 | 271 | 51 | 287 | 99 | 3% | 5% | 5% | 1% | 8% | 1% | 3% |
| | Average | 156 | 89 | 141 | **267** | 53 | 293 | 102 | 4% | 7% | 7% | 2% | **10%** | 1% | 3% |
| A2 | Class I | 200 | 152 | 120 | 205 | 117 | 161 | 146 | 8% | 12% | 14% | 9% | 11% | 4% | 6% |
| | Class II | 194 | 162 | 122 | 210 | 118 | 152 | 140 | 10% | 14% | 15% | 11% | 12% | 4% | 6% |
| | Class III | 229 | 137 | 110 | 205 | 106 | 165 | 148 | 10% | 13% | 15% | 11% | 12% | 4% | 6% |
| | Class IV | 206 | 124 | 123 | 204 | 110 | 177 | 156 | 7% | 10% | 14% | 8% | 9% | 3% | 7% |
| | Average | **207** | 144 | 119 | 206 | 113 | 164 | 147 | 9% | 12% | **14%** | 10% | 11% | 4% | 6% |
| A3 | Class I | 161 | 182 | 154 | 208 | 172 | 107 | 116 | 11% | 15% | 12% | 13% | 11% | 6% | 4% |
| | Class II | 155 | 216 | 142 | 192 | 170 | 99 | 125 | 12% | 17% | 13% | 15% | 14% | 8% | 7% |
| | Class III | 179 | 180 | 140 | 211 | 172 | 104 | 115 | 13% | 17% | 13% | 17% | 14% | 9% | 7% |
| | Class IV | 161 | 128 | 153 | 219 | 181 | 127 | 132 | 10% | 13% | 11% | 11% | 9% | 7% | 6% |
| | Average | 164 | 176 | 147 | **208** | 174 | 109 | 122 | 12% | **16%** | 12% | 14% | 12% | 7% | 6% |

reasons are that this group can be expanded when the large search space is applied and thus have more opportunity to improve a solution, and the modification schemes in the *Selection with modification* operators generally lead to a worse solution. For the transformation operators, the *Single-node-multiple-periods transformation* performs well on the Archetti et al. instances which are the small to medium size instances,

Table 4.XII: Performance of Selection Operators for the Boudia et al. Instances

| Prob | Average number of calls | | | | | | | Average % of calls to improve a solution | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| set | *S1* | *S2* | *S3* | *S4* | *S5* | *S6* | *S7* | *S1* | *S2* | *S3* | *S4* | *S5* | *S6* | *S7* |
| B1 | **255** | 198 | 143 | 141 | 129 | 117 | 115 | **15%** | 12% | 8% | 10% | 7% | 7% | 3% |
| B2 | **127** | 126 | 93 | 81 | 84 | 96 | 77 | **18%** | 13% | 10% | 8% | 9% | 9% | 5% |
| B3 | **82** | 80 | 66 | 60 | 52 | 56 | 49 | **13%** | 12% | 10% | 7% | 7% | 12% | 3% |

Table 4.XIII: Performance of Transformation Operators for the Boudia et al. Instances

| Prob | Average number of calls | | | | | | | Average % of calls to improve a solution | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| set | *T1* | *T2* | *T3* | *T4* | *T5* | *T6* | *T7* | *T1* | *T2* | *T3* | *T4* | *T5* | *T6* | *T7* |
| B1 | 168 | 161 | 156 | **231** | 185 | 89 | 109 | 7% | 11% | 8% | **14%** | 12% | 3% | 8% |
| B2 | 109 | **128** | 92 | 108 | 116 | 70 | 63 | 12% | **14%** | 9% | 13% | **14%** | 4% | 4% |
| B3 | 71 | **104** | 58 | 63 | 63 | 42 | 44 | 6% | **14%** | 8% | **14%** | 11% | 2% | 6% |

while both the *Single-node-multiple-periods transformation* and the *Multiple-nodes-single-period transformation* have similar performances for the large to very large size instances of Boudia et al.

Table 4.XIV: Average % of Calls Leading to an Improvement

| **Operator type** | **Instance** | |
|---|---|---|
| | Archetti et al. | Boudia et al. |
| *Selection* | | |
| *Selection by ranking* | 9% | 11% |
| *Selection with modification* | 6% | 6% |
| *Transformation* | | |
| *Single-node-multiple-periods* | 10% | 10% |
| *Multiple-nodes-single-period* | 9% | 13% |
| *Multiple-nodes-multiple-periods* | 4% | 5% |

## 4.6 Conclusion

In this chapter, we have discussed the production routing problem (PRP) which is a very complicated combinatorial problem that combines the multi-level lot-sizing problem and the vehicle routing problem. We have developed an efficient heuristic using enumeration, adaptive large neighborhood search and network flow techniques to solve the problem. The experiments conducted on small to medium size instances (Archetti et al., 2011) and on large to very large size instances (Boudia et al., 2005)

show that our algorithm (Op-ALNS) outperforms the former heuristic approaches on the two benchmark test sets and provides high quality solutions. This approach can also be adapted to solve the PRP with multiple products where we can still handle the binary variables representing setup and routing decisions by the enumeration scheme and the operators of the ALNS. The MCF, however, has to be replaced by a capacitated multi-commodity network flow problem to determine the continuous variables associated with the quantity decisions.

# Chapter 5
# Production Routing Under Demand Uncertainty

This chapter is based on the following article.

- Adulyasak, Y., Cordeau, J.-F., Jans, R. Benders Decomposition for Production Routing under Demand Uncertainty. GERAD Tech Rep. G-2012-57. 35 pages. Submitted to *Operations Research* in October 2012.

This chapter addresses the stochastic PRP (SPRP) with demand uncertainty in a two-stage decision process. We introduce a novel approach based on Benders decomposition to solve the problem. We further discuss the benefits of reoptimization capabilities that can be useful in two practical settings in stochastic environments.

## 5.1 Introduction

Demand uncertainty is a major issue in supply chain management as some of the information required for decision making is often known only approximately in the form of forecasts. In these circumstances, solving a deterministic model using point estimates can lead to wrong and costly decisions. One should thus explicitly take the uncertainty into account in the decision process.

In this chapter, we consider the stochastic PRP (SPRP) under demand uncertainty in a two-stage decision process, where the distribution of the demand is assumed to be known. In the first stage, setup and customer visit decisions, as well as the assignment of vehicles to customers in the case of multiple vehicles must be determined. This is in line with real-world practice, where some decisions such as production setups are decided in advance and these plans remain fixed in order to avoid large disruptions (Hopp and Spearman, 2000). Planned visits must be communicated to the customers (and sometimes to the drivers) in advance in order to

prepare the workforce, equipment and materials. The replenishment schedules are hence also fixed before the delivery is made. The second stage involves production, inventory and delivery quantity decisions made when the demand becomes known. Since the decisions regarding customer visits and customer-vehicle assignments are made in the first stage, the routing decisions consist of constructing a tour for each vehicle to visit the set of assigned customers. This can be determined in the first stage regardless of the demand realization in the second stage. If some demand is left unmet at the end of a period, a unit penalty cost has to be paid. This penalty cost can be viewed as the cost of purchasing and delivering the product from an outsourced manufacturer or an opportunity cost associated with the unmet demand. As such, all tours are feasible since not all the demand has to be delivered by the regular vehicles. This problem is an important variant and a practical enhancement to the deterministic PRP and IRP and can also be seen as a generalization of these problems when taking into account the uncertainty of demand. To the best of our knowledge, this problem has not been discussed before. Most of the previous studies on the SIRP consider a Markov decision process where all the decisions are taken independently in each discrete time period and the outcomes of the decisions in each stage affect the subsequent stage. These problems consist of finding optimal decisions in each time period, which lead to the minimal expected total cost in the planning horizon. The only research studying a two-stage decision problem is the RIRP addressed by Solyalı et al. (2012). They focused on robust optimization which attempts to find a solution that is immune to any realization of demand. Their approach is appropriate when the distribution of demand is unknown. However, since information such as historical demands is typically available, one can construct demand profiles and use a two-stage stochastic model to solve the problem. Moreover, both the SPRP and SIRP concern operational planning where the decision process is repeated a large number of times over a long horizon. Instead of using robust optimization to find the solution with the minimal worst case cost, one can possibly benefit to a greater degree in the long term by using the two-stage stochastic programming approach to find the solution where the total expected operating cost is minimized.

The main contributions of this chapter are fourfold. First, we introduce the two-stage stochastic PRP and provide a standard formulation which is an extension of the deterministic PRP formulation presented in Chapter 3. Second, we propose two Benders reformulations for the SPRP. In the first decomposition, the second-stage quantity decision variables are projected out, while the routing variables are also projected out in the second decomposition. Third, we develop exact algorithms for the Benders reformulations based on a branch-and-cut framework, called branch-and-Benders-cut (BBC), where the Benders cuts are used in conjunction with subtour elimination constraints. This algorithm is then compared to a classical Benders algorithm. Several computational enhancements are proposed: lower bound lifting inequalities, scenario group cuts and Pareto-optimal cuts, and extensive computational results are provided. A comparison with an extension of the successful branch-and-cut (BC) approach for the deterministic problems in Chapter 3 indicates the superiority of the best version of the BBC approach using the first decomposition for problems with a large number of scenarios. Fourth, we demonstrate the benefits of the reoptimization capabilities of Benders decomposition for the SPRP in two stochastic environments, i.e., in a sample average approximation method (SAA) and in a rolling horizon (RH) framework. In these two settings, we obtain improvements in CPU time of 50% (for SAA) and 41% (for RH) compared to the Benders approach without reoptimization, and of 83% (for SAA) and 81% (for RH) compared to the branch-and-cut approach.

The rest of the chapter is organized as follows. Section 5.2 presents notation and formulations for the SPRP. Section 5.3 describes the Benders decomposition approaches to solve the problem. The details of the solution algorithms are provided in Section 5.4. This is followed by the computational experiments in Sections 4.5 and 5.6, and by the conclusion.

## 5.2 SPRP Formulations

### 5.2.1 Notation

Let $\Omega$ denote the finite set of demand scenarios (this set can be enumerated explicitly if the number of possible demand realizations is not too large or it can be constructed

by a sampling method). Also let $\omega \in \Omega$ be the index of the demand scenarios. The two-stage SPRP can be defined on a complete undirected graph $G = (N, E)$. We further use the following notation in addition to the previously defined notation in Chapter 3.

Decision variables:

$p_{t\omega}$ production quantity in period $t$ under scenario $\omega$;

$I_{it\omega}$ inventory at node $i$ at the end of period $t$ under scenario $\omega$;

$q_{ikt\omega}$ quantity delivered to customer $i$ with vehicle $k$ in period $t$ under scenario $\omega$;

$e_{it\omega}$ amount of unmet demand at customer $i$ in period $t$ associated with scenario $\omega$.

Parameters:

$\sigma_i$ unit cost of unmet demand of customer $i$;

$d_{it\omega}$ demand of customer $i$ in period $t$ under scenario $\omega$ (we assume throughout that $d_{it\omega} \leq L_i, \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega$);

$\rho_\omega$ probability of scenario $\omega$.

Let also $I_{i0\omega} = I_{i0}, \forall \omega \in \Omega$, $M_{t\omega} = \min\left\{C, \sum_{j=t}^{l} \sum_{i \in N_c} d_{ij\omega}\right\}$ and $M'_{it\omega} = \min\left\{L_i, Q, \sum_{j=t}^{l} d_{ij\omega}\right\}$.

### 5.2.2 Two-Stage SPRP Formulation

We first present a two-stage SPRP formulation which is an extension of the formulation used in the branch-and-cut approaches for the deterministic problems in Chapter 3. The SPRP can be formulated as follows:

$$\min \sum_{t \in T} \left( f y_t + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} x_{ijkt} + \sum_{\omega \in \Omega} \rho_\omega \left( u p_{t\omega} + \sum_{i \in N} h_i I_{it\omega} + \sum_{i \in N_c} \sigma_i e_{it\omega} \right) \right) \quad (5.1)$$

s.t.

$$I_{0,t-1,\omega} + p_{t\omega} = \sum_{i \in N_c} \sum_{k \in K} q_{ikt\omega} + I_{0t\omega} \qquad \forall t \in T, \forall \omega \in \Omega \tag{5.2}$$

$$I_{i,t-1,\omega} + \sum_{k \in K} q_{ikt\omega} + e_{it\omega} = d_{it\omega} + I_{it\omega} \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega \tag{5.3}$$

$$I_{0t\omega} \leq L_0 \qquad \forall t \in T, \forall \omega \in \Omega \tag{5.4}$$

$$I_{it\omega} + d_{it\omega} \leq L_i \qquad \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega \tag{5.5}$$

$$p_{t\omega} \leq M_{t\omega} y_t \qquad \forall t \in T, \forall \omega \in \Omega \tag{5.6}$$

$$\sum_{i \in N_c} q_{ikt\omega} \leq Q z_{0kt} \qquad \forall k \in K, \forall t \in T, \forall \omega \in \Omega \tag{5.7}$$

$$q_{ikt\omega} \leq M'_{it\omega} z_{ikt} \quad \forall i \in N_c, \forall k \in K, \forall t \in T, \forall \omega \in \Omega \tag{5.8}$$

$$\sum_{k \in K} z_{ikt} \leq 1 \qquad \forall i \in N_c, \forall t \in T \tag{5.9}$$

$$\sum_{(j,j') \in \delta(i)} x_{jj'kt} = 2 z_{ikt} \qquad \forall i \in N, \forall k \in K, \forall t \in T \tag{5.10}$$

$$\sum_{(i,j) \in E(S)} x_{ijkt} \leq \sum_{i \in S} z_{ikt} - z_{ekt} \qquad \forall S \subseteq N_c : |S| \geq 2, \forall e \in S, \forall k \in K, \forall t \in T \tag{5.11}$$

$$p_{t\omega}, I_{it\omega}, q_{ikt\omega} \geq 0 \qquad \forall i \in N, \forall k \in K, \forall t \in T, \forall \omega \in \Omega \tag{5.12}$$

$$y_t, z_{ikt} \in \{0,1\} \qquad \forall i \in N, \forall k \in K, \forall t \in T \tag{5.13}$$

$$x_{ijkt} \in \{0,1\} \qquad \forall (i,j) \in E : i \neq 0, \forall k \in K, \forall t \in T \tag{5.14}$$

$$x_{0jkt} \in \{0,1,2\} \quad \forall j \in N_c, \forall k \in K, \forall t \in T. \tag{5.15}$$

The objective function (5.1) minimizes the total cost of the first stage decisions and the expected total cost of the second stage decisions. Constraints (5.2) and (5.3) control the inventory flow balance for each scenario at the plant and customers, respectively. The maximum inventory level at the plant and customers is imposed by constraints (5.4) and (5.5), respectively. Constraints (5.6) allow a positive production quantity only if a setup is made; this quantity cannot exceed the minimum of the production capacity and the total demand in the remaining periods. The delivery

quantity in each vehicle cannot exceed the vehicle capacity imposed by constraints (5.7) and a positive delivery quantity is allowed only if the customer is visited according to constraints (5.8). Each customer cannot be visited more than once per period following constraints (5.9). Constraints (5.10) require the number of edges incident to be 2 if the node is visited and constraints (5.11) are subtour elimination constraints (SECs) for each vehicle. We remark that constraints (5.5) impose the inventory capacity at customers when the delivery is made prior to demand consumption. These constraints can also be written as $I_{i,t-1,\omega} + \sum_{k \in K} q_{ikt\omega} + e_{it\omega} \leq L_i$. Note that this formulation becomes the vehicle index formulation for the deterministic PRP in Chapter 3 when the number of scenarios is equal to one and the extra delivery quantity is forced to be zero.

The formulation (5.1)-(5.15), together with the inequalities (3.36)-(3.37) and the valid vehicle symmetry breaking constraints SBC0 and SBC3 presented in Chapter 3, will be referred to as the basic formulation ($BF$).

A simple modification can be made to formulate the two-stage SIRP where the production quantity in each period is fixed. Denote by $B_t$ the production quantity made available in each period. One can set all the production setup variables $y_t$ to one ($y_t = \bar{y}_t = 1$). Constraints (5.6) and the parameters $M'_{it\omega}$ are replaced with $p_{t\omega} = B_t \bar{y}_t, \forall t \in T, \forall \omega \in \Omega$ and $\min\{L_i, Q\}$, respectively. The rest of the formulation remains unchanged.

## 5.3 Benders Decomposition Approaches

We now introduce exact algorithms based on the Benders decomposition scheme (Benders, 1962) to solve the two-stage SPRP. In Benders decomposition, the original problem is reformulated into a *master problem*, and a number of *subproblems* which are typically easier to solve than the original problem. By using linear programming duality, all the variables that belong to the subproblems are projected out and the master problem contains the remaining variables and an artificial variable representing a lower bound on the cost of the subproblem. In the original concept of Benders decomposition, the problem is solved by a cutting plane algorithm. At each iteration, the values of the master problem variables are first determined and

the subproblems are solved by holding these variables fixed. If the subproblems are feasible and bounded, an *optimality cut* is added to the master problem, otherwise a *feasibility cut* is added. An upper bound can be computed from the subproblems and a lower bound is obtained if the master problem is solved to optimality. The process continues until an optimal solution is found or the optimality gap is smaller than a given threshold value.

In the two-stage SPRP, we observe that the integer variables are the first-stage decisions while the continuous variables belong to the second-stage. If the decisions in the first stage are fixed, the resulting subproblem is a network flow problem which can be decomposed by scenario. This follows the original idea of applying Benders decomposition to stochastic integer programs, also known as the $L$-shaped method (see Van Slyke and Wets, 1969; Birge and Louveaux, 2011). In this section, we present two different Benders decomposition schemes. The first one projects out the flow variables of the second stage, i.e., production, inventory, delivery and unmet demand quantities, while the second one also projects out the routing variables.

### 5.3.1 Benders Reformulation 1 (BR1)

The first Benders reformulation is constructed by separating the first-stage and second-stage decisions into a master problem and subproblems, respectively. We let $\bar{\mathbf{x}}$, $\bar{\mathbf{y}}$ and $\bar{\mathbf{z}}$ denote the vectors of fixed $x_{ijkt}$, $y_t$ and $z_{ikt}$ variables, respectively. One can observe that the second-stage decisions are independent of $\bar{\mathbf{x}}$. The expected total cost of the second-stage decisions, denoted by $\upsilon(\bar{\mathbf{y}}, \bar{\mathbf{z}})$, can be calculated as $\upsilon(\bar{\mathbf{y}}, \bar{\mathbf{z}}) = \sum_{\omega \in \Omega} \rho_\omega \upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$, where $\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$ is the total second-stage cost of scenario $\omega$ which can itself be obtained by solving the following *primal flow subproblem (PFS)*:

$$\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}}) = \min \sum_{t \in T} \left( up_{t\omega} + \sum_{i \in N} h_i I_{it\omega} + \sum_{i \in N_c} \sigma_i e_{it\omega} \right) \tag{5.16}$$

s.t. (5.12) and

$$I_{0,t-1,\omega} + p_{t\omega} = \sum_{i \in N_c} \sum_{k \in K} q_{ikt\omega} + I_{0t\omega} \quad \forall t \in T \tag{5.17}$$

$$I_{i,t-1,\omega} + \sum_{k \in K} q_{ikt\omega} + e_{it\omega} = I_{it\omega} + d_{it\omega} \qquad \forall i \in N_c, \forall t \in T \tag{5.18}$$

$$I_{0t\omega} \leq L_0 \qquad \forall t \in T \tag{5.19}$$

$$I_{it\omega} + d_{it\omega} \leq L_i \qquad \forall i \in N_c, \forall t \in T \tag{5.20}$$

$$p_{t\omega} \leq M_{t\omega} \bar{y}_t \qquad \forall t \in T \tag{5.21}$$

$$\sum_{i \in N_c} q_{ikt\omega} \leq Q \bar{z}_{0kt} \qquad \forall k \in K, \forall t \in T \tag{5.22}$$

$$q_{ikt\omega} \leq M'_{it\omega} \bar{z}_{ikt} \qquad \forall i \in N_c, \forall k \in K, \forall t \in T. \tag{5.23}$$

Due to the presence of the variables $e_{it\omega}$, the PFS is always feasible because the demand can be left unmet. Furthermore, since the cost parameters $u$, $h_i$ and $\sigma_i$ are finite and due to constraints (5.17)-(5.20), any feasible solution of the PFS must be bounded. We let $\boldsymbol{\alpha} = (\alpha_{t\omega} | \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\beta} = (\beta_{it\omega} | \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\gamma} = (\gamma_{t\omega} \geq 0 | \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\theta} = (\theta_{it\omega} \geq 0 | \forall i \in N_c, \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\delta} = (\delta_{t\omega} \geq 0 | \forall t \in T, \forall \omega \in \Omega)$, $\boldsymbol{\kappa} = (\kappa_{kt\omega} \geq 0 | \forall k \in K, \forall t \in T, \forall \omega \in \Omega)$ and $\boldsymbol{\zeta} = (\zeta_{ikt\omega} \geq 0 | \forall i \in N_c, \forall k \in K, \forall t \in T, \forall \omega \in \Omega)$ be the vectors of the dual variables associated with constraints (5.17)-(5.23), respectively, and let also $\alpha_{l+1,\omega} = 0$ and $\beta_{i,l+1,\omega} = 0$. The dual of the primal subproblem for each scenario $\omega$, called the *dual flow subproblem (DFS)*, can be formulated as follows.

$$\begin{aligned} \upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}}) = \max \; & -I_{00}\alpha_{1\omega} + \sum_{i \in N_c} (d_{i1\omega} - I_{i0})\beta_{i1\omega} + \sum_{i \in N_c} \sum_{t=2}^{l} d_{it\omega}\beta_{it\omega} - \sum_{t \in T} L_0 \gamma_{t\omega} \\ & - \sum_{t \in T} \sum_{i \in N_c} (L_i - d_{it\omega})\theta_{it\omega} - \sum_{t \in T} M_{t\omega} \bar{y}_t \delta_{t\omega} - \sum_{t \in T} \sum_{k \in K} Q \bar{z}_{0kt} \kappa_{kt\omega} \\ & - \sum_{t \in T} \sum_{k \in K} \sum_{i \in N_c} M'_{it\omega} \bar{z}_{ikt} \zeta_{ikt\omega} \end{aligned} \tag{5.24}$$

s.t.

$$\alpha_{t\omega} - \delta_{t\omega} \leq u \qquad \forall t \in T \tag{5.25}$$

$$-\alpha_{t\omega} + \alpha_{t+1,\omega} - \gamma_{t\omega} \leq h_0 \qquad \forall t \in T \tag{5.26}$$

$$-\beta_{it\omega} + \beta_{i,t+1,\omega} - \theta_{it\omega} \leq h_i \qquad \forall i \in N_c, \forall t \in T \tag{5.27}$$

$$-\alpha_{t\omega} + \beta_{it\omega} - \kappa_{kt\omega} - \zeta_{ikt\omega} \leq 0 \qquad \forall i \in N_c, \forall k \in K, \forall t \in T \tag{5.28}$$

$$\beta_{it\omega} \leq \sigma_i \qquad \forall i \in N_c, \forall t \in T. \tag{5.29}$$

Let $\Delta_\omega$ denote the polyhedron defined by constraints (5.25)-(5.29). Since the PFS is always feasible and bounded, by strong duality, the DFS is feasible and bounded. We further let $\Delta = \bigcup_{\omega \in \Omega} \Delta_\omega$ and $P_\Delta$ be the set of extreme points of $\Delta$.

To formulate the *Benders master problem*, we define $\pi_\omega(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}) = -I_{00}\alpha_{1\omega} + \sum_{i \in N_c}(d_{i1\omega} - I_{i0})\beta_{i1\omega} + \sum_{i \in N_c}\sum_{t=2}^{l} d_{it\omega}\beta_{it\omega} - \sum_{t \in T} L_0\gamma_{t\omega} - \sum_{t \in T}\sum_{i \in N_c}(L_i - d_{it\omega})\theta_{it\omega}$ and we introduce an artificial variable $\eta$ representing the expected total flow cost. The original model (5.1)-(5.15) can be reformulated as follows.

$$\min \sum_{t \in T} \left( f y_t + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} x_{ijkt} \right) + \eta \tag{5.30}$$

s.t. (5.9)-(5.11), (5.13)-(5.15) and

$$\sum_{\omega \in \Omega} \rho_\omega \left( -\sum_{t \in T} M_{t\omega}\delta_{t\omega} y_t - \sum_{t \in T}\sum_{k \in K} Q\kappa_{kt\omega} z_{0kt} - \sum_{t \in T}\sum_{k \in K}\sum_{i \in N_c} M'_{it\omega}\zeta_{ikt\omega} z_{ikt} + \pi_\omega(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}) \right) \leq \eta \; \forall (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}, \boldsymbol{\delta}, \boldsymbol{\zeta}, \boldsymbol{\kappa}) \in P_\Delta. \tag{5.31}$$

This formulation, together with the valid inequalities (3.36)-(3.37), SBC0 and SBC3, will be referred to as BMP1.

Observe that BMP1 contains a large number of Benders cuts (5.31) as well as the SECs (5.11). Thus, a natural solution approach is to start from a relaxed BMP1 where these constraints are dropped. Next, violated constraints are detected and iteratively added to the problem. The solution approaches to handle this reformulation will be explained in Section 5.4.

### 5.3.2 Benders Reformulation 2 (BR2)

Due to the presence of the SECs (5.11) in BMP1, the Benders master problem cannot be solved by a standard branch-and-bound procedure as is done in a typical Benders

decomposition algorithm. One alternative is to reformulate the problem by keeping only the $y_t$ and $z_{ikt}$ variables in the master problem, which results in two separate Benders subproblems: the previous PFS plus the following subproblem associated with the routing variables $x_{ijkt}$:

$$r(\bar{\mathbf{z}}) = \min \sum_{t \in T} \left( \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} x_{ijkt} \right) \tag{5.32}$$

s.t. (5.14)-(5.15) and

$$\sum_{(j,j') \in \delta(i)} x_{jj'kt} = 2\bar{z}_{ikt} \qquad \forall i \in N, \forall k \in K, \forall t \in T \tag{5.33}$$

$$\sum_{(i,j) \in E(S)} x_{ijkt} \leq \sum_{i \in S} \bar{z}_{ikt} - \bar{z}_{ekt} \quad \forall S \subseteq N_c : |S| \geq 2, \forall e \in S, \forall k \in K, \forall t \in T. \tag{5.34}$$

This subproblem can be separated by vehicle and time period, and reduces to independent TSPs. The number of routing subproblems is then equal to the number of vehicles dispatched during the planning horizon. Denote by $N_c^{kt}(\bar{\mathbf{z}})$ the set of customers visited by vehicle $k$ in period $t$ corresponding to the fixed decision vector $\bar{\mathbf{z}}$, i.e., $N_c^{kt}(\bar{\mathbf{z}}) = \{i | \bar{z}_{ikt} = 1\}$ and $N^{kt}(\bar{\mathbf{z}}) = N_c^{kt}(\bar{\mathbf{z}}) \cup \{0\}$. This subproblem can be rewritten as $r(\bar{\mathbf{z}}) = \sum_{t \in T} \sum_{k \in K} r_{kt}(\bar{\mathbf{z}})$, where $r_{kt}(\bar{\mathbf{z}})$ is the total routing cost of vehicle $k$ in period $t$, which can be determined by solving the following problem:

$$r_{kt}(\bar{\mathbf{z}}) = \min \sum_{(i,j) \in E} c_{ij} x_{ijkt} \tag{5.35}$$

s.t. (5.14)-(5.15) and

$$\sum_{(j,j') \in \delta(i)} x_{jj'kt} = 2\bar{z}_{ikt} \qquad \forall i \in N \tag{5.36}$$

$$\sum_{(i,j) \in E(S^{kt})} x_{ijkt} \leq |S^{kt}| - 1 \qquad \forall S^{kt} \subseteq N_c^{kt}(\bar{\mathbf{z}}) : |S^{kt}| \geq 2. \tag{5.37}$$

To dualize this model, we relax the integrality constraints (5.14) and (5.15) by replacing them with the following constraints:

$$0 \leq x_{ijkt} \leq 1 \qquad \forall (i,j) \in E : i \neq 0 \tag{5.38}$$

$$0 \leq x_{0jkt} \leq 2 \qquad \forall j \in N_c. \tag{5.39}$$

Model (5.35) and (5.36)-(5.39) will be referred to as the *primal routing subproblem (PRS)*.

Denote by $\boldsymbol{\varphi} = (\varphi_{ikt} | \forall i \in N, \forall k \in K, \forall t \in T)$, $\boldsymbol{\vartheta} = (\vartheta_{S^{kt}} \geq 0 | \forall k \in K, \forall t \in T, \forall S^{kt} \subseteq N_c^{kt}(\bar{\mathbf{z}}) : |S^{kt}| \geq 2)$, $\boldsymbol{\mu} = (\mu_{ijkt} \geq 0 | \forall (i,j) \in E : i \neq 0, \forall k \in K, \forall t \in T)$ and $\boldsymbol{\mu}^0 = (\mu^0_{jkt} \geq 0 | \forall j \in N_c, \forall k \in K, \forall t \in T)$ the vectors of the dual variables associated with constraints (5.36)-(5.39), respectively. The *dual routing subproblem (DRS)* can be formulated as follows.

$$r_{kt}(\bar{\mathbf{z}}) = \max \sum_{i \in N} 2\bar{z}_{ikt}\varphi_{ikt} - \sum_{S^{kt} \subseteq N_c^{kt}(\bar{\mathbf{z}}) : |S^{kt}| \geq 2} (|S^{kt}| - 1)\vartheta_{S^{kt}} - \sum_{(i,j) \in E : i \neq 0} \mu_{ijkt} - 2 \sum_{j \in N_c} \mu^0_{jkt} \tag{5.40}$$

$$\text{s.t.} \qquad \varphi_{ikt} + \varphi_{jkt} - \sum_{S^{kt} : i,j \in S^{kt}} \vartheta_{S^{kt}} - \mu_{ijkt} \leq c_{ij} \qquad (i,j) \in E : i \neq 0 \tag{5.41}$$

$$\varphi_{ikt} + \varphi_{jkt} - \sum_{S^{kt} : i,j \in S^{kt}} \vartheta_{S^{kt}} - \mu^0_{jkt} \leq c_{0j} \qquad (i,j) \in E : i = 0. \tag{5.42}$$

Denote by $\boldsymbol{\Theta}_{kt}$ the polyhedron defined by constraints (5.41)-(5.42), (5.36)-(5.39). Since the cost $c_{ij}$ is finite, both the PRS and DRS are feasible and bounded. Let also $\boldsymbol{\Theta} = \bigcup_{k \in K, t \in T} \boldsymbol{\Theta}_{kt}$ and $P_{\boldsymbol{\Theta}}$ be the set of extreme points of $\boldsymbol{\Theta}$. To formulate the Benders master problem, we introduce an artificial variable $\tau$ representing the total routing cost and we define $\psi_{kt}(\boldsymbol{\vartheta}, \boldsymbol{\mu}, \boldsymbol{\mu^0}) = -\sum_{S^{kt} \subseteq N_c^{kt}(\bar{\mathbf{z}}) : |S^{kt}| \geq 2} (|S^{kt}| - 1)\vartheta_{S^{kt}} - \sum_{(i,j) \in E : i \neq 0} \mu_{ijkt} -$

$2 \sum_{j \in N_c} \mu^0_{jkt}$. The following Benders master problem is obtained:

$$\min \sum_{t \in T} f y_t + \eta + \tau \tag{5.43}$$

s.t. (5.9), (5.13), (5.31) and

$$\sum_{t \in T} \sum_{k \in K} \left( \sum_{i \in N} 2 z_{ikt} \varphi_{ikt} + \psi_{kt}(\boldsymbol{\vartheta}, \boldsymbol{\mu}, \boldsymbol{\mu^0}) \right) \leq \tau \qquad \forall (\boldsymbol{\varphi}, \boldsymbol{\vartheta}, \boldsymbol{\mu}, \boldsymbol{\mu^0}) \in P_{\boldsymbol{\Theta}}. \tag{5.44}$$

This problem with the valid inequalities (3.19), SBC0 and SBC3 will be referred to as BMP2. Note that BMP2 is equivalent to a relaxation of the original problem $BF$ in which integrality is not imposed on the $x_{ijkt}$ variables. In Section 5.4.1, we explain how an upper bound to the original formulation can be obtained in the classical Benders algorithm when solving BR2. In Section 5.4.2, we further explain how an optimal integer solution to the original problem can be guaranteed by the branch-and-Benders-cut implementation with an embedded enumeration algorithm.

Both BR1 and BR2 can be easily modified to handle the two-stage SIRP as follows. First, all the production setup variables $y_t$ in the BMP1 and BMP2 must be set to one, i.e., $y_t = \bar{y}_t = 1$. Second, the production quantity variables $p_{t\omega}$ and the parameters $M'_{it\omega}$ are substituted by $p_{t\omega} = B_t \bar{y}_t, \forall t \in T, \forall \omega \in \Omega$ and $\min\{L_i, Q\}$, respectively. Finally, constraints (5.21) and the dual variables $\delta_{t\omega}$ are removed from the PFS and DFS, respectively. The rest of the formulation remains unchanged.

## 5.4 Benders Decomposition Algorithms

In this section, we describe the algorithms we have developed to handle the Benders reformulations BR1 and BR2. We first present the classical algorithm and then introduce a Benders decomposition algorithm that is embedded in a branch-and-cut framework to solve the SPRP.

### 5.4.1 Classical Benders Decomposition Algorithm

In classical Benders decomposition, the master problem is solved to optimality at each iteration to obtain a lower bound on the optimal objective function value of the problem. Then, the subproblem is solved to generate a Benders cut and compute an upper bound. This cut is added to the master problem and the process is repeated until an optimal solution is found. In BR1, since the master problem BMP1 contains the SECs, it must be solved by a branch-and-cut process. At each node of the branch-and-bound tree, a so-called *separation algorithm* is applied to detect violated SECs and these cuts are then added to the problem. As in the algorithm in Chapter 3, we use the minimum $s-t$ algorithm of the Concorde callable library (Applegate et al., 2011) as the exact separation algorithm. For BR2, since the routing part is handled by the PRS with the relaxed integrality constraints on the arc variables, the solution can be fractional. In this case, even when the $y_t$ and $z_{ikt}$ variables in the BMP2 take integer values, the value of $f\bar{y}_t$ plus the cost of the subproblems does not provide a valid upper bound on the original problem $BF$ because of the fractional $x_{ijkt}$ variables. However, an upper bound can be computed by solving the integer PRS either exactly or with a heuristic. We use the following heuristic to compute a valid upper bound when a fractional solution is obtained. TSP tours are constructed by the *GENIUS* procedure (Gendreau et al., 1992) and improved by *3-opt* routes (Lin, 1965). This process is adequate since our experiments have shown that fractional solutions rarely occur and the gap between the heuristic solution and the optimal solution to the integer PRS is usually negligible because the number of customers in each tour is small. Note that even if we solve the integer PRS to optimality, a gap may still exist between the lower bound of the BMP2 and the best upper bound found by solving the integer PRS. If one wishes to guarantee an optimal solution using BR2, one can instead embed the Benders decomposition algorithm in a branch-and-cut algorithm as we will discuss in detail in the next section.

Denote by $P'$ the set of extreme points obtained so far by solving the Benders subproblems, i.e., DFS for BMP1, and DFS and PRS for BMP2. Note that BMP is used to represent the master problem (both BMP1 and BMP2). We define $Z^b$ as the set of integer and binary variable values obtained by solving the BMP (consisting of $x, y, z$

for BMP1 and $y, z$ for BMP2). Let $BMP(P')$ be the Benders master problem with the Benders cuts generated from $P'$ and $SP(Z^b)$ be the Benders subproblems given the set of fixed integer and binary variables $Z^b$. Let also $v(BMP(P'))$, $v(SP(Z^b))$ and $\xi(Z^b)$ denote the objective function value of the Benders master problem, the objective function value of the Benders subproblems, and the cost associated with the fixed variables in the set $Z^b$, respectively. We further define $\tilde{v}(SP(Z^b))$ as the objective function value of the heuristic solution if the solution of the PRS is fractional in BR2, while we simply set $\tilde{v}(SP(Z^b)) = v(SP(Z^b))$ for BR1 and also for BR2 if the solution of the PRS is not fractional. Note that the algorithm stops when an optimal solution is found or a maximum CPU time is reached. The original Benders decomposition (BD) algorithm is shown in Algorithm 5.1.

**Algorithm 5.1** Original Benders Decomposition Algorithm (BD)

$ub, gap \leftarrow \infty, lb \leftarrow -\infty$ and $b \leftarrow 0$
$P' \leftarrow \varnothing$
**while** stopping criterion not satisfied **do**
  Solve $BMP(P')$ to obtain $Z^b$
  **if** $v(BMP(P')) > lb$ **then** $lb \leftarrow v(BMP(P'))$
  Solve $SP(Z^b)$ to obtain extreme points, generate Benders cuts and add to $BMP(P')$
  (for BMP2) Solve the TSP heuristic if $x$ is fractional
  **if** $\xi(Z^b) + \tilde{v}(SP(Z^b)) < ub$ **then** $ub \leftarrow \xi(Z^b) + \tilde{v}(SP(Z^b))$
  $gap \leftarrow (ub - lb)/ub$
  $b \leftarrow b + 1$
**end while**

When applying a Benders decomposition technique to a mixed-integer problem, one typically retains the integer variables in the BMP and solves the master problem from scratch by a branch-and-bound procedure at each iteration. This process may be highly inefficient. Furthermore, for BR1 where the master problem also contains the SECs, a branch-and-cut approach must be employed and the problem can be much more difficult to solve. To overcome these difficulties, we propose an alternative implementation of the Benders decomposition.

Table 5.I: Benders Cut Generation Strategies of the Benders Algorithms

| **Stage** | **Benders Reformulation** | | | |
|---|---|---|---|---|
| | **BR1** | | **BR2** | |
| | **BD** | **BBC** | **BD** | **BBC** |
| Any B&B Node | S | S | - | - |
| Integer | - | F, S | - | F,R |
| Optimal | F | n/a | F,R | n/a |
| **No. of B&B Trees** | $b$ | 1 | $b$ | 1 |

S - SECs
F - Benders flow cuts
R - Benders route cuts
$b$ - The number of iterations in Benders algorithm

### 5.4.2 Branch-and-Cut Based Benders Algorithm

Since Benders cuts generated from any solution of the master problem (including the linear relaxation) are valid (McDaniel and Devine, 1977), one can generate Benders cuts at any node of the branch-and-bound tree of the Benders master problem. This way, Benders decomposition can be embedded in a standard branch-and-cut framework where the Benders subproblems are solved and the generated Benders cuts are added to the master problem at any node of the branch-and-bound tree of the master problem. We refer to this approach as branch-and-Benders-cut (BBC). Other implementations of Benders decomposition in a branch-and-cut framework were discussed by Codato and Fischetti (2006) and Fortz and Poss (2009) to deal with feasibility cuts, as well as by Naoum-Sawaya and Elhedhli (2010) and de Camargo et al. (2011) who employed an interior point method and an outer-approximation method to solve the Benders reformulations, respectively. In our algorithm, to avoid generating a large number of Benders cuts, the cuts are added to the branch-and-bound tree of the master problem only when an integer solution of the BMP, i.e., a solution where all the integrality constraints and the current Benders cuts of the BMP are satisfied, is found. The process is described in Algorithm 2. A comparison of the cut generation strategy of each algorithm is shown in Table 5.I. It describes the stage at which each type of cuts is generated during the branch-and-bound process when solving the Benders master problem. Note that the Benders flow cuts (F) are generated by solving the DFS and the Benders route cuts (R) are generated by solving the PRS in our implementation.

**Algorithm 5.2** Branch-and-Benders-Cut (BBC) Algorithm

$ub, gap \leftarrow \infty, lb \leftarrow -\infty$ and $b \leftarrow 0$
$P' \leftarrow \varnothing$
**begin** solving $BMP(P')$ by a branch-and-bound and applying the following steps at each node of the branch-and-bound tree
  $lb \leftarrow$ the best overall lower bound of $BMP(P')$
  (for BMP1) Solve the separation algorithm and add SECs to the $BMP(P')$
  **if** stopping criterion not satisfied **and** an integer solution found **then**
    Set $Z^b$ using the integer solution
    Solve $SP(Z^b)$ to obtain extreme points, generate Benders cuts and add to $BMP(P')$
    (for BMP2) Solve the TSP heuristic if $x$ is fractional
    **if** $\xi(Z^b) + \tilde{v}(SP(Z^b)) < ub$ **then** $ub \leftarrow \xi(Z^b) + \tilde{v}(SP(Z^b))$
    $gap \leftarrow (ub - lb)/ub$
    $b \leftarrow b + 1$
  **endif**
**end**

As shown in Table 5.I, in BD, the Benders cuts are only generated and added when an optimal solution of the Benders master problem is found and the master problem is then solved from scratch with a new branch-and-bound tree in the next iteration. We thus explore $b$ branch-and-bound trees for the master problem where $b$ is the number of iterations in the Benders algorithm. In the BBC, however, a single branch-and-bound tree for the master problem is explored during the whole process. The SECs are added at each node. Each time a new integer solution of the BMP is found, the subproblems resulting from this integer solution are solved and Benders cuts are added. The branch-and-bound process continues until it reaches a stopping criterion. Both upper and lower bounds are computed at the nodes of the tree and an optimal solution to the original formulation $BF$ is found when the BMP has been solved to optimality. This process follows the branch-and-cut framework.

The fact that one tree is explored in the BBC algorithm is also particularly useful for BR2 when an integrality gap on the routing part exists and one must find an optimal solution for the original problem. This can be done by adding a few steps to the BBC. During the solution process, when a fractional PRS solution is found,

the set $Z^b$ and the value $\xi(Z^b) + v(SP(Z^b))$ must be stored. This value is a lower bound on the solution cost corresponding to the set $Z^b$ for the original formulation and therefore the solution can be pruned later if this value exceeds the value of the current best overall feasible solution ($ub$). At the end of the process, one has a current best solution for the original problem and a (possibly empty) set of fractional solutions including the integer solution of the BMP2 if it is fractional. Then, all the remaining fractional solutions must be evaluated again by solving to optimality the PRS with the imposed integrality constraints (5.14) and (5.15). The optimal solution is then the best solution among all these solutions. A numerical example of the BBC algorithm is provided in the Appendix C.

We also remark that this Benders cut generation strategy is different from those of Naoum-Sawaya and Elhedhli (2010) and de Camargo et al. (2011) since the Benders cuts are added only when an integer solution of the BMP is found in our BBC algorithm, while Naoum-Sawaya and Elhedhli (2010) and de Camargo et al. (2011) generate cuts at each node of the tree using a relaxed BMP solution. The latter strategy, however, appears to be inefficient for our problem as we show in the computational results reported in Section 5.5.1.

### 5.4.3 Computational Enhancements

#### 5.4.3.1 Lower Bound Lifting Inequalities

Since parts of the objective function (5.1) are projected out in the Benders reformulations, the optimality gap may be large in the initial stage of the algorithm due to the low quality of the lower bound. A large number of Benders cuts are thus needed to close the gap. To address this issue, we can lift the lower bound of the Benders master problem by using initial cuts, called lower bound lifting inequalities (LBL), that contain some information about the parts of the original objective function that are removed. First, we can add cuts to represent a lower bound on the flow costs, i.e., unit production, inventory and penalty costs. We observe that, for the periods between two consecutive deliveries to a customer, the minimum flow cost can be calculated by considering the quantity that must be supplied between the two visits

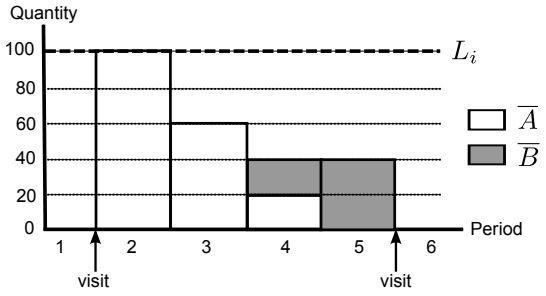



Figure 5.1: Inventory Level Corresponding to the Two Consecutive Visits in Periods 2 and 6

to satisfy the demand. Figure 5.1 illustrates the inventory level when customer $i$ is visited in period 2 and then in period 6 while the demand in each period is equal to 40 so that the minimum total quantity required to satisfy the demand between these periods is equal to 160. Given the maximum inventory capacity $L_i = 100$, this total demand quantity can be separated into two parts, i.e., the amount $\overline{A} = 100$ under $L_i$ which can be supplied and can be also left unmet if the cost of this is lower, and the amount $\overline{B} = 60$ which cannot be supplied and has to be left unmet due to the inventory capacity limit.

We define the periods 0 and $l+1$ as dummy periods at the beginning and the end of the planning horizon (used for calculation purposes) and $d_{i0\omega} = d_{i,l+1,\omega} = 0, \forall i \in N_c, \forall \omega \in \Omega$. Let $\lambda_{ivt}$ be a binary variable equal to one if customer $i$ is visited in period $v < t$ and the next visit is in period $t$, and the parameter $\phi_{ivt}$ be the minimum possible sum of unit production, inventory and penalty costs over the period $v$ to $t-1$ associated with the variable $\lambda_{ivt}$, calculated as,

$$\phi_{ivt} = \sum_{\omega \in \Omega} \rho_\omega \left( \sum_{s=v}^{t-1} \min \left\{ H^P_{ivs\omega}, H^\sigma_{ivs\omega} \right\} + c^h_{ivt} + \sigma_i \left( \sum_{s=v}^{t-1} d_{is\omega} - \varphi_{iv} \right)^+ \right)$$

where

$$H^P_{ivs\omega} = \begin{cases} (h_i(s-1)+u) \min \left\{ d_{is\omega}, \left( \varphi_{i0} - \sum_{w=1}^{s-1} d_{iw\omega} \right)^+ \right\} & \text{if } v = 0 \\ (h_i(s-v)+u) \min \left\{ d_{is\omega}, \left( \varphi_{iv} - \sum_{w=v}^{s-1} d_{iw\omega} \right)^+ \right\} & \text{if } v > 0, \end{cases}$$

$$H^{\sigma}_{ivs\omega} = \sigma_i \min\left\{ d_{is\omega}, \left( \varphi_{iv} - \sum_{w=v}^{s-1} d_{iw\omega} \right)^+ \right\},$$

$$\varphi_{iv} = \begin{cases} I_{i0} & \text{if } v = 0 \\ L_i & \text{if } 0 < v < t \leq l+1 \end{cases}, \text{ and}$$

$$c^h_{ivt} = \begin{cases} h_i(t-1)\left(I_{i0} - \sum_{w=1}^{t-1} d_{iw\omega}\right)^+ & \text{if } v = 0 \\ h_i(t-v)\left(I_{i0} - \sum_{w=1}^{t-1} d_{iw\omega}\right)^+ & \text{if } 0 < v < t \leq l+1. \end{cases}$$

Note that $c^h_{ivt}$ is the inventory cost at customer $i$ incurred over the period $v$ to $t-1$ for the part of the initial inventory that is not used up at the end of period $t-1$. The following cuts can be added to the master problem:

$$\sum_{t=1}^{l+1} \sum_{v=0}^{t-1} \sum_{i \in N_c} \phi_{ivt} \lambda_{ivt} - u \sum_{i \in N_c} I_{i0} \leq \eta \tag{5.45}$$

$$\sum_{v=0}^{t-1} \lambda_{ivt} = \sum_{k \in K} z_{ikt} \qquad \forall i \in N_c, \forall t \in T \tag{5.46}$$

$$\sum_{v=0}^{t-1} \sum_{s=t}^{l+1} \lambda_{ivs} = 1 \qquad \forall i \in N_c, \forall t \in T \cup \{l+1\} \tag{5.47}$$

$$\lambda_{ivt} \in \{0,1\} \qquad \forall i \in N_c, \forall 0 \leq v < t \leq l+1. \tag{5.48}$$

Constraint (5.45) provides a lower bound for the flow cost. Constraints (5.46) link the $\lambda_{ivt}$ and $z_{ikt}$ variables and constraints (5.47) enforce that one replenishment plan $\lambda_{ivt}$ be selected in each period. Although the new variables $\lambda_{ivt}$ are added to the formulation, one can observe that when all the $z_{ikt}$ variables are fixed, the variables $\lambda_{ivt}$ can be easily set by inspection. These constraints can be added to both formulations BMP1 and BMP2.

For BR2, where the routing cost is also projected out, we let $\chi_{ijkt}$ be additional routing variables. The following inequalities can be added to the BMP2:

$$\sum_{(i,j) \in E} \sum_{k \in K} c_{ij} \chi_{ijkt} \leq \tau \tag{5.49}$$

$$\sum_{(j,j')\in\delta(i)} \chi_{jj'kt} = 2z_{ikt} \qquad \forall i \in N, \forall k \in K, \forall t \in T \tag{5.50}$$

$$\chi_{ijkt} \in \{0,1\} \qquad \forall (i,j) \in E : i \neq 0, \forall k \in K, \forall t \in T \tag{5.51}$$

$$\chi_{0jkt} \in \{0,1,2\} \qquad \forall j \in N_c, \forall k \in K, \forall t \in T. \tag{5.52}$$

#### 5.4.3.2 Scenario Group Cuts

In BR1 and BR2, the Benders subproblem is decomposed into many subproblems. Many Benders cuts, one for each subproblem, can be added at once instead of just one Benders cut to accelerate the convergence (Birge and Louveaux, 1988). However, adding too many cuts at each iteration can lead to a decreased performance of the Benders algorithm (de Camargo et al., 2008). In the SPRP, where the number of subproblems is equal to the number of scenarios, the number of Benders cuts generated at each iteration can be very large. To overcome this issue, we can instead create groups of scenarios using some similarity and aggregate the Benders cuts in each group to reduce the number of cuts added per iteration. To create scenario groups, we first define the number of scenario groups $n_G \leq |\Omega|$ and groups of scenarios $G(g)$, indexed by $g$. Then, the range between the maximum and minimum total demand among all scenarios is calculated and separated equally into $n_G$ groups. For each scenario, if the total demand falls into the total demand range of a group, the scenario is then assigned to this group. Denote by $\eta_g$ the expected total flow cost corresponding to group $g$. The variable $\eta$ in the objective function (5.30) and (5.43) is replaced by $\sum_{g=1}^{n_G} \eta_g$ and constraints (5.31) are replaced by the following constraints:

$$\sum_{\omega\in G(g)} \rho_\omega \left( -\sum_{t\in T} M_{t\omega}\delta_{t\omega} y_t - \sum_{t\in T}\sum_{k\in K} Q\kappa_{kt\omega} z_{0kt} - \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} M'_{it\omega}\zeta_{ikt\omega} z_{ikt} \right.$$
$$\left. + \pi_\omega(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}) \right) \leq \eta_g \ \forall 1 \leq g \leq n_G, \forall (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}, \boldsymbol{\delta}, \boldsymbol{\zeta}, \boldsymbol{\kappa}) \in P_\Delta. \tag{5.53}$$

In this way, we can control the number of Benders cuts added at each iteration. For the formulation BMP2, where the routing subproblems have to be solved, we can

also add the cut for each vehicle in each period separately. Let $\tau_{kt}$ be the routing cost of vehicle $k$ in period $t$. The variable $\tau$ in the objective function (5.43) is replaced with $\sum_{t\in T}\sum_{k\in K}\tau_{kt}$ and constraints (5.44) are replaced with

$$\left(\sum_{i\in N} 2z_{ikt}\varphi_{ikt} + \psi_{kt}(\boldsymbol{\vartheta},\boldsymbol{\mu},\boldsymbol{\mu^0})\right) \leq \tau_{kt} \quad \forall k\in K, \forall t\in T, \forall(\boldsymbol{\varphi},\boldsymbol{\vartheta},\boldsymbol{\mu},\boldsymbol{\mu^0})\in P_{\boldsymbol{\Theta}}. \quad (5.54)$$

#### 5.4.3.3 Pareto-Optimal Cuts

The performance of a Benders decomposition approach depends largely on the quality of the cuts. Magnanti and Wong (1981) introduced the concept of non-dominated cuts, called *Pareto-optimal cuts*. Let $\mathbf{Y}$ and $\mathbf{Z}$ be the sets of vectors associated with the $y_t$ and $z_{ikt}$ variables, respectively. For the flow cut obtained by the DFS corresponding to the scenario $\omega$, the cut generated from the extreme point $(\boldsymbol{\alpha}^1,\boldsymbol{\beta}^1,\boldsymbol{\gamma}^1,\boldsymbol{\theta}^1,\boldsymbol{\delta}^1,\boldsymbol{\zeta}^1,\boldsymbol{\kappa}^1)$ dominates the cut generated from the extreme point $(\boldsymbol{\alpha}^2,\boldsymbol{\beta}^2,\boldsymbol{\gamma}^2,\boldsymbol{\theta}^2,\boldsymbol{\delta}^2,\boldsymbol{\zeta}^2,\boldsymbol{\kappa}^2)$ if and only if

$$\begin{aligned}-\sum_{t\in T} M_{t\omega}\delta^1_{t\omega}y_t - \sum_{t\in T}\sum_{k\in K} Q\kappa^1_{kt\omega}z_{0kt} - \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} M'_{it\omega}\zeta^1_{ikt\omega}z_{ikt} + \pi_\omega(\boldsymbol{\alpha}^1,\boldsymbol{\beta}^1,\boldsymbol{\gamma}^1,\boldsymbol{\theta}^1) \geq \\ -\sum_{t\in T} M_{t\omega}\delta^2_{t\omega}y_t - \sum_{t\in T}\sum_{k\in K} Q\kappa^2_{kt\omega}z_{0kt} - \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} M'_{it\omega}\zeta^2_{ikt\omega}z_{ikt} + \pi_\omega(\boldsymbol{\alpha}^2,\boldsymbol{\beta}^2,\boldsymbol{\gamma}^2,\boldsymbol{\theta}^2) \\ \forall \mathbf{y}\in\mathbf{Y}, \forall \mathbf{z}\in\mathbf{Z},\end{aligned}$$

with strict inequality for at least one point. Denote by $\mathbf{Y}^{LP}$ the polyhedron defined by $0 \leq y_t \leq 1, \forall t\in T$ and $\mathbf{Z}^{LP}$ the polyhedron defined by constraints (5.9) and $0\leq z_{ikt}\leq 1, \forall i\in N, \forall k\in K, \forall t\in T$. Let $ri(\mathbf{Y}^{LP})$ and $ri(\mathbf{Z}^{LP})$ be the relative interior of $\mathbf{Y}^{LP}$ and $\mathbf{Z}^{LP}$, respectively. A Pareto-optimal cut can be computed by using the dual solution obtained by solving the following subproblem, where $\mathbf{y}^0\in ri(\mathbf{Y}^{LP})$ and $\mathbf{z}^0\in ri(\mathbf{Z}^{LP})$:

$$\max \; -I_{00}\alpha_{1\omega} + \sum_{i\in N_c}(d_{i1\omega} - I_{i0})\beta_{i1\omega} + \sum_{i\in N_c}\sum_{t=2}^{l} d_{it\omega}\beta_{it} - \sum_{t\in T} L_0\gamma_{t\omega}$$
$$- \sum_{t\in T}\sum_{i\in N_c}(L_i - d_{it})\theta_{it\omega} - \sum_{t\in T} M_{t\omega}y_t^0\delta_{t\omega} - \sum_{t\in T}\sum_{k\in K} Qz_{0kt}^0\kappa_{kt\omega}$$
$$- \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} \overline{M}_{it\omega}z_{ikt}^0\zeta_{ikt\omega} \tag{5.55}$$

$$\text{s.t.} \; -I_{00}\alpha_{1\omega} + \sum_{i\in N_c}(d_{i1\omega} - I_{i0})\beta_{i1\omega} + \sum_{i\in N_c}\sum_{t=2}^{l} d_{it\omega}\beta_{it} - \sum_{t\in T} L_0\gamma_{t\omega}$$
$$- \sum_{t\in T}\sum_{i\in N_c}(L_i - d_{it})\theta_{it\omega} - \sum_{t\in T} M_{t\omega}\bar{y}_t\delta_{t\omega} - \sum_{t\in T}\sum_{k\in K} Q\bar{z}_{0kt}\kappa_{kt\omega}$$
$$- \sum_{t\in T}\sum_{k\in K}\sum_{i\in N_c} \overline{M}_{it\omega}\bar{z}_{ikt}\zeta_{ikt\omega} = \upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}}) \tag{5.56}$$
$$(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\theta}, \boldsymbol{\delta}, \boldsymbol{\zeta}, \boldsymbol{\kappa}) \in \Delta_\omega. \tag{5.57}$$

Constraints (5.56) and (5.57) ensure that we select a feasible dual solution that was optimal for the original DFS objective function $\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$. To generate a Pareto-optimal cut, any core point $(y^0, z^0)$, where $\mathbf{y}^0 \in ri(\mathbf{Y}^{LP})$ and $\mathbf{z}^0 \in ri(\mathbf{Z}^{LP})$, can be used (Magnanti and Wong, 1981). We employ a method similar to that of Cordeau et al. (2001) to ensure that the core point lies within the relative interior of the master problem polyhedron. The details of this method are provided in the Appendix C.

The generation of Pareto-optimal cuts usually improves the convergence of the algorithm but requires solving two different linear programs sequentially for each subproblem, i.e., first the DFS to find $\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$, and then the problem (5.55)-(5.57). We expedite the process by using a network flow algorithm to determine the value $\upsilon_\omega(\bar{\mathbf{y}}, \bar{\mathbf{z}})$.

## 5.5 Computational Experiments

We have performed experiments using the PRP instances introduced in Chapter 3, which were themselves generated from the instances of Archetti et al. (2011). The test set consists of instances with $n = 10$, 15, 20, 25 and 30 customers and $l = 3$ and 6 periods. There are four instances per instance size and each of them has different characteristics. More details on the instances are provided in the Appendix. We generate scenarios by a Monte-Carlo simulation in which the demand in each period is independent and varies in the range $[\bar{d}_{it}(1-\epsilon), \bar{d}_{it}(1+\epsilon)]$, where $\bar{d}_{it}$ is the demand of the nominal case from the original test set. Unless stated otherwise, we assume that the demands are uniformly distributed and the value of $\epsilon$ is set to 0.1 and 0.25, which represents a maximum demand variation of 10% and 25%, respectively. We set the cost of unmet demand $\sigma_i$ proportional to the production and transportation costs, calculated as $\sigma_i = u + 5\,[f/C + 2c_{0i}/Q]$, where $[x]$ denotes the nearest integer to $x$, and the multiplier 5 is used to ensure that the cost of unmet demand is sufficiently large. The vehicle capacity is calculated as $Q = \left\lfloor 1.5 \max_{i \in N_c}\{\bar{L}_i\}/m \right\rfloor$ where $\bar{L}_i$ is the original value in the Archetti et al. test set. The sets $\mathcal{S}$ and $\mathcal{L}$ are used to represent the instance sizes $n = 10$, 15 and $n = 20$, 25, 30, respectively. Columns *Gap*, *CPU* and *B.Cuts* show the average optimality gap (%), the average CPU time in seconds and the average number of generated Benders cuts, respectively. To indicate the best results, boldface letters are put on the smallest average computing time if all instances of a problem size are solved to optimality, otherwise they are put on the best average optimality gap. The experiments were conducted on a workstation with an Intel Xeon 2.67GHz processor and 24GB of RAM under Scientific Linux 6.1 using CPLEX 12.3. The algorithms were coded in C and C# on MonoDevelop 3.0 under OpenSUSE Linux 12.1. In all experiments, branching priority was given first to $y$ variables and then to $z$ and $x$ (for BR1) variables, respectively.

### 5.5.1 Performance of the Branch-and-Benders-Cut

This section explores the performance of the BBC algorithm and the effect of the computational enhancements. We chose the test set with $l = 3$ for these experiments.

The number of vehicles is set to one and the number of scenarios is set to 100 and 500. The maximum computing time is set to two hours for each instance.

#### 5.5.1.1 Comparison Between the Original Benders Algorithm and the Branch-and-Benders-Cut Algorithm

We first examine the performance of the BD and BBC algorithms for BR1 and BR2. For the BBC, we also explore the performance of the following different cut generation strategies:

***Every*** Benders cuts are generated at every node in the branch-and-bound tree;

***Root/Integer*** Benders cuts are only generated at the root node and when an integer solution of the BMP is found;

***Integer*** Benders cuts are generated only when an integer solution of the BMP is found.

Note that the *Every* strategy was used by Naoum-Sawaya and Elhedhli (2010) and de Camargo et al. (2011) while the *Integer* is the strategy we proposed in Algorithm 5.2. The *Root/Integer* can be seen as a combination of the previous two strategies. Tables 5.II and 5.III provide the average results for each instance set for BR1 and BR2, respectively.

Table 5.II: Comparison of the BD and BBC Algorithms on BR1

| Set | $\|\Omega\|$ | $\epsilon$ | BD | | | | BBC | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | *Every* | | | | *Root/Integer* | | | | *Integer* | | | |
| | | | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts |
| $\mathcal{S}$ | 100 | 0.10 | 8/8 | 0.0 | 1555.4 | 112.3 | 7/8 | 0.0 | 1748.6 | 9995.0 | 8/8 | 0.0 | **110.8** | 263.5 | 8/8 | 0.0 | 114.5 | 252.3 |
| | | 0.25 | 7/8 | 0.0 | 3062.3 | 178.6 | 7/8 | 0.0 | 1879.8 | 10642.9 | 8/8 | 0.0 | **106.3** | 360.4 | 8/8 | 0.0 | 203.5 | 328.5 |
| | 500 | 0.10 | 8/8 | 0.0 | 1524.4 | 114.0 | 6/8 | 0.8 | 3189.8 | 3900.9 | 8/8 | 0.0 | **225.7** | 271.9 | 8/8 | 0.0 | 302.4 | 250.3 |
| | | 0.25 | 7/8 | 0.2 | 2816.4 | 144.4 | 4/8 | 1.6 | 3901.8 | 5015.9 | 8/8 | 0.0 | 316.8 | 324.5 | 8/8 | 0.0 | **287.8** | 268.6 |
| Average | | | 30/32 | 0.0 | 2239.6 | 137.3 | 24/32 | 0.6 | 2680.0 | 7388.7 | 32/32 | 0.0 | **189.9** | 305.1 | 32/32 | 0.0 | 227.1 | 274.9 |
| $\mathcal{L}$ | 100 | 0.10 | 0/12 | 7.4 | 7200.0 | 79.7 | 0/12 | 7.2 | 7200.0 | 34401.6 | 0/12 | 4.8 | 7200.0 | 3392.3 | 0/12 | **4.7** | 7200.0 | 2181.7 |
| | | 0.25 | 0/12 | 7.3 | 7200.0 | 83.4 | 0/12 | 7.7 | 7200.0 | 34209.6 | 0/12 | 5.7 | 7200.0 | 4022.5 | 0/12 | **4.8** | 7200.0 | 2435.7 |
| | 500 | 0.10 | 0/12 | 7.6 | 7200.0 | 76.7 | 0/12 | 9.9 | 7200.0 | 10267.9 | 1/12 | **4.8** | 7131.8 | 2916.0 | 0/12 | 5.1 | 7200.0 | 2001.0 |
| | | 0.25 | 0/12 | 8.7 | 7200.0 | 88.3 | 0/12 | 10.6 | 7200.0 | 10240.8 | 0/12 | 6.0 | 7200.0 | 3360.1 | 0/12 | **5.2** | 7200.0 | 2371.2 |
| Average | | | 0/48 | 7.7 | 7200.0 | 82.0 | 0/48 | 8.9 | 7200.0 | 22280.0 | 1/48 | 5.3 | 7183.0 | 3422.7 | 0/48 | **4.9** | 7200.0 | 2247.4 |
| Total | | | 30/80 | 4.7 | 5215.8 | 104.1 | 24/80 | 5.6 | 5392.0 | 16323.5 | 33/80 | 3.2 | 4385.7 | 2175.7 | 32/80 | **3.0** | 4410.8 | 1458.4 |

Table 5.III: Comparison of the BD and BBC Algorithms on BR2

| Set | $\|\Omega\|$ | $\epsilon$ | BD | | | | BBC | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | Every | | | | Root/Integer | | | | Integer | | | |
| | | | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts |
| $\mathcal{S}$ | 100 | 0.10 | 2/8 | 7.3 | 5981.4 | 575.0 | 3/8 | 7.3 | 5271.4 | 42714.3 | 4/8 | 5.1 | 3994.3 | 9328.3 | 4/8 | **4.5** | 3906.0 | 8967.8 |
| | | 0.25 | 1/8 | 7.2 | 6348.0 | 607.8 | 4/8 | 7.3 | 5441.7 | 45463.0 | 4/8 | 5.7 | 3798.5 | 10305.8 | 4/8 | **5.1** | 3731.0 | 8429.3 |
| | 500 | 0.10 | 2/8 | 6.9 | 6117.7 | 585.5 | 1/8 | 11.9 | 6639.7 | 17163.8 | 4/8 | **4.8** | 4105.7 | 6601.0 | 4/8 | 5.2 | 4089.8 | 5806.8 |
| | | 0.25 | 1/8 | 8.0 | 6370.7 | 608.5 | 1/8 | 10.7 | 6840.4 | 17860.0 | 4/8 | **5.5** | 4186.5 | 7321.5 | 4/8 | 6.2 | 4052.9 | 7102.3 |
| Average | | | 6/32 | 7.3 | 6204.5 | 594.2 | 9/32 | 9.3 | 6048.3 | 30800.3 | 16/32 | **5.3** | 4021.2 | 8389.1 | 16/32 | **5.3** | 3944.9 | 7576.5 |
| $\mathcal{L}$ | 100 | 0.10 | 0/12 | 25.0 | 7200.0 | 627.7 | 0/12 | 25.2 | 7200.0 | 34952.8 | 0/12 | 17.3 | 7200.0 | 13226.0 | 0/12 | **16.3** | 7200.0 | 13859.5 |
| | | 0.25 | 0/12 | 26.9 | 7200.0 | 480.0 | 0/12 | 25.7 | 7200.0 | 34744.8 | 0/12 | 19.0 | 7200.0 | 14767.7 | 0/12 | **13.8** | 7200.0 | 11261.7 |
| | 500 | 0.10 | 0/12 | 25.3 | 7200.0 | 540.2 | 0/12 | 32.7 | 7200.0 | 11935.7 | 0/12 | **17.8** | 7200.0 | 8491.5 | 0/12 | 18.4 | 7200.0 | 8573.5 |
| | | 0.25 | 0/12 | 25.9 | 7200.0 | 692.7 | 0/12 | 34.4 | 7200.0 | 11605.5 | 0/12 | 19.0 | 7200.0 | 8235.3 | 0/12 | **18.8** | 7200.0 | 8312.2 |
| Average | | | 0/48 | 25.8 | 7200.0 | 585.1 | 0/48 | 29.5 | 7200.0 | 23309.7 | 0/48 | 18.3 | 7200.0 | 11180.1 | 0/48 | **16.8** | 7200.0 | 10501.7 |
| Total | | | 6/80 | 18.4 | 6801.8 | 588.8 | 9/80 | 21.4 | 6739.3 | 26305.9 | 16/80 | 13.1 | 5928.5 | 10063.7 | 16/80 | **12.2** | 5898.0 | 9331.6 |

Among the three Benders cut generation strategies for the BBC algorithm, the *Every* strategy is the worst performing and is even worse than the BD algorithm. The *Root/Integer* and *Integer* strategies have similar performance but the *Integer* is better overall for both BR1 and BR2. The number of Benders cuts generated by the *Every* strategy is much higher than those generated by *Root/Integer* and *Integer*. When comparing the BBC with *Integer* and the BD algorithm, the results clearly indicate the benefits of using the branch-and-cut framework to solve the Benders reformulations. The average optimality gaps after two hours of CPU time as well as the computing times on the instances solved to optimality are improved considerably. It should also be noted that the average number of Benders cuts in the BBC is much larger than in the BD and this could strengthen the master problem and expedite the process. For these reasons, we use only the BBC with *Integer* for the remaining computational experiments. We also observe that the performance of BR2 is significantly worse than BR1 because the routing parts are removed from the master problem. However, since the methods presented in Section 5.4.3 can further improve the quality of the cuts and the process, we still perform the experiments on BR2 using the BBC to see the impact of the computational enhancements.

#### 5.5.1.2 Impact of the Lower Bound Lifting Inequalities

Table 5.IV reports the performance of the BBC algorithm when the lower bound lifting cuts (LBL) of Section 5.4.3.1 are added. The results show that the LBL

substantially improves the performance of the Benders algorithm. For the instance set $\mathcal{S}$ solved to optimality by BR1, the average computing time is reduced by 75%. The average optimality gap is also significantly decreased to approximately one third of the previous average gap and the BBC algorithm could solve an additional 16 and 13 instances to optimality for BR1 and BR2, respectively. The results clearly indicate the benefits of the LBL. These lower bound lifting inequalities are thus used in the remaining parts of the computational experiments.

Table 5.IV: Average Results of the BBC Algorithms Using the Lower Bound Lifting Inequalities (LBL)

| Set | $\lvert\Omega\rvert$ | $\epsilon$ | Benders Reformulation 1 (BR1) | | | | | | | | Benders Reformulation 2 (BR2) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | None | | | | LBL | | | | None | | | | LBL | | | |
| | | | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts | #Opt | Gap | CPU | B.Cuts |
| $\mathcal{S}$ | 100 | 0.10 | 8/8 | 0.0 | 114.5 | 252.3 | 8/8 | 0.0 | **27.0** | 97.1 | 4/8 | 4.5 | 3906.0 | 8967.8 | 7/8 | **0.0** | 1361.3 | 3471.0 |
| | | 0.25 | 8/8 | 0.0 | 203.5 | 328.5 | 8/8 | 0.0 | **33.5** | 124.1 | 4/8 | 5.1 | 3731.0 | 8429.3 | 8/8 | 0.0 | **845.7** | 2467.5 |
| | 500 | 0.10 | 8/8 | 0.0 | 302.4 | 250.3 | 8/8 | 0.0 | **72.9** | 89.9 | 4/8 | 5.2 | 4089.8 | 5806.8 | 7/8 | **0.0** | 1855.9 | 3002.3 |
| | | 0.25 | 8/8 | 0.0 | 287.8 | 268.6 | 8/8 | 0.0 | **94.6** | 119.6 | 4/8 | 6.2 | 4052.9 | 7102.3 | 7/8 | **0.1** | 1515.7 | 2479.8 |
| Average | | | 32/32 | 0.0 | 227.1 | 274.9 | 32/32 | 0.0 | **57.0** | 107.7 | 16/32 | 5.3 | 3944.9 | 7576.5 | 29/32 | **0.0** | 1394.7 | 2855.1 |
| $\mathcal{L}$ | 100 | 0.10 | 0/12 | 4.7 | 7200.0 | 2181.7 | 4/12 | **1.7** | 5617.0 | 1762.8 | 0/12 | 16.3 | 7200.0 | 13859.5 | 0/12 | **5.4** | 7200.0 | 17612.2 |
| | | 0.25 | 0/12 | 4.8 | 7200.0 | 2435.7 | 4/12 | **1.7** | 5602.3 | 1904.9 | 0/12 | 13.8 | 7200.0 | 11261.7 | 0/12 | **5.9** | 7200.0 | 18055.5 |
| | 500 | 0.10 | 0/12 | 5.1 | 7200.0 | 2001.0 | 5/12 | **1.6** | 5523.3 | 1535.7 | 0/12 | 18.4 | 7200.0 | 8573.5 | 0/12 | **5.9** | 7200.0 | 9058.3 |
| | | 0.25 | 0/12 | 5.2 | 7200.0 | 2371.2 | 3/12 | **2.1** | 6044.8 | 1873.4 | 0/12 | 18.8 | 7200.0 | 8312.2 | 0/12 | **6.3** | 7200.0 | 8933.2 |
| Average | | | 0/48 | 4.9 | 7200.0 | 2247.4 | 16/48 | **1.8** | 5696.8 | 1769.2 | 0/48 | 16.8 | 7200.0 | 10501.7 | 0/48 | **5.9** | 7200.0 | 13414.8 |
| Total | | | 32/80 | 3.0 | 4410.8 | 1458.4 | 48/80 | **1.1** | 3440.9 | 1104.6 | 16/80 | 12.2 | 5898.0 | 9331.6 | 29/80 | **3.6** | 4877.9 | 9190.9 |

#### 5.5.1.3 Impact of the Scenario Group Cuts and Pareto-Optimal Cuts

The average results of different settings are shown in Tables 5.V and 5.VI for BR1 and BR2, respectively. Although we have tested several different choices for the number of groups in the scenario group cuts, we report results only for $n_G = 5$ and $n_G = |\Omega|$ because a change of $n_G$ in a small range (e.g., less than 10) has little impact on the performance. Note that the latter case (i.e., $n_G = |\Omega|$) is the same as the multicut strategy proposed by Birge and Louveaux (1988). The multiple route cuts (5.43)-(5.44) are also applied to BR2 when using the scenario group cuts. Table 5.VII provides details on the average number of nodes (in thousands), and average number of Benders cuts.

Table 5.V: Average Results of the BBC Algorithm Using the Scenario Group Cuts and Pareto-Optimal Cuts on BR1

| Set | $\|\Omega\|$ | $\epsilon$ | Single cut | | | | | | 5-Cut | | | | | | $\|\Omega\|$-Cut | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Non-Pareto | | | Pareto | | | Non-Pareto | | | Pareto | | | Non-Pareto | | | Pareto | | |
| | | | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU |
| $\mathcal{S}$ | 100 | 0.10 | 8/8 | 0.0 | 27.0 | 8/8 | 0.0 | 50.9 | 8/8 | 0.0 | **25.4** | 8/8 | 0.0 | 38.4 | 8/8 | 0.0 | 33.5 | 8/8 | 0.0 | 35.1 |
| | | 0.25 | 8/8 | 0.0 | 33.5 | 8/8 | 0.0 | 46.9 | 8/8 | 0.0 | **23.8** | 8/8 | 0.0 | 41.3 | 8/8 | 0.0 | 40.3 | 8/8 | 0.0 | 51.5 |
| | 500 | 0.10 | 8/8 | 0.0 | **72.9** | 8/8 | 0.0 | 151.3 | 8/8 | 0.0 | 82.5 | 8/8 | 0.0 | 131.7 | 8/8 | 0.0 | 256.5 | 8/8 | 0.0 | 210.8 |
| | | 0.25 | 8/8 | 0.0 | 94.6 | 8/8 | 0.0 | 180.5 | 8/8 | 0.0 | **75.9** | 8/8 | 0.0 | 158.4 | 8/8 | 0.0 | 243.8 | 8/8 | 0.0 | 299.3 |
| Average | | | 32/32 | 0.0 | 57.0 | 32/32 | 0.0 | 107.4 | 32/32 | 0.0 | **51.9** | 32/32 | 0.0 | 92.5 | 32/32 | 0.0 | 143.5 | 32/32 | 0.0 | 149.2 |
| $\mathcal{L}$ | 100 | 0.10 | 4/12 | 1.7 | 5617.0 | 6/12 | 1.3 | 5218.9 | 3/12 | 1.9 | 5665.8 | 6/12 | **1.2** | 5176.2 | 2/12 | 2.8 | 6301.5 | 4/12 | 1.7 | 6030.0 |
| | | 0.25 | 4/12 | 1.7 | 5602.3 | 5/12 | 1.6 | 5473.5 | 4/12 | 2.0 | 5756.3 | 4/12 | **1.5** | 5239.1 | 3/12 | 2.4 | 6116.6 | 2/12 | 2.0 | 6125.3 |
| | 500 | 0.10 | 5/12 | **1.6** | 5523.3 | 4/12 | 1.7 | 6093.8 | 3/12 | 2.1 | 5781.4 | 5/12 | **1.6** | 5615.4 | 1/12 | 4.0 | 7049.1 | 2/12 | 3.4 | 6630.5 |
| | | 0.25 | 3/12 | 2.1 | 6044.8 | 2/12 | 2.4 | 6379.1 | 3/12 | 2.0 | 5927.1 | 3/12 | **1.9** | 6033.0 | 2/12 | 3.6 | 6971.1 | 2/12 | 3.3 | 6708.0 |
| Average | | | 16/48 | 1.8 | 5696.8 | 17/48 | 1.8 | 5791.3 | 13/48 | 2.0 | 5782.7 | 18/48 | **1.6** | 5515.9 | 8/48 | 3.2 | 6609.6 | 10/48 | 2.6 | 6373.5 |
| Total | | | 48/80 | 1.1 | 3440.9 | 49/80 | 1.1 | 3517.8 | 45/80 | 1.2 | 3490.4 | 50/80 | **0.9** | 3346.5 | 40/80 | 1.9 | 4023.2 | 42/80 | 1.6 | 3883.7 |

Table 5.VI: Average Results of the BBC Algorithm Using the Scenario Group Cuts and Pareto-Optimal Cuts on BR2

| Set | $\|\Omega\|$ | $\epsilon$ | Single cut | | | | | | 5-Cut | | | | | | $\|\Omega\|$-Cut | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Non-Pareto | | | Pareto | | | Non-Pareto | | | Pareto | | | Non-Pareto | | | Pareto | | |
| | | | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU | #Opt | Gap | CPU |
| $\mathcal{S}$ | 100 | 0.10 | 7/8 | 0.0 | 1361.3 | 7/8 | 0.0 | 1547.4 | 8/8 | 0.0 | 367.3 | 8/8 | 0.0 | **285.6** | 6/8 | 0.1 | 2196.1 | 6/8 | 0.1 | 1950.2 |
| | | 0.25 | 8/8 | 0.0 | 845.7 | 8/8 | 0.0 | 930.7 | 8/8 | 0.0 | 260.8 | 8/8 | 0.0 | **230.8** | 7/8 | 0.2 | 1599.7 | 7/8 | 0.2 | 1195.2 |
| | 500 | 0.10 | 7/8 | 0.0 | 1855.9 | 6/8 | 0.4 | 2320.4 | 8/8 | 0.0 | **506.5** | 8/8 | 0.0 | 776.5 | 4/8 | 1.0 | 3660.7 | 6/8 | 0.7 | 2859.3 |
| | | 0.25 | 7/8 | 0.1 | 1515.7 | 7/8 | 0.4 | 1922.0 | 8/8 | 0.0 | **386.1** | 8/8 | 0.0 | 705.6 | 4/8 | 1.2 | 3656.1 | 5/8 | 0.7 | 3380.3 |
| Average | | | 29/32 | 0.0 | 1394.7 | 28/32 | 0.2 | 1680.1 | 32/32 | 0.0 | **380.2** | 32/32 | 0.0 | 499.6 | 21/32 | 0.6 | 2778.1 | 24/32 | 0.4 | 2346.3 |
| $\mathcal{L}$ | 100 | 0.10 | 0/12 | 5.4 | 7200.0 | 0/12 | 5.4 | 7200.0 | 0/12 | 5.5 | 7200.0 | 0/12 | **5.0** | 7200.0 | 0/12 | 6.4 | 7200.0 | 0/12 | 6.1 | 7200.0 |
| | | 0.25 | 0/12 | 5.9 | 7200.0 | 0/12 | 5.6 | 7200.0 | 0/12 | 5.7 | 7200.0 | 0/12 | **5.3** | 7200.0 | 0/12 | 6.5 | 7200.0 | 0/12 | 6.0 | 7200.0 |
| | 500 | 0.10 | 0/12 | 5.9 | 7200.0 | 0/12 | 6.2 | 7200.0 | 0/12 | 5.8 | 7200.0 | 0/12 | **5.7** | 7200.0 | 0/12 | 7.7 | 7200.0 | 0/12 | 7.3 | 7200.0 |
| | | 0.25 | 0/12 | 6.3 | 7200.0 | 0/12 | 6.7 | 7200.0 | 0/12 | **5.9** | 7200.0 | 0/12 | 6.0 | 7200.0 | 0/12 | 7.6 | 7200.0 | 0/12 | 7.4 | 7200.0 |
| Average | | | 0/48 | 5.9 | 7200.0 | 0/48 | 6.0 | 7200.0 | 0/48 | 5.7 | 7200.0 | 0/48 | **5.5** | 7200.0 | 0/48 | 7.1 | 7200.0 | 0/48 | 6.7 | 7200.0 |
| Total | | | 29/80 | 3.6 | 4877.9 | 28/80 | 3.7 | 4992.1 | 32/80 | 3.4 | 4472.1 | 32/80 | **3.3** | 4519.9 | 21/80 | 4.5 | 5431.3 | 24/80 | 4.2 | 5258.5 |

Table 5.VII: Average Number of Nodes (in Thousands) and Benders Cuts Using the Scenario Group Cuts and Pareto-Optimal Cuts

| | Set | Single cut | | | | 5-Cut | | | | $\|\Omega\|$-Cut | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Non-Pareto | | Pareto | | Non-Pareto | | Pareto | | Non-Pareto | | Pareto | |
| | | kNodes | B.Cuts | kNodes | B.Cuts | kNodes | B.Cuts | kNodes | B.Cuts | kNodes | B.Cuts | kNodes | B.Cuts |
| BR1 | $\mathcal{S}$ | 1.9 | 107.7 | 1.1 | 64.4 | 1.7 | 436.3 | 0.8 | 279.1 | 0.8 | 15584.4 | 6.0 | 12081.3 |
| | $\mathcal{L}$ | 81.9 | 1769.2 | 56.2 | 1080.3 | 55.1 | 7105.8 | 37.2 | 4280.9 | 10.5 | 90631.3 | 9.9 | 75758.3 |
| BR2 | $\mathcal{S}$ | 87.0 | 2855.1 | 52.9 | 1901.6 | 73.7 | 1929.4 | 43.8 | 1416.2 | 18.7 | 46649.6 | 18.2 | 42679.3 |
| | $\mathcal{L}$ | 144.3 | 13414.8 | 68.5 | 6657.5 | 201.1 | 23247.3 | 152.1 | 14052.1 | 22.9 | 131657.8 | 21.7 | 114537.4 |

According to the results reported in Tables 5.V and 5.VI, the performance of the $|\Omega|$-cut indicates that a large number of scenario group cuts can lead to a significant increase in computational effort because the problem size is too large to be solved

efficiently. However, choosing an appropriate number of groups can lead to a better performance compared to a single cut as one can see in the results on the set $\mathcal{S}$ for both BR1 and BR2 and the set $\mathcal{L}$ for BR2. Pareto-optimal cuts can result in increased computing time on the small instance set $\mathcal{S}$ but the gap on the large instance set $\mathcal{L}$ generally decreases. Combining the 5-cut and Pareto-optimal cuts provides the best results compared to the other options. Table 5.VII clearly explains the impact of the scenario group cuts and Pareto-optimal cuts that lead to a faster convergence. The number of Benders cuts when using the $|\Omega|$-cut significantly increases compared to the single cut especially when the number of scenarios is large, while the number of Benders cuts grows to a much smaller extent when using the 5-cut. The Pareto-optimal cuts, however, could reduce the number of generated Benders cuts compared to the non-Pareto-optimal cuts on the instance set $\mathcal{S}$ solved to optimality and could provide smaller gaps on the remaining instances while fewer Benders cuts are generated. We also observe that, on the small instance set $\mathcal{S}$, the number of times in which the Benders subproblems are called to generate a set of Benders cuts associated with a BMP solution $(\bar{\mathbf{y}}, \bar{\mathbf{z}})$ decreased by 52% and 20% for BR1 and decreased by 90% and 80% for BR2 by using the $|\Omega|$-cut and 5-cut, respectively, compared to the single cut. We can also conclude that the performance of the BBC on BR1 is far superior to BR2. Thus, we choose the BBC with LBL, 5-cut and Pareto-optimal cut of BR1 as the preferred setting of the BBC algorithm for the remaining computational experiments.

### 5.5.2 Comparisons with the Branch-and-Cut Procedure (BC)

In this section, we compare the results of the BBC with a branch-and-cut algorithm (BC) similar to the approaches of Archetti et al. (2007, 2011); Solyalı and Süral (2011); Solyalı et al. (2012) to solve several variants of the deterministic PRP and IRP. We adapted the branch-and-cut algorithm for the vehicle index formulation presented in Chapter 3, which provided the best results for the deterministic problem, to deal with stochastic version. This BC algorithm is applied to solve the formulation $BF$, where the SECs (5.11) are added in a branch-and-cut fashion following the approach for the vehicle index formulations described in Section 3.4.1. The results are shown in Table 5.VIII. Columns *N.Cplex* and *N.SECs* show the number of CPLEX cuts

and SECs generated in the branch-and-bound tree, respectively. To better explore the differences between the two approaches, we performed the test with a number of scenarios $|\Omega| = 100$, 200, 500 and 1000. Note that the average optimality gap was computed only using instances where a feasible solution is found.

Table 5.VIII: Average Results of the BC and the BBC Algorithms

| Set | $\|\Omega\|$ | $\epsilon$ | BC | | | | | | BBC | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #Opt | Gap | CPU | Nodes | N.Cplex | N.SECs | #Opt | Gap | CPU | Nodes | N.Cplex | N.SECs | B.Cuts |
| $\mathcal{S}$ | 100 | 0.10 | 8/8 | 0.0 | **27.1** | 32.4 | 2873.3 | 74.3 | 8/8 | 0.0 | 38.4 | 704.5 | 14.8 | 196.4 | 270.0 |
| | | 0.25 | 8/8 | 0.0 | **30.5** | 43.1 | 2966.4 | 77.5 | 8/8 | 0.0 | 41.3 | 712.0 | 13.3 | 211.0 | 308.1 |
| | 200 | 0.10 | 8/8 | 0.0 | 85.7 | 41.8 | 5569.5 | 76.0 | 8/8 | 0.0 | **60.7** | 849.4 | 18.8 | 209.6 | 250.6 |
| | | 0.25 | 8/8 | 0.0 | 88.4 | 42.8 | 6133.8 | 76.0 | 8/8 | 0.0 | **87.8** | 948.0 | 14.3 | 222.0 | 358.1 |
| | 500 | 0.10 | 8/8 | 0.0 | 499.8 | 53.1 | 13413.3 | 87.5 | 8/8 | 0.0 | **131.7** | 988.9 | 20.8 | 212.1 | 241.9 |
| | | 0.25 | 8/8 | 0.0 | 475.2 | 47.8 | 15143.4 | 91.9 | 8/8 | 0.0 | **158.4** | 773.8 | 17.0 | 195.8 | 296.3 |
| | 1000 | 0.10 | 8/8 | 0.0 | 2068.2 | 62.9 | 25796.9 | 93.5 | 8/8 | 0.0 | **281.3** | 700.4 | 18.1 | 202.6 | 292.5 |
| | | 0.25 | 8/8 | 0.0 | 1994.8 | 53.4 | 28207.3 | 91.1 | 8/8 | 0.0 | **357.6** | 856.3 | 17.1 | 237.9 | 366.9 |
| Average | | | 64/64 | 0.0 | 658.7 | 47.1 | 12513.0 | 83.5 | 64/64 | 0.0 | **144.7** | 816.6 | 16.8 | 210.9 | 298.0 |
| $\mathcal{L}$ | 100 | 0.10 | 12/12 | 0.0 | **669.8** | 425.8 | 6154.7 | 470.8 | 6/12 | 1.2 | 5176.2 | 47558.3 | 22.2 | 1910.2 | 4594.2 |
| | | 0.25 | 12/12 | 0.0 | **660.9** | 337.0 | 6603.4 | 455.9 | 4/12 | 1.5 | 5239.1 | 45597.8 | 22.0 | 1824.1 | 5073.3 |
| | 200 | 0.10 | 12/12 | 0.0 | **1706.9** | 322.2 | 13051.6 | 436.0 | 6/12 | 1.4 | 5237.7 | 40121.6 | 21.6 | 1885.9 | 4060.0 |
| | | 0.25 | 12/12 | 0.0 | **1844.0** | 386.1 | 13072.0 | 452.8 | 5/12 | 1.7 | 5513.1 | 36761.8 | 24.4 | 1838.3 | 5033.8 |
| | 500 | 0.10 | 7/12 | 0.4$^{(3)}$ | 4957.6 | 161.1 | 30650.6 | 335.8 | 5/12 | **1.6** | 5615.4 | 28579.6 | 21.1 | 1714.2 | 3484.2 |
| | | 0.25 | 7/12 | 3.7$^{(1)}$ | 5169.1 | 175.4 | 31821.2 | 332.5 | 3/12 | **1.9** | 6033.0 | 26865.2 | 21.6 | 1757.9 | 3972.1 |
| | 1000 | 0.10 | 2/12 | 12.8$^{(6)}$ | 6918.1 | 27.8 | 58806.1 | 108.1 | 4/12 | **2.1** | 6185.7 | 22172.7 | 22.7 | 1626.4 | 2634.2 |
| | | 0.25 | 2/12 | 13.0$^{(7)}$ | 6860.6 | 21.6 | 57049.8 | 90.6 | 3/12 | **2.5** | 6307.3 | 19517.9 | 21.1 | 1494.3 | 2763.8 |
| Average | | | 66/96 | 3.7$^{(17)}$ | 3598.4 | 232.1 | 27151.2 | 335.3 | 36/96 | **1.8** | 5663.4 | 33396.9 | 22.1 | 1756.4 | 3951.9 |
| Total | | | 130/160 | 2.2$^{(17)}$ | 2422.5 | 158.1 | 21295.9 | 234.6 | 100/160 | **1.1** | 3455.9 | 20364.8 | 19.9 | 1138.2 | 2490.4 |

$^{(-)}$ the number of instance where a feasible solution could not be found

When the number of scenarios is not large ($|\Omega| = 100$ and 200), the BC algorithm can still solve the instances with $n = 30$ to optimality, while the BBC algorithm could not find the optimal solution for the large instances within the time limit. When the number of scenarios is larger ($|\Omega| = 500$ and 1000), however, the BC is inferior to the BBC and it could not even find a feasible solution for several instances. We observe that the BBC is far less sensitive to the number of scenarios.

In addition to this test, we solved similar instances with more vehicles ($m = 2$) and a longer planning horizon ($l = 6$) to see the impact of these parameters. The results are provided in Table 5.IX. When the number of vehicles or time periods increases, the performance of the BC algorithm drops significantly. The BBC algorithm outperforms the BC algorithm on most of the instances except on those with $|\Omega| = 100$ and on the instance set $\mathcal{S}$ with 6 periods and $|\Omega| = 200$. The BC algorithm, however, cannot handle the instances with $|\Omega| \geq 500$ and could not find a feasible solution for many

of these instances. For the set $\mathcal{L}$ with $|\Omega| = 1000$, the root node could not even be processed within two hours.

Table 5.IX: Average Results of the BC and the BBC Algorithms on Instances with More Vehicles and a Longer Horizon

| Set | $\|\Omega\|$ | 2 vehicles ($m = 2$) | | | | | | | | 6 periods ($l = 6$) | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | BC | | | | BBC | | | | BC | | | | BBC | | | |
| | | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes |
| $\mathcal{S}$ | 100 | 16/16 | 0.0 | 144.6 | 84.4 | 16/16 | 0.0 | **74.7** | 1193.1 | 15/16 | 0.0 | **1229.0** | 1029.7 | 10/16 | 0.6 | 3519.5 | 29655.0 |
| | 200 | 16/16 | 0.0 | 377.6 | 84.9 | 16/16 | 0.0 | **123.3** | 1286.6 | 14/16 | **0.2** | 2724.0 | 637.5 | 8/16 | 0.8 | 3843.4 | 23895.9 |
| | 500 | 15/16 | 0.5 | 2755.7 | 100.8 | 16/16 | 0.0 | **214.9** | 1276.1 | 6/16 | 7.1 | 5893.5 | 155.1 | 8/16 | **1.3** | 4129.0 | 11724.4 |
| | 1000 | 9/16 | 13.7$^{(3)}$ | 4533.9 | 17.4 | 16/16 | 0.0 | **352.8** | 1261.2 | 0/16 | 19.0$^{(14)}$ | 7200.0 | 4.5 | 8/16 | **1.5** | 4453.5 | 6582.5 |
| Average | | 56/64 | 3.5 | 1953.0 | 71.9 | 64/64 | 0.0 | **191.4** | 1254.2 | 35/64 | 6.6 | 4261.6 | 456.7 | 34/64 | **1.1** | 3986.3 | 17964.5 |
| $\mathcal{L}$ | 100 | 16/24 | **0.6** | 3834.4 | 697.2 | 10/24 | 1.6 | 5451.9 | 21881.2 | 4/24 | **3.5** | 6403.6 | 1085.8 | 0/24 | 7.1 | 7200.0 | 17084.2 |
| | 200 | 12/24 | 6.7 | 5368.2 | 286.6 | 8/24 | **1.9** | 5702.0 | 19840.0 | 3/24 | 13.3 | 6916.7 | 327.2 | 0/24 | **7.7** | 7200.0 | 12486.7 |
| | 500 | 1/24 | 23.3$^{(13)}$ | 7198.8 | 33.9 | 5/24 | **2.0** | 6167.5 | 16510.9 | 0/24 | 32.1$^{(18)}$ | 7200.0 | 1.0 | 0/24 | **8.6** | 7200.0 | 6501.3 |
| | 1000 | 0/24 | n/a$^{(24)}$ | 7200.0 | 0.0 | 6/24 | **2.3** | 6274.4 | 10856.0 | 0/24 | n/a$^{(24)}$ | 7200.0 | 0.0 | 0/24 | **9.7** | 7200.0 | 3721.6 |
| Average | | 29/96 | 10.2$^{(37)}$ | 5900.4 | 254.4 | 29/96 | **1.9** | 5899.0 | 17272.0 | 7/96 | 16.3$^{(42)}$ | 6930.1 | 353.5 | 0/96 | **8.3** | 7200.0 | 9948.4 |
| Total | | 85/160 | 7.5$^{(40)}$ | 4321.4 | 181.4 | 93/160 | **1.2** | 3616.0 | 10864.9 | 42/160 | 12.4$^{(42)}$ | 5862.7 | 394.8 | 34/160 | **5.4** | 5914.5 | 13154.8 |

$^{(-)}$ the number of instance where a feasible solution could not be found

It should also be noted that a large number of CPLEX cuts is generated when solving the BC and the majority of them, approximately 90%, are *flow cover* cuts that strengthen the network structure of the problem. In Table 5.X, we further examine the impact of turning off CPLEX cuts on the instances with $m = 1$ and $l = 3$. The results clearly show the impact of the CPLEX cuts for the BC algorithm. Without these cuts, the performance of the BC algorithm decreases significantly. The number of optimal solutions found by the BC algorithm after two hours on the instance set $\mathcal{L}$ decreased from 66 to 31 and the instances with $|\Omega| \leq 200$ were not solved to optimality. This can be explained by the substantial increase in the average number of nodes of the BC algorithm when the CPLEX cuts are not used. However, more feasible solutions on large instances could be found because the BC explored a larger number of nodes in the branch-and-bound tree. For the BBC, there is almost no impact on the performance and the algorithm could also provide better results on the instance set $\mathcal{L}$ with $|\Omega| = 200$ compared to the BC when the CPLEX cuts are turned off.

Table 5.X: Performance of the BC and the BBC Algorithms without CPLEX Cuts

| Set | $\|\Omega\|$ | BC | | | | | | | | BBC | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | with CPLEX cuts | | | | w/o CPLEX cuts | | | | with CPLEX cuts | | | | w/o CPLEX cuts | | | |
| | | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes | #Opt | Gap | CPU | Nodes |
| $\mathcal{S}$ | 100 | 16/16 | 0.0 | 28.8 | 37.8 | 16/16 | 0.0 | 37.0 | 242.1 | 16/16 | 0.0 | 39.9 | 708.3 | 16/16 | 0.0 | 39.9 | 714.9 |
| | 200 | 16/16 | 0.0 | 87.1 | 42.3 | 16/16 | 0.0 | 102.5 | 283.4 | 16/16 | 0.0 | 74.2 | 898.7 | 16/16 | 0.0 | 63.4 | 754.7 |
| | 500 | 16/16 | 0.0 | 487.5 | 50.4 | 16/16 | 0.0 | 522.7 | 307.3 | 16/16 | 0.0 | 145.1 | 881.3 | 16/16 | 0.0 | 155.3 | 828.3 |
| | 1000 | 16/16 | 0.0 | 2031.5 | 58.1 | 16/16 | 0.0 | 2490.1 | 301.9 | 16/16 | 0.0 | 319.5 | 778.3 | 16/16 | 0.0 | 296.5 | 762.8 |
| Average | | 64/64 | 0.0 | 658.7 | 47.1 | 64/64 | 0.0 | 788.1 | 283.7 | 64/64 | 0.0 | 144.7 | 816.6 | 64/64 | 0.0 | 138.8 | 765.2 |
| $\mathcal{L}$ | 100 | 24/24 | 0.0 | 665.4 | 381.4 | 15/24 | 0.8 | 3876.0 | 5035.1 | 10/24 | 1.4 | 5207.6 | 46578.0 | 10/24 | 1.4 | 5033.8 | 44197.3 |
| | 200 | 24/24 | 0.0 | 1775.5 | 354.1 | 12/24 | 1.8 | 4781.5 | 2723.0 | 11/24 | 1.5 | 5375.4 | 38441.7 | 9/24 | 1.5 | 5479.1 | 41797.5 |
| | 500 | 14/24 | $2.0^{(4)}$ | 5063.4 | 168.3 | 4/24 | 4.8 | 6434.8 | 984.9 | 8/24 | 1.8 | 5824.2 | 27722.4 | 8/24 | 1.9 | 5871.3 | 29406.7 |
| | 1000 | 4/24 | $12.9^{(13)}$ | 6889.4 | 24.7 | 0/24 | $5.1^{(11)}$ | 7200.0 | 217.0 | 7/24 | 2.3 | 6246.5 | 20845.3 | 7/24 | 2.3 | 6294.8 | 24777.6 |
| Average | | 66/96 | $3.7^{(17)}$ | 3598.4 | 232.1 | 31/96 | $3.1^{(11)}$ | 5573.1 | 2240.0 | 36/96 | 1.8 | 5663.4 | 33396.9 | 34/96 | 1.8 | 5669.7 | 35044.8 |
| Total | | 130/160 | $2.2^{(17)}$ | 2422.5 | 158.1 | 95/160 | $1.9^{(11)}$ | 3659.1 | 1457.5 | 100/160 | 1.1 | 3455.9 | 20364.8 | 98/160 | 1.1 | 3457.3 | 21332.9 |

$^{(-)}$ the number of instance where a feasible solution could not be found

## 5.6 Reoptimization Capabilities of the Benders Decomposition Algorithm in Stochastic Environments

The reoptimization capabilities provided by Benders decomposition can be very useful in "what-if" analyses (Geoffrion and Graves, 1974; Cordeau et al., 2006). In situations where the changes made to the problem data affect only the master problem, the previously generated Benders cuts are still valid and one can warm-start the algorithm with these cuts. A new optimal solution is typically obtained in a few iterations. Examples of this situation include forcing variables to take specific values or adding logical constraints on the master problem variables. If one instead employs a branch-and-cut algorithm, it must restart from scratch every time a change is made, which can be very time consuming. Furthermore, if the changes affect only the objective function but not the polyhedron of the Benders dual subproblem, one can also recalculate the Benders cuts from the already generated extreme points. This is the case for the two-stage SPRP, where changes in demand affect only the objective function of the DFS. We now discuss how the reoptimization capabilities of the Benders algorithm can be particularly useful in two practical settings, i.e., in a sample average approximation (SAA) scheme and in a rolling horizon (RH) framework.

In both cases, the procedure starts by solving the first problem from scratch using the BBC and keeping the dual solutions generated during the solution process. When the problem is reoptimized, Benders cuts corresponding to the new demand scenarios (and also new initial inventory levels in RH) are computed using the dual solutions that were already generated in the previous replications. These cuts are added to the Benders master problem at the start of each replication. To avoid adding too many cuts, we use a preprocessing step to select the cuts. This process starts by solving the BMP without any Benders cuts to obtain an initial solution along with the values of $\eta_g$. Then, the cuts with a non-negative left-hand-side value for the initial solution are used as initial Benders cuts. The number of initial Benders cuts is limited to 5000. If the number of cuts exceeds the limit, those with the largest left-hand-side value are selected first. In Table 5.XI, the results obtained by the BBC algorithm with reoptimization are shown in Column *BBC-ReOpt* and the average percentage of the number of initial Benders cuts in each replication is shown in Column *%I.Cuts*. Column *T.CPU* indicates the average total time spent to solve all replications of the same instance and boldface letters are put on the smallest time. Columns *CPU*, *Nodes* and *B.Cuts* are the same as in the previous section and they show the average results per replication.

### 5.6.1 Reoptimization for a Sample Average Approximation Scheme

Sample average approximation (SAA) (Kleywegt et al., 2002b) is a Monte-Carlo simulation-based sampling method developed to solve problems where the number of scenarios is very large. It can also be applied to problems with continuous distributions or with an infinite number of scenarios. Given a large scenario set, denoted by $\Omega'$, which is intractable, the SAA consists of solving a number of smaller and tractable problems with a sample of size $|\Omega| \ll |\Omega'|$. In the SAA process, one can calculate the SAA gap which is the estimated difference between the solution obtained by solving the replications of the sample size $|\Omega|$ and a statistical lower bound on the optimal value for the large scenario set $\Omega'$. This gap can be determined by a sample average function. In practical applications, one can choose a sample size $|\Omega|$ and the number

of replications $|M|$ that are most appropriate for the problem in terms of solution quality and computing time. In the SPRP, where different demand scenarios are considered in each replication, one can take advantage of the reoptimization capabilities of Benders decomposition to solve the problem more efficiently. We provide more details on the SAA scheme and steps to estimate the optimality gap in the Appendix C.

Tests were performed with sample sizes $|\Omega| = 100$, 200, 500 and 1000 and a number of replications $|M| = 100$, 50, 20 and 10, respectively, which makes the total number of evaluated scenarios equal to 10,000 for every sample size. Because the experiments in this section focus on the benefits of the reoptimization capabilities, we performed the tests on the instance set $\mathcal{S}$ with $l = 3$ and $m = 1$ to avoid excessive computing times. In addition to the uniform distribution, experiments were also performed with normal and gamma distributions with parameters chosen so that the current demand range $[\bar{d}_{it}(1-\epsilon), \bar{d}_{it}(1+\epsilon)]$ corresponds approximately to a 99.5% confidence interval. The size of the large scenario set was chosen equal to $|\Omega'| = 10{,}000$ to evaluate the SAA gap. In the first set of experiments, we compared the results provided by different algorithms. Scenarios were generated a priori and all the algorithms were applied to the same scenario sets to ensure a fair comparison. The results are provided in Table 5.XI.

Table 5.XI: Performance of the Algorithms Using the SAA Method

| Distribution | $\|\Omega\|$ | $\|M\|$ | $\epsilon$ | BC | | | BBC | | | | BBC-ReOpt | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | T.CPU | CPU | Nodes | T.CPU | CPU | Nodes | B.Cuts | T.CPU | CPU | Nodes | B.Cuts | %I.Cuts |
| Uniform | 100 | 100 | 0.10 | 2703.6 | 27.0 | 35.3 | 3622.7 | 36.2 | 750.0 | 259.6 | **2090.0** | 20.9 | 303.7 | 2571.7 | 97.4 |
| | | | 0.25 | 2777.1 | 27.8 | 35.9 | 3972.9 | 39.7 | 776.7 | 289.1 | **2668.1** | 26.7 | 360.7 | 3135.2 | 97.1 |
| | 200 | 50 | 0.10 | 4348.9 | 87.0 | 44.9 | 3140.4 | 62.8 | 742.4 | 258.6 | **1439.4** | 28.8 | 333.4 | 1904.5 | 96.2 |
| | | | 0.25 | 4373.6 | 87.5 | 43.1 | 3686.3 | 73.7 | 801.1 | 302.0 | **1827.2** | 36.5 | 357.7 | 2427.8 | 96.1 |
| | 500 | 20 | 0.10 | 9596.9 | 479.8 | 52.6 | 2778.8 | 138.9 | 787.8 | 264.4 | **1131.3** | 56.6 | 380.8 | 1007.4 | 91.5 |
| | | | 0.25 | 9372.4 | 468.6 | 50.2 | 3160.5 | 158.0 | 773.9 | 300.1 | **1437.3** | 71.9 | 396.7 | 1287.6 | 91.5 |
| | 1000 | 10 | 0.10 | 18891.8 | 1889.2 | 53.6 | 2531.7 | 253.2 | 819.7 | 268.4 | **1197.4** | 119.7 | 432.5 | 686.3 | 83.6 |
| | | | 0.25 | 19639.4 | 1963.9 | 53.3 | 3223.0 | 322.3 | 808.7 | 339.0 | **1466.7** | 146.7 | 451.2 | 901.4 | 84.7 |
| Average | | | | 8963.0 | 628.9 | 46.1 | 3264.5 | 135.6 | 782.5 | 285.1 | **1657.2** | 63.5 | 377.1 | 1740.3 | 94.5 |
| Normal | 100 | 100 | 0.10 | 2815.3 | 28.2 | 41.8 | 2898.9 | 29.0 | 597.7 | 210.5 | **1664.3** | 16.6 | 235.7 | 2390.4 | 97.6 |
| | | | 0.25 | 2862.2 | 28.6 | 48.4 | 3444.1 | 34.4 | 791.7 | 244.7 | **2331.1** | 23.3 | 368.8 | 2662.7 | 97.1 |
| | 200 | 50 | 0.10 | 4326.1 | 86.5 | 52.7 | 2596.0 | 51.9 | 613.9 | 215.4 | **1044.6** | 20.9 | 243.0 | 1486.1 | 96.4 |
| | | | 0.25 | 4425.6 | 88.5 | 58.1 | 3188.0 | 63.8 | 843.9 | 260.4 | **1639.7** | 32.8 | 416.3 | 1986.4 | 95.9 |
| | 500 | 20 | 0.10 | 9480.4 | 474.0 | 59.7 | 2377.4 | 118.9 | 644.6 | 230.3 | **940.0** | 47.0 | 278.0 | 865.0 | 91.5 |
| | | | 0.25 | 9138.1 | 456.9 | 69.9 | 2712.2 | 135.6 | 839.7 | 260.5 | **1280.2** | 64.0 | 451.6 | 1138.6 | 91.6 |
| | 1000 | 10 | 0.10 | 19107.4 | 1910.7 | 58.2 | 2244.3 | 224.4 | 670.9 | 241.7 | **1008.6** | 100.9 | 317.7 | 634.4 | 84.3 |
| | | | 0.25 | 19069.0 | 1906.9 | 68.6 | 2663.6 | 266.4 | 943.6 | 286.4 | **1257.2** | 125.7 | 450.3 | 760.6 | 84.5 |
| Average | | | | 8903.0 | 622.5 | 57.2 | 2765.6 | 115.5 | 743.2 | 243.7 | **1395.7** | 53.9 | 345.2 | 1490.5 | 94.5 |
| Gamma | 100 | 100 | 0.10 | 2654.1 | 26.5 | 37.1 | 2821.5 | 28.2 | 558.3 | 204.7 | **1536.4** | 15.4 | 213.5 | 2178.1 | 97.6 |
| | | | 0.25 | 2759.2 | 27.6 | 34.6 | 3729.1 | 37.3 | 782.0 | 271.9 | **2385.5** | 23.9 | 338.6 | 3043.5 | 97.3 |
| | 200 | 50 | 0.10 | 4058.4 | 81.2 | 44.3 | 2455.4 | 49.1 | 564.9 | 204.8 | **1022.1** | 20.4 | 213.4 | 1408.3 | 96.4 |
| | | | 0.25 | 4305.7 | 86.1 | 42.4 | 3444.2 | 68.9 | 815.8 | 286.3 | **1705.3** | 34.1 | 373.8 | 2324.9 | 96.2 |
| | 500 | 20 | 0.10 | 8992.3 | 449.6 | 56.5 | 2184.2 | 109.2 | 595.5 | 210.4 | **851.9** | 42.6 | 258.6 | 802.2 | 91.8 |
| | | | 0.25 | 9520.2 | 476.0 | 54.0 | 3027.3 | 151.4 | 826.3 | 292.3 | **1281.0** | 64.1 | 389.4 | 1229.4 | 91.6 |
| | 1000 | 10 | 0.10 | 18456.3 | 1845.6 | 59.0 | 2092.2 | 209.2 | 631.1 | 225.6 | **894.4** | 89.4 | 277.9 | 538.6 | 84.4 |
| | | | 0.25 | 19368.8 | 1936.9 | 57.9 | 2718.1 | 271.8 | 786.4 | 292.9 | **1533.0** | 153.3 | 471.5 | 944.3 | 84.2 |
| Average | | | | 8764.4 | 616.2 | 48.2 | 2809.0 | 115.6 | 695.0 | 248.6 | **1401.2** | 55.4 | 317.1 | 1558.7 | 94.6 |
| Total Average | | | | 8876.8 | 622.5 | 50.5 | 2946.4 | 122.3 | 740.3 | 259.2 | **1484.7** | 57.6 | 346.4 | 1596.5 | 94.5 |

These results clearly indicate the benefits of reoptimization when solving the SPRP in a SAA method. The BBC algorithm using reoptimization could reduce the average total computing time by approximately 50% and 83% compared to the BBC without reoptimization and BC, respectively. The average number of Benders cuts generated by the BBC-ReOpt is significantly larger than for the BBC. However, the majority of them (94.5%) are the initial cuts generated from previous replications and only 5.5% of the number of Benders cuts or 87.7 cuts on average were newly generated at each replication. To further demonstrate the benefits of reoptimization, Figure 5.2 shows the average computing time spent in each replication to solve the instances with $n = 15$, $\epsilon = 0.10$, $|\Omega| = 200$ and a uniform distribution. The computing times after the first replication are significantly reduced when using the BBC-ReOpt.

To evaluate the impact of the sample size, we further performed experiments to compare the solution quality obtained with different sizes using the BBC-ReOpt.

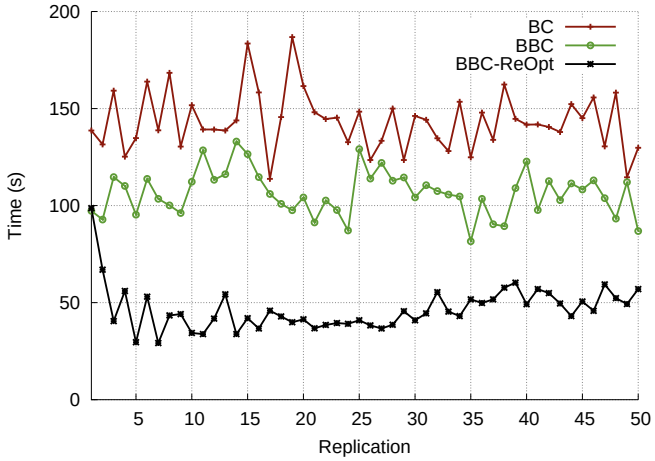



Figure 5.2: Computing Time Spent at Each Replication in the SAA Method

The results are shown in Table 5.XII. These tests follow the procedure presented by Kleywegt et al. (2002b) in which scenarios are generated and the SAA gap is calculated during the SAA process. We report the results at the end of the process after all the replications are solved. Column *ETC (%Abs)* shows the average absolute gap (%) computed by $\sum_{i=1}^{|M|}\left((\nu_{\Omega}^{i}-\hat{\nu}_{\Omega})/\nu_{\Omega}^{i}\right)$, where $\nu_{\Omega}^{i}$ is the expected total cost in replication $i$ corresponding to the objective function (5.1) and $\hat{\nu}_{\Omega}$ is the SAA lower bound. Column *SAA (%Abs)* indicates the average absolute SAA gap (%) computed by $\varepsilon^{SAA}(\Omega,\Omega')/\hat{\nu}_{\Omega}$, where $\varepsilon^{SAA}(\Omega,\Omega')$ is the SAA gap obtained by the SAA method as discussed in the Appendix C. Column $\pm$ *SAA (%) at 95% CI* indicates the average range of the gap ($\pm$%) at the 95% confidence interval (CI) where it is computed with the standard deviation of the SAA gaps and we assume that the gap is normally distributed.

Table 5.XII: Solution Quality and Computing Times of the SAA Using Different Sample Size

| | | ETC (%Abs) | | | | SAA (%Abs) | | | | ± SAA (%) at CI 95% | | | | T.CPU | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Distribution | $\|\Omega\| =$ | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 | 100 | 200 | 500 | 1000 |
| | $\|M\| =$ | 100 | 50 | 20 | 100 | 100 | 50 | 20 | 100 | 100 | 50 | 20 | 100 | 100 | 50 | 20 | 100 |
| Uniform | $\epsilon =$0.10 | 0.52 | 0.35 | 0.15 | **0.07** | 0.09 | 0.04 | 0.03 | **0.02** | 0.12 | 0.11 | 0.08 | **0.07** | 1769.1 | 1276.7 | 1113.0 | **1090.6** |
| | 0.25 | 0.98 | 0.59 | 0.29 | **0.13** | 0.13 | 0.12 | 0.04 | **0.04** | 0.25 | 0.23 | 0.17 | **0.14** | 2259.1 | 1616.4 | **1374.2** | 1436.9 |
| | Average | 0.75 | 0.47 | 0.22 | **0.10** | 0.11 | 0.08 | 0.04 | **0.03** | 0.19 | 0.17 | 0.13 | **0.11** | 2014.1 | 1446.6 | **1243.6** | 1263.7 |
| Normal | 0.10 | 0.06 | 0.05 | 0.02 | **0.01** | 0.02 | **0.01** | **0.01** | **0.01** | 0.02 | 0.02 | 0.02 | **0.01** | 1833.2 | 1365.6 | **1224.2** | 1228.4 |
| | 0.25 | 0.10 | 0.06 | 0.04 | **0.03** | 0.02 | 0.02 | **0.01** | **0.01** | 0.03 | 0.02 | 0.02 | **0.01** | 2752.3 | 2079.1 | **1501.0** | 1600.0 |
| | Average | 0.08 | 0.05 | 0.03 | **0.02** | 0.02 | **0.01** | **0.01** | **0.01** | 0.03 | 0.02 | 0.02 | **0.01** | 2292.7 | 1722.3 | **1362.6** | 1414.2 |
| Gamma | 0.10 | 0.03 | 0.02 | **0.01** | **0.01** | 0.01 | 0.01 | **0.00** | **0.00** | 0.02 | **0.01** | **0.01** | **0.01** | 1815.7 | 1315.5 | **1188.5** | 1256.0 |
| | 0.25 | 0.09 | 0.08 | 0.04 | **0.03** | 0.01 | 0.02 | **0.01** | **0.01** | 0.03 | 0.03 | **0.02** | **0.02** | 2662.0 | 2009.1 | **1587.1** | 1801.0 |
| | Average | 0.06 | 0.05 | 0.03 | **0.02** | 0.01 | 0.01 | 0.01 | **0.00** | 0.02 | 0.02 | 0.02 | **0.01** | 2238.8 | 1662.3 | **1387.8** | 1528.5 |
| | Total Average | 0.30 | 0.19 | 0.09 | **0.05** | 0.04 | 0.03 | **0.02** | **0.02** | 0.08 | 0.07 | **0.05** | **0.05** | 2181.9 | 1610.4 | **1331.3** | 1402.2 |

We observe that the largest sample size $|\Omega| = 1000$ can provide the best average ETC gap as well as the best average SAA gap with the least variation compared to the other sample sizes but the average computing time is larger than for the sample size $|\Omega| = 500$. The sample size $|\Omega| = 500$ is, however, generally the best in terms of the trade-off between solution quality and computing time.

### 5.6.2 Reoptimization for the Dynamic and Stochastic PRP in a Rolling Horizon Framework

In practice, dynamic planning problems are solved repeatedly over time as new information becomes available. This is often done in a rolling horizon framework: a problem is solved using the information for a limited number of periods and in a later period, the problem is solved again using updated information. Usually, there is some overlap between the periods considered in two subsequent problems in order to reduce the end of horizon effect. This version of the SPRP can be seen as a dynamic and stochastic PRP (DSPRP).

In a rolling horizon, one must solve the problem repeatedly with updated initial inventory levels and a new set of demand scenarios. Reoptimization can be particularly useful in this context since the changes do not affect the Benders subproblem polyhedron. To explore the benefits of the reoptimization of the BBC, we performed experiments using the data set used in the previous section but extended to 52 time periods (i.e., one year). We assume that the demand in the current period is more

accurate with $\epsilon = 0.1$ and the next periods are more uncertain with $\epsilon = 0.25$. The nominal demand is assumed to increase 5% for every quarter (i.e., every 13 periods). The number of rolling periods to optimize is set to 3 and 4 and the overlapping period is set to one, which results in a total number of times that the problem has to be solved being equal to $|M| = 26$ and 17 for 3 and 4 rolling periods, respectively. We assume that the initial inventory for the current rolling periods is equal to the rounded expected value of the corresponding inventory level over all scenarios resulting from the previous rolling periods. The total computing time per instance is limited to 30 hours and the experiments are performed on the instances with $n = 10$ since it is very time consuming to solve larger instance sizes for the whole planning horizon (the total computing time usually exceeds 30 hours).

The results are shows in Table 5.XIII and we indicate the variants by $t_r/t_o$ where $t_r$ and $t_o$ are the number of rolling and overlapping periods, respectively. According to these results, the BBC-ReOpt could reduce the average total computing time by approximately 41% and 81% compared to the BBC without reoptimization and BC, respectively. We can also observe that the reduction in computing time is less for the BBC using reoptimization in the rolling horizon framework compared to the results of the SAA. This is due to the fact that, in each subsequent problem, demands are computed from different nominal cases and initial inventory levels are changed. Thus, the initial cuts generated in the rolling horizon framework are not as strong as in the SAA where demand scenarios of each customer are generated from the same demand distribution.

Table 5.XIII: Performance of the Algorithms Using the Rolling Horizon Method

| $t_r/t_o$ | $\|M\|$ | $\|\Omega\|$ | $\epsilon$ | BC | | | BBC | | | | BBC-ReOpt | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | T.CPU | CPU | Nodes | T.CPU | CPU | Nodes | B.Cuts | T.CPU | CPU | Nodes | B.Cuts | %I.Cuts |
| 3/1 | 26 | 100 | 0.10 | 485.8 | 18.7 | 54.0 | 962.6 | 37.0 | 1361.2 | 365.2 | **439.0** | 16.9 | 567.9 | 1674.4 | 91.3 |
| | | 200 | 0.10 | 1270.4 | 48.9 | 60.4 | 1658.5 | 63.8 | 1385.7 | 377.2 | **863.8** | 33.3 | 585.4 | 1611.6 | 91.0 |
| | | 500 | 0.10 | 6813.9 | 262.1 | 66.9 | 3569.3 | 137.3 | 1418.1 | 380.0 | **1853.7** | 74.0 | 615.7 | 1824.6 | 91.7 |
| | | 1000 | 0.10 | 26577.2 | 1022.2 | 69.5 | 6774.5 | 260.6 | 1300.3 | 395.6 | **3529.7** | 130.6 | 598.8 | 1740.2 | 91.1 |
| Average | | | | 8786.8 | 338.0 | 62.7 | 3241.2 | 124.7 | 1366.3 | 379.5 | **1671.6** | 64.3 | 592.0 | 1712.7 | 91.3 |
| 4/1 | 17 | 100 | 0.10 | **889.7** | 52.3 | 133.1 | 2013.3 | 118.4 | 4733.2 | 914.2 | 1376.2 | 81.0 | 2931.0 | 3340.0 | 87.1 |
| | | 200 | 0.10 | 2655.0 | 156.2 | 133.7 | 3177.8 | 186.9 | 3913.4 | 879.3 | **1909.5** | 112.3 | 2474.3 | 3341.1 | 88.9 |
| | | 500 | 0.10 | 15110.2 | 888.8 | 140.3 | 6587.8 | 387.5 | 4024.9 | 862.2 | **4051.2** | 238.3 | 2485.8 | 3555.7 | 88.8 |
| | | 1000 | 0.10 | 65006.9† | 3823.9 | 121.8 | 13537.6 | 796.3 | 4131.4 | 910.7 | **8489.2** | 499.4 | 2478.2 | 3714.9 | 88.1 |
| Average | | | | 20915.4† | 1230.3 | 132.2 | 6329.1 | 372.3 | 4200.7 | 891.6 | **3956.5** | 232.7 | 2592.3 | 3487.9 | 88.2 |
| Total | | | | 14851.1† | 784.1 | 97.5 | 4785.2 | 248.5 | 2783.5 | 635.6 | **2814.0** | 148.5 | 1592.1 | 2600.3 | 89.8 |

†15 (out of 68) instances were not solved to optimality and the average optimality gap is 0.4%

## 5.7 Conclusion

We have addressed demand uncertainty in the production routing problem and proposed Benders reformulations to handle the problem. The Benders algorithms are implemented in a branch-and-cut framework, called branch-and-Benders-cut (BBC), and several computational enhancements, namely, lower bound lifting inequalities, scenario group cuts and Pareto-optimal cuts, are used to improve the performance of the algorithms. The BBC with the best version of the Benders reformulation outperformed the branch-and-cut (BC) algorithm on the standard formulation when solving instances with 500 and 1000 scenarios. The results also indicate that the performance of the BBC does not depend on the CPLEX cuts, while the performance of the branch-and-cut algorithm relies heavily on these cuts. We further discuss reoptimization capabilities in the context of the sample average approximation method and the dynamic and stochastic production routing problem in a rolling horizon framework. The results show that reoptimization can substantially improve the performance of the BBC and BC algorithms for which the computing time is reduced by 50% and 83%, respectively, for the sample average approximation method, and by 41% and 81%, respectively, for the rolling horizon framework.

# Chapter 6
# Conclusion

This thesis addresses the production routing problem (PRP), which is an integrated supply chain operational planning problem that jointly optimizes production, inventory, distribution and routing decisions. The problem is also a generalization and natural extension of the inventory routing problem (IRP) obtained by taking production decisions into account. The PRP has been extensively studied in the past two decades starting from Chandra and Fisher (1994) who explicitly discussed the cost benefit of integrating the production aspect into the IRP. In this thesis, we have discussed several issues that have not been addressed in the previous literature. We first introduced formulations and exact algorithms that can handle the PRP with multiple vehicles and which can also be applied to the IRP. We next developed a novel heuristic algorithm that can handle the PRP efficiently and outperformed all former approaches. Finally, we specifically addressed the PRP with demand uncertainty and developed efficient approaches to solve the problem. This section outlines the contributions made in the thesis and future research opportunities.

In Chapter 2, we first presented a general perspective of integrated supply chain operational planning problems and provided a brief review of the three integrated problems, i.e., the lot-sizing problem with direct shipment, the IRP and the PRP, with the PRP being a generalization of the other two. Next, we reviewed the models and different formulation schemes, followed by heuristics and exact algorithms that have been proposed for the PRP. We found that, despite a growing body of literature, there is still a lack of exact algorithms to efficiently handle the PRP. We finally discussed studies that have addressed demand uncertainty in the PRP and related problems.

In Chapter 3, we introduced formulations and exact algorithms for the PRP with multiple vehicles, which is an issue that has been neglected in the past due to its complexity. Two formulation schemes, with and without a vehicle index, are proposed to

solve the PRP and IRP under both the maximum level (ML) and the order-up-to level (OU) inventory replenishment policies. The vehicle index formulations are further improved using symmetry breaking constraints, while the non-vehicle index formulations are strengthened by several cuts. We developed branch-and-cut approaches to solve the problems. The results show that the vehicle index formulations are superior in finding optimal solutions, while the non-vehicle index formulations are generally better at providing good lower bounds on larger instances. IRP and PRP instances with up to 35 customers, 3 periods and 3 vehicles can be solved to optimality within two hours for the ML policy. By using parallel computing, the algorithms could solve the instances for the same policy with up to 45 and 50 customers, 3 periods and 3 vehicles for the IRP and PRP, respectively. For the more difficult IRP (resp. PRP) under the OU policy, the algorithms could handle instances with up to 30 customers, 3 (resp. 6) periods and 3 vehicles on a single core machine, and up to 45 (resp. 35) customers, 3 (resp. 6) periods and 3 vehicles on a multi-core machine. These results thus fill an important gap in the PRP and IRP literature.

In Chapter 4, we proposed a novel heuristic, called an optimization-based adaptive large neighborhood search (Op-ALNS), to solve large instances which cannot be handled by the developed exact algorithms. In the Op-ALNS, binary variables representing setup and routing decisions are handled by an enumeration scheme and upper-level search operators, respectively, while continuous variables associated with production, inventory and shipment quantities are set by solving a network flow subproblem. Extensive computational experiments have been performed on benchmark instances from the literature. The results show that our algorithm generally outperforms existing heuristics for the PRP and can produce high quality solutions in short computing times. We further adapted the Op-ALNS to the PRP with the OU policy and the IRP with the ML and OU policies and performed experiments to evaluate the solution quality of the Op-ALNS on these variants. The Op-ALNS could provide high quality solutions for the PRP instances with deviations from the optimal solutions of 1.2% and 0.9% for the ML and OU policies, respectively, while it could obtain good quality solutions for the IRP in very short computing times compared to existing approaches.

In Chapter 5, we addressed the PRP with demand uncertainty, called the stochastic PRP (SPRP), which is a practical enhancement to the deterministic PRP. We considered the problem in a two-stage decision process. The decisions in the first stage include production setups and customer visit schedules, while the production, inventory and delivery quantities are determined in the second stage. We developed exact solution approaches based on Benders decomposition together with several refinements. Two different Benders reformulation schemes were proposed. Furthermore, we compared a standard implementation of the Benders algorithm to one that integrates the Benders cuts within a branch-and-cut framework. This latter implementation is called branch-and-Benders-cut (BBC) and uses only one enumeration tree for the Benders master problem. We also showed how this can be integrated with a branch-and-cut algorithm for subtour elimination constraints. The Benders master is further enhanced with lower bound lifting inequalities, scenario group cuts and Pareto-optimal cuts. The best version of the Benders algorithm provides superior results to a branch-and-cut algorithm on the standard formulation when solving a large number of scenarios while the performance of the branch-and-cut algorithm heavily relies on the CPLEX cuts. We further exploit the reoptimization capability of the Benders approach in two stochastic settings, namely, a sample average approximation scheme to handle a large number of scenarios, and a rolling horizon framework for a dynamic and stochastic variant of the PRP. The results show that the computing time of the Benders algorithm using reoptimization is reduced by approximately 40-50% and more than 80% compared to the Benders algorithm without reoptimization and the branch-and-cut algorithm, respectively.

As an integration of several areas in production and distribution planning, there are many future research directions that have not been explored. We summarize them as follows:

**Other variants of the LSP.** As a major component of the PRP, one may consider other interesting variants of the LSP. For example, the multi-product problem where the production setup for each product must be done separately. Although some heuristics have been proposed for this variant (Fumero and Vercellis, 1999 and Armentano et al., 2011), no studies have discussed exact methods to solve

the problem. One can also consider the multi-product variant where production startups must also be taken into account.

**PRP with customer visit and delivery time windows.** Time windows are one of the most common issues in transportation operations. There are two different types of time windows that can be considered in the PRP. The first type, called customer visit time windows, is imposed on periods where some customers should be visited to satisfy operational requirements. The second type, called delivery time window, is imposed during the day of delivery and is a well-known variant of the VRP.

**Robust PRP (RPRP).** In stochastic environments, it is possible that a probabilistic description of the uncertainty is not available and one cannot use the SPRP to solve the problem. In this case, instead of minimizing the expected total cost, one is interested in obtaining a solution that is immune to any realization of the uncertainty in a given set and therefore the product availability must be guaranteed.

**Tactical PRP.** In many applications, customers must be clustered and a driver is assigned to serve a specific cluster. The cluster decisions are fixed for a long term horizon. Examples of this variant can be found in Michel and Vanderbeck (2012) and Coelho et al. (2012a). One possible approach is to decompose the problem. The first problem is used to identify clusters, while the other problem is a modified PRP to determine the remaining decisions.

In this thesis, various contributions for the PRP have been made to support a growing interest in this research area and real-world industrial applications. There are, however, still a number of interesting issues that have not been addressed. We are encouraging researchers in this field to pursue further developments in this promising research area.

# Bibliography

T. F. Abdelmaguid and M. M. Dessouky. A genetic algorithm approach to the integrated inventory-distribution problem. *Int. J. Prod. Res.*, 44(21):4445–4464, 2006.

D. Adelman. A price-directed approach to stochastic inventory/routing. *Oper. Res.*, 52(4):499–514, 2004.

O. Alp, N. K. Erkip, and R. Güllü. Optimal lot-sizing/vehicle-dispatching policies under stochastic lead times and stepwise fixed costs. *Oper. Res.*, 51(1):160–166, 2003.

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

D. Applegate, R. Bixby, V. Chvátal, and W. Cook. Concorde TSP solver. 2011. http://www.tsp.gatech.edu/concorde.html.

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

C. Archetti, L. Bertazzi, G. Paletta, and M. G. Speranza. Analysis of the maximum level policy in a production-distribution system. *Comput. Oper. Res.*, 38(12):1731–1746, 2011.

C. Archetti, L. Bertazzi, A. Hertz, and M. G. Speranza. A hybrid heuristic for an inventory routing problem. *INFORMS J. Comput.*, 24(1):101–116, 2012.

V. A. Armentano, A. L. Shiguemoto, and A. Løkketangen. Tabu search with path relinking for an integrated production-distribution problem. *Comput. Oper. Res.*, 38(8):1199–1209, 2011.

N. Azi, M. Gendreau, and J.-Y. Potvin. An adaptive large neighborhood search for a vehicle routing problem with multiple trips. CIRRELT Tech. Rep. 2010-08, Université de Montréal, Canada, 2010.

J. F. Bard and N. Nananukul. The integrated production-inventory-distribution-routing problem. *J. Sched.*, 12(3):257–280, 2009a.

J. F. Bard and N. Nananukul. Heuristics for a multiperiod inventory routing problem with production decisions. *Comput. Ind. Eng.*, 57(3):713–723, 2009b.

J. F. Bard and N. Nananukul. A branch-and-price algorithm for an integrated production and inventory routing problem. *Comput. Oper. Res.*, 37(12):2202–2217, 2010.

C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Oper. Res.*, 46(3):316–329, 1998.

W. J. Bell, L. M. Dalberto, M. L. Fisher, A. J. Greenfield, R. Jaikumar, P. Kedia, R. G. Mack, and P. J. Prutzman. Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. *Interfaces*, 13(6): 4–23, 1983.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

L. Bertazzi, A. Bosco, F. Guerriero, and D. Laganà. A stochastic inventory routing problem with stock-out. *Transport. Res. C-Emer.*, Article in Press, 2011.

J. R. Birge and F. V. Louveaux. A multicut algorithm for two-stage stochastic linear programs. *Eur. J. Oper. Res.*, 34(3):384–392, 1988.

J. R. Birge and F. V. Louveaux. Two-stage recourse problems. In *Introduction to Stochastic Programming*, Springer Series in Operations Research and Financial Engineering, pages 181–263. Springer New York, 2011.

M. Boudia and C. Prins. A memetic algorithm with dynamic population management for an integrated production-distribution problem. *Eur. J. Oper. Res.*, 195(3):703–715, 2009.

M. Boudia, M. A. O. Louly, and C. Prins. Combined optimization of production and distribution. In *CD-ROM Proceedings of the International Conference on Industrial Engineering and Systems Management (IESM'05), Marrakech, Morocco*, 16-19 May 2005.

M. Boudia, M. A. O. Louly, and C. Prins. A reactive GRASP and path relinking for a combined production-distribution problem. *Comput. Oper. Res.*, 34(11):3402–3419, 2007.

M. Boudia, M. A. O. Louly, and C. Prins. Fast heuristics for a combined production planning and vehicle routing problem. *Prod. Plan. Control*, 19(2):85–96, 2008.

J. Bramel and D. Simchi-Levi. Set-covering-based algorithms for the capacitated vrp. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problems*, pages 85–108. Society for Industrial and Applied Mathematics, 2001.

G. Brown, J. Keegan, B. Vigus, and K. Wood. The Kellogg company optimizes production, inventory, and distribution. *Interfaces*, 31(6):1–15, 2001.

A. M. Campbell and M. W. P. Savelsbergh. A decomposition approach for the inventory-routing problem. *Transportation Sci.*, 38(4):488–502, 2004.

W. B. Carlton and J. W. Barnes. Solving the traveling-salesman problem with time windows using tabu search. *IIE Trans.*, 28(8):617–629, 1996.

M. Carter, J. Farvolden, G. Laporte, and J. Xu. Solving an integrated logistics problem arising in grocery distribution. *INFOR*, 34:290–306, 1996.

S. Çetinkaya, H. Uster, G. Easwaran, and B. B. Keskin. An integrated outbound logistics model for Frito-Lay: Coordinating aggregate-level production and distribution decisions. *Interfaces*, 39(5):460–475, September 1, 2009.

S. Chand, V. N. Hsu, S. Sethi, and V. Deshpande. A dynamic lot sizing problem with multiple customers: Customer-specific shipping and backlogging costs. *IIE Trans.*, 39(11):1059–1069, 2007.

P. Chandra. A dynamic distribution model with warehouse and customer replenishment requirements. *J. Oper. Res. Soc.*, 44(7):681–692, 1993.

P. Chandra and M. Fisher. Coordination of production and distribution planning. *Eur. J. Oper. Res.*, 72(3):503–517, 1994.

M. Christiansen. Decomposition of a combined inventory and time constrained ship routing problem. *Transportation Sci.*, 33(1):3–16, 1999.

G. Clarke and J. W. Wright. Scheduling of vehicles from a central depot to a number of delivery points. *Oper. Res.*, 12(4):568–581, 1964.

G. Codato and M. Fischetti. Combinatorial Benders' cuts for mixed-integer linear programming. *Oper. Res.*, 54(4):756–766, 2006.

L. C. Coelho, J.-F. Cordeau, and G. Laporte. Consistency in multi-vehicle inventory-routing. *Transport. Res. C-Emer.*, 24:270–287, 2012a.

L. C. Coelho, J.-F. Cordeau, and G. Laporte. Dynamic and stochastic inventory-routing. CIRRELT Tech. Rep. 2012-37, HEC Montréal, Canada, 2012b.

L. C. Coelho, J.-F. Cordeau, and G. Laporte. The inventory-routing problem with transshipment. *Comput. Oper. Res.*, 39(11):2537–2548, 2012c.

J.-F. Cordeau, F. Soumis, and J. Desrosiers. Simultaneous assignment of locomotives and cars to passenger trains. *Oper. Res.*, 49(4):531–548, 2001.

J.-F. Cordeau, F. Pasin, and M. Solomon. An integrated model for logistics network design. *Ann. Oper. Res.*, 144(1):59–82, 2006.

G. B. Dantzig and J. H. Ramser. The truck dispatching problem. *Management Sci.*, 6(1):80–91, 1959.

R. de Camargo, G. de Miranda Jr., and H. Luna. Benders decomposition for the uncapacitated multiple allocation hub location problem. *Comput. Oper. Res.*, 35 (4):1047–1064, 2008.

R. S. de Camargo, G. de Miranda Jr., and R. P. Ferreira. A hybrid outer-approximation/Benders decomposition algorithm for the single allocation hub location problem under congestion. *Oper. Res. Lett.*, 39(5):329–337, 2011.

Z. Degraeve, W. Gochet, and R. Jans. Alternative formulations for a layout problem in the fashion industry. *Eur. J. Oper. Res.*, 143(1):80–93, 2002.

G. D. Eppen and R. K. Martin. Solving multi-item capacitated lot-sizing problems using variable redefinition. *Oper. Res.*, 35(6):832–848, 1987.

A. Federgruen and M. Tzur. Time-partitioning heuristics: Application to one warehouse, multi-item, multi-retailer lot-sizing problems. *Naval Res. Logist.*, 46(5): 463–486, 1999.

A. Federgruen and P. Zipkin. A combined vehicle routing and inventory allocation problem. *Oper. Res.*, 32(5):1019–1037, 1984.

T. A. Feo and M. G. C. Resende. Greedy randomized adaptive search procedures. *J. Global Optim.*, 6(2):109–133, 1995.

M. Fischetti and A. Lodi. Local branching. *Math. Programming*, 98(1):23–47, 2003.

M. Fischetti, J. J. Salazar-González, and P. Toth. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Oper. Res.*, 45(3):378–394, 1997.

M. L. Fisher. The Lagrangian relaxation method for solving integer programming problems. *Management Sci.*, 27(1):1–18, 1981.

B. Fortz and M. Poss. An improved Benders decomposition applied to a multi-layer network design problem. *Oper. Res. Lett.*, 37(5):359–364, 2009.

F. Fumero and C. Vercellis. Synchronized development of production, inventory, and distribution schedules. *Transportation Sci.*, 33(3):330–340, 1999.

V. Gaur and M. L. Fisher. A periodic inventory routing problem at a supermarket chain. *Oper. Res.*, 52(6):813–822, 2004.

M. Gendreau, A. Hertz, and G. Laporte. New insertion and postoptimization procedures for the traveling salesman problem. *Oper. Res.*, 40(6):1086–1094, 1992.

M. Gendreau, G. Laporte, and F. Semet. A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32(4):263–273, 1998.

M. Gendreau, G. Laporte, and J.-Y. Potvin. Metaheuristics for the capacitated VRP. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problems*, pages 129–154. Society for Industrial and Applied Mathematics, 2001.

A. M. Geoffrion and G. W. Graves. Multicommodity distribution system design by Benders decomposition. *Management Sci.*, 20(5):822–844, 1974.

F. Glover. Tabu search–Part I. *ORSA J. Comput.*, 1(3):190–206, 1989.

F. Glover. Tabu search and adaptive memory programing - advances, applications and challenges. In R. Barr, R. Helgason, and J. Kennington, editors, *Interfaces in Computer Science and Operations Research*, pages 1–75. Kluwer Academic Publishers, Boston, 1996.

Y. T. Herer and M. Tzur. The dynamic transshipment problem. *Naval Res. Logist.*, 48(5):386–408, 2001.

W. Hopp and M. Spearman. *Factory Physics.* Irwin/McGraw-Hill, NY, USA, second edition, 2000.

V. N. Hsu, C.-L. Li, and W.-Q. Xiao. Dynamic lot size problems with one-way product substitution. *IIE Trans.*, 37(3):201–215, 2005.

L. M. Hvattum and A. Løkketangen. Using scenario trees and progressive hedging for stochastic inventory routing problems. *J. Heuristics*, 15(6):527–557, 2009.

L. M. Hvattum, A. Løkketangen, and G. Laporte. Scenario tree-based heuristics for stochastic inventory-routing problems. *INFORMS J. Comput.*, 21(2):268–285, 2009.

P. Jaillet, J. F. Bard, L. Huang, and M. Dror. Delivery cost approximations for inventory routing problems in a rolling horizon framework. *Transportation Sci.*, 36 (3):292–300, 2002.

R. Jans. Solving lot-sizing problems on parallel identical machines using symmetry-breaking constraints. *INFORMS J. Comput.*, 21(1):123–136, 2009.

R. Jans and Z. Degraeve. Modeling industrial lot sizing problems: A review. *Int. J. Prod. Res.*, 46(6):1619–1643, 2008.

W. Jaruphongsa and C.-Y. Lee. Dynamic lot-sizing problem with demand time windows and container-based transportation cost. *Optim. Lett.*, 2(1):39–51, 2008.

W. Jaruphongsa, S. Çetinkaya, and C.-Y. Lee. Outbound shipment mode considerations for integrated inventory and delivery lot-sizing decisions. *Oper. Res. Lett.*, 35(6):813–822, 2007.

Y. Jin and A. Muriel. Single-warehouse multi-retailer inventory systems with full truckload shipments. *Naval Res. Logist.*, 56(5):450–464, 2009.

A. J. Kleywegt, V. S. Nori, and M. W. P. Savelsbergh. The stochastic inventory routing problem with direct deliveries. *Transportation Sci.*, 36(1):94–118, 2002a.

A. J. Kleywegt, A. Shapiro, and T. H. de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM J. Optim.*, 12(2):479–502, 2002b.

A. J. Kleywegt, V. S. Nori, and M. W. P. Savelsbergh. Dynamic programming approximations for a stochastic inventory routing problem. *Transportation Sci.*, 38 (1):42–70, 2004.

J. Krarup and O. Bilde. Plant location, set covering and economic lot size: An $O(mn)$-algorithm for structured problems. In L. Collatz and W. Wetterling, editors, *International Series of Numerical Mathematics*, volume 36, pages 155–180. Birkhaeuser, Basel, 1977.

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

G. Laporte, R. Musmanno, and F. Vocaturo. An adaptive large neighbourhood search heuristic for the capacitated arc-routing problem with stochastic demands. *Transportation Sci.*, 44(1):125–135, 2010.

W.-S. Lee, J.-H. Han, and S.-J. Cho. A heuristic algorithm for a multi-product dynamic lot-sizing and shipping problem. *Int. J. Prod. Econ.*, 98(2):204–214, 2005.

L. Lei, S. Liu, A. Ruszczynski, and S. Park. On the integrated production, inventory, and distribution routing problem. *IIE Trans.*, 38(11):955–970, 2006.

A. N. Letchford and J. J. Salazar-González. Projection results for vehicle routing. *Math. Programming*, 105(2):251–274, 2006.

C.-L. Li, V. N. Hsu, and W.-Q. Xiao. Dynamic lot sizing with batch ordering and truckload discounts. *Oper. Res.*, 52(4):639–654, 2004.

S. Lin. Computer solutions of the traveling salesman problem. *Bell Syst. Tech. J.*, 44 (10):2245–2269, 1965.

M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Oper. Res.*, 53(6):1007–1023, 2005.

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

T. L. Magnanti and R. T. Wong. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Oper. Res.*, 29(3):464–484, 1981.

D. McDaniel and M. Devine. A modified Benders' partitioning algorithm for mixed integer programming. *Management Sci.*, 24(3):312–319, 1977.

S. Michel and F. Vanderbeck. A column-generation based tactical planning method for inventory routing. *Oper. Res.*, 60(2):382–397, 2012.

C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *J. Assoc. Comput. Mach.*, 7(4):326–329, 1960.

P. Moscato. Memetic algorithms: A short introduction. In *New Ideas in Optimization*, pages 219–234. McGraw-Hill Ltd., UK, 1999.

N. Nananukul. *Lot-Sizing and Inventory Routing for Production, Inventory and Distribution Systems.* PhD thesis, Graduate Program in Operations Research and Industrial Engineering, The University of Texas, Austin, 2008.

J. Naoum-Sawaya and S. Elhedhli. An interior-point Benders based branch-and-cut algorithm for mixed integer programs. *Ann. Oper. Res.*, Article in Press:1–23, 2010.

D. Pisinger and S. Ropke. A general heuristic for vehicle routing problems. *Comput. Oper. Res.*, 34(8):2403–2435, 2007.

Y. Pochet and L. A. Wolsey. Single-item uncapacitated lot-sizing. In *Production Planning by Mixed Integer Programming*, Springer Series in Operations Research and Financial Engineering, pages 207–234. Springer New York, 2006.

C. Prins. Private communication. 2012.

G. M. Ribeiro and G. Laporte. An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Comput. Oper. Res.*, 39(3): 728–735, 2012.

N. Rizk, A. Martel, and A. Ramudhin. A Lagrangean relaxation algorithm for multi-item lot-sizing problems with joint piecewise linear resource costs. *Int. J. Prod. Econ.*, 102(2):344–357, 2006.

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

M. Ruokokoski, O. Solyalı, J.-F. Cordeau, R. Jans, and H. Süral. Efficient formulations and a branch-and-cut algorithm for a production-routing problem. GERAD Tech. Rep. G-2010-66, HEC Montréal, Canada, 2010.

A. Rusdiansyah and D. Tsao. An integrated model of the periodic delivery problems for vending-machine supply chains. *J. Food. Eng.*, 70(3):421–434, 2005.

M. W. P. Savelsbergh and J.-H. Song. Inventory routing with continuous moves. *Comput. Oper. Res.*, 34(6):1744–1763, 2007.

M. W. P. Savelsbergh and J.-H. Song. An optimization algorithm for the inventory routing problem with continuous moves. *Comput. Oper. Res.*, 35(7):2266–2282, 2008.

P. Shaw. A new local search algorithm providing high quality solutions to vehicle routing problems. Technical report, Department of Computer Science, University of Strathclyde, Scotland, 1997.

H. D. Sherali and J. C. Smith. Improving discrete model representations via symmetry considerations. *Management Sci.*, 47(10):1396–1407, 2001.

O. Solyalı and H. Süral. A relaxation based solution approach for the inventory control and vehicle routing problem in vendor managed systems. In S. K. Neogy, A. K. Das, and R. B. Bapat, editors, *Modeling, Computation and Optimization*, pages 171–189. World Scientific, 2009.

O. Solyalı and H. Süral. A branch-and-cut algorithm using a strong formulation and an a priori tour-based heuristic for an inventory-routing problem. *Transportation Sci.*, 45(3):335–345, 2011.

O. Solyalı and H. Süral. The one-warehouse multi-retailer problem: Reformulation, classification, and computational results. *Ann. Oper. Res.*, 196(1):517–541, 2012.

O. Solyalı, J.-F. Cordeau, and G. Laporte. Robust inventory routing under demand uncertainty. *Transportation Sci.*, 46(3):327–340, 2012.

K. Sörensen and M. Sevaux. MA|PM: Memetic algorithms with population management. *Comput. Oper. Res.*, 33(5):1214–1225, 2006.

P. Toth and D. Vigo. An overview of vehicle routing problems. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, pages 1–26. SIAM Monographs on Discrete Mathematics and Applications, PA, 2001.

L. van Norden and S. van de Velde. Multi-product lot-sizing with a transportation capacity reservation contract. *Eur. J. Oper. Res.*, 165(1):127–138, 2005.

R. Van Slyke and R. Wets. *L*-shaped linear programs with applications to optimal control and stochastic programming. *SIAM J. Appl. Math.*, 17(4):638–663, 1969.

H. M. Wagner and T. M. Whitin. Dynamic version of the economic lot size model. *Management Sci.*, 5(1):89–96, 1958.

# Appendix A
# Supplement to Chapter 3

Section A.1 provides the details of MVPRP and MVIRP instances and Section A.2 presents the detailed results of the computational experiments discussed in Section 3.6 of Chapter 3. More details on the instances and the solutions can be found on the website https://sites.google.com/site/YossiriAdulyasak/publications.

## A.1 Details of the Instances

This section presents the details of the two test sets used in our computational experiments.

### A.1.1 Details of the MVPRP Instances

The Archetti et al. (2011) dataset consists of instances with six periods. Each problem size contains four classes and each instance type has five instances with different node coordinates. The first class contains standard instances. The second and the third classes are identical to the first but with higher unit production costs and higher transportation costs, respectively. The fourth class consists of instances from the first and second classes but with no customer inventory cost. We generated our instances using those from the four classes to ensure that we consider different problem characteristics. Instances with 50 customers were used to generate our dataset. We created instances from $n = 10$ to 50, to 40 and to 30 customers with an increment of 5 for time horizons with $l = 3, 6$ and 9 periods, respectively. The number of vehicles is set to $m = 2$ or 3 for the instances with $n \leq 25$ and to $m = 3$ or 4 for the instances with $25 < n \leq 50$. There are four instances per problem size which results in $42 \times 4 = 168$ instances in total or $168 \times 2 = 336$ instances for both the ML and OU policy.

The MVPRP instances are generated using the instances presented in Archetti et al. (2011). We select the instances with 50 customers to generate our test bed. The details are as follows.

1. **Instance size**. We create instances from $n = 10$ to 50, to 40 and to 30 customers with an increment of 5 for the time periods $l = 3, 6$ and 9, respectively.

2. **Customers**: To generate a new instance with $n$ customers, we select customer numbers from 1 to $n$ from the Archetti et al. instance. For the instances with 3 periods, initial inventory levels at customers are all reduced by a factor of 2 to prevent the case where initial inventory levels are already sufficient to satisfy the demand during the full planning horizon. Also, since the target stock level in the Archetti et al. test set is defined by the inventory after consumption, we simply set $L_i = \bar{L}_i + d_{it}$, where $\bar{L}_i$ is the original value in the Archetti et al. test set. This does not have any effect since the values of $d_{it}$ for each customer in the Archetti et al. test set are constant during the planning horizon.

3. **Vehicles**. Since the number of vehicles in the original instances is unlimited, we set the number of vehicles equal to $m = 2$ or 3 for the instances with $n \leq 25$, and $m = 3$ or 4 for the instances with $25 < n \leq 50$. Similar to Archetti et al., we set the vehicle capacity related to the maximum inventory level at the customers. The vehicle capacity is calculated as $Q = \lfloor 1.5\alpha \max_{i \in N_c}\{\bar{L}_i\}/m \rfloor$, where $\alpha = \lfloor n/10 \rfloor + 1$. In our test, this setting is appropriate since a smaller vehicle capacity could lead to infeasible solutions, especially for the MVPRP-OU.

4. **Instance number.** We generate four instances per instance size. Each instance is taken from a different instance class in the Archetti et al. dataset to ensure that different characteristics are captured. We select the original instance type 1, 25, 49 and 73 to create our test bed.

5. **Production capacity.** In the Archetti et al. dataset, they considered uncapacitated production and unlimited inventory capacity at the plant. Since we also consider production capacity in our study, the production capacity is set to

$C = \left\lfloor 2 \sum_{i \in N_c} \sum_{t \in T} d_{it}/l \right\rfloor$ and plant inventory capacity is set to $L_0 = \lfloor C/2 \rfloor$. In our preliminary tests, we found that a smaller production capacity could lead to stockouts in the MVPRP-OU.

We summarize the parameters of the instance set in Table A.I.

Table A.I: Characteristics of the MVPRP Instance Sets

| $n$ | $l$ | $m$ | $C$ | $L_0$ | $Q$ |
|---|---|---|---|---|---|
| 10 | 3/6/9 | 2 | 304 | 152 | 198 |
| 10 | 3/6/9 | 3 | 304 | 152 | 132 |
| 15 | 3/6/9 | 2 | 470 | 235 | 198 |
| 15 | 3/6/9 | 3 | 470 | 235 | 132 |
| 20 | 3/6/9 | 2 | 540 | 270 | 283 |
| 20 | 3/6/9 | 3 | 540 | 270 | 189 |
| 25 | 3/6/9 | 2 | 700 | 350 | 283 |
| 25 | 3/6/9 | 3 | 700 | 350 | 189 |
| 30 | 3/6/9 | 3 | 768 | 384 | 228 |
| 30 | 3/6/9 | 4 | 768 | 384 | 171 |
| 35 | 3/6 | 3 | 948 | 474 | 276 |
| 35 | 3/6 | 4 | 948 | 474 | 207 |
| 40 | 3/6 | 3 | 1256 | 628 | 360 |
| 40 | 3/6 | 4 | 1256 | 628 | 216 |
| 45 | 3 | 3 | 1438 | 719 | 360 |
| 45 | 3 | 4 | 1438 | 719 | 207 |
| 50 | 3 | 3 | 1348 | 674 | 360 |
| 50 | 3 | 4 | 1348 | 674 | 270 |

### A.1.2 Details of the MVIRP Instances

We have also adapted the IRP instances for the single vehicle case which are presented in Archetti et al. (2007). This instance set was used in several studies, e.g., Archetti et al. (2007, 2012); Solyalı and Süral (2011) and Coelho et al. (2012a,c). The set consists of instances with 5 to 50 customers (with an increment of 5) with 3 periods, and 5 to 30 customers with 6 periods. There are two main groups, i.e., low inventory costs and high inventory costs, and five instances per instance size in each group. We used the same method as Coelho et al. (2012a) to generate the MVIRP instances, i.e., the original vehicle capacity is divided by the desired number of vehicles and rounded to the nearest integer value. We used the instances with 5 to 50 customers from the instances with 3 periods and 5 to 25 customers from the instances with 6 periods in our experiments. The number of vehicles is set using the same method as for the

MVPRP instances. There are $30 \times 10 = 300$ instances in total or $300 \times 2 = 600$ instances for both the ML and OU policy.

As we mentioned earlier, the timing of the replenishment process in our paper and Archetti et al. (2007) are different. In Archetti et al., the inventory level is considered at the beginning of the period. We denote by $s_{it}$ the beginning inventory level at node $i$ in period $t$ and we can observe that

$$s_{it} = I_{i,t-1} \quad \forall i \in N, \forall t \in T. \tag{A.1}$$

The following modifications are used to convert the instances of Archetti et al. (2007) for our formulation. For ease of presentation, we use the variables without vehicle index and they can be easily converted to vehicle index formulations using $q_{it} = \sum_{k \in K} q_{ikt}$.

First, in the Archetti et al. paper, the supplier can only use the inventory at the beginning of the period to replenish the customers and the supplier must ensure that the total amount shipped to retailers in a period cannot exceed the available amount at the supplier in the beginning of that period, i.e.,

$$s_{0t} \geq \sum_{i \in N_c} q_{it} \quad \forall t \in T. \tag{A.2}$$

Replacing $s_{0t}$ by $I_{0,t-1}$ and from (3.22) and the production capacity constraints for the IRP in Section 3.2.4, the constraints above can be rewritten as

$$I_{0t} \geq B_t \quad \forall t \in T, \tag{A.3}$$

and these constraints are added to our formulations.

Second, since the inventory costs are charged at the beginning of the period starting from period one, where $s_{i1} = I_{i0}$, the fixed cost $\sum_{i \in N} h_i I_{i0}$ is simply added to the objective function in our formulations.

The other parts of the formulations remain unchanged.

## A.2 Detailed Computational Results of the Exact Algorithms

The results on the effect of the valid inequalities are presented in Section A.2. Section A.2 presents the results of the experiments performed on a multi-core processor and Section A.2 presents the results on the single-vehicle instances.

### A.2.1 Detailed Results on the Effect of the Valid Inequalities

#### A.2.1.1 Effect of Vehicle Symmetry Breaking Constraints on the Vehicle Index Formulations

The results on the MVPRP and MVIRP instances that are solved to optimality are shown in Tables A.II and A.III, respectively, and the results on the instances that could not be solved to optimality are reported in Tables A.IV and A.V, respectively. The column $h_i$ in Tables A.III and A.V indicates the group of the MVIRP instances, i.e., low (L) or high (H) inventory costs.

Table A.II: Results of Using Different SBCs on the MVPRP Instances Solved to Optimality

| $n$ $l$ $m$ | None | | SBC0 | | SBC0+1 | | SBC0+2 | | SBC0+3 | | SBC0+4 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CPU | Node | CPU | Node | CPU | Node | CPU | Node | CPU | Node | CPU | Node |
| MVPRP-ML | | | | | | | | | | | | |
| 10 3 2 | 0.4 | 12 | 0.2 | 3 | **0.2** | 3 | 0.3 | 2 | **0.2** | 1 | 0.3 | 1 |
| 10 3 3 | 2.1 | 88 | 1.2 | 24 | 1.1 | 13 | 1.1 | 15 | **0.6** | 11 | 0.6 | 14 |
| 15 3 2 | 4.6 | 109 | 3.2 | 55 | **2.1** | 21 | 3.2 | 58 | 2.8 | 63 | 2.8 | 66 |
| 15 3 3 | 163.8 | 3614 | 54.9 | 884 | 34.7 | 486 | 55.6 | 839 | 33.6 | 537 | **28.7** | 486 |
| 10 6 2 | 9.0 | 343 | 3.2 | 31 | 4.0 | 40 | 3.2 | 33 | **1.7** | 20 | 2.4 | 34 |
| 10 6 3 | 137.8 | 3588 | 31.2 | 675 | 23.1 | 337 | 23.9 | 423 | **11.8** | 221 | 14.9 | 295 |
| 15 6 2 | 227.7 | 2435 | 190.7 | 1971 | 85.6 | 719 | 176.0 | 1438 | 91.2 | 957 | **94.4** | 873 |
| 15 6 3 | 2370.9$^{(2)}$ | 12239 | 2036.2$^{(2)}$ | 8669 | 1426.4$^{(1)}$ | 5449 | 1615.6$^{(1)}$ | 6709 | 926.7 | 5090 | **829.8** | 4586 |
| 10 9 2 | 83.0 | 1815 | 51.5 | 981 | 26.5 | 299 | 36.7 | 513 | **18.2** | 274 | 23.4 | 350 |
| 10 9 3 | 3600.0$^{(4)}$ | 37039 | 1799.3$^{(1)}$ | 22449 | 558.2 | 6377 | 1406.8$^{(1)}$ | 16613 | 529.3 | 7557 | **474.8** | 6967 |
| Average | 659.9 | 6128 | 417.1 | 3574 | 216.2 | 1374 | 332.2 | 2664 | 161.6 | 1473 | **147.2** | 1367 |
| MVPRP-OU | | | | | | | | | | | | |
| 10 3 2 | 0.6 | 12 | 0.6 | 13 | 0.6 | 10 | 0.6 | 19 | 0.6 | 19 | **0.5** | 8 |
| 10 3 3 | 1.4 | 85 | 1.2 | 77 | **1.1** | 51 | 1.7 | 80 | 1.2 | 47 | 1.2 | 36 |
| 15 3 2 | 43.4 | 1775 | 38.1 | 1557 | 30.7 | 1037 | 56.7 | 2033 | **21.3** | 745 | 30.7 | 1064 |
| 15 3 3 | 62.1 | 1616 | 59.6 | 1395 | 68.4 | 1249 | 60.6 | 1166 | 20.0 | 328 | **18.1** | 287 |
| 10 6 2 | 0.7 | 7 | **0.6** | 3 | **0.6** | 5 | 0.7 | 3 | **0.6** | 4 | **0.6** | 3 |
| 10 6 3 | 2.3 | 24 | 2.0 | 13 | 2.1 | 15 | 2.0 | 11 | 1.7 | 11 | **1.5** | 8 |
| 15 6 2 | 15.4 | 165 | 13.6 | 136 | 14.7 | 101 | 15.0 | 119 | **10.8** | 86 | 13.0 | 123 |
| 15 6 3 | 188.7 | 1466 | 191.7 | 1372 | 97.5 | 540 | 165.7 | 1020 | 95.4 | 587 | **85.9** | 493 |
| 10 9 2 | 26.1 | 519 | 18.6 | 355 | 17.5 | 226 | 15.4 | 226 | **11.6** | 164 | 14.1 | 203 |
| 10 9 3 | 499.5 | 7228 | 339.7 | 4987 | 81.4 | 758 | 233.8 | 2448 | **96.2** | 915 | 94.0 | 973 |
| 15 9 2 | 1847.8$^{(1)}$ | 10466 | 1982.8$^{(2)}$ | 9917 | 1165.0 | 5943 | 2006.4$^{(2)}$ | 11154 | **1038.4** | 5740 | 1868.6$^{(1)}$ | 10087 |
| Average | 244.3 | 2124 | 240.8 | 1802 | 134.5 | 903 | 232.6 | 1662 | **118.0** | 786 | 193.5 | 1208 |

$^{(-)}$ indicates the number of instances (out of 4) were not solved to optimality

Table A.III: Results of Using Different SBCs on the MVIRP Instances Solved to Optimality

| $n$ | $l$ | $m$ | $h_i$ | None | | SBC0 | | SBC0+1 | | SBC0+2 | | SBC0+3 | | SBC0+4 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | CPU | Node | CPU | Node | CPU | Node | CPU | Node | CPU | Node | CPU | Node |
| MVIRP-ML | | | | | | | | | | | | | | | |
| 5 | 3 | 2 | L | 0.3 | 145 | 0.2 | 76 | 0.2 | 45 | 0.2 | 63 | **0.2** | 31 | **0.2** | 34 |
| 5 | 3 | 3 | L | 3.7 | 2377 | 1.1 | 361 | 0.7 | 167 | 1.2 | 391 | **0.4** | 130 | **0.4** | 159 |
| 10 | 3 | 2 | L | 2.6 | 280 | 2.0 | 169 | 2.1 | 178 | 2.1 | 161 | **1.4** | 89 | 1.7 | 132 |
| 10 | 3 | 3 | L | 140.2 | 14127 | 51.9 | 4281 | 28.5 | 1850 | 42.3 | 2955 | **11.7** | 847 | 13.1 | 951 |
| 15 | 3 | 2 | L | 16.3 | 622 | 7.0 | 225 | 9.9 | 215 | 9.9 | 215 | **5.9** | 146 | 7.4 | 187 |
| 15 | 3 | 3 | L | 723.6 | 18118 | 186.8 | 4929 | 101.0 | 2180 | 225.1 | 5329 | **39.5** | 1060 | 41.7 | 1047 |
| 5 | 6 | 2 | L | 25.6 | 10400 | 17.7 | 6009 | 5.6 | 1854 | 11.7 | 4352 | **4.0** | 1177 | **4.0** | 1241 |
| 5 | 6 | 3 | L | 3600.0$^{(5)}$ | 694135 | 3019.3$^{(2)}$ | 645199 | 217.5 | 55557 | 2470.2$^{(2)}$ | 503803 | 130.5 | 34304 | **117.7** | 32258 |
| 10 | 6 | 2 | L | 1048.9 | 53129 | 1111.7 | 50608 | 231.6 | 11070 | 550.3 | 23550 | **117.0** | 6229 | 168.7 | 7936 |
| 15 | 6 | 2 | L | 2956.0$^{(3)}$ | 33269 | 2258.2$^{(2)}$ | 24193 | 1408.5 | 15004 | 2269.1$^{(1)}$ | 27991 | **718.8** | 10593 | 843.3 | 11515 |
| 5 | 3 | 2 | H | 0.3 | 123 | 0.3 | 86 | 0.2 | 52 | 0.3 | 53 | 0.2 | 20 | **0.1** | 28 |
| 5 | 3 | 3 | H | 2.9 | 1893 | 1.1 | 411 | 0.8 | 221 | 1.1 | 305 | **0.3** | 80 | 0.4 | 139 |
| 10 | 3 | 2 | H | 2.7 | 270 | 2.4 | 225 | 2.1 | 164 | 2.2 | 136 | 1.9 | 128 | **1.2** | 76 |
| 10 | 3 | 3 | H | 92.1 | 8841 | 38.2 | 2534 | 24.9 | 1653 | 45.0 | 2928 | **10.3** | 744 | 12.4 | 976 |
| 15 | 3 | 2 | H | 13.3 | 458 | 7.8 | 214 | 7.4 | 147 | 9.9 | 264 | **4.8** | 99 | 7.1 | 168 |
| 15 | 3 | 3 | H | 565.0 | 14989 | 174.5 | 4171 | 75.3 | 1662 | 162.1 | 4383 | 42.5 | 891 | **38.9** | 896 |
| 5 | 6 | 2 | H | 21.2 | 8365 | 16.1 | 5201 | 4.2 | 1274 | 9.0 | 3164 | **2.7** | 851 | 3.8 | 1113 |
| 5 | 6 | 3 | H | 3423.6$^{(4)}$ | 713140 | 2561.2$^{(1)}$ | 530888 | 219.7 | 42621 | 2326.6$^{(2)}$ | 486730 | **80.6** | 21290 | 84.2 | 25457 |
| 10 | 6 | 2 | H | 763.6 | 38379 | 706.9 | 33873 | 164.5 | 8268 | 554.7 | 24460 | **118.4** | 6185 | 165.6 | 7878 |
| 15 | 6 | 2 | H | 2746.0$^{(2)}$ | 32886 | 1993.5$^{(2)}$ | 24270 | 958.9 | 10837 | 1714.7$^{(1)}$ | 18797 | 476.8 | 6461 | **466.1** | 6888 |
| Average | | | | 807.4 | 82297 | 607.9 | 66896 | 173.2 | 7751 | 520.4 | 55501 | **88.4** | 4568 | 98.9 | 4954 |
| MVIRP-OU | | | | | | | | | | | | | | | |
| 5 | 3 | 2 | L | 0.1 | 2 | 0.1 | 0 | 0.1 | 0 | 0.2 | 0 | **0.0** | 0 | 0.1 | 0 |
| 5 | 3 | 3 | L | 0.1 | 5 | 0.1 | 3 | 0.2 | 0 | 0.2 | 4 | **0.0** | 0 | **0.0** | 0 |
| 10 | 3 | 2 | L | 3.6 | 336 | 3.3 | 277 | 3.3 | 221 | 4.2 | 328 | **2.4** | 156 | 2.7 | 182 |
| 10 | 3 | 3 | L | 15.1 | 806 | 10.7 | 483 | 13.7 | 531 | 16.8 | 673 | **5.2** | 171 | 5.5 | 205 |
| 15 | 3 | 2 | L | 10.8 | 239 | 10.4 | 220 | 10.0 | 181 | 15.7 | 286 | **6.8** | 104 | 7.8 | 140 |
| 15 | 3 | 3 | L | 111.0 | 1338 | 51.5 | 657 | 58.4 | 621 | 79.0 | 998 | 29.6 | 298 | **20.6** | 204 |
| 5 | 6 | 2 | L | 2.3 | 236 | 2.6 | 265 | 1.7 | 120 | 2.0 | 217 | **1.4** | 85 | **1.4** | 100 |
| 5 | 6 | 3 | L | 1.1 | 153 | 1.1 | 142 | 0.6 | 29 | 0.6 | 36 | **0.1** | 0 | 0.2 | 1 |
| 10 | 6 | 2 | L | 957.8 | 31479 | 607.4 | 19279 | 285.8 | 9416 | 553.7 | 18265 | **247.8** | 7758 | 279.2 | 9554 |
| 15 | 6 | 2 | L | 2764.9$^{(2)}$ | 23445 | 1948.2$^{(1)}$ | 15498 | 1182.7 | 9800 | 1920.3 | 16378 | 777.2 | 6281 | **757.4** | 6352 |
| 5 | 3 | 2 | H | 0.1 | 0 | 0.1 | 0 | 0.2 | 0 | 0.2 | 0 | **0.0** | 0 | **0.0** | 0 |
| 5 | 3 | 3 | H | 0.2 | 3 | 0.1 | 0 | 0.1 | 0 | 0.2 | 5 | **0.0** | 0 | 0.1 | 0 |
| 10 | 3 | 2 | H | 3.8 | 352 | 3.6 | 319 | 3.3 | 236 | 4.4 | 317 | 2.5 | 168 | **2.2** | 161 |
| 10 | 3 | 3 | H | 18.2 | 1010 | 10.4 | 544 | 8.7 | 331 | 12.5 | 507 | **5.5** | 192 | 6.5 | 246 |
| 15 | 3 | 2 | H | 11.9 | 276 | 9.5 | 230 | 11.5 | 212 | 12.9 | 299 | **6.4** | 118 | 7.8 | 128 |
| 15 | 3 | 3 | H | 94.9 | 1380 | 48.4 | 639 | 52.4 | 500 | 101.4 | 1244 | **21.3** | 216 | 23.6 | 242 |
| 5 | 6 | 2 | H | 2.2 | 262 | 2.5 | 263 | 1.6 | 100 | 2.2 | 230 | 1.4 | 101 | **1.1** | 86 |
| 5 | 6 | 3 | H | 0.6 | 37 | 0.8 | 80 | 0.7 | 30 | 0.6 | 21 | **0.2** | 0 | 0.2 | 1 |
| 10 | 6 | 2 | H | 772.4 | 27040 | 483.0 | 15476 | 280.2 | 9565 | 502.8 | 17132 | 265.5 | 8542 | **205.3** | 6690 |
| 15 | 6 | 2 | H | 2621.4$^{(2)}$ | 21957 | 2578.0$^{(2)}$ | 21691 | 927.6 | 7649 | 2163.0$^{(2)}$ | 17711 | **605.2** | 5012 | 769.5 | 6077 |
| Average | | | | 369.6 | 5518 | 288.6 | 3803 | 142.1 | 1977 | 269.6 | 3733 | **98.9** | 1460 | 104.6 | 1518 |

$^{(-)}$ indicates the number of instances (out of 5) were not solved to optimality

Table A.IV: Results of Using Different SBCs on the MVPRP Instances Not Solved to Optimality

| $n$ $l$ $m$ | None | | SBC0 | | SBC0+1 | | SBC0+2 | | SBC0+3 | | SBC0+4 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | %LB | Node | %LB | Node | %LB | Node | %LB | Node | %LB | Node | %LB | Node |
| MVPRP-ML | | | | | | | | | | | | |
| 15 9 2 | 99.1$^{(2)}$ | 15212 | 99.1$^{(2)}$ | 11403 | **99.5**$^{(2)}$ | 8103 | 99.3$^{(2)}$ | 10375 | 99.3$^{(2)}$ | 8247 | 99.3$^{(2)}$ | 8747 |
| 15 9 3 | 96.5$^{(4)}$ | 7090 | 96.6$^{(4)}$ | 7369 | 96.6$^{(4)}$ | 6511 | 96.6$^{(4)}$ | 6476 | **97.0**$^{(4)}$ | 7996 | 96.9$^{(4)}$ | 8381 |
| Average | 97.8 | 11151 | 97.9 | 9386 | **98.1** | 7307 | 97.9 | 8425 | **98.1** | 8121 | **98.1** | 8564 |
| MVPRP-OU | | | | | | | | | | | | |
| 15 3 9 | 97.6$^{(4)}$ | 8421 | 97.5$^{(4)}$ | 8978 | 97.7$^{(4)}$ | 7974 | 97.3$^{(4)}$ | 7991 | **98.2**$^{(4)}$ | 9041 | 97.8$^{(4)}$ | 7893 |

$^{(-)}$ indicates the number of instances (out of 4) were not solved to optimality

Table A.V: Results of Using Different SBCs on the MVIRP Instances Not Solved to Optimality

| $n$ $l$ $m$ $h_i$ | None | | SBC0 | | SBC0+1 | | SBC0+2 | | SBC0+3 | | SBC0+4 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | %LB | Node | %LB | Node | %LB | Node | %LB | Node | %LB | Node | %LB | Node |
| MVIRP-ML | | | | | | | | | | | | |
| 10 6 3 L | 87.5$^{(5)}$ | 59901 | 88.9$^{(5)}$ | 56175 | 90.5$^{(5)}$ | 55147 | 89.3$^{(5)}$ | 58882 | **95.3**$^{(3)}$ | 68276 | 91.5$^{(4)}$ | 70671 |
| 15 6 3 L | 85.4$^{(5)}$ | 13555 | 86.5$^{(5)}$ | 15036 | 88.7$^{(5)}$ | 12938 | 88.1$^{(5)}$ | 13729 | 93.1$^{(5)}$ | 16758 | **94.0**$^{(5)}$ | 17734 |
| 10 6 3 H | 91.9$^{(5)}$ | 58381 | 93.2$^{(5)}$ | 59372 | 94.6$^{(5)}$ | 56604 | 93.6$^{(5)}$ | 63415 | 96.8$^{(3)}$ | 64053 | **97.2**$^{(3)}$ | 75831 |
| 15 6 3 H | 92.3$^{(5)}$ | 13772 | 92.7$^{(5)}$ | 14097 | 94.1$^{(5)}$ | 12966 | 93.5$^{(5)}$ | 13312 | **97.2**$^{(5)}$ | 17237 | 96.7$^{(5)}$ | 18032 |
| Average | 89.3 | 36402 | 90.3 | 36170 | 92.0 | 34414 | 91.1 | 37335 | **95.6** | 41581 | 94.9 | 45567 |
| MVIRP-OU | | | | | | | | | | | | |
| 10 6 3 L | 89.0$^{(5)}$ | 47138 | 90.8$^{(5)}$ | 48530 | 92.5$^{(5)}$ | 41738 | 89.7$^{(5)}$ | 45128 | **97.4**$^{(2)}$ | 32919 | 97.0$^{(2)}$ | 35196 |
| 15 6 3 L | 82.8$^{(5)}$ | 12227 | 84.5$^{(5)}$ | 12943 | 84.0$^{(5)}$ | 11522 | 83.7$^{(5)}$ | 13729 | **90.9**$^{(5)}$ | 12041 | **90.9**$^{(5)}$ | 13393 |
| 10 6 3 H | 93.1$^{(5)}$ | 47267 | 94.3$^{(5)}$ | 50628 | 94.8$^{(5)}$ | 40254 | 93.4$^{(5)}$ | 45406 | 98.1$^{(2)}$ | 35484 | **98.6**$^{(2)}$ | 38091 |
| 15 6 3 H | 90.5$^{(5)}$ | 12820 | 91.1$^{(5)}$ | 13295 | 91.3$^{(5)}$ | 12103 | 90.7$^{(5)}$ | 13312 | **95.2**$^{(5)}$ | 12684 | 94.5$^{(5)}$ | 12392 |
| Average | 88.8 | 29863 | 90.2 | 31349 | 90.7 | 26404 | 89.4 | 29394 | **95.4** | 23282 | 95.3 | 24768 |

$^{(-)}$ indicates the number of instances (out of 5) were not solved to optimality

### A.2.1.2 Effect of Valid Inequalities for the Non-Vehicle Index Formulations

The average lower bounds of each instance size are shown in Tables A.VI and A.VII. The numbers presented are equal to the average lower bounds at the root node compared to the optimal solutions or the best upper bounds if the instances were not solved to optimality. Each column shows the results of using each cut presented in Section 3.3.2, where the columns *None* and *All* present the results without using any additional cuts (i.e., only GFSECs (3.28) and (3.53) are applied) and all cuts together, respectively. Note that there are some cases where using all cuts produces worse results than using one type of cut since heuristic separation algorithms are employed for the non-vehicle index formulations.

Table A.VI: Effects of the Valid Inequalities on Lower Bounds at the Root Node on the MVPRP Instances

| $n$ | $l$ | $m$ | MVPRP-ML | | | | | | MVPRP-OU | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | None | (3.55) | (3.57) | (3.58) | (3.60) | All | None | (3.56) | (3.57) | (3.59) | (3.60) | All |
| 10 | 3 | 2 | 98.0 | 98.0 | 98.8 | 98.0 | 98.9 | **100.0** | 96.9 | 97.1 | 97.2 | 97.2 | 96.9 | **97.3** |
| 10 | 3 | 3 | 97.2 | 96.9 | 98.2 | 97.2 | 97.5 | **98.7** | 96.6 | **96.8** | 96.6 | **96.8** | 96.6 | **96.8** |
| 15 | 3 | 2 | 96.4 | 96.4 | **98.2** | 96.5 | 96.4 | **98.2** | 88.0 | 91.1 | 88.3 | 89.0 | 88.0 | **91.8** |
| 15 | 3 | 3 | 95.1 | 95.2 | 97.1 | 95.4 | 95.7 | **97.6** | 88.4 | 90.9 | 88.3 | 89.4 | 88.2 | **91.3** |
| 10 | 6 | 2 | 89.5 | 89.5 | 91.3 | 89.8 | 89.7 | **91.6** | 94.5 | 94.5 | **94.8** | 94.5 | 94.5 | **94.8** |
| 10 | 6 | 3 | 88.9 | 88.9 | 90.4 | 89.1 | 89.5 | **91.0** | 94.4 | 94.4 | 94.6 | 94.4 | 94.5 | **94.7** |
| 15 | 6 | 2 | 90.4 | 90.5 | 92.1 | 90.6 | 90.6 | **92.4** | 94.5 | 94.6 | 94.9 | 94.6 | 94.6 | **94.9** |
| 15 | 6 | 3 | 89.2 | 89.5 | 90.9 | 90.1 | 89.6 | **91.5** | 93.9 | 94.0 | **94.3** | 94.1 | 93.9 | **94.3** |
| 10 | 9 | 2 | 95.0 | 95.0 | 96.2 | 95.1 | 95.3 | **96.5** | 96.2 | 96.2 | 97.2 | 96.5 | 96.3 | **97.3** |
| 10 | 9 | 3 | 94.3 | 94.3 | 95.4 | 94.5 | 94.5 | **96.3** | 95.5 | 95.6 | 96.3 | 96.0 | 95.7 | **96.6** |
| 15 | 9 | 2 | 93.3 | 93.3 | 94.6 | 93.4 | 93.5 | **94.9** | 94.9 | 95.2 | 96.1 | 95.3 | 95.1 | **96.5** |
| 15 | 9 | 3 | 92.2 | 92.1 | 93.1 | 92.7 | 92.8 | **94.0** | 94.4 | 94.5 | 95.3 | 94.8 | 94.7 | **95.6** |
| Average | | | 93.3 | 93.3 | 94.7 | 93.5 | 93.7 | **95.2** | 94.0 | 94.6 | 94.5 | 94.4 | 94.1 | **95.2** |

Tables A.VIII and A.IX present the effects of using all these cuts on our exact algorithms. The results of the branch-and-cut without and with the additional valid inequalities are shown in columns $F(ML)|nk$, $F(OU)|nk$ and in columns $F(ML)|nk^+$, $F(OU)|nk^+$, respectively.

## A.2.2 Detailed Results of the Branch-and-Cut Algorithm on Multi-Core Processors

In this section, we present the detailed results of the branch-and-cut algorithm using the vehicle index formulations on the remaining MVPRP and MVIRP instances in Tables A.X and A.XI. We report the average total aggregate CPU time and average wallclock time in hours (h) in columns *CPU(h)* and *WC(h)*, respectively, while the computing time of the Op-ALNS is reported in seconds (s). Column $C/W$ shows the ratio of the aggregate CPU time and wallclock time (CPU/WC). Note that the average CPU/WC ratio is 1.6 and 2.2 for the MVPRP and MVIRP, respectively.

Table A.VII: Effects of the Valid Inequalities on Lower Bounds at the Root Node on the MVIRP Instances

| $n$ | $l$ | $m$ | $h_i$ | MVPRP-ML | | | | | | MVPRP-OU | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | None | (3.55) | (3.57) | (3.58) | (3.60) | All | None | (3.56) | (3.57) | (3.59) | (3.60) | All |
| 5 | 3 | 2 | L | 86.6 | 86.6 | 87.6 | 86.8 | 92.9 | **93.9** | 92.1 | 92.0 | 93.7 | 92.5 | 94.1 | **95.8** |
| 5 | 3 | 3 | L | 83.2 | 83.7 | 84.3 | 84.7 | 88.2 | **89.5** | 86.6 | 86.0 | 87.1 | 86.2 | 88.3 | **88.8** |
| 10 | 3 | 2 | L | 86.0 | 86.0 | 86.9 | 86.0 | 90.7 | **91.3** | 81.0 | 82.1 | 84.7 | 82.7 | 83.2 | **85.2** |
| 10 | 3 | 3 | L | 78.8 | 79.9 | 83.7 | 81.0 | 88.1 | **89.1** | 82.3 | 83.5 | 87.0 | 85.8 | 85.5 | **89.5** |
| 15 | 3 | 2 | L | 86.3 | 86.5 | 89.4 | 86.5 | 90.1 | **92.1** | 80.2 | 81.5 | **86.5** | 81.6 | 82.1 | 85.4 |
| 15 | 3 | 3 | L | 81.4 | 81.4 | 86.2 | 81.8 | 85.4 | **88.9** | 80.7 | 81.3 | **87.7** | 82.0 | 81.4 | 86.9 |
| 5 | 6 | 2 | L | 75.1 | 75.6 | 76.4 | 79.5 | 82.4 | **83.1** | 83.4 | 85.0 | 86.3 | 86.2 | 86.3 | **87.8** |
| 5 | 6 | 3 | L | 82.3 | 83.0 | 83.8 | 83.9 | **87.3** | 86.4 | 85.4 | 84.5 | 84.5 | 85.0 | 85.8 | **86.6** |
| 10 | 6 | 2 | L | 72.8 | 73.3 | 74.8 | 75.5 | 78.1 | **79.4** | 80.0 | 80.0 | 81.8 | 82.5 | 84.1 | **86.6** |
| 10 | 6 | 3 | L | 73.2 | 72.8 | 75.6 | 76.9 | 80.1 | **82.7** | 76.6 | 78.1 | 80.8 | 81.5 | 80.9 | **83.2** |
| 15 | 6 | 2 | L | 72.2 | 71.9 | 75.7 | 73.2 | 74.7 | **79.2** | 81.1 | 81.4 | 85.2 | 82.4 | 82.4 | **87.2** |
| 15 | 6 | 3 | L | 69.9 | 69.2 | 72.7 | 71.8 | 71.3 | **77.6** | 75.8 | 75.3 | 78.8 | 77.2 | 76.1 | **81.1** |
| 5 | 3 | 2 | H | 91.5 | 91.5 | 92.2 | 91.7 | 95.5 | **96.0** | 94.6 | 94.6 | 95.7 | 95.0 | 96.1 | **97.2** |
| 5 | 3 | 3 | H | 88.6 | 88.6 | 89.4 | 89.0 | 92.0 | **93.0** | 90.4 | 90.1 | 90.9 | 90.2 | 91.8 | **92.2** |
| 10 | 3 | 2 | H | 93.3 | 93.3 | 93.8 | 93.3 | 95.7 | **95.7** | 90.1 | 90.6 | 92.0 | 90.9 | 91.1 | **92.4** |
| 10 | 3 | 3 | H | 88.8 | 89.3 | 91.4 | 89.9 | 93.2 | **94.2** | 89.7 | 90.5 | 92.6 | 92.0 | 91.4 | **94.3** |
| 15 | 3 | 2 | H | 94.0 | 94.1 | 95.5 | 93.9 | 95.4 | **96.6** | 90.6 | 91.2 | 93.6 | 91.2 | 91.4 | **93.1** |
| 15 | 3 | 3 | H | 91.0 | 91.1 | 93.4 | 91.2 | 92.5 | **94.5** | 90.1 | 90.2 | 93.8 | 90.9 | 90.6 | **93.6** |
| 5 | 6 | 2 | H | 84.7 | 85.7 | 85.4 | 87.2 | 88.0 | **88.8** | 89.2 | 90.3 | 91.1 | 90.9 | 91.0 | **92.1** |
| 5 | 6 | 3 | H | 88.8 | 88.4 | 88.8 | 89.0 | **91.0** | 90.3 | 89.5 | 88.9 | 88.9 | 89.1 | 89.7 | **90.3** |
| 10 | 6 | 2 | H | 83.7 | 84.1 | 85.3 | 86.4 | 87.3 | **87.8** | 87.8 | 87.8 | 88.9 | 89.4 | 90.4 | **91.8** |
| 10 | 6 | 3 | H | 83.2 | 81.9 | 85.0 | 85.9 | 86.8 | **89.1** | 84.6 | 84.9 | 87.4 | 88.2 | 87.1 | **89.5** |
| 15 | 6 | 2 | H | 85.8 | 85.6 | 87.5 | 86.4 | 86.8 | **89.3** | 90.0 | 90.1 | 92.2 | 90.8 | 90.8 | **93.3** |
| 15 | 6 | 3 | H | 83.5 | 83.4 | 85.3 | 84.2 | 84.5 | **86.9** | 86.0 | 86.2 | 87.9 | 87.1 | 86.6 | **89.8** |
| Average | | | | 83.5 | 83.6 | 85.4 | 84.8 | 87.4 | **89.0** | 85.7 | 86.1 | 88.3 | 87.1 | 87.4 | **89.7** |

### A.2.3 Detailed Results of the Exact Algorithms on Single-Vehicle Instances

In this section, we test our branch-and-cut algorithms on the single vehicle PRP and IRP instances from the literature to see the performance of the algorithms compared to the multi-vehicle case in the previous section. Since the SBCs are dropped when $m < 2$, the formulations $F(ML)|k$ and $F(OU)|k$ become equivalent to the formulations proposed by Archetti et al. (2007, 2011) and Solyalı and Süral (2011), respectively. We make a few remarks on the two formulation schemes for the single vehicle case. Without additional valid inequalities, the constraints in the $F(ML)|k$ and $F(OU)|k$ that do not appear in the $F(ML)|nk$ and $F(OU)|nk$ are constraints (3.7) and (3.41) for the ML and OU, respectively, and constraints (3.11), while the

Table A.VIII: Effects of the Valid Inequalities on the Branch-and-Cut Algorithm on MVPRP Instances

| $n$ $l$ $m$ | MVPRP-ML | | | | | | MVPRP-OU | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $F(ML)\|nk$ | | | $F(ML)\|nk^+$ | | | $F(OU)\|nk$ | | | $F(OU)\|nk^+$ | | |
| | %LB | CPU | Node | %LB | CPU | Node | %LB | CPU | Node | %LB | CPU | Node |
| 10 3 2 | 100.0 | 0.3 | 12 | 100.0 | **0.1** | 0 | 100.0 | **0.3** | 13 | 100.0 | **0.3** | 8 |
| 10 3 3 | 100.0 | 0.6 | 71 | 100.0 | **0.4** | 17 | 100.0 | 0.4 | 13 | 100.0 | **0.3** | 5 |
| 15 3 2 | 100.0 | 40.8 | 4203 | 100.0 | **15.6** | 918 | 100.0 | 329.8 | 17432 | 100.0 | **139.8** | 10223 |
| 15 3 3 | 100.0 | 275.0 | 30747 | 100.0 | **58.9** | 3039 | 100.0 | 967.3 | 39767 | 100.0 | **428.6** | 28676 |
| 10 6 2 | 100.0 | 6.9 | 716 | 100.0 | **1.0** | 6 | 100.0 | 0.5 | 5 | 100.0 | **0.3** | 0 |
| 10 6 3 | 100.0 | 40.6 | 4687 | 100.0 | **19.8** | 1506 | 100.0 | **0.6** | 5 | 100.0 | 0.7 | 6 |
| 15 6 2 | 99.4$^{(4)}$ | 3600.0 | 100687 | **99.8**$^{(1)}$ | 1048.8 | 16217 | 100.0 | 75.3 | 2208 | 100.0 | **53.1** | 1333 |
| 15 6 3 | 98.9$^{(4)}$ | 3600.0 | 77402 | **99.3**$^{(4)}$ | 3600.0 | 64402 | 100.0 | 322.4 | 10196 | 100.0 | **117.8** | 2802 |
| 10 9 2 | 100.0 | 182.4 | 10872 | 100.0 | **53.5** | 1907 | 100.0 | 15.7 | 566 | 100.0 | **9.0** | 200 |
| 10 9 3 | 99.7$^{(3)}$ | 2767.0 | 125261 | 100.0 | **1454.5** | 77948 | 100.0 | 58.3 | 3045 | 100.0 | **43.6** | 1789 |
| 15 9 2 | 98.8$^{(4)}$ | 3600.0 | 33291 | **99.4**$^{(4)}$ | 3600.0 | 27380 | 99.0$^{(4)}$ | 3600.0 | 45068 | **99.3**$^{(3)}$ | 3600.0 | 34038 |
| 15 9 3 | 98.3$^{(4)}$ | 3600.0 | 32155 | **98.6**$^{(4)}$ | 3600.0 | 23677 | **98.6**$^{(4)}$ | 3600.0 | 33266 | **98.6**$^{(4)}$ | 3600.0 | 23592 |
| Optimal | 100.0 | 397.3 | 28639 | 100.0 | **161.3** | 10337 | 100.0 | 109.7 | 7325 | 100.0 | **69.6** | 5105 |
| Not optimal | 98.9 | 3600.0 | 83703 | **99.3** | 2954.0 | 47366 | 98.8 | 3530.5 | 65816 | **98.9** | 3255.2 | 43935 |
| Total | 99.6 | 1464.9 | 46994 | **99.8** | 1092.2 | 22680 | **99.8** | 679.8 | 17073 | **99.8** | 600.6 | 11576 |

$^{(-)}$ indicates the number of instances (out of 4) were not solved to optimality

constraints (3.28) and (3.53) in the $F(ML)|nk$ and $F(OU)|nk$, respectively, do not appear in the $F(ML)|k$ and $F(OU)|k$. With the valid inequalities in Section 3.3.2 for the single vehicle case, all the constraints in $F(ML)|k$ and $F(OU)|k$ are included in the $F(ML)|nk$ and $F(OU)|nk$ because the inequalities (3.55), (3.56) and (3.57) are equivalent to (3.7), (3.41) and (3.11), respectively. Thus, the formulations $F(ML)|nk$ and $F(OU)|nk$ become the $F(ML)|k$ and $F(OU)|k$, respectively, plus additional valid inequalities.

We test the single vehicle case using the MVPRP and MVIRP instances of the previous section. For the single-vehicle PRP, we use the same method as described in Section A.1 and simply set the number of vehicles to one when calculating the vehicle capacity. For the single-vehicle IRP, the instances are actually the original single vehicle IRP instances presented in Archetti et al. (2007). The results on the single-vehicle PRP and IRP are shown in Tables A.XII and A.XIII, respectively. Column *Cuts* shows the number of cuts generated in the branch-and-cut process.

We can see that the single-vehicle instances are much easier to solve and all instances were solved to optimality. The vehicle index formulations are slightly better

Table A.IX: Effects of the Valid Inequalities on the Branch-and-Cut Algorithm on MVIRP Instances

| $n$ | $l$ | $m$ | $h_i$ | MVIRP-ML | | | | | | MVIRP-OU | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $F(ML)\|nk$ | | | $F(ML)\|nk^+$ | | | $F(OU)\|nk$ | | | $F(OU)\|nk^+$ | | |
| | | | | %LB | CPU | Node | %LB | CPU | Node | %LB | CPU | Node | %LB | CPU | Node |
| 5 | 3 | 2 | L | 100.0 | **0.1** | 57 | 100.0 | **0.1** | 44 | 100.0 | **0.1** | 8 | 100.0 | **0.1** | 5 |
| 5 | 3 | 3 | L | 100.0 | 0.2 | 197 | 100.0 | **0.1** | 102 | 100.0 | **0.1** | 16 | 100.0 | **0.1** | 9 |
| 10 | 3 | 2 | L | 100.0 | 5.1 | 1125 | 100.0 | **3.9** | 754 | 100.0 | 4.2 | 1033 | 100.0 | **2.6** | 445 |
| 10 | 3 | 3 | L | 100.0 | 36.2 | 8666 | 100.0 | **25.0** | 4358 | 100.0 | 8.0 | 1779 | 100.0 | **5.2** | 726 |
| 15 | 3 | 2 | L | 100.0 | 125.0 | 10947 | 100.0 | **44.9** | 3117 | 100.0 | 210.7 | 18948 | 100.0 | **50.5** | 2643 |
| 15 | 3 | 3 | L | 99.1$^{(1)}$ | 1382.1 | 89506 | 100.0 | **964.1** | 67451 | 100.0 | 289.0 | 20756 | 100.0 | **49.0** | 2248 |
| 5 | 6 | 2 | L | 100.0 | 17.6 | 10589 | 100.0 | **16.7** | 8827 | 100.0 | **3.0** | 1527 | 100.0 | 4.4 | 2275 |
| 5 | 6 | 3 | L | 100.0 | 297.6 | 195135 | 100.0 | **232.9** | 132762 | 100.0 | **2.4** | 1056 | 100.0 | 3.6 | 1614 |
| 10 | 6 | 2 | L | 97.2$^{(2)}$ | 1789.8 | 149734 | **98.0**$^{(2)}$ | 1630.5 | 120769 | 97.7$^{(2)}$ | 2008.1 | 180764 | **98.9**$^{(1)}$ | 1521.5 | 111403 |
| 10 | 6 | 3 | L | 95.4$^{(4)}$ | 3332.8 | 306752 | **95.6**$^{(4)}$ | 3485.9 | 236854 | 95.4$^{(4)}$ | 3284.0 | 291015 | **96.0**$^{(4)}$ | 3239.2 | 184098 |
| 15 | 6 | 2 | L | 92.7$^{(5)}$ | 3600.0 | 93068 | **95.7**$^{(5)}$ | 3600.0 | 75164 | 91.9$^{(5)}$ | 3600.0 | 86215 | **95.5**$^{(5)}$ | 3600.0 | 69853 |
| 15 | 6 | 3 | L | 90.9$^{(5)}$ | 3600.0 | 75639 | **92.6**$^{(5)}$ | 3600.0 | 65202 | 89.3$^{(5)}$ | 3600.0 | 77423 | **91.5**$^{(5)}$ | 3600.0 | 54325 |
| 5 | 3 | 2 | H | 100.0 | **0.1** | 44 | 100.0 | **0.1** | 24 | 100.0 | **0.1** | 11 | 100.0 | **0.1** | 1 |
| 5 | 3 | 3 | H | 100.0 | 0.2 | 153 | 100.0 | **0.1** | 61 | 100.0 | **0.1** | 23 | 100.0 | **0.1** | 3 |
| 10 | 3 | 2 | H | 100.0 | **5.4** | 1381 | 100.0 | 7.8 | 1535 | 100.0 | 5.4 | 1250 | 100.0 | **2.7** | 386 |
| 10 | 3 | 3 | H | 100.0 | 39.0 | 10468 | 100.0 | **22.0** | 3500 | 100.0 | 9.5 | 2208 | 100.0 | **4.4** | 651 |
| 15 | 3 | 2 | H | 100.0 | 123.5 | 10397 | 100.0 | **49.3** | 3593 | 100.0 | 293.9 | 22258 | 100.0 | **39.8** | 2032 |
| 15 | 3 | 3 | H | 99.4$^{(1)}$ | 1187.9 | 87227 | 100.0 | **536.8** | 38901 | 100.0 | 227.0 | 15888 | 100.0 | **53.7** | 2721 |
| 5 | 6 | 2 | H | 100.0 | **16.9** | 9977 | 100.0 | 17.2 | 8707 | 100.0 | **2.7** | 1418 | 100.0 | 4.6 | 2301 |
| 5 | 6 | 3 | H | 100.0 | 215.6 | 146203 | 100.0 | **173.4** | 99959 | 100.0 | **2.1** | 988 | 100.0 | 2.7 | 1377 |
| 10 | 6 | 2 | H | 98.4$^{(2)}$ | 1664.0 | 154960 | **99.0**$^{(2)}$ | 1566.8 | 114063 | 98.5$^{(2)}$ | 2044.6 | 183224 | **99.6**$^{(1)}$ | 1669.2 | 123457 |
| 10 | 6 | 3 | H | **97.1**$^{(4)}$ | 3128.9 | 295580 | **97.1**$^{(4)}$ | 3540.2 | 242110 | **97.2**$^{(3)}$ | 3109.8 | 268698 | 97.1$^{(4)}$ | 3284.1 | 234901 |
| 15 | 6 | 2 | H | 96.4$^{(5)}$ | 3600.0 | 82539 | **98.0**$^{(5)}$ | 3600.0 | 78883 | 95.7$^{(5)}$ | 3600.0 | 81206 | **97.6**$^{(5)}$ | 3600.0 | 68426 |
| 15 | 6 | 3 | H | 95.2$^{(5)}$ | 3600.0 | 76099 | **96.0**$^{(5)}$ | 3600.0 | 62961 | 94.0$^{(5)}$ | 3600.0 | 71871 | **95.4**$^{(5)}$ | 3600.0 | 58152 |
| Optimal | | | | 99.9 | 215.8 | 36379 | 100.0 | **130.9** | 23356 | 100.0 | 66.1 | 5573 | 100.0 | **14.0** | 1215 |
| Not optimal | | | | 95.4 | 3039.4 | 154296 | **96.5** | 3077.9 | 124501 | 95.0 | 3105.8 | 155052 | **96.5** | 3014.3 | 113077 |
| Total | | | | 98.4 | 1157.0 | 75685 | **98.8** | 1113.2 | 57071 | 98.3 | 1079.4 | 55399 | **98.8** | 1014.1 | 38502 |

$^{(-)}$ indicates the number of instances (out of 5) were not solved to optimality

on small instances but the non-vehicle index formulations could significantly reduce computing times on some large instances. The non-vehicle index formulation outperforms on average the vehicle index formulation for the PRP, while both formulations have a similar performance for the IRP.

Note that the IRP instances with $50c/3p/1v$ in Archetti et al. (2007) are now solved to optimality. We can also observe that the PRP is more difficult to solve than the IRP for the single vehicle case, which is similar to the results presented by Archetti et al. (2007) and Archetti et al. (2011) for the IRP and PRP, respectively.

We also ran the algorithm on the single-vehicle PRP instances with uncapacitated production of Archetti et al. (2011) which consist of 480 PRP instances with

Table A.X: Average Results on MVPRP Instances Using Muli-Core Processors

| $n$ | $l$ | $m$ | MVPRP-ML | | | | | | MVPRP-OU | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | $F(ML)\|k$ | | | | Op-ALNS | | $F(OU)\|k$ | | | | Op-ALNS | |
| | | | %LB | CPU(h) | WC(h) | $\frac{\text{CPU}}{\text{WC}}$ | %DIFF | CPU(s) | %LB | CPU(h) | WC(h) | $\frac{\text{CPU}}{\text{WC}}$ | %DIFF | CPU(s) |
| 10 | 3 | 2 | 100.0 | 0.0 | 0.0 | 1.4 | 0.4 | 4.6 | 100.0 | 0.0 | 0.0 | 1.2 | 0.0 | 4.2 |
| 10 | 3 | 3 | 100.0 | 0.0 | 0.0 | 1.3 | 1.1 | 4.3 | 100.0 | 0.0 | 0.0 | 1.2 | 0.0 | 4.4 |
| 15 | 3 | 2 | 100.0 | 0.0 | 0.0 | 1.5 | 0.9 | 6.6 | 100.0 | 0.0 | 0.0 | 1.4 | 2.0 | 5.8 |
| 15 | 3 | 3 | 100.0 | 0.0 | 0.0 | 1.5 | 1.0 | 6.6 | 100.0 | 0.0 | 0.0 | 1.2 | 1.0 | 6.7 |
| 20 | 3 | 2 | 100.0 | 0.0 | 0.0 | 1.4 | 1.0 | 9.8 | 100.0 | 0.0 | 0.0 | 1.2 | 1.5 | 7.4 |
| 20 | 3 | 3 | 100.0 | 0.0 | 0.0 | 1.4 | 1.7 | 10.0 | 100.0 | 0.2 | 0.1 | 1.5 | 1.3 | 8.4 |
| 25 | 3 | 2 | 100.0 | 0.0 | 0.0 | 1.2 | 1.5 | 12.4 | 100.0 | 1.1 | 0.6 | 1.9 | 1.5 | 11.1 |
| 25 | 3 | 3 | 100.0 | 0.0 | 0.0 | 1.6 | 1.2 | 14.3 | 99.8$^{(1)}$ | 20.9 | 7.6 | 2.8 | 2.9 | 14.3 |
| 30 | 3 | 3 | 100.0 | 0.0 | 0.0 | 1.5 | 1.8 | 23.7 | 98.6$^{(4)}$ | 27.1 | 12.0 | 2.3 | 2.4 | 19.9 |
| 30 | 3 | 4 | 100.0 | 0.5 | 0.3 | 1.8 | 1.4 | 28.1 | 97.0$^{(4)}$ | 22.7 | 12.0 | 1.9 | 2.2 | 23.3 |
| 35 | 3 | 3 | 100.0 | 0.3 | 0.2 | 1.5 | 2.4 | 36.7 | 96.2$^{(4)}$ | 29.9 | 12.0 | 2.5 | 0.3 | 28.8 |
| 35 | 3 | 4 | 100.0 | 7.8 | 3.1 | 2.5 | 2.3 | 43.0 | 94.8$^{(4)}$ | 19.8 | 12.0 | 1.6 | 0.4 | 37.7 |
| 40 | 3 | 3 | 100.0 | 1.5 | 0.8 | 1.8 | 1.3 | 51.7 | 98.0$^{(4)}$ | 24.5 | 12.0 | 2.0 | 1.1 | 32.0 |
| 40 | 3 | 4 | 99.0$^{(2)}$ | 15.9 | 7.8 | 2.0 | 0.6 | 52.4 | 95.7$^{(4)}$ | 18.4 | 12.0 | 1.5 | 0.2 | 42.9 |
| 45 | 3 | 3 | 100.0 | 7.9 | 3.6 | 2.2 | 1.2 | 67.5 | 97.2$^{(4)}$ | 20.9 | 12.0 | 1.7 | 0.8 | 45.9 |
| 45 | 3 | 4 | 97.9$^{(3)}$ | 23.9 | 11.9 | 2.0 | 0.3 | 72.1 | 95.3$^{(4)}$ | 18.7 | 12.0 | 1.6 | 0.0 | 56.7 |
| 50 | 3 | 3 | 100.0 | 3.3 | 2.1 | 1.6 | 1.4 | 90.4 | 96.5$^{(4)}$ | 21.7 | 12.0 | 1.8 | 0.1 | 59.2 |
| 50 | 3 | 4 | 99.0$^{(3)}$ | 16.5 | 9.8 | 1.7 | 1.6 | 85.1 | 94.7$^{(4)}$ | 19.6 | 12.0 | 1.6 | 0.0 | 65.9 |
| 10 | 6 | 2 | 100.0 | 0.0 | 0.0 | 1.3 | 0.6 | 7.3 | 100.0 | 0.0 | 0.0 | 1.5 | 0.1 | 9.5 |
| 10 | 6 | 3 | 100.0 | 0.0 | 0.0 | 1.5 | 0.4 | 8.4 | 100.0 | 0.0 | 0.0 | 1.1 | 0.1 | 14.0 |
| 15 | 6 | 2 | 100.0 | 0.0 | 0.0 | 1.7 | 1.0 | 13.8 | 100.0 | 0.0 | 0.0 | 1.2 | 0.4 | 14.1 |
| 15 | 6 | 3 | 100.0 | 0.2 | 0.1 | 1.7 | 1.6 | 14.0 | 100.0 | 0.0 | 0.0 | 1.2 | 0.7 | 17.6 |
| 20 | 6 | 2 | 100.0 | 0.0 | 0.0 | 1.5 | 1.3 | 22.1 | 100.0 | 0.0 | 0.0 | 1.2 | 0.4 | 22.4 |
| 20 | 6 | 3 | 100.0 | 0.3 | 0.2 | 1.8 | 1.5 | 20.6 | 100.0 | 0.1 | 0.1 | 1.2 | 0.5 | 32.7 |
| 25 | 6 | 2 | 100.0 | 0.0 | 0.0 | 1.3 | 1.8 | 28.4 | 100.0 | 0.1 | 0.1 | 1.2 | 1.2 | 30.1 |
| 25 | 6 | 3 | 100.0 | 1.0 | 0.6 | 1.6 | 1.7 | 34.2 | 100.0 | 4.8 | 3.6 | 1.3 | 1.8 | 48.4 |
| 30 | 6 | 3 | 100.0 | 5.7 | 2.4 | 2.4 | 1.6 | 52.3 | 100.0 | 0.6 | 0.4 | 1.3 | 0.5 | 58.7 |
| 30 | 6 | 4 | 98.9$^{(2)}$ | 15.1 | 7.6 | 2.0 | 1.3 | 59.3 | 100.0 | 7.3 | 5.2 | 1.4 | 1.3 | 90.3 |
| 35 | 6 | 3 | 99.3$^{(2)}$ | 19.0 | 7.6 | 2.5 | 1.3 | 78.4 | 100.0 | 0.9 | 0.8 | 1.2 | 0.7 | 74.1 |
| 35 | 6 | 4 | 97.7$^{(4)}$ | 21.5 | 12.0 | 1.8 | 0.8 | 78.4 | 98.9$^{(3)}$ | 14.6 | 11.7 | 1.2 | 0.9 | 123.1 |
| 40 | 6 | 3 | 98.8$^{(4)}$ | 20.5 | 12.0 | 1.7 | 0.8 | 110.2 | 99.6$^{(2)}$ | 14.7 | 11.1 | 1.3 | 1.6 | 98.4 |
| 40 | 6 | 4 | 97.2$^{(4)}$ | 19.7 | 12.0 | 1.6 | 0.1 | 109.5 | 97.7$^{(4)}$ | 14.6 | 12.0 | 1.2 | 1.0 | 169.0 |
| 10 | 9 | 2 | 100.0 | 0.0 | 0.0 | 1.6 | 1.8 | 13.8 | 100.0 | 0.0 | 0.0 | 1.4 | 1.4 | 18.4 |
| 10 | 9 | 3 | 100.0 | 0.1 | 0.1 | 1.5 | 1.8 | 12.9 | 100.0 | 0.0 | 0.0 | 1.3 | 1.3 | 26.7 |
| 15 | 9 | 2 | 100.0 | 1.4 | 0.7 | 2.0 | 1.6 | 24.1 | 100.0 | 0.2 | 0.2 | 1.5 | 1.2 | 28.2 |
| 15 | 9 | 3 | 98.1$^{(4)}$ | 18.5 | 12.0 | 1.5 | 1.7 | 25.7 | 99.8$^{(1)}$ | 10.4 | 5.8 | 1.8 | 2.0 | 55.7 |
| 20 | 9 | 2 | 100.0 | 0.4 | 0.3 | 1.5 | 1.0 | 36.8 | 100.0 | 0.1 | 0.1 | 1.3 | 0.8 | 42.6 |
| 20 | 9 | 3 | 99.5$^{(2)}$ | 9.8 | 6.3 | 1.6 | 1.3 | 39.2 | 99.7$^{(1)}$ | 7.3 | 5.2 | 1.4 | 0.4 | 70.4 |
| 25 | 9 | 2 | 99.8$^{(1)}$ | 8.3 | 4.9 | 1.7 | 0.6 | 56.4 | 100.0 | 0.8 | 0.6 | 1.2 | 0.5 | 65.4 |
| 25 | 9 | 3 | 98.3$^{(4)}$ | 16.5 | 12.0 | 1.4 | 0.8 | 66.7 | 98.5$^{(3)}$ | 12.0 | 9.9 | 1.2 | 0.6 | 129.8 |
| 30 | 9 | 3 | 97.8$^{(4)}$ | 19.4 | 12.0 | 1.6 | 0.9 | 105.4 | 99.0$^{(3)}$ | 14.7 | 10.8 | 1.4 | 1.0 | 136.5 |
| 30 | 9 | 4 | 95.9$^{(4)}$ | 18.4 | 12.0 | 1.5 | 0.3 | 115.7 | 97.2$^{(4)}$ | 15.3 | 12.0 | 1.3 | 0.1 | 238.2 |
| Optimal | | | 100.0 | 1.1 | 0.5 | 1.6 | 1.4 | 24.9 | 100.0 | 0.7 | 0.5 | 1.3 | 0.9 | 27.9 |
| Not optimal | | | 98.3 | 17.4 | 10.0 | 1.8 | 0.9 | 75.3 | 97.6 | 18.3 | 10.9 | 1.7 | 1.0 | 76.2 |
| Total | | | 99.4 | 6.5 | 3.7 | 1.7 | 1.2 | 41.7 | 98.9 | 8.7 | 5.2 | 1.5 | 0.9 | 49.7 |

n/a indicates the instance size was already solved to optimality using a single core processor
$^{(-)}$ indicates the number of instances (out of 4) were not solved to optimality

14 customers and 6 periods. Since these instances were designed to test a heuristic, several different parameter settings in terms of inventory, production and transportation costs are used to generate the test set. In Archetti et al. (2011), they consider the PRP-ML with uncapacitated production, where the following valid inequalities

for the uncapacitated lot-sizing problem are also added:

$$I_{0,t-1} \leq \sum_{i \in N_c} \sum_{j=t}^{l} d_{ij}(1 - y_t) \quad \forall t \in T \tag{A.4}$$

$$p_t \geq \frac{f}{h_{0j}}(y_{t-j} + y_t - 1) \quad 2 \leq t \leq l, 1 \leq j \leq t-1. \tag{A.5}$$

These inequalities are also added in the $F(ML)|k$ and $F(ML)|nk$ formulations. The results are shown Table A.XIV. Since there are 480 instances, we report the average results of each class. Similar to the results on the single-vehicle instances in the previous experiments, the vehicle index formulation is slightly better because the instance size is relatively small. Note that all instances that were not solved to optimality in Archetti et al. (2011) are solved to optimality with this implementation.

### A.2.4 Results of the Exact Algorithms when Allowing Multiple Visits

We further explore the results when multiple visits to a customer in each time period is allowed. This can be done by replacing constraints (3.8) with the following constraints:

$$\sum_{k \in K} z_{ikt} \leq n_v \qquad \forall i \in N_c, \forall t \in T,$$

where $n_v$ is the maximum number of visits to a customer in a time period. Note that this modification can be done for the $F(ML)|k$ formulation only since the shortest path constraints of the $F(OU)|k$ formulation do not allow multiple visits. Also, this feature cannot be incorporated in the non-vehicle index formulations.

We perform the tests on the MVPRP and MVIRP instances with $n \leq 30$ and $m = 3$ and the results are shown in Table A.XV. Column *Cost index (%)* shows the average total cost in percentage of the cost when allowing maximum one visit per period per customer, calculated by $\hat{c}_{n_v}/\hat{c}_1$, where $\hat{c}_{n_v}$ and $\hat{c}_1$ are the total costs obtained by allowing maximum $n_v$ visits and 1 visit per per period per customer, respectively. Column *Best* indicates the number of instances in which the minimum

cost is found in each setting. We found that, by allowing multiple visits per period, all the total costs remain the same for the MVPRP while multiple visits are very rarely used. The average CPU time, however, increased when the maximum number of visits increased. For the MVIRP, more multiple visits are used while the cases where $n_v = 2$ and $n_v = 3$ could provide slight improvements in the total costs (0.6% on average). The average CPU time by using multiple visits increased compared to the standard case with a single visit.

Table A.XI: Average Results on MVIRP Instances Using Muli-Core Processors

| $n$ | $l$ | $m$ | $h_i$ | MVIRP-ML | | | | | | MVIRP-OU | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $F(ML)\|k$ | | | | Op-ALNS | | $F(OU)\|k$ | | | | Op-ALNS | |
| | | | | %LB | CPU(h) | WC(h) | $\frac{\text{CPU}}{\text{WC}}$ | %DIFF | CPU(s) | %LB | CPU(h) | WC(h) | $\frac{\text{CPU}}{\text{WC}}$ | %DIFF | CPU(s) |
| 5 | 3 | 2 | L | 100.0 | 0.0 | 0.0 | 1.5 | 1.3 | 3.5 | 100.0 | 0.0 | 0.0 | 1.2 | 0.5 | 3.4 |
| 5 | 3 | 3 | L | 100.0 | 0.0 | 0.0 | 1.5 | 0.5 | 3.7 | 100.0 | 0.0 | 0.0 | 1.1 | 0.2 | 3.8 |
| 10 | 3 | 2 | L | 100.0 | 0.0 | 0.0 | 1.5 | 3.0 | 5.5 | 100.0 | 0.0 | 0.0 | 1.5 | 2.1 | 6.6 |
| 10 | 3 | 3 | L | 100.0 | 0.0 | 0.0 | 1.6 | 4.1 | 6.0 | 100.0 | 0.0 | 0.0 | 1.5 | 1.7 | 7.4 |
| 15 | 3 | 2 | L | 100.0 | 0.0 | 0.0 | 1.5 | 2.3 | 7.9 | 100.0 | 0.0 | 0.0 | 1.5 | 5.5 | 8.1 |
| 15 | 3 | 3 | L | 100.0 | 0.0 | 0.0 | 1.5 | 7.0 | 10.3 | 100.0 | 0.0 | 0.0 | 1.6 | 6.7 | 13.7 |
| 20 | 3 | 2 | L | 100.0 | 0.0 | 0.0 | 1.6 | 5.0 | 10.9 | 100.0 | 0.0 | 0.0 | 1.9 | 5.8 | 10.6 |
| 20 | 3 | 3 | L | 100.0 | 0.1 | 0.1 | 2.0 | 4.9 | 11.6 | 100.0 | 0.2 | 0.1 | 2.0 | 10.0 | 14.1 |
| 25 | 3 | 2 | L | 100.0 | 0.0 | 0.0 | 1.5 | 5.4 | 18.0 | 100.0 | 0.0 | 0.0 | 1.9 | 9.1 | 4.6 |
| 25 | 3 | 3 | L | 100.0 | 0.4 | 0.2 | 2.0 | 6.0 | 17.2 | 100.0 | 0.2 | 0.1 | 1.9 | 8.8 | 22.9 |
| 30 | 3 | 3 | L | 100.0 | 0.9 | 0.5 | 1.9 | 7.3 | 24.8 | 100.0 | 0.9 | 0.4 | 2.2 | 9.4 | 29.0 |
| 30 | 3 | 4 | L | 100.0 | 19.5 | 6.2 | 3.1 | 8.1 | 27.3 | 100.0 | 15.6 | 4.5 | 3.4 | 11.4 | 37.4 |
| 35 | 3 | 3 | L | 100.0 | 1.1 | 0.6 | 1.7 | 6.0 | 32.7 | 100.0 | 4.5 | 1.6 | 2.9 | 10.8 | 41.3 |
| 35 | 3 | 4 | L | 97.6$^{(2)}$ | 25.8 | 8.6 | 3.0 | 7.8 | 36.1 | 95.0$^{(4)}$ | 40.5 | 10.2 | 4.0 | 10.6 | 51.6 |
| 40 | 3 | 3 | L | 100.0 | 2.9 | 1.5 | 1.9 | 8.4 | 47.0 | 100.0 | 3.5 | 1.5 | 2.4 | 10.8 | 47.7 |
| 40 | 3 | 4 | L | 94.2$^{(4)}$ | 28.8 | 11.6 | 2.5 | 7.4 | 49.9 | 93.1$^{(4)}$ | 40.0 | 11.9 | 3.4 | 13.1 | 62.0 |
| 45 | 3 | 3 | L | 100.0 | 12.7 | 5.2 | 2.5 | 10.9 | 107.4 | 100.0 | 17.2 | 5.3 | 3.3 | 10.5 | 98.5 |
| 45 | 3 | 4 | L | 91.3$^{(4)}$ | 27.2 | 10.0 | 2.7 | 6.4 | 127.0 | 89.0$^{(4)}$ | 43.9 | 12.0 | 3.7 | 8.7 | 122.3 |
| 50 | 3 | 3 | L | 95.1$^{(5)}$ | 36.2 | 12.0 | 3.0 | 5.6 | 122.8 | 95.0$^{(4)}$ | 44.9 | 11.8 | 3.8 | 10.9 | 119.5 |
| 50 | 3 | 4 | L | 79.3$^{(5)}$ | 27.4 | 12.0 | 2.3 | 2.3 | 142.8 | 79.3$^{(5)}$ | 36.5 | 12.0 | 3.0 | 4.6 | 136.0 |
| 5 | 6 | 2 | L | 100.0 | 0.0 | 0.0 | 1.6 | 3.6 | 5.7 | 100.0 | 0.0 | 0.0 | 1.2 | 2.7 | 5.9 |
| 5 | 6 | 3 | L | 100.0 | 0.0 | 0.0 | 1.4 | 3.3 | 6.8 | 100.0 | 0.0 | 0.0 | 1.2 | 1.2 | 7.6 |
| 10 | 6 | 2 | L | 100.0 | 0.0 | 0.0 | 1.8 | 4.2 | 9.7 | 100.0 | 0.1 | 0.0 | 2.0 | 7.5 | 9.7 |
| 10 | 6 | 3 | L | 99.5$^{(1)}$ | 7.3 | 4.3 | 1.7 | 3.9 | 11.4 | 100.0 | 1.1 | 0.5 | 2.3 | 5.0 | 12.0 |
| 15 | 6 | 2 | L | 100.0 | 0.2 | 0.1 | 1.9 | 3.9 | 18.0 | 100.0 | 0.2 | 0.1 | 1.9 | 9.1 | 16.5 |
| 15 | 6 | 3 | L | 99.2$^{(1)}$ | 11.3 | 5.8 | 1.9 | 4.5 | 18.8 | 100.0 | 16.9 | 5.5 | 3.1 | 10.3 | 20.6 |
| 20 | 6 | 2 | L | 100.0 | 1.4 | 0.7 | 2.0 | 6.2 | 29.0 | 99.6$^{(1)}$ | 10.2 | 4.1 | 2.5 | 10.4 | 27.8 |
| 20 | 6 | 3 | L | 91.8$^{(5)}$ | 22.3 | 12.0 | 1.9 | 4.8 | 34.3 | 84.1$^{(5)}$ | 25.0 | 12.0 | 2.1 | 7.2 | 33.0 |
| 25 | 6 | 2 | L | 100.0 | 8.0 | 3.3 | 2.4 | 6.8 | 54.6 | 98.3$^{(2)}$ | 16.2 | 6.4 | 2.5 | 9.3 | 46.4 |
| 25 | 6 | 3 | L | 90.0$^{(5)}$ | 21.8 | 12.0 | 1.8 | 3.9 | 59.3 | 84.6$^{(5)}$ | 26.2 | 12.0 | 2.2 | 6.3 | 56.2 |
| 5 | 3 | 2 | H | 100.0 | 0.0 | 0.0 | 1.4 | 0.9 | 3.2 | 100.0 | 0.0 | 0.0 | 1.7 | 1.1 | 3.3 |
| 5 | 3 | 3 | H | 100.0 | 0.0 | 0.0 | 1.5 | 0.5 | 3.7 | 100.0 | 0.0 | 0.0 | 1.2 | 0.1 | 4.0 |
| 10 | 3 | 2 | H | 100.0 | 0.0 | 0.0 | 1.5 | 2.1 | 5.3 | 100.0 | 0.0 | 0.0 | 1.4 | 0.9 | 6.2 |
| 10 | 3 | 3 | H | 100.0 | 0.0 | 0.0 | 1.6 | 2.6 | 6.0 | 100.0 | 0.0 | 0.0 | 1.5 | 2.3 | 7.1 |
| 15 | 3 | 2 | H | 100.0 | 0.0 | 0.0 | 1.5 | 2.0 | 7.8 | 100.0 | 0.0 | 0.0 | 1.6 | 4.2 | 9.4 |
| 15 | 3 | 3 | H | 100.0 | 0.0 | 0.0 | 1.5 | 4.0 | 8.5 | 100.0 | 0.0 | 0.0 | 1.6 | 3.5 | 13.9 |
| 20 | 3 | 2 | H | 100.0 | 0.0 | 0.0 | 1.6 | 2.3 | 11.4 | 100.0 | 0.0 | 0.0 | 1.8 | 3.3 | 11.9 |
| 20 | 3 | 3 | H | 100.0 | 0.1 | 0.1 | 1.7 | 4.6 | 11.3 | 100.0 | 0.1 | 0.1 | 2.0 | 5.5 | 15.9 |
| 25 | 3 | 2 | H | 100.0 | 0.0 | 0.0 | 1.6 | 2.8 | 16.8 | 100.0 | 0.0 | 0.0 | 1.8 | 4.1 | 5.4 |
| 25 | 3 | 3 | H | 100.0 | 0.3 | 0.2 | 1.8 | 4.3 | 18.2 | 100.0 | 0.3 | 0.2 | 2.0 | 3.9 | 22.0 |
| 30 | 3 | 3 | H | 100.0 | 0.5 | 0.3 | 1.8 | 3.4 | 25.9 | 100.0 | 1.3 | 0.6 | 2.3 | 6.0 | 28.2 |
| 30 | 3 | 4 | H | 100.0 | 11.7 | 3.8 | 3.1 | 3.6 | 29.0 | 100.0 | 18.3 | 5.2 | 3.5 | 5.4 | 34.8 |
| 35 | 3 | 3 | H | 100.0 | 0.8 | 0.5 | 1.9 | 4.4 | 35.3 | 100.0 | 3.4 | 1.2 | 2.7 | 5.0 | 41.1 |
| 35 | 3 | 4 | H | 99.2$^{(2)}$ | 24.1 | 7.6 | 3.2 | 4.0 | 36.1 | 96.9$^{(4)}$ | 38.3 | 10.2 | 3.7 | 4.7 | 50.7 |
| 40 | 3 | 3 | H | 100.0 | 2.8 | 1.6 | 1.8 | 4.6 | 46.0 | 100.0 | 2.9 | 1.1 | 2.5 | 6.3 | 48.9 |
| 40 | 3 | 4 | H | 98.1$^{(4)}$ | 24.3 | 10.8 | 2.3 | 2.8 | 51.5 | 96.6$^{(5)}$ | 38.7 | 12.0 | 3.2 | 5.8 | 61.5 |
| 45 | 3 | 3 | H | 100.0 | 9.0 | 3.9 | 2.3 | 4.7 | 110.2 | 100.0 | 20.4 | 5.9 | 3.5 | 5.8 | 98.6 |
| 45 | 3 | 4 | H | 97.0$^{(4)}$ | 27.0 | 9.9 | 2.7 | 2.5 | 140.6 | 95.5$^{(5)}$ | 40.6 | 12.0 | 3.4 | 4.9 | 119.3 |
| 50 | 3 | 3 | H | 98.5$^{(5)}$ | 31.4 | 12.0 | 2.6 | 3.2 | 135.2 | 98.6$^{(3)}$ | 41.0 | 10.1 | 4.1 | 3.0 | 120.9 |
| 50 | 3 | 4 | H | 93.1$^{(5)}$ | 28.7 | 12.0 | 2.4 | 1.2 | 162.9 | 91.3$^{(5)}$ | 35.3 | 12.0 | 2.9 | 1.8 | 137.0 |
| 5 | 6 | 2 | H | 100.0 | 0.0 | 0.0 | 1.6 | 2.2 | 5.7 | 100.0 | 0.0 | 0.0 | 1.3 | 2.4 | 5.7 |
| 5 | 6 | 3 | H | 100.0 | 0.0 | 0.0 | 1.4 | 1.8 | 6.6 | 100.0 | 0.0 | 0.0 | 1.3 | 1.3 | 7.5 |
| 10 | 6 | 2 | H | 100.0 | 0.0 | 0.0 | 1.7 | 3.3 | 9.8 | 100.0 | 0.1 | 0.0 | 1.9 | 5.7 | 9.1 |
| 10 | 6 | 3 | H | 99.4$^{(1)}$ | 7.8 | 4.7 | 1.7 | 2.5 | 11.4 | 100.0 | 1.2 | 0.5 | 2.2 | 5.9 | 11.6 |
| 15 | 6 | 2 | H | 100.0 | 0.2 | 0.1 | 1.8 | 1.8 | 17.6 | 100.0 | 0.2 | 0.1 | 1.9 | 5.4 | 15.7 |
| 15 | 6 | 3 | H | 99.8$^{(1)}$ | 9.4 | 4.7 | 2.0 | 2.8 | 17.3 | 100.0 | 16.5 | 6.0 | 2.7 | 4.7 | 19.7 |
| 20 | 6 | 2 | H | 100.0 | 1.1 | 0.5 | 2.1 | 4.2 | 32.3 | 99.8 | 11.6 | 4.6 | 2.5 | 4.7 | 27.5 |
| 20 | 6 | 3 | H | 96.2$^{(4)}$ | 21.3 | 11.2 | 1.9 | 2.6 | 36.8 | 91.5$^{(5)}$ | 24.8 | 12.0 | 2.1 | 2.5 | 34.1 |
| 25 | 6 | 2 | H | 100.0 | 5.9 | 2.4 | 2.5 | 2.3 | 54.1 | 99.0$^{(2)}$ | 12.1 | 5.4 | 2.2 | 4.1 | 48.3 |
| 25 | 6 | 3 | H | 95.2$^{(5)}$ | 22.2 | 12.0 | 1.8 | 2.0 | 54.9 | 91.8$^{(5)}$ | 24.7 | 12.0 | 2.1 | 3.4 | 54.4 |
| Optimal | | | | 100.0 | 2.1 | 0.9 | 1.8 | 4.1 | 22.0 | 100.0 | 3.0 | 1.0 | 2.0 | 5.3 | 20.0 |
| Not optimal | | | | 95.2 | 22.5 | 9.6 | 2.3 | 3.9 | 69.4 | 93.8 | 28.9 | 9.7 | 2.9 | 6.4 | 72.7 |
| Total | | | | 98.6 | 8.0 | 3.4 | 2.0 | 4.0 | 36.2 | 98.0 | 11.4 | 3.8 | 2.3 | 5.6 | 35.8 |

n/a indicates the instance size was already solved to optimality using a single core processor
$^{(-)}$ indicates the number of instances (out of 5) were not solved to optimality

Table A.XII: Average Results on the Single-Vehicle PRP Instances

| $n$ | $l$ | $F(ML)\|k$ | | | $F(ML)\|nk$ | | | $F(OU)\|k$ | | | $F(OU)\|nk$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | CPU | Nodes | Cuts | CPU | Nodes | Cuts | CPU | Nodes | Cuts | CPU | Nodes | Cuts |
| 10 | 3 | **0.1** | 0 | 6 | **0.1** | 0 | 12 | **0.1** | 1 | 7 | 0.2 | 5 | 12 |
| 15 | 3 | **0.3** | 0 | 26 | 0.4 | 0 | 40 | **1.2** | 57 | 84 | 1.4 | 52 | 99 |
| 20 | 3 | **0.3** | 1 | 21 | 0.5 | 1 | 34 | **1.0** | 26 | 51 | 1.2 | 19 | 58 |
| 25 | 3 | **0.4** | 0 | 25 | 0.5 | 0 | 32 | **7.6** | 156 | 218 | 10.2 | 250 | 280 |
| 30 | 3 | **5.1** | 6 | 122 | 7.7 | 5 | 140 | 30.3 | 286 | 456 | **24.0** | 233 | 439 |
| 35 | 3 | **9.3** | 15 | 142 | 10.0 | 12 | 145 | 219.9 | 1730 | 1123 | **177.4** | 1674 | 1360 |
| 40 | 3 | **12.3** | 2 | 111 | 19.1 | 2 | 151 | 109.8 | 496 | 632 | **81.1** | 393 | 667 |
| 45 | 3 | 67.8 | 57 | 249 | **42.1** | 15 | 205 | 323.2 | 849 | 1386 | **238.6** | 736 | 1293 |
| 50 | 3 | **47.4** | 15 | 215 | 54.8 | 11 | 211 | 2114.9 | 4912 | 3190 | **1222.6** | 3205 | 3066 |
| 10 | 6 | **0.2** | 0 | 26 | 0.3 | 1 | 40 | **0.2** | 0 | 12 | 0.2 | 1 | 14 |
| 15 | 6 | **1.1** | 3 | 85 | 2.1 | 2 | 111 | **0.3** | 0 | 20 | 0.4 | 0 | 28 |
| 20 | 6 | **2.0** | 2 | 102 | 6.6 | 2 | 163 | **1.1** | 4 | 26 | 2.3 | 7 | 40 |
| 25 | 6 | **10.3** | 2 | 226 | 11.5 | 1 | 199 | **8.3** | 34 | 115 | 12.7 | 43 | 142 |
| 30 | 6 | **50.1** | 176 | 402 | 71.2 | 132 | 444 | **2.5** | 1 | 40 | 3.8 | 1 | 65 |
| 35 | 6 | **74.3** | 136 | 446 | 108.1 | 87 | 477 | **4.9** | 1 | 53 | 8.7 | 1 | 92 |
| 40 | 6 | **80.5** | 4 | 384 | 105.3 | 5 | 407 | **36.1** | 5 | 187 | 53.6 | 4 | 198 |
| 10 | 9 | **0.8** | 10 | 66 | 1.1 | 9 | 83 | **0.6** | 8 | 41 | 1.0 | 8 | 49 |
| 15 | 9 | **21.8** | 292 | 371 | 22.2 | 201 | 387 | **9.5** | 147 | 162 | 12.0 | 153 | 185 |
| 20 | 9 | **14.4** | 64 | 237 | 22.5 | 74 | 298 | **8.0** | 26 | 140 | 12.6 | 29 | 148 |
| 25 | 9 | **178.4** | 500 | 741 | 234.2 | 650 | 745 | **39.7** | 138 | 258 | 67.7 | 183 | 347 |
| 30 | 9 | 1715.6 | 4151 | 1504 | **899.8** | 1160 | 1496 | **91.9** | 67 | 578 | 150.4 | 77 | 706 |
| Average | | 109.2 | 259 | 262 | **77.1** | 113 | 277 | 143.4 | 426 | 418 | **99.2** | 337 | 442 |

Table A.XIII: Average Results on the Single-Vehicle IRP Instances

| $n$ | $l$ | $h_i$ | $F(ML)\|k$ | | | $F(ML)\|nk$ | | | $F(OU)\|k$ | | | $F(OU)\|nk$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | CPU | Nodes | Cuts | CPU | Nodes | Cuts | CPU | Nodes | Cuts | CPU | Nodes | Cuts |
| 5 | 3 | L | **0.0** | 0 | 4 | **0.0** | 0 | 6 | **0.0** | 0 | 7 | 0.1 | 0 | 7 |
| 10 | 3 | L | **0.1** | 0 | 13 | **0.1** | 0 | 9 | **0.4** | 15 | 43 | 0.6 | 15 | 52 |
| 15 | 3 | L | 0.4 | 1 | 27 | **0.2** | 1 | 18 | **1.5** | 52 | 114 | 1.6 | 31 | 95 |
| 20 | 3 | L | 1.6 | 28 | 74 | **1.5** | 32 | 63 | **3.6** | 40 | 166 | 7.2 | 84 | 242 |
| 25 | 3 | L | **2.0** | 17 | 61 | 2.8 | 25 | 63 | **8.6** | 51 | 269 | 14.0 | 71 | 323 |
| 30 | 3 | L | 12.8 | 117 | 139 | **9.6** | 99 | 150 | **21.9** | 82 | 384 | 27.5 | 66 | 404 |
| 35 | 3 | L | **2.9** | 7 | 47 | 4.1 | 7 | 39 | **44.8** | 136 | 464 | 54.3 | 118 | 511 |
| 40 | 3 | L | 28.0 | 57 | 181 | **24.2** | 68 | 144 | **137.0** | 213 | 847 | 150.5 | 305 | 826 |
| 45 | 3 | L | 45.0 | 44 | 186 | **27.5** | 23 | 110 | 188.6 | 225 | 759 | **176.2** | 159 | 700 |
| 50 | 3 | L | 154.5 | 73 | 436 | **138.7** | 84 | 305 | 546.9 | 277 | 1448 | **487.9** | 218 | 1316 |
| 5 | 6 | L | **0.1** | 21 | 16 | 0.2 | 28 | 25 | **0.3** | 17 | 19 | 0.4 | 19 | 27 |
| 10 | 6 | L | **0.8** | 26 | 61 | **0.8** | 24 | 65 | **1.5** | 68 | 114 | 2.2 | 81 | 127 |
| 15 | 6 | L | **3.6** | 77 | 143 | 5.4 | 98 | 211 | **4.9** | 97 | 211 | 6.3 | 103 | 220 |
| 20 | 6 | L | **19.7** | 242 | 340 | 29.3 | 283 | 438 | **24.7** | 258 | 510 | 34.9 | 314 | 591 |
| 25 | 6 | L | **45.7** | 250 | 447 | 49.3 | 278 | 440 | **38.4** | 144 | 500 | 40.1 | 122 | 522 |
| 30 | 6 | L | **169.0** | 616 | 829 | 177.6 | 598 | 966 | **151.5** | 337 | 1056 | 188.8 | 426 | 1187 |
| 5 | 3 | H | **0.0** | 0 | 4 | 0.0 | 0 | 7 | **0.0** | 0 | 6 | 0.0 | 0 | 9 |
| 10 | 3 | H | **0.1** | 0 | 12 | 0.1 | 0 | 11 | **0.5** | 21 | 44 | 0.6 | 17 | 50 |
| 15 | 3 | H | **0.3** | 2 | 22 | **0.3** | 3 | 23 | **1.2** | 28 | 91 | 1.7 | 34 | 95 |
| 20 | 3 | H | **1.7** | 37 | 72 | 2.1 | 39 | 72 | **3.4** | 37 | 160 | 4.9 | 38 | 186 |
| 25 | 3 | H | 2.3 | 23 | 63 | **2.1** | 6 | 60 | **8.2** | 51 | 246 | 12.1 | 44 | 266 |
| 30 | 3 | H | 11.3 | 99 | 142 | **10.1** | 92 | 169 | **26.1** | 132 | 423 | 24.8 | 115 | 422 |
| 35 | 3 | H | **3.5** | 8 | 52 | 4.6 | 4 | 46 | **39.4** | 105 | 405 | 64.7 | 105 | 575 |
| 40 | 3 | H | **26.0** | 54 | 173 | 32.3 | 45 | 175 | **107.0** | 144 | 745 | 114.5 | 139 | 746 |
| 45 | 3 | H | **37.7** | 41 | 197 | 40.1 | 20 | 163 | **158.9** | 153 | 720 | 160.1 | 147 | 660 |
| 50 | 3 | H | 157.1 | 87 | 450 | **156.9** | 92 | 428 | 472.0 | 211 | 1296 | **462.2** | 231 | 1342 |
| 5 | 6 | H | **0.1** | 22 | 17 | 0.1 | 13 | 24 | **0.3** | 16 | 21 | 0.4 | 12 | 30 |
| 10 | 6 | H | **0.7** | 21 | 68 | 0.8 | 13 | 67 | **1.5** | 56 | 111 | 2.1 | 74 | 135 |
| 15 | 6 | H | **3.2** | 59 | 146 | 4.6 | 73 | 179 | **5.7** | 120 | 241 | 5.8 | 81 | 227 |
| 20 | 6 | H | **15.3** | 135 | 330 | 17.8 | 130 | 334 | **33.6** | 397 | 572 | 41.3 | 403 | 620 |
| 25 | 6 | H | **24.1** | 89 | 317 | 24.4 | 55 | 328 | **32.2** | 112 | 469 | 50.0 | 169 | 543 |
| 30 | 6 | H | 106.2 | 200 | 722 | **98.8** | 175 | 746 | 171.7 | 418 | 1105 | **134.0** | 272 | 1037 |
| Average | | | 27.4 | 76.6 | 181 | **27.1** | 75.2 | 183.9 | **69.9** | 125.4 | 424 | 71.0 | 125.4 | 440 |

Table A.XIV: Results on Archetti et al. (2011) PRP-ML Instances with 14 Customers and 6 Periods.

| Class | $F(ML)\|k$ | | | $F(ML)\|nk$ | | |
|---|---|---|---|---|---|---|
| | CPU | Nodes | Cuts | CPU | Nodes | Cuts |
| I | **5.9** | 167 | 183 | 6.7 | 139 | 207 |
| II | **4.7** | 128 | 162 | 5.3 | 103 | 177 |
| III | **10.9** | 367 | 241 | 11.4 | 284 | 248 |
| IV | **6.8** | 208 | 202 | 8.0 | 183 | 223 |
| Avg | **7.1** | 217 | 197 | 7.9 | 177 | 214 |

Table A.XV: Results on MVPRP and MVIRP when Allowing Multiple Visits

| Problem | Maximum number of visits | Cost index (%) | Best | CPU (s) | Number of visits (%) | | |
|---|---|---|---|---|---|---|---|
| | | | | | 1 | 2 | 3 |
| PRP | 1 | 100.00 | 30 | 87.5 | 100.00 | n/a | n/a |
| | 2 | 100.00 | 30 | 105.1 | 99.75 | 0.25 | n/a |
| | 3 | 100.00 | 30 | 134.2 | 99.83 | 0.17 | 0.00 |
| IRP | 1 | 100.00 | 24 | 552.6 | 100.00 | n/a | n/a |
| | 2 | 99.40 | 58 | 729.5 | 94.26 | 5.74 | n/a |
| | 3 | 99.40 | 60 | 683.0 | 94.64 | 5.15 | 0.21 |

# Appendix B
# Supplement to Chapter 4

Tables B.I-B.VI show the detailed average results of the benchmark A (Archetti et al., 2011) by problem type and each of them contains 5 instances. The number of new solutions for each problem type can be found in the column *New*. Tables B.VII-B.IX show the detailed results of the benchmark B (Boudia et al., 2005) for each instance. Similar to the previous section, we use boldface symbols if the total cost obtained by Op-ALNS is lower than the best solution in the literature, and the symbol ***** to indicate the best solution. More details on the instances and the solutions can be found on the website https://sites.google.com/site/YossiriAdulyasak/publications.

Table B.I: Detailed average results of the set A1 (14 Customers) - number 1-48.

| | | Total cost | | | | %diff from current best solutions | | | | New |
|---|---|---|---|---|---|---|---|---|---|---|
| Prob type | $\mathcal{H}$ | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | |
| 1 | 35786* | 36604 | 36219 | 36218 | 36218 | 2.3% | 1.2% | 1.2% | 1.2% | 0 |
| 2 | 36045* | 36498 | 36411 | 36245 | 36245 | 1.3% | 1.0% | 0.6% | 0.6% | 0 |
| 3 | 36604* | 37116 | 36971 | 36816 | 36795 | 1.4% | 1.0% | 0.6% | 0.5% | 1 |
| 4 | 73212 | 73190 | **73071** | **72937** | **72937*** | 0.0% | -0.2% | -0.4% | -0.4% | 3 |
| 5 | 73810 | **73593** | **73343** | **73250** | **73250*** | -0.3% | -0.6% | -0.8% | -0.8% | 5 |
| 6 | 74558 | **74161** | **73954** | **73912** | **73912*** | -0.5% | -0.8% | -0.9% | -0.9% | 5 |
| 7 | 28309 | 28557 | 28311 | **28217** | **28217*** | 0.9% | 0.0% | -0.3% | -0.3% | 2 |
| 8 | 28495 | 28731 | **28445** | **28445** | **28445*** | 0.8% | -0.2% | -0.2% | -0.2% | 3 |
| 9 | 29440 | 29525 | **29401** | **29395** | **29395*** | 0.3% | -0.1% | -0.2% | -0.2% | 3 |
| 10 | 65119 | **61992** | **61931** | **61931** | **61931*** | -4.8% | -4.9% | -4.9% | -4.9% | 5 |
| 11 | 64866 | **62864** | **62721** | **62721** | **62721*** | -3.1% | -3.3% | -3.3% | -3.3% | 5 |
| 12 | 66307 | **65109** | **64966** | **64964** | **64964*** | -1.8% | -2.0% | -2.0% | -2.0% | 3 |
| 13 | 38944* | 40694 | 40111 | 39561 | 39461 | 4.5% | 3.0% | 1.6% | 1.3% | 0 |
| 14 | 39706* | 40702 | 40538 | 40397 | 40397 | 2.5% | 2.1% | 1.7% | 1.7% | 1 |
| 15 | 41221* | 42245 | 42061 | 41765 | 41704 | 2.5% | 2.0% | 1.3% | 1.2% | 0 |
| 16 | 75951* | 76706 | 76521 | 76394 | 76394 | 1.0% | 0.8% | 0.6% | 0.6% | 2 |
| 17 | 76761 | 77434 | 76805 | **76615** | **76615*** | 0.9% | 0.1% | -0.2% | -0.2% | 3 |
| 18 | 78348* | 78945 | 78708 | 78502 | 78502 | 0.8% | 0.5% | 0.2% | 0.2% | 2 |
| 19 | 31134* | 32183 | 31977 | 31578 | 31578 | 3.4% | 2.7% | 1.4% | 1.4% | 2 |
| 20 | 31695* | 32478 | 31928 | 31915 | 31907 | 2.5% | 0.7% | 0.7% | 0.7% | 2 |
| 21 | 33163* | 33903 | 33794 | 33720 | 33611 | 2.2% | 1.9% | 1.7% | 1.4% | 2 |
| 22 | 67724 | **65540** | **65334** | **65334** | **65334*** | -3.2% | -3.5% | -3.5% | -3.5% | 5 |
| 23 | 68162 | **66612** | **66393** | **66291** | **66291*** | -2.3% | -2.6% | -2.7% | -2.7% | 5 |
| 24 | 70126 | **69575** | **69224** | **69141** | **69141*** | -0.8% | -1.3% | -1.4% | -1.4% | 4 |
| 25 | 209667* | 210944 | 210367 | 210246 | 210246 | 0.6% | 0.3% | 0.3% | 0.3% | 0 |
| 26 | 209939* | 210889 | 210556 | 210465 | 210465 | 0.5% | 0.3% | 0.3% | 0.3% | 0 |
| 27 | 210376* | 210879 | 210647 | 210603 | 210603 | 0.2% | 0.1% | 0.1% | 0.1% | 1 |
| 28 | 536626 | 537183 | 536823 | **536595** | **536595*** | 0.1% | 0.0% | -0.0% | -0.0% | 2 |
| 29 | 537332 | 537147 | **536862** | **536816** | **536816*** | 0.0% | -0.1% | -0.1% | -0.1% | 5 |
| 30 | 537950 | **537645** | **537383** | **537383** | **537383*** | -0.1% | -0.1% | -0.1% | -0.1% | 5 |
| 31 | 202086* | 202812 | 202314 | 202202 | 202202 | 0.4% | 0.1% | 0.1% | 0.1% | 2 |
| 32 | 202267* | 202839 | 202555 | 202555 | 202555 | 0.3% | 0.1% | 0.1% | 0.1% | 1 |
| 33 | 203212 | 203406 | 203142 | **203107** | **203107*** | 0.1% | 0.0% | -0.1% | -0.1% | 3 |
| 34 | 528501 | **525535** | **525347** | **525347** | **525347*** | -0.6% | -0.6% | -0.6% | -0.6% | 5 |
| 35 | 528258 | **526202** | **526086** | **526086** | **526086*** | -0.4% | -0.4% | -0.4% | -0.4% | 5 |
| 36 | 529699 | **528416** | **528286** | **528286** | **528286*** | -0.2% | -0.3% | -0.3% | -0.3% | 3 |
| 37 | 212721* | 215861 | 215252 | 215182 | 215182 | 1.5% | 1.2% | 1.2% | 1.2% | 0 |
| 38 | 213523* | 216079 | 215265 | 214995 | 214995 | 1.2% | 0.8% | 0.7% | 0.7% | 1 |
| 39 | 214993* | 216333 | 215821 | 215606 | 215606 | 0.6% | 0.4% | 0.3% | 0.3% | 1 |
| 40 | 539356* | 541316 | 540264 | 540253 | 540253 | 0.4% | 0.2% | 0.2% | 0.2% | 1 |
| 41 | 540157 | 541098 | **540081** | **540065** | **540065*** | 0.2% | -0.0% | -0.0% | -0.0% | 3 |
| 42 | 541740* | 542234 | 541965 | 541965 | 541965 | 0.1% | 0.0% | 0.0% | 0.0% | 2 |
| 43 | 204882* | 207039 | 206001 | 205990 | 205990 | 1.1% | 0.5% | 0.5% | 0.5% | 1 |
| 44 | 205467* | 206656 | 205969 | 205969 | 205969 | 0.6% | 0.2% | 0.2% | 0.2% | 1 |
| 45 | 206935* | 207762 | 207437 | 207437 | 207437 | 0.4% | 0.2% | 0.2% | 0.2% | 2 |
| 46 | 531028 | **528942** | **528748** | **528741** | **528741*** | -0.4% | -0.4% | -0.4% | -0.4% | 5 |
| 47 | 531554 | **530011** | **529840** | **529840** | **529840*** | -0.3% | -0.3% | -0.3% | -0.3% | 5 |
| 48 | 533518 | **533136** | **532673** | **532673** | **532673*** | -0.1% | -0.2% | -0.2% | -0.2% | 3 |

Table B.II: Detailed average results of the set A1 (14 Customers) - number 49-96.

| | | Total cost | | | | %diff from current best solutions | | | | New |
|---|---|---|---|---|---|---|---|---|---|---|
| Prob type | $\mathcal{H}$ | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | |
| 49 | 62233* | 70513 | 69666 | 67808 | 67771 | 13.3% | 11.9% | 9.0% | 8.9% | 0 |
| 50 | 65434* | 72447 | 71505 | 71183 | 70551 | 10.7% | 9.3% | 8.8% | 7.8% | 0 |
| 51 | 71018* | 78757 | 77906 | 77464 | 76300 | 10.9% | 9.7% | 9.1% | 7.4% | 0 |
| 52 | 99002* | 103563 | 103103 | 102448 | 102448 | 4.6% | 4.1% | 3.5% | 3.5% | 1 |
| 53 | 101673* | 107564 | 106268 | 103494 | 103494 | 5.8% | 4.5% | 1.8% | 1.8% | 2 |
| 54 | 107892* | 113871 | 113358 | 112861 | 112464 | 5.5% | 5.1% | 4.6% | 4.2% | 0 |
| 55 | 54058* | 58456 | 58313 | 57153 | 57104 | 8.1% | 7.9% | 5.7% | 5.6% | 1 |
| 56 | 56922* | 60545 | 59210 | 58235 | 58234 | 6.4% | 4.0% | 2.3% | 2.3% | 2 |
| 57 | 62479* | 68769 | 68132 | 67186 | 66901 | 10.1% | 9.0% | 7.5% | 7.1% | 1 |
| 58 | 90406* | 92099 | 91572 | 91410 | 91410 | 1.9% | 1.3% | 1.1% | 1.1% | 1 |
| 59 | 93196* | 95329 | 95248 | 93962 | 93962 | 2.3% | 2.2% | 0.8% | 0.8% | 2 |
| 60 | 99464* | 104531 | 104096 | 104096 | 104089 | 5.1% | 4.7% | 4.7% | 4.6% | 1 |
| 61 | 91043* | 106203 | 101917 | 100350 | 100350 | 16.7% | 11.9% | 10.2% | 10.2% | 1 |
| 62 | 97162* | 114882 | 110954 | 107528 | 104326 | 18.2% | 14.2% | 10.7% | 7.4% | 0 |
| 63 | 108344* | 126152 | 124305 | 121159 | 117566 | 16.4% | 14.7% | 11.8% | 8.5% | 0 |
| 64 | 127563* | 137538 | 136336 | 134500 | 131804 | 7.8% | 6.9% | 5.4% | 3.3% | 1 |
| 65 | 133615* | 144858 | 138194 | 137975 | 137152 | 8.4% | 3.4% | 3.3% | 2.6% | 2 |
| 66 | 145850* | 155434 | 150855 | 149930 | 149930 | 6.6% | 3.4% | 2.8% | 2.8% | 3 |
| 67 | 82868* | 91294 | 89906 | 88163 | 87521 | 10.2% | 8.5% | 6.4% | 5.6% | 1 |
| 68 | 88285* | 97683 | 95834 | 94304 | 94073 | 10.6% | 8.6% | 6.8% | 6.6% | 1 |
| 69 | 98829* | 108521 | 106493 | 105546 | 105112 | 9.8% | 7.8% | 6.8% | 6.4% | 1 |
| 70 | 119167* | 123995 | 122557 | 121104 | 121098 | 4.1% | 2.8% | 1.6% | 1.6% | 1 |
| 71 | 124633* | 129341 | 128032 | 127632 | 127312 | 3.8% | 2.7% | 2.4% | 2.1% | 1 |
| 72 | 135642* | 145457 | 145449 | 144163 | 142528 | 7.2% | 7.2% | 6.3% | 5.1% | 1 |
| 73 | 23691 | **23604** | **23501** | **23501** | **23501*** | -0.4% | -0.8% | -0.8% | -0.8% | 4 |
| 74 | 24066* | 24132 | 24070 | **24066** | **24066*** | 0.3% | 0.0% | 0.0% | 0.0% | 2 |
| 75 | 25022* | 25707 | 25642 | 25628 | 25598 | 2.7% | 2.5% | 2.4% | 2.3% | 1 |
| 76 | 58367 | **57196** | **57172** | **57172** | **57172*** | -2.0% | -2.0% | -2.0% | -2.0% | 5 |
| 77 | 58937 | **58332** | **58181** | **58166** | **58166*** | -1.0% | -1.3% | -1.3% | -1.3% | 5 |
| 78 | 60704* | 61285 | 61161 | 61149 | 61149 | 1.0% | 0.8% | 0.7% | 0.7% | 1 |
| 79 | 26583* | 27094 | 27009 | 26959 | 26959 | 1.9% | 1.6% | 1.4% | 1.4% | 1 |
| 80 | 27507* | 27716 | 27575 | 27525 | 27525 | 0.8% | 0.3% | 0.1% | 0.1% | 2 |
| 81 | 28623* | 29702 | 29505 | 29236 | 29227 | 3.8% | 3.1% | 2.1% | 2.1% | 2 |
| 82 | 61261 | **60426** | **60423** | **60423** | **60423*** | -1.4% | -1.4% | -1.4% | -1.4% | 5 |
| 83 | 62249 | **61791** | **61676** | **61676** | **61676*** | -0.7% | -0.9% | -0.9% | -0.9% | 4 |
| 84 | 64309* | 65771 | 65552 | 65455 | 65455 | 2.3% | 1.9% | 1.8% | 1.8% | 1 |
| 85 | 197463 | 197578 | **197295** | **197295** | **197295*** | 0.1% | -0.1% | -0.1% | -0.1% | 4 |
| 86 | 197838 | 197987 | 197853 | **197796** | **197796*** | 0.1% | 0.0% | -0.0% | -0.0% | 3 |
| 87 | 198794* | 199678 | 199466 | 199392 | 199392 | 0.4% | 0.3% | 0.3% | 0.3% | 1 |
| 88 | 521759 | **520576** | **520535** | **520535** | **520535*** | -0.2% | -0.2% | -0.2% | -0.2% | 5 |
| 89 | 522329 | **521756** | **521576** | **521573** | **521573*** | -0.1% | -0.1% | -0.1% | -0.1% | 5 |
| 90 | 524096* | 524830 | 524737 | 524652 | 524652 | 0.1% | 0.1% | 0.1% | 0.1% | 1 |
| 91 | 200355* | 200798 | 200667 | 200665 | 200665 | 0.2% | 0.2% | 0.2% | 0.2% | 1 |
| 92 | 201279 | 201473 | 201205 | **201009** | **201009*** | 0.1% | 0.0% | -0.1% | -0.1% | 4 |
| 93 | 202395* | 203869 | 203673 | 203503 | 203503 | 0.7% | 0.6% | 0.5% | 0.5% | 1 |
| 94 | 524653 | **524096** | **523824** | **523824** | **523824*** | -0.1% | -0.2% | -0.2% | -0.2% | 5 |
| 95 | 525641 | 525395 | **525186** | **525129** | **525129*** | 0.0% | -0.1% | -0.1% | -0.1% | 4 |
| 96 | 527701* | 529395 | 528992 | 528912 | 528912 | 0.3% | 0.2% | 0.2% | 0.2% | 0 |
| Avg | 180830* | 182951 | 182339 | 181975 | 181803 | 1.2% | 0.8% | 0.6% | 0.5% | 211 |

Table B.III: Detailed average results of the set A2 (50 Customers) - number 1-48.

| | | Total cost | | | | %diff from current best solutions | | | | New |
|---|---|---|---|---|---|---|---|---|---|---|
| Prob type | $\mathcal{H}$ | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | |
| 1 | 111142 | **111025** | **110415** | **110194** | **110194*** | -0.1% | -0.7% | -0.9% | -0.9% | 5 |
| 2 | 112138 | **111914** | **111452** | **111344** | **111344*** | -0.2% | -0.6% | -0.7% | -0.7% | 5 |
| 3 | 114514 | **113822** | **113426** | **113385** | **113385*** | -0.6% | -1.0% | -1.0% | -1.0% | 5 |
| 4 | 232117 | 232680 | **232065** | **231719** | **231715*** | 0.2% | -0.0% | -0.2% | -0.2% | 4 |
| 5 | 233384 | 233838 | 233391 | **233145** | **233145*** | 0.2% | 0.0% | -0.1% | -0.1% | 3 |
| 6 | 235759 | 235774 | **235432** | **235414** | **235414*** | 0.0% | -0.1% | -0.1% | -0.1% | 3 |
| 7 | 91607 | 91631 | **90789** | **90456** | **90456*** | 0.0% | -0.9% | -1.3% | -1.3% | 4 |
| 8 | 91737* | 92491 | 92255 | 92023 | 92023 | 0.8% | 0.6% | 0.3% | 0.3% | 2 |
| 9 | 93917* | 94460 | 94094 | 94060 | 94060 | 0.6% | 0.2% | 0.2% | 0.2% | 2 |
| 10 | 211665 | **205500** | **205404** | **205315** | **205315*** | -2.9% | -3.0% | -3.0% | -3.0% | 5 |
| 11 | 211008 | **206542** | **206514** | **206514** | **206514*** | -2.1% | -2.1% | -2.1% | -2.1% | 5 |
| 12 | 212683 | **209203** | **209106** | **209054** | **209054*** | -1.6% | -1.7% | -1.7% | -1.7% | 5 |
| 13 | 120795* | 122369 | 121605 | 121245 | 121184 | 1.3% | 0.7% | 0.4% | 0.3% | 2 |
| 14 | 122216* | 124390 | 123333 | 123024 | 122995 | 1.8% | 0.9% | 0.7% | 0.6% | 0 |
| 15 | 127897 | 128448 | 127865 | **127236** | **127112*** | 0.4% | 0.0% | -0.5% | -0.6% | 4 |
| 16 | 240333 | 242056 | 240491 | **239740** | **239740*** | 0.7% | 0.1% | -0.2% | -0.2% | 4 |
| 17 | 243371 | 244019 | **243235** | **242907** | **242907*** | 0.3% | -0.1% | -0.2% | -0.2% | 3 |
| 18 | 248541 | 249014 | **247791** | **247723** | **247723*** | 0.2% | -0.3% | -0.3% | -0.3% | 2 |
| 19 | 98637 | 100621 | 99489 | **97999** | **97999*** | 2.0% | 0.9% | -0.6% | -0.6% | 4 |
| 20 | 101752 | 102216 | **101030** | **100604** | **100602*** | 0.5% | -0.7% | -1.1% | -1.1% | 3 |
| 21 | 106330 | 106861 | **106206** | **105804** | **105804*** | 0.5% | -0.1% | -0.5% | -0.5% | 4 |
| 22 | 218948 | **213040** | **212829** | **212775** | **212629*** | -2.7% | -2.8% | -2.8% | -2.9% | 5 |
| 23 | 222183 | **215312** | **215292** | **215289** | **215289*** | -3.1% | -3.1% | -3.1% | -3.1% | 5 |
| 24 | 226431 | **220247** | **220039** | **220006** | **220006*** | -2.7% | -2.8% | -2.8% | -2.8% | 5 |
| 25 | 720389 | 720343 | **719550** | **719412** | **719412*** | 0.0% | -0.1% | -0.1% | -0.1% | 5 |
| 26 | 721422 | 721558 | **720944** | **720638** | **720638*** | 0.0% | -0.1% | -0.1% | -0.1% | 5 |
| 27 | 723653 | 723505 | **722946** | **722781** | **722781*** | 0.0% | -0.1% | -0.1% | -0.1% | 5 |
| 28 | 1856295 | **1855162** | **1854341** | **1853939** | **1853871*** | -0.1% | -0.1% | -0.1% | -0.1% | 5 |
| 29 | 1857632 | 1857080 | **1855924** | **1855350** | **1855335*** | 0.0% | -0.1% | -0.1% | -0.1% | 5 |
| 30 | 1860048 | **1859112** | **1858226** | **1858056** | **1858056*** | -0.1% | -0.1% | -0.1% | -0.1% | 5 |
| 31 | 697989 | 698133 | 697688 | **697334** | **697223*** | 0.0% | 0.0% | -0.1% | -0.1% | 5 |
| 32 | 699413 | 699385 | **698575** | **698432** | **698432*** | 0.0% | -0.1% | -0.1% | -0.1% | 5 |
| 33 | 701478 | 701833 | **700750** | **700439** | **700439*** | 0.1% | -0.1% | -0.1% | -0.1% | 5 |
| 34 | 1828171 | **1814660** | **1814454** | **1814338** | **1814338*** | -0.7% | -0.8% | -0.8% | -0.8% | 5 |
| 35 | 1831926 | **1815701** | **1815610** | **1815610** | **1815610*** | -0.9% | -0.9% | -0.9% | -0.9% | 5 |
| 36 | 1832113 | **1818037** | **1817832** | **1817797** | **1817797*** | -0.8% | -0.8% | -0.8% | -0.8% | 5 |
| 37 | 729581* | 732852 | 731343 | 730306 | 730306 | 0.4% | 0.2% | 0.1% | 0.1% | 1 |
| 38 | 731219* | 735939 | 734231 | 733465 | 733465 | 0.6% | 0.4% | 0.3% | 0.3% | 1 |
| 39 | 736006* | 738403 | 737225 | 736748 | 736748 | 0.3% | 0.2% | 0.1% | 0.1% | 1 |
| 40 | 1863404 | 1864927 | 1863636 | **1862905** | **1862905*** | 0.1% | 0.0% | -0.0% | -0.0% | 4 |
| 41 | 1867045 | 1866567 | **1865638** | **1865377** | **1865377*** | 0.0% | -0.1% | -0.1% | -0.1% | 5 |
| 42 | 1871014 | 1871670 | 1870141 | **1869943** | **1869943*** | 0.0% | 0.0% | -0.1% | -0.1% | 4 |
| 43 | 705334* | 707790 | 706461 | 706233 | 706233 | 0.3% | 0.2% | 0.1% | 0.1% | 0 |
| 44 | 707738* | 709812 | 708853 | 708603 | 708603 | 0.3% | 0.2% | 0.1% | 0.1% | 2 |
| 45 | 712176* | 714411 | 712956 | 712884 | 712884 | 0.3% | 0.1% | 0.1% | 0.1% | 2 |
| 46 | 1836554 | **1823326** | **1822754** | **1822462** | **1822462*** | -0.7% | -0.8% | -0.8% | -0.8% | 5 |
| 47 | 1837671 | **1825488** | **1825036** | **1825014** | **1825014*** | -0.7% | -0.7% | -0.7% | -0.7% | 5 |
| 48 | 1841394 | **1830200** | **1829866** | **1829824** | **1829824*** | -0.6% | -0.6% | -0.6% | -0.6% | 5 |

Table B.IV: Detailed average results of the set A2 (50 Customers) - number 49-96.

| | | Total cost | | | | %diff from current best solutions | | | | New |
|---|---|---|---|---|---|---|---|---|---|---|
| Prob type | $\mathcal{H}$ | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | |
| 49 | 147245 | 150579 | 148867 | **147036** | **146885*** | 2.3% | 1.1% | -0.1% | -0.2% | 3 |
| 50 | 152864 | 156498 | 154976 | 153289 | **152850*** | 2.4% | 1.4% | 0.3% | -0.0% | 2 |
| 51 | 164246* | 168206 | 165904 | 164684 | 164414 | 2.4% | 1.0% | 0.3% | 0.1% | 2 |
| 52 | 267056 | 269044 | 267273 | **265573** | **265498*** | 0.7% | 0.1% | -0.6% | -0.6% | 3 |
| 53 | 273852 | 274217 | **271971** | **271084** | **271084*** | 0.1% | -0.7% | -1.0% | -1.0% | 3 |
| 54 | 285729 | 286694 | **285160** | **283709** | **283473*** | 0.3% | -0.2% | -0.7% | -0.8% | 3 |
| 55 | 120446* | 125405 | 123455 | 122396 | 121603 | 4.1% | 2.5% | 1.6% | 1.0% | 1 |
| 56 | 127804 | 131277 | 128567 | 128056 | **127548*** | 2.7% | 0.6% | 0.2% | -0.2% | 1 |
| 57 | 139491* | 143245 | 140710 | 139842 | 139511 | 2.7% | 0.9% | 0.3% | 0.0% | 2 |
| 58 | 240508 | **236240** | **235443** | **234692** | **234692*** | -1.8% | -2.1% | -2.4% | -2.4% | 5 |
| 59 | 247618 | **241597** | **240877** | **240714** | **240714*** | -2.4% | -2.7% | -2.8% | -2.8% | 5 |
| 60 | 259026 | **253656** | **253129** | **252871** | **252871*** | -2.1% | -2.3% | -2.4% | -2.4% | 5 |
| 61 | 184494* | 196694 | 194400 | 192075 | 192017 | 6.6% | 5.4% | 4.1% | 4.1% | 0 |
| 62 | 197563* | 205861 | 203692 | 201916 | 201462 | 4.2% | 3.1% | 2.2% | 2.0% | 1 |
| 63 | 220629* | 231307 | 226364 | 224272 | 224261 | 4.8% | 2.6% | 1.7% | 1.6% | 0 |
| 64 | 305026* | 312300 | 310802 | 307379 | 306756 | 2.4% | 1.9% | 0.8% | 0.6% | 2 |
| 65 | 319038 | 322927 | 321141 | 319315 | **318089*** | 1.2% | 0.7% | 0.1% | -0.3% | 3 |
| 66 | 341606* | 349446 | 345513 | 341640 | 341640 | 2.3% | 1.1% | 0.0% | 0.0% | 3 |
| 67 | 158277* | 173745 | 165893 | 163697 | 163027 | 9.8% | 4.8% | 3.4% | 3.0% | 1 |
| 68 | 169990* | 183552 | 179799 | 175475 | 173593 | 8.0% | 5.8% | 3.2% | 2.1% | 2 |
| 69 | 193984* | 206670 | 201947 | 200746 | 199122 | 6.5% | 4.1% | 3.5% | 2.6% | 0 |
| 70 | 278033 | **275528** | **273597** | **272581** | **272557*** | -0.9% | -1.6% | -2.0% | -2.0% | 5 |
| 71 | 290009 | **287866** | **287301** | **284135** | **284135*** | -0.7% | -0.9% | -2.0% | -2.0% | 5 |
| 72 | 314116 | **313384** | **311735** | **310503** | **309402*** | -0.2% | -0.8% | -1.2% | -1.5% | 5 |
| 73 | 77974 | **77197** | **76980** | **76887** | **76863*** | -1.0% | -1.3% | -1.4% | -1.4% | 5 |
| 74 | 79234 | **78326** | **78172** | **78120** | **78120*** | -1.1% | -1.3% | -1.4% | -1.4% | 5 |
| 75 | 82270 | **80676** | **80523** | **80521** | **80488*** | -1.9% | -2.1% | -2.1% | -2.2% | 5 |
| 76 | 195134 | **192585** | **192440** | **192330** | **192330*** | -1.3% | -1.4% | -1.4% | -1.4% | 5 |
| 77 | 196947 | **193594** | **193530** | **193446** | **193446*** | -1.7% | -1.7% | -1.8% | -1.8% | 5 |
| 78 | 200769 | **195958** | **195808** | **195808** | **195808*** | -2.4% | -2.5% | -2.5% | -2.5% | 5 |
| 79 | 85392 | **85042** | **84493** | **84144** | **84130*** | -0.4% | -1.1% | -1.5% | -1.5% | 5 |
| 80 | 88075 | **87089** | **86627** | **86577** | **86577*** | -1.1% | -1.6% | -1.7% | -1.7% | 5 |
| 81 | 92810 | **92168** | **91559** | **91109** | **91033*** | -0.7% | -1.3% | -1.8% | -1.9% | 5 |
| 82 | 204185 | **200697** | **200096** | **200032** | **200032*** | -1.7% | -2.0% | -2.0% | -2.0% | 5 |
| 83 | 205559 | **203044** | **202670** | **202596** | **202570*** | -1.2% | -1.4% | -1.4% | -1.5% | 5 |
| 84 | 209703 | **207708** | **207376** | **207113** | **207113*** | -1.0% | -1.1% | -1.2% | -1.2% | 5 |
| 85 | 681071 | **678781** | **678547** | **678448** | **678448*** | -0.3% | -0.4% | -0.4% | -0.4% | 5 |
| 86 | 682086 | **679869** | **679652** | **679604** | **679604*** | -0.3% | -0.4% | -0.4% | -0.4% | 5 |
| 87 | 684877 | **682268** | **682057** | **682008** | **682008*** | -0.4% | -0.4% | -0.4% | -0.4% | 5 |
| 88 | 1801653 | **1795588** | **1795465** | **1795445** | **1795445*** | -0.3% | -0.3% | -0.3% | -0.3% | 5 |
| 89 | 1802527 | **1796604** | **1796537** | **1796536** | **1796536*** | -0.3% | -0.3% | -0.3% | -0.3% | 5 |
| 90 | 1806032 | **1798984** | **1798852** | **1798852** | **1798852*** | -0.4% | -0.4% | -0.4% | -0.4% | 5 |
| 91 | 688473 | **687285** | **686304** | **686264** | **686264*** | -0.2% | -0.3% | -0.3% | -0.3% | 5 |
| 92 | 690933 | **689758** | **689142** | **688920** | **688920*** | -0.2% | -0.3% | -0.3% | -0.3% | 5 |
| 93 | 695258 | **694271** | **693811** | **693631** | **693631*** | -0.1% | -0.2% | -0.2% | -0.2% | 5 |
| 94 | 1811928 | **1804420** | **1803750** | **1803749** | **1803749*** | -0.4% | -0.5% | -0.5% | -0.5% | 5 |
| 95 | 1814102 | **1806440** | **1806143** | **1806044** | **1806044*** | -0.4% | -0.4% | -0.4% | -0.4% | 5 |
| 96 | 1815969 | **1811258** | **1810716** | **1810641** | **1810641*** | -0.3% | -0.3% | -0.3% | -0.3% | 5 |
| Avg | 592608 | **591968** | **590920** | **590327** | **590210*** | -0.1% | -0.3% | -0.4% | -0.4% | 366 |

Table B.V: Detailed average results of the set A3 (100 Customers) - number 1-48.

| | | Total cost | | | | %diff from current best solutions | | | | New |
|---|---|---|---|---|---|---|---|---|---|---|
| Prob type | $\mathcal{H}$ | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | |
| 1 | 203837 | **200932** | **200506** | **200364** | **200339*** | -1.4% | -1.6% | -1.7% | -1.7% | 5 |
| 2 | 205823 | **203186** | **202801** | **202690** | **202505*** | -1.3% | -1.5% | -1.5% | -1.6% | 4 |
| 3 | 209930 | **207840** | **207506** | **207375** | **207355*** | -1.0% | -1.2% | -1.2% | -1.2% | 4 |
| 4 | 424668 | **422951** | **422237** | **422036** | **421609*** | -0.4% | -0.6% | -0.6% | -0.7% | 4 |
| 5 | 426709 | **424444** | **424065** | **423976** | **423951*** | -0.5% | -0.6% | -0.6% | -0.6% | 4 |
| 6 | 429843 | **429371** | **429068** | **428985** | **428985*** | -0.1% | -0.2% | -0.2% | -0.2% | 4 |
| 7 | 166160 | 166297 | **165852** | **165374** | **165353*** | 0.1% | -0.2% | -0.5% | -0.5% | 4 |
| 8 | 169153 | **168174** | **167798** | **167724** | **167724*** | -0.6% | -0.8% | -0.8% | -0.8% | 4 |
| 9 | 174620 | **173739** | **173323** | **172925** | **172756*** | -0.5% | -0.7% | -1.0% | -1.1% | 4 |
| 10 | 383901 | **375202** | **374872** | **374834** | **374834*** | -2.3% | -2.4% | -2.4% | -2.4% | 5 |
| 11 | 386738 | **376772** | **376718** | **376641** | **376641*** | -2.6% | -2.6% | -2.6% | -2.6% | 5 |
| 12 | 393261 | **382199** | **381934** | **381887** | **381887*** | -2.8% | -2.9% | -2.9% | -2.9% | 5 |
| 13 | 220312 | **219959** | **218833** | **218325** | **218235*** | -0.2% | -0.7% | -0.9% | -0.9% | 4 |
| 14 | 224492 | 224721 | **223563** | **223273** | **222975*** | 0.1% | -0.4% | -0.5% | -0.7% | 4 |
| 15 | 234290 | 234744 | **233631** | **233046** | **232930*** | 0.2% | -0.3% | -0.5% | -0.6% | 4 |
| 16 | 440834 | **439635** | **438333** | **437764** | **437591*** | -0.3% | -0.6% | -0.7% | -0.7% | 4 |
| 17 | 443582 | 444278 | 443633 | **443028** | **442954*** | 0.2% | 0.0% | -0.1% | -0.1% | 4 |
| 18 | 453472 | 454973 | 454343 | 453787 | **453347*** | 0.3% | 0.2% | 0.1% | 0.0% | 3 |
| 19 | 180038 | 181501 | 180440 | 180049 | **179631*** | 0.8% | 0.2% | 0.0% | -0.2% | 3 |
| 20 | 185450 | 186197 | 185702 | **185237** | **184689*** | 0.4% | 0.1% | -0.1% | -0.4% | 3 |
| 21 | 195769 | 196504 | **195621** | **194467** | **194348*** | 0.4% | -0.1% | -0.7% | -0.7% | 4 |
| 22 | 398836 | **388737** | **388285** | **387987** | **387929*** | -2.5% | -2.6% | -2.7% | -2.7% | 5 |
| 23 | 403532 | **392286** | **391971** | **391948** | **391948*** | -2.8% | -2.9% | -2.9% | -2.9% | 5 |
| 24 | 414468 | **403005** | **402877** | **402553** | **402553*** | -2.8% | -2.8% | -2.9% | -2.9% | 5 |
| 25 | 1330842* | 1335123 | 1334136 | 1333711 | 1333711 | 0.3% | 0.2% | 0.2% | 0.2% | 4 |
| 26 | 1333933* | 1335814 | 1335359 | 1335135 | 1335112 | 0.1% | 0.1% | 0.1% | 0.1% | 4 |
| 27 | 1338874* | 1341212 | 1340297 | 1340000 | 1340000 | 0.2% | 0.1% | 0.1% | 0.1% | 4 |
| 28 | 3431183* | 3438882 | 3436893 | 3434891 | 3434656 | 0.2% | 0.2% | 0.1% | 0.1% | 4 |
| 29 | 3433231* | 3440899 | 3439355 | 3438014 | 3437944 | 0.2% | 0.2% | 0.1% | 0.1% | 4 |
| 30 | 3438241* | 3446924 | 3445494 | 3444000 | 3443932 | 0.3% | 0.2% | 0.2% | 0.2% | 4 |
| 31 | 1291166* | 1293807 | 1292418 | 1291373 | 1291243 | 0.2% | 0.1% | 0.0% | 0.0% | 4 |
| 32 | 1292485* | 1296851 | 1295237 | 1294321 | 1293987 | 0.3% | 0.2% | 0.1% | 0.1% | 4 |
| 33 | 1297691* | 1301155 | 1299675 | 1299018 | 1299017 | 0.3% | 0.2% | 0.1% | 0.1% | 4 |
| 34 | 3382493 | **3359099** | **3358635** | **3358590** | **3358590*** | -0.7% | -0.7% | -0.7% | -0.7% | 4 |
| 35 | 3382040 | **3360795** | **3360674** | **3360647** | **3360647*** | -0.6% | -0.6% | -0.6% | -0.6% | 4 |
| 36 | 3387158 | **3366447** | **3366029** | **3365921** | **3365921*** | -0.6% | -0.6% | -0.6% | -0.6% | 4 |
| 37 | 1347052* | 1353810 | 1351810 | 1351393 | 1351364 | 0.5% | 0.4% | 0.3% | 0.3% | 2 |
| 38 | 1351738* | 1358624 | 1356393 | 1355921 | 1355921 | 0.5% | 0.3% | 0.3% | 0.3% | 1 |
| 39 | 1361323* | 1368371 | 1365944 | 1364622 | 1364533 | 0.5% | 0.3% | 0.2% | 0.2% | 3 |
| 40 | 3445419* | 3455163 | 3452321 | 3450388 | 3450184 | 0.3% | 0.2% | 0.1% | 0.1% | 4 |
| 41 | 3448846* | 3458010 | 3456636 | 3455612 | 3455578 | 0.3% | 0.2% | 0.2% | 0.2% | 4 |
| 42 | 3457897* | 3472264 | 3468394 | 3465713 | 3465485 | 0.4% | 0.3% | 0.2% | 0.2% | 4 |
| 43 | 1303028* | 1310379 | 1307813 | 1306593 | 1306245 | 0.6% | 0.4% | 0.3% | 0.2% | 2 |
| 44 | 1308437* | 1315667 | 1312173 | 1311414 | 1311284 | 0.6% | 0.3% | 0.2% | 0.2% | 3 |
| 45 | 1318013* | 1326202 | 1322178 | 1321095 | 1320962 | 0.6% | 0.3% | 0.2% | 0.2% | 3 |
| 46 | 3393407 | **3374419** | **3373496** | **3373419** | **3373419*** | -0.6% | -0.6% | -0.6% | -0.6% | 4 |
| 47 | 3397753 | **3377742** | **3377399** | **3377311** | **3377311*** | -0.6% | -0.6% | -0.6% | -0.6% | 4 |
| 48 | 3406711 | **3388904** | **3388179** | **3388128** | **3388128*** | -0.5% | -0.5% | -0.5% | -0.5% | 4 |

Table B.VI: Detailed average results of the set A3 (100 Customers) - number 49-96.

| | | Total cost | | | | %diff from current best solutions | | | | New |
|---|---|---|---|---|---|---|---|---|---|---|
| Prob type | $\mathcal{H}$ | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | |
| 49 | 266982* | 272974 | 270521 | 267889 | 267705 | 2.2% | 1.3% | 0.3% | 0.3% | 2 |
| 50 | 282368 | 283963 | **281278** | **279636** | **279046*** | 0.6% | -0.4% | -1.0% | -1.2% | 4 |
| 51 | 305103 | 308916 | **304665** | **302852** | **302692*** | 1.2% | -0.1% | -0.7% | -0.8% | 3 |
| 52 | 486203 | **486176** | **484763** | **482968** | **482374*** | -0.0% | -0.3% | -0.7% | -0.8% | 3 |
| 53 | 501738 | **498393** | **497183** | **495983** | **495503*** | -0.7% | -0.9% | -1.1% | -1.2% | 4 |
| 54 | 525235 | **522708** | **521388** | **520687** | **519271*** | -0.5% | -0.7% | -0.9% | -1.1% | 4 |
| 55 | 220444* | 229262 | 224815 | 222435 | 222402 | 4.0% | 2.0% | 0.9% | 0.9% | 2 |
| 56 | 233338* | 241217 | 238944 | 236277 | 234847 | 3.4% | 2.4% | 1.3% | 0.6% | 2 |
| 57 | 256709* | 263874 | 262167 | 261048 | 260041 | 2.8% | 2.1% | 1.7% | 1.3% | 2 |
| 58 | 438271 | **430873** | **429239** | **427059** | **426682*** | -1.7% | -2.1% | -2.6% | -2.6% | 5 |
| 59 | 451326 | **444065** | **442330** | **440620** | **440427*** | -1.6% | -2.0% | -2.4% | -2.4% | 4 |
| 60 | 474444 | **469439** | **467559** | **466103** | **465949*** | -1.1% | -1.5% | -1.8% | -1.8% | 4 |
| 61 | 335314* | 351022 | 348517 | 345459 | 343234 | 4.7% | 3.9% | 3.0% | 2.4% | 1 |
| 62 | 361350* | 375601 | 371189 | 367495 | 366176 | 3.9% | 2.7% | 1.7% | 1.3% | 1 |
| 63 | 410028* | 430360 | 423798 | 420479 | 418485 | 5.0% | 3.4% | 2.5% | 2.1% | 2 |
| 64 | 552990* | 561902 | 559494 | 556003 | 556003 | 1.6% | 1.2% | 0.5% | 0.5% | 3 |
| 65 | 578930 | 586793 | 580584 | **576882** | **576694*** | 1.4% | 0.3% | -0.4% | -0.4% | 3 |
| 66 | 627085* | 637770 | 632819 | 630680 | 629662 | 1.7% | 0.9% | 0.6% | 0.4% | 3 |
| 67 | 287624* | 305050 | 297355 | 292894 | 292647 | 6.1% | 3.4% | 1.8% | 1.7% | 1 |
| 68 | 308413* | 331633 | 325526 | 319001 | 317849 | 7.5% | 5.5% | 3.4% | 3.1% | 0 |
| 69 | 358909* | 381375 | 374666 | 371290 | 371091 | 6.3% | 4.4% | 3.4% | 3.4% | 0 |
| 70 | 505587 | **500393** | **495203** | **493022** | **492953*** | -1.0% | -2.1% | -2.5% | -2.5% | 5 |
| 71 | 525305 | **524308** | **521348** | **519015** | **518321*** | -0.2% | -0.8% | -1.2% | -1.3% | 4 |
| 72 | 575984 | **575348** | **571279** | **570768** | **570768*** | -0.1% | -0.8% | -0.9% | -0.9% | 4 |
| 73 | 142234 | **139644** | **139397** | **139392** | **139354*** | -1.8% | -2.0% | -2.0% | -2.0% | 5 |
| 74 | 143539 | **141910** | **141791** | **141647** | **141604*** | -1.1% | -1.2% | -1.3% | -1.3% | 4 |
| 75 | 149071 | **147301** | **147042** | **146947** | **146900*** | -1.2% | -1.4% | -1.4% | -1.5% | 4 |
| 76 | 358306 | **350528** | **350459** | **350231** | **350193*** | -2.2% | -2.2% | -2.3% | -2.3% | 5 |
| 77 | 357617 | **352841** | **352725** | **352657** | **352608*** | -1.3% | -1.4% | -1.4% | -1.4% | 4 |
| 78 | 363981 | **358267** | **357876** | **357739** | **357739*** | -1.6% | -1.7% | -1.7% | -1.7% | 4 |
| 79 | 155127 | **152782** | **152421** | **151946** | **151932*** | -1.5% | -1.7% | -2.1% | -2.1% | 5 |
| 80 | 158823 | **157957** | **157430** | **157254** | **157224*** | -0.5% | -0.9% | -1.0% | -1.0% | 4 |
| 81 | 169366 | **168381** | **167853** | **167474** | **167203*** | -0.6% | -0.9% | -1.1% | -1.3% | 4 |
| 82 | 371576 | **363549** | **363199** | **362871** | **362825*** | -2.2% | -2.3% | -2.3% | -2.4% | 5 |
| 83 | 373758 | **368188** | **367842** | **367538** | **367523*** | -1.5% | -1.6% | -1.7% | -1.7% | 5 |
| 84 | 384356 | **378740** | **378303** | **378143** | **377756*** | -1.5% | -1.6% | -1.6% | -1.7% | 4 |
| 85 | 1257213 | **1254851** | **1254298** | **1254136** | **1254136*** | -0.2% | -0.2% | -0.2% | -0.2% | 4 |
| 86 | 1258627 | **1256423** | **1256282** | **1256260** | **1256260*** | -0.2% | -0.2% | -0.2% | -0.2% | 4 |
| 87 | 1262940 | **1262133** | **1261478** | **1261408** | **1261408*** | -0.1% | -0.1% | -0.1% | -0.1% | 4 |
| 88 | 3324944 | **3320838** | **3320158** | **3320044** | **3320044*** | -0.1% | -0.1% | -0.1% | -0.1% | 4 |
| 89 | 3326391 | **3322243** | **3322124** | **3322121** | **3322121*** | -0.1% | -0.1% | -0.1% | -0.1% | 4 |
| 90 | 3330552 | **3327987** | **3327573** | **3327557** | **3327557*** | -0.1% | -0.1% | -0.1% | -0.1% | 4 |
| 91 | 1270777 | **1270126** | **1269115** | **1268279** | **1268246*** | -0.1% | -0.1% | -0.2% | -0.2% | 4 |
| 92 | 1274350 | **1273343** | **1272897** | **1272890** | **1272890*** | -0.1% | -0.1% | -0.1% | -0.1% | 4 |
| 93 | 1284854 | **1284550** | **1283273** | **1283000** | **1283000*** | -0.0% | -0.1% | -0.1% | -0.1% | 4 |
| 94 | 3340334 | **3335770** | **3335034** | **3334864** | **3334864*** | -0.1% | -0.2% | -0.2% | -0.2% | 4 |
| 95 | 3346257 | **3339123** | **3338757** | **3338754** | **3338754*** | -0.2% | -0.2% | -0.2% | -0.2% | 4 |
| 96 | 3357505 | **3350310** | **3349496** | **3349305** | **3349305*** | -0.2% | -0.2% | -0.2% | -0.2% | 4 |
| Avg | 1092509 | **1092348** | **1090774** | **1089818** | **1089589*** | -0.0% | -0.2% | -0.2% | -0.3% | 353 |

Table B.VII: Detailed results of the instance set B1 (50 Customers).

| | Total cost | | | | | | | %diff from current best solutions | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Ins | MA\|PM | RTS | TSPR | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> |
| 1 | 378378 | 398795 | 373550 | **348656** | **346999** | **343588** | **343588*** | -6.7% | -7.1% | -8.0% | -8.0% |
| 2 | 403913 | 373374 | 369303 | **361205** | **358313** | **357680** | **357680*** | -2.2% | -3.0% | -3.1% | -3.1% |
| 3 | 409573 | 353058 | 343960* | 350482 | 348727 | 345584 | 345584 | 1.9% | 1.4% | 0.5% | 0.5% |
| 4 | 399220 | 361176 | 354935 | **347876** | **345975** | **343096** | **343096*** | -2.0% | -2.5% | -3.3% | -3.3% |
| 5 | 422279 | 364819 | 357718 | **352234** | **349423** | **348270** | **348270*** | -1.5% | -2.3% | -2.6% | -2.6% |
| 6 | 407122 | 368082 | 367426 | **353839** | **353224** | **353224** | **353224*** | -3.7% | -3.9% | -3.9% | -3.9% |
| 7 | 414977 | 369963 | 363049 | **354466** | **352356** | **350847** | **350847*** | -2.4% | -2.9% | -3.4% | -3.4% |
| 8 | 379744 | 370822 | 364096 | **351383** | **349891** | **349891** | **349891*** | -3.5% | -3.9% | -3.9% | -3.9% |
| 9 | 407935 | 379379 | 374301 | **360673** | **355905** | **352559** | **350651*** | -3.6% | -4.9% | -5.8% | -6.3% |
| 10 | 396258 | 370655 | 363462 | **341463** | **339257** | **338837** | **338837*** | -6.1% | -6.7% | -6.8% | -6.8% |
| 11 | 402475 | 354025 | 352689 | **340887** | **337607** | **337607** | **337607*** | -3.3% | -4.3% | -4.3% | -4.3% |
| 12 | 358702 | 354981 | 351572 | **345503** | **344288** | **344003** | **343774*** | -1.7% | -2.1% | -2.2% | -2.2% |
| 13 | 371030 | 365432 | 361130 | **349799** | **346342** | **343004** | **342681*** | -3.1% | -4.1% | -5.0% | -5.1% |
| 14 | 406114 | 363404 | 353829 | 355776 | **352579** | **351877** | **351877*** | 0.6% | -0.4% | -0.6% | -0.6% |
| 15 | 373076 | 367659 | 361234 | **356600** | **354042** | **352749** | **351130*** | -1.3% | -2.0% | -2.3% | -2.8% |
| 16 | 379404 | 360534 | 356096 | **352768** | **350218** | **346237** | **344017*** | -0.9% | -1.7% | -2.8% | -3.4% |
| 17 | 406353 | 398442 | 389912 | **358963** | **352491** | **349582** | **349582*** | -7.9% | -9.6% | -10.3% | -10.3% |
| 18 | 401179 | 368533 | 361888 | **360090** | **354951** | **353211** | **353211*** | -0.5% | -1.9% | -2.4% | -2.4% |
| 19 | 406893 | 377073 | 368279 | **353187** | **350587** | **350587** | **350587*** | -4.1% | -4.8% | -4.8% | -4.8% |
| 20 | 398508 | 372141 | 364818 | **349125** | **347339** | **344176** | **343576*** | -4.3% | -4.8% | -5.7% | -5.8% |
| 21 | 397112 | 374743 | 370731 | **343918** | **340713** | **339136** | **337984*** | -7.2% | -8.1% | -8.5% | -8.8% |
| 22 | 358749 | 347329 | 335477 | 339743 | 336848 | **332674** | **332674*** | 1.3% | 0.4% | -0.8% | -0.8% |
| 23 | 407369 | 362619 | 360303 | **354172** | **351442** | **349456** | **349355*** | -1.7% | -2.5% | -3.0% | -3.0% |
| 24 | 369784 | 375022 | 359697 | **343776** | **341859** | **340057** | **340012*** | -4.4% | -5.0% | -5.5% | -5.5% |
| 25 | 411556 | 374682 | 361125 | **355535** | **354638** | **353163** | **350535*** | -1.5% | -1.8% | -2.2% | -2.9% |
| 26 | 408704 | 366167 | 358213 | **355124** | **353467** | **352237** | **352237*** | -0.9% | -1.3% | -1.7% | -1.7% |
| 27 | 366197 | 375261 | 360714 | **344520** | **343440** | **340832** | **340647*** | -4.5% | -4.8% | -5.5% | -5.6% |
| 28 | 401032 | 373155 | 362754 | **345993** | **342970** | **341061** | **341019*** | -4.6% | -5.5% | -6.0% | -6.0% |
| 29 | 384282 | 379320 | 371205 | **369262** | **368313** | **367812** | **367423*** | -0.5% | -0.8% | -0.9% | -1.0% |
| 30 | 369959 | 369223 | 357664 | **346408** | **345268** | **344754** | **344754*** | -3.1% | -3.5% | -3.6% | -3.6% |
| Avg | 393263 | 369662 | 361704 | **351448** | **348982** | **347260** | **346878*** | -2.8% | -3.5% | -3.9% | -4.1% |

Table B.VIII: Detailed results of the instance set B2 (100 Customers)

| | Total cost | | | | | | | %diff from current best solutions | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Ins | MA\|PM | RTS | TSPR | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> |
| 1 | 714401 | 711671 | 698538 | **639206** | **635154** | **633535** | **633535*** | -8.5% | -9.1% | -9.3% | -9.3% |
| 2 | 722047 | 694694 | 663692 | **641533** | **640920** | **640920** | **640920*** | -3.3% | -3.4% | -3.4% | -3.4% |
| 3 | 677598 | 683270 | 667862 | **631518** | **630169** | **626364** | **626168*** | -5.4% | -5.6% | -6.2% | -6.2% |
| 4 | 710552 | 718252 | 688698 | **626929** | **625060** | **623816** | **623816*** | -9.0% | -9.2% | -9.4% | -9.4% |
| 5 | 733040 | 731260 | 708950 | **650730** | **650535** | **645350** | **645335*** | -8.2% | -8.2% | -9.0% | -9.0% |
| 6 | 696146 | 744927 | 688060 | **634773** | **634088** | **633468** | **633468*** | -7.7% | -7.8% | -7.9% | -7.9% |
| 7 | 705322 | 695728 | 663719 | **630570** | **625995** | **624866** | **624682*** | -5.0% | -5.7% | -5.9% | -5.9% |
| 8 | 679210 | 706058 | 661189 | **627557** | **625355** | **622243** | **621973*** | -5.1% | -5.4% | -5.9% | -5.9% |
| 9 | 699518 | 705035 | 679439 | **638534** | **636533** | **634924** | **634924*** | -6.0% | -6.3% | -6.6% | -6.6% |
| 10 | 705778 | 696521 | 676606 | **638073** | **636102** | **635594** | **635594*** | -5.7% | -6.0% | -6.1% | -6.1% |
| 11 | 709122 | 711895 | 697770 | **642361** | **638365** | **636948** | **636948*** | -7.9% | -8.5% | -8.7% | -8.7% |
| 12 | 755726 | 703162 | 669059 | **647338** | **645784** | **645019** | **645019*** | -3.2% | -3.5% | -3.6% | -3.6% |
| 13 | 695466 | 721066 | 682208 | **637910** | **636368** | **633971** | **633971*** | -6.5% | -6.7% | -7.1% | -7.1% |
| 14 | 718260 | 698548 | 673717 | **639592** | **637694** | **636407** | **636171*** | -5.1% | -5.3% | -5.5% | -5.6% |
| 15 | 736041 | 711506 | 684363 | **644342** | **642593** | **639156** | **639156*** | -5.8% | -6.1% | -6.6% | -6.6% |
| 16 | 715209 | 714873 | 678969 | **643905** | **643112** | **642220** | **641644*** | -5.2% | -5.3% | -5.4% | -5.5% |
| 17 | 737832 | 702314 | 690434 | **655364** | **653512** | **651782** | **651782*** | -5.1% | -5.3% | -5.6% | -5.6% |
| 18 | 723413 | 720238 | 698709 | **641060** | **638127** | **636631** | **636631*** | -8.3% | -8.7% | -8.9% | -8.9% |
| 19 | 720218 | 748734 | 700400 | **649480** | **647811** | **646046** | **646046*** | -7.3% | -7.5% | -7.8% | -7.8% |
| 20 | 724727 | 729099 | 707717 | **663569** | **661841** | **659157** | **659097*** | -6.2% | -6.5% | -6.9% | -6.9% |
| 21 | 724328 | 738746 | 702766 | **645868** | **640986** | **637662** | **637662*** | -8.1% | -8.8% | -9.3% | -9.3% |
| 22 | 701506 | 702849 | 685149 | **653058** | **651954** | **651954** | **651954*** | -4.7% | -4.8% | -4.8% | -4.8% |
| 23 | 710033 | 712717 | 691659 | **631617** | **630273** | **627739** | **627739*** | -8.7% | -8.9% | -9.2% | -9.2% |
| 24 | 734327 | 727741 | 707566 | **646945** | **645543** | **645202** | **645202*** | -8.6% | -8.8% | -8.8% | -8.8% |
| 25 | 725446 | 725869 | 711439 | **646600** | **644016** | **641643** | **641643*** | -9.1% | -9.5% | -9.8% | -9.8% |
| 26 | 718939 | 700719 | 694508 | **636446** | **634025** | **630061** | **630030*** | -8.4% | -8.7% | -9.3% | -9.3% |
| 27 | 715068 | 686382 | 659865 | **629629** | **625496** | **623147** | **623147*** | -4.6% | -5.2% | -5.6% | -5.6% |
| 28 | 685117 | 700980 | 653260 | **633835** | **630575** | **627640** | **627565*** | -3.0% | -3.5% | -3.9% | -3.9% |
| 29 | 722571 | 725030 | 708137 | **647602** | **646374** | **645068** | **644471*** | -8.5% | -8.7% | -8.9% | -9.0% |
| 30 | 721850 | 698942 | 682501 | **636809** | **634925** | **632569** | **632569*** | -6.7% | -7.0% | -7.3% | -7.3% |
| Avg | 714627 | 712294 | 685898 | **641092** | **638976** | **637037** | **636962*** | -6.5% | -6.8% | -7.1% | -7.1% |

Table B.IX: Detailed results of the instance set B3 (200 Customers)

| | | | | Total cost | | | | %diff from current best solutions | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Ins | MA\|PM | RTS | TSPR | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> | Op-ALNS <FS> | Op-ALNS <I-100> | Op-ALNS <I-500> | Op-ALNS <I-1000> |
| 1 | 996151 | 1030684 | 951277 | **895166** | **891180** | **890195*** | - | -5.9% | -6.3% | -6.4% | - |
| 2 | 978373 | 1010158 | 898671 | **877999** | **875578** | **874170*** | - | -2.3% | -2.6% | -2.7% | - |
| 3 | 986147 | 1016681 | 929493 | **878190** | **874533** | **870588*** | - | -5.5% | -5.9% | -6.3% | - |
| 4 | 962937 | 1042854 | 953349 | **879897** | **879052** | **876436*** | - | -7.7% | -7.8% | -8.1% | - |
| 5 | 970638 | 1023680 | 941598 | **873178** | **868228** | **866448*** | - | -7.3% | -7.8% | -8.0% | - |
| 6 | 965646 | 1025262 | 935639 | **885610** | **883897** | **883797*** | - | -5.3% | -5.5% | -5.5% | - |
| 7 | 980562 | 1038746 | 967666 | **888909** | **885014** | **882653*** | - | -8.1% | -8.5% | -8.8% | - |
| 8 | 1014809 | 1066068 | 988430 | **874503** | **872089** | **871609*** | - | -11.5% | -11.8% | -11.8% | - |
| 9 | 967738 | 1018420 | 920347 | **861212** | **857984** | **855020*** | - | -6.4% | -6.8% | -7.1% | - |
| 10 | 1093230 | 1035240 | 990212 | **877964** | **875189** | **871739*** | - | -11.3% | -11.6% | -12.0% | - |
| 11 | 1008080 | 1037705 | 942341 | **881462** | **880409** | **879384*** | - | -6.5% | -6.6% | -6.7% | - |
| 12 | 998951 | 1035350 | 929992 | **887589** | **886232** | **885161*** | - | -4.6% | -4.7% | -4.8% | - |
| 13 | 984918 | 1063024 | 949453 | **895239** | **892511** | **888886*** | - | -5.7% | -6.0% | -6.4% | - |
| 14 | 964301 | 1024491 | 947613 | **886077** | **883983** | **881609*** | - | -6.5% | -6.7% | -7.0% | - |
| 15 | 981167 | 1026787 | 946464 | **879939** | **875989** | **873130*** | - | -7.0% | -7.4% | -7.7% | - |
| 16 | 1017777 | 1033656 | 965143 | **883321** | **878215** | **877192*** | - | -8.5% | -9.0% | -9.1% | - |
| 17 | 1073640 | 1022250 | 969354 | **878438** | **873935** | **872275*** | - | -9.4% | -9.8% | -10.0% | - |
| 18 | 1003670 | 1063306 | 969355 | **887448** | **884647** | **882181*** | - | -8.4% | -8.7% | -9.0% | - |
| 19 | 997348 | 1065705 | 968497 | **884680** | **883594** | **880700*** | - | -8.7% | -8.8% | -9.1% | - |
| 20 | 981788 | 1027134 | 931517 | **879213** | **876107** | **874986*** | - | -5.6% | -5.9% | -6.1% | - |
| 21 | 974384 | 1044771 | 957666 | **879386** | **875416** | **875395*** | - | -8.2% | -8.6% | -8.6% | - |
| 22 | 1065780 | 1045790 | 978963 | **876062** | **872850** | **872284*** | - | -10.5% | -10.8% | -10.9% | - |
| 23 | 1070520 | 1027042 | 981539 | **892064** | **887369** | **885757*** | - | -9.1% | -9.6% | -9.8% | - |
| 24 | 978491 | 1045014 | 952625 | **874739** | **872294** | **870611*** | - | -8.2% | -8.4% | -8.6% | - |
| 25 | 1029327 | 1024239 | 921784 | **887814** | **884761** | **882360*** | - | -3.7% | -4.0% | -4.3% | - |
| 26 | 961728 | 1043128 | 953756 | **871228** | **865240** | **864760*** | - | -8.7% | -9.3% | -9.3% | - |
| 27 | 1028006 | 1030753 | 971152 | **892641** | **889707** | **888957*** | - | -8.1% | -8.4% | -8.5% | - |
| 28 | 1011689 | 1032478 | 936095 | **875459** | **869403** | **868974*** | - | -6.5% | -7.1% | -7.2% | - |
| 29 | 1015741 | 1019371 | 962296 | **878833** | **875770** | **875568*** | - | -8.7% | -9.0% | -9.0% | - |
| 30 | 985496 | 1027915 | 936838 | **885333** | **884163** | **880018*** | - | -5.5% | -5.6% | -6.1% | - |
| Avg | 1001634 | 1034923 | 951638 | **881653** | **878511** | **876761*** | - | -7.3% | -7.6% | -7.8% | - |

# Appendix C
# Supplement to Chapter 5

This is the online supplement of the paper: Benders Decomposition for Production Routing under Demand Uncertainty. Section C.1 provides a numerical example of the BBC algorithm. The details of the approach to generate a core point for Pareto-optimal cuts are shown in Section C.2. Section C.3 provides the details of the SAA method. More details on the instances and the solutions can be found on the website https://sites.google.com/site/YossiriAdulyasak/publications.

## C.1 Numerical Example of the Branch-and-Benders-Cut Algorithm

We give a numerical example of the BBC algorithm, specifically for the case when the Benders subproblem contains integer variables. Note that the variable definition in this section is only used for this example and these variables are not related to other sections.

**Example**. The BBC algorithm is applied to solve the following problem:

$$
\begin{aligned}
P: \ \min\ & 6x_1 + 10x_2 + y_1 + 2y_2 \\
& -15x_1 - 22x_2 + 5y_1 + 8y_2 \leq 0 \\
& y_1 + y_2 \geq 1.5 \\
& x_1, x_2 \in \{0,1\} \\
& y_1, y_2 \in \{0,1,2\}.
\end{aligned}
$$

Given that $x_1$ and $x_2$ are fixed to $\bar{x}_1$ and $\bar{x}_2$, respectively, and the integrality constraints on the variables $y_1$ and $y_2$ are relaxed, the following *primal subproblem (PSP)* is obtained:

$$\begin{aligned} PSP: \min\ & y_1 + 2y_2 \\ & 5y_1 + 8y_2 \leq 15\bar{x}_1 + 22\bar{x}_2 \\ & y_1 + y_2 \geq 1.5 \\ & 0 \leq y_1 \leq 2 \\ & 0 \leq y_2 \leq 2. \end{aligned}$$

Denote by $\lambda_1$ to $\lambda_4$ the dual variables associated with the first to the last constraint of the primal subproblem, respectively. The *dual subproblem (DSP)* can be written as follows.

$$\begin{aligned} DSP: \max\ & -(15\bar{x}_1 + 22\bar{x}_2)\lambda_1 + 1.5\lambda_2 - 2\lambda_3 - 2\lambda_4 \\ & -5\lambda_1 + \lambda_2 - \lambda_3 \leq 1 \\ & -8\lambda_1 + \lambda_2 - \lambda_4 \leq 2 \\ & \lambda_1, \lambda_2, \lambda_3, \lambda_4 \geq 0. \end{aligned}$$

Let $\Delta$ be the polyhedron defined by the constraints of the dual subproblem. Also let $P_\Delta$ and $Q_\Delta$ be the set of extreme points and extreme rays of $\Delta$, respectively. Denote by $\eta$ an artificial variable representing the lower bound on the cost of the subproblem. The Benders master problem is as follows.

$$\begin{aligned} BP: \min\ & 6x_1 + 10x_2 + \eta \\ & -15\lambda_1 x_1 - 22\lambda_1 x_2 + 1.5\lambda_2 - 2\lambda_3 - 2\lambda_4 \leq \eta && \forall (\lambda_1, \lambda_2, \lambda_3, \lambda_4) \in P_\Delta \\ & -15\lambda_1 x_1 - 22\lambda_1 x_2 + 1.5\lambda_2 - 2\lambda_3 - 2\lambda_4 \leq 0 && \forall (\lambda_1, \lambda_2, \lambda_3, \lambda_4) \in Q_\Delta \\ & x_1, x_2 \in \{0, 1\}. \end{aligned}$$

Note that $BP$ is a relaxation of the original problem $P$. Recall that $Z^b = \{\bar{x}_1, \bar{x}_2\}$ is the set of binary variable values $x_1$ and $x_2$ and $\xi(Z^b) = 6\bar{x}_1 + 10\bar{x}_2$, $\upsilon(SP(Z^b))$ is the objective function obtained by solving the $DSP$. We also let $BP_i$ denote the relaxed Benders master problem (i.e., the integrality constraints on $x_1$ and $x_2$ variables are relaxed) at node $i$ in the branch-and-bound tree. Next we explain in detail all the steps in the BBC solution process of this problem. The illustration of the branch-and-bound tree is shown in Figure C.1.

1. The upper bound ($ub$) and the lower bound ($lb$) of the original problem $P$ are set to $\infty$ and 0, respectively. At the root node (node 0), the following problem is solved:

$$\begin{aligned} BP_0 : \min\, & 6x_1 + 10x_2 \\ & 0 \leq x_1 \leq 1 \\ & 0 \leq x_1 \leq 1. \end{aligned}$$

   The solution $x_1 = 0$, $x_2 = 0$ is obtained and it satisfies the integrality constraints. We set $Z^0 = \{0, 0\}$ and solve the DSP. Since the DSP is unbounded, we obtain the extreme ray direction $(0.2, 1, 0, 0)$ and generate a Benders feasibility cut. The $BP_0$ becomes

$$\begin{aligned} BP_0 : \min\, & 6x_1 + 10x_2 \\ & -3x_1 - 4.4x_2 + 1.5 \leq 0 \\ & 0 \leq x_1 \leq 1 \\ & 0 \leq x_1 \leq 1. \end{aligned}$$

   We solve the $BP_0$ and the solution $x_1 = 0.5$, $x_2 = 0$, and the objective function 3 is obtained. We can also set $lb = 3$ because there is only one active node. Since $x_1$ is fractional, branching is performed on this variable and node 1 ($x_1 = 0$) and node 2 ($x_1 = 1$) are created.

2. At node 1, the following problem is solved:

$$\begin{aligned} BP_1 : \min 6x_1 + 10x_2 \\ -3x_1 - 4.4x_2 + 1.5 \leq 0 \\ x_1 = 0 \\ 0 \leq x_1 \leq 1 \\ 0 \leq x_1 \leq 1. \end{aligned}$$

   The solution $x_1 = 0$, $x_2 = 0.3409$ is obtained. Therefore, node 3 ($x_2 = 1$) is created.

3. We assume that the search follows the depth-first-search scheme. Then, at node 3, the following problem is solved:

$$\begin{aligned} BP_1 : \min 6x_1 + 10x_2 \\ -3x_1 - 4.4x_2 + 1.5 \leq 0 \\ x_1 = 0 \\ x_2 = 1 \\ 0 \leq x_1 \leq 1 \\ 0 \leq x_1 \leq 1. \end{aligned}$$

   The solution $x_1 = 0$, $x_2 = 1$ is obtained. We set $Z^1 = \{0, 1\}$ and solve the DSP. The extreme point $(0, 1, 0, 0)$, the solution $y_1 = 1.5$ and $y_2 = 0$ and $\upsilon(SP(Z^1)) = 1.5$ are obtained. Since $y_1$ is fractional, we store the solution $Z^1$ together with its lower bound on the objective function value $\xi(Z^1) + \upsilon(SP(Z^1)) = 10 + 1.5 = 11.5$. To obtain an upper bound, we simply round up the value of $y_1$ to 2 and this solution is feasible. We set $ub = 12$ which is also a valid upper bound for the original problem $P$. Since both $x_1$ and $x_2$ are now fixed, there is no further branching to be performed on this node. The Benders optimality cut generated from the extreme point, i.e., $1.5 \leq \eta$, is added to the Benders master and the process continues at node 2.

4. At node 2, the following problem is solved:

$$\begin{aligned} BP_2 : \min 6x_1 + 10x_2 + \eta \\ -3x_1 - 4.4x_2 + 1.5 \leq 0 \\ 1.5 \leq \eta \\ x_1 = 1 \\ 0 \leq x_1 \leq 1 \\ 0 \leq x_1 \leq 1. \end{aligned}$$

   The solution $x_1 = 1$, $x_2 = 0$ is obtained. We set $Z^2 = \{1, 0\}$ and solve the DSP. The extreme point $(0, 1, 0, 0)$, the solution $y_1 = 1.5$ and $y_2 = 0$ and $\upsilon(SP(Z^2)) = 1.5$ are obtained, which also gives the same Benders cut $1.5 \leq \eta$. Since $y_1$ is fractional, we store the solution $Z^2$ together with its lower bound on the objective function value $\xi(Z^2) + \upsilon(SP(Z^2)) = 6 + 1.5 = 7.5$. To obtain an upper bound, we round up the value of $y_1$ to 2 and $ub = 8$. Also, since the lower bound of the solution $Z^1$, i.e., $\xi(Z^1) + \upsilon(SP(Z^1)) = 11.5 > ub$, the solution $Z^1$ can be pruned. Then, node 4 ($x_2 = 1$) is created.

5. At node 4, the following problem is solved:

$$\begin{aligned} BP_4 : \min 6x_1 + 10x_2 + \eta \\ -3x_1 - 4.4x_2 + 1.5 \leq 0 \\ 1.5 \leq \eta \\ x_1 = 1 \\ x_2 = 1 \\ 0 \leq x_1 \leq 1 \\ 0 \leq x_1 \leq 1. \end{aligned}$$

   The solution $x_1 = 1$, $x_2 = 1$ is obtained and we set $Z^3 = \{1, 1\}$. However, since $\xi(Z^3) + \upsilon(SP(Z^3)) \geq \xi(Z^3) = 16 > ub$. This solution can be pruned and the

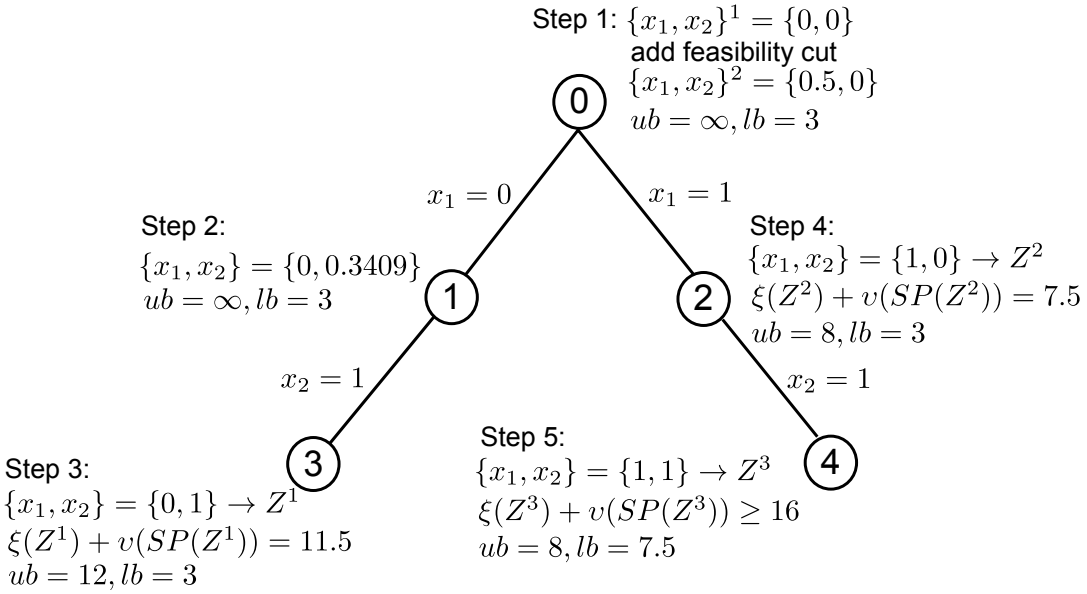



Figure C.1: Illustration of the branch-and-bound tree

process is stopped since there is no active node in the tree. We can also update the lower bound $lb = 7.5$.

6. After the process, we obtain the $ub = 8$ and also the solution $Z^2 = \{1, 0\}$. Then, the integrality constraints on $y_1$ and $y_2$ variables are imposed and the following problem is solved to optimality:

$$\begin{aligned} \min\ & y_1 + 2y_2 \\ & 5y_1 + 8y_2 \leq 15(1) + 22(0) \\ & y_1 + y_2 \geq 1.5 \\ & y_1, y_2 \in \{0, 1, 2\}. \end{aligned}$$

This gives the solution $\{y_1, y_2\} = \{2, 0\}$ and the objective function corresponding to the original problem $P$: $6(1) + 10(0) + 1(2) + 2(0) = 8$. This becomes the optimal solution.

## C.2 Approach to Determine a Core Point for the Pareto-optimal Cut

We follow a procedure similar to that of Cordeau et al. (2001) to determine a core point for the Pareto-optimal cuts. Recall that $\bar{d}_{it}$ is the nominal demand value and let $\hat{\varepsilon}$ be a positive small value. We solve the problem with one scenario $|\Omega| = 1$ associated with the nominal demand value without the routing variable to determine a core point $\mathbf{y}^0 \in ri(\mathbf{Y}^{LP})$ and $\mathbf{z}^0 \in ri(\mathbf{Z}^{LP})$. The problem is the following.

$$\max \sum_{t \in T} \varrho_t^y + \sum_{i \in N} \sum_{k \in K} \sum_{t \in T} \varrho_{ikt}^z$$

s.t. (5.2)-(5.9) and,

$$\begin{aligned}
y_t - \hat{\varepsilon}\varrho_t^y &\geq 0 && \forall t \in T \\
y_t + \hat{\varepsilon}\varrho_t^y &\leq 1 && \forall t \in T \\
z_{ikt} - \hat{\varepsilon}\varrho_{ikt}^z &\geq 0 && \forall i \in N, \forall k \in K, \forall t \in T \\
z_{ikt} + \hat{\varepsilon}\varrho_{ikt}^z &\leq 1 && \forall i \in N, \forall k \in K, \forall t \in T \\
\varrho_t^y, \varrho_{ikt}^z &\in \{0,1\} && \forall i \in N, \forall k \in K, \forall t \in T.
\end{aligned}$$

One can ensure that the core point lies in the relative interior of $\mathbf{Y}^{LP}$ and $\mathbf{Z}^{LP}$. The problem is typically easy to solve and the optimal solution can be obtained in less than one second by using CPLEX. Additionally, it is not necessary to obtain the optimal solution as a feasible solution of the problem is sufficient.

## C.3 Details on Sample Average Approximation (SAA) Method for the SPRP

We adapt the SAA procedure of Kleywegt et al. (2002b) to calculate the approximate optimality gap (SAA gap) as follows.

1. A sample size $|\Omega|$ and the number of replications $|M|$ are determined. A larger sample size $|\Omega'| \gg |\Omega|$ is also selected to compute the estimated objective value

of the optimal solution of the SAA problem. The probability of the scenario $\omega$ associated with the sample size $|\Omega|$ is equal to $\rho_\omega = 1/|\Omega|, \forall \omega \in \Omega$

2. For $s = 1 \rightarrow |M|$, do the following steps:

   (a) Generate a sample $\Omega$ and solve the problem to obtain the objective value, denoted by $\nu^s_\Omega$, and the optimal solution, denoted by $Z^s$ (which consists of the vectors $\bar{\mathbf{x}}, \bar{\mathbf{y}}$ and $\bar{\mathbf{z}}$) , of the replication $s$.

   (b) Use the solution $Z^s$ to obtain an upper bound on the optimal solution for the generated large sample size $|\Omega'|$, denoted by $\nu_{\Omega'}(Z^s)$, which can be calculated as

$$\nu_{\Omega'}(Z^s) = \sum_{t \in T} \left( f\bar{y}_t + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} \bar{x}_{ijkt} + \sum_{\omega \in \Omega'} \rho_\omega \left( up_{t\omega} + \sum_{i \in N} h_i I_{it\omega} + \sum_{i \in N_c} \sigma_i e_{it\omega} \right) \right).$$

   Denote by $\hat{Z}^s$ the solution providing the best value of the upper bound $\nu_{\Omega'}(\hat{Z}^s)$ found after $s$ replications. Let also $\hat{\mathbf{x}}, \hat{\mathbf{y}}$ and $\hat{\mathbf{z}}$ denote the vectors corresponding to the solution $\hat{Z}^s$. The variance of this estimated upper bound can be calculated as

$$\sigma^2_{\Omega'}(\hat{Z}^s) = \frac{1}{|\Omega'|(|\Omega'| - 1)} \sum_{\omega \in \Omega'} \left( G_\omega(\hat{Z}^s) - \nu_{\Omega'}(\hat{Z}^s) \right)^2$$

   where

$$G_\omega(\hat{Z}^s) = \sum_{t \in T} \left( f\hat{y}_t + \sum_{(i,j) \in E} \sum_{k \in K} c_{ij} \hat{x}_{ijkt} + up_{t\omega} + \sum_{i \in N} h_i I_{it\omega} + \sum_{i \in N_c} \sigma_i e_{it\omega} \right).$$

   (c) Compute the average of the objective values obtained in the previous step, denoted by $\hat{\nu}^s_\Omega$, and their corresponding variance as follows

$$\hat{\nu}^s_\Omega = \frac{1}{s} \sum_{i=1}^{s} \nu^i_\Omega$$

$$\sigma^2_{\hat{\nu}^s_\Omega} = \frac{1}{s(s-1)} \sum_{i=1}^{s} (\nu^i_\Omega - \hat{\nu}^s_\Omega)^2.$$

The value $\hat{\nu}^s_\Omega$ is a statistical lower bound for the optimal solution of the sample $\Omega'$.

(d) Compute the SAA gap, denoted by $\varepsilon^{SAA}(\Omega, \Omega')$ and the variance of the gap as follows.

$$\varepsilon^{SAA}(\Omega, \Omega') = \nu_{\Omega'}(\hat{Z}^s) - \hat{\nu}^s_\Omega$$
$$\sigma^2_{\varepsilon^{SAA}(\Omega,\Omega')} = \sigma^2_{\Omega'}(\hat{Z}^s) + \sigma^2_{\hat{\nu}^s_\Omega}.$$

The best value of $\varepsilon^{SAA}(\Omega, \Omega')$ and its corresponding $\sigma^2_{\varepsilon^{SAA}(\Omega,\Omega')}$ are kept track of during the process.

3. Choose the solution $\hat{Z}^s$ as the best solution.