UNIVERSITÀ DELLA CALABRIA

Department of Mathematics and Computer Science

PhD Thesis in Mathematics and Computer Science

Cycle XXXII

SSD: Mat/09

Optimizing inventory and distribution in supply chain management

Supervisor

Prof. Demetrio Laganà

Candidate

Rosario Paradiso

Coordinator

Prof. Nicola Leone

Academic Year: 2019-2020

A mia nonna Vincenzina, che mi ha sempre incoraggiato a seguire la mia strada.

Contents

List of Tables

List of Figures

Abstract

Nella presente tesi di dottorato sono studiati due problemi (e relative generalizzazioni e varianti) di particolare interesse nel campo della Logistica e della Ricerca Operativa.

Il primo problema trattato è il Multi Trip Vehicle Routing Problem with Time Windows (MTVRP). Si tratta di una generalizzazione del classico Vehicle Routing Problem (VRP) in cui ogni veicolo può essere utilizzato più volte durante l'orizzonte di pianificazione. Questa caratteristica, che complica notevolmente il problema, trova applicazione in diversi contesti di *last mile deliveries* o *urban logistics* nei quali è preferibile utilizzare vettori elettrici, con limitata capacità di trasporto. Queste caratteristiche si traducono in un numero limitato di visite o in una ridotta lunghezza (temporale o spaziale) delle rotte che questi veicoli possono percorrere, rendendo necessario il riutilizzo degli stessi veicoli nella pianificazione.

Nel Capitolo 2 viene proposto un metodo esatto, capace di risolvere cinque diverse varianti del problema che incorporano diverse caratteristiche motivate da applicazioni reali. Per ogni variante, il metodo proposto risulta esse migliore degli algoritmi esatti presenti in letteratura, in termini di tempo computazionale e numero di istanze risolte. Inoltre, per le prima volta vengono risolte istanze fino a 50 clienti.

Nel Capitolo 3, viene studiato e definito un problema affine al MTVRP, denominato Installation Planning of Offshore Wind Farm Problem (IPOWF). Il problema nasce nel contesto della costruzione di parchi eolici off-shore: dopo la fase di progettazione in cui vengono definite tutte le specifiche (numero, tipo e posizione delle turbine), il parco deve essere costruito avendo a disposizione una flotta di vascelli che vengono utilizzati per eseguire le operazioni necessarie alla costruzione di ogni turbina. L'obbiettivo è schedulare le operazioni e definire le rotte dei vascelli in modo da completare il parco, minimizzando il costo dovuto all'utilizzo dei vascelli stessi (costo di noleggio) e il costo di mancata produzione energetica derivante dalle turbine non completate. Inoltre, le condizioni meteo devono essere tenute in considerazione, dal momento che alcune operazioni possono essere eseguite solo in presenza di condizioni controllate di agitazione ondosa. La fase di definizione delle rotte dei vascelli risulta, di fatto, un MTVRP dal momento che i vascelli hanno bisogno di tornare nel porto di provenienza per caricare risorse, prima di procedere verso la turbina successiva. Il problema è stato studiato in collaborazione con Vattenfall, una delle maggiore aziende a livello mondiale operanti nel settore. Grazie a questa collaborazione è stato possibile e testare l'approccio su dati reali.

Per risolvere il problema sono stati definiti dei modelli di programmazione lineare intera (*Mixed Integer Linear Programming models*, MILPs). Alcuni di questi sono utilizzati per trovare soluzioni ammissibili in tempi ragionevoli. Le ipotesi alla base di tali modelli non incidono sull'ammissibilità delle soluzioni del problema originario. Tuttavia, per valutare la qualità delle soluzioni sono definiti dei modelli di lower bound basati su rilassamenti del problema. L'approccio è in grado di fornire soluzioni con gap di ottimalità inferiori del 2%.

I Capitoli 4 e 5 sono dedicati al secondo problema trattato in questa tesi, ovvero a una variante del classico *Inventory Routing Problem* (IRP). IRP è il problema di ottimizzazione sotteso al paradigma gestionale *Vendor Management System*, una politica di gestione di parte della catena logistica in cui il vendor ha il controllo dei livelli di inventario dei propri clienti e del relativo rifornimento, assicurando il soddisfacimento della domanda. Si tratta quindi di un problema in cui decisioni relative al livello di inventario e al trasporto vengono prese in modo integrato al fine di minimizzare i costi del sistema.

L'IRP è un problema definito su un orizzonte temporale finito e su una rete logistica caratterizzata da un nodo vendor (deposito o impianto di produzione) e da un insieme di nodi clienti che "consumano" un prodotto ad un tasso noto, e che devono essere riforniti per non andare in *stock out*. Anche il tasso di produzione dei prodotti presso il vendor è noto. La domanda presso i clienti deve essere soddisfatta utilizzando le quantità rese disponibili dal vendor che sono distribuite mediante una flotta di veicoli

capacitati, posizionati presso il vendor stesso. Inoltre, sia clienti che il vendor possono accumulare scorte di inventario senza eccedere una capacità massima. L'obbiettivo è quello di stabilire le rotte dei veicoli e le quantità da trasportare presso i clienti durante il periodo di pianificazione, garantendo il soddisfacimento della domanda per ogni periodo e minimizzando il costo di trasporto e il costo di inventario totale (presso i clienti e presso il vendor).

In questa tesi, viene studiata una versione di IRP in cui la pianificazione periodica è fatta su un intervallo di tempo "illimitato". Questo tipo di problema viene generalmente identificato come il *Cyclic Inventory Routing Problem* (CIRP). La pianificazione periodica è tipica nei sistemi nei produzione in cui si cerca di standardizzare le operazioni. Sistemi logistici che sposano politiche manageriali di *lean production* ne sono un esempio.

Nel Capitolo 4 viene studiata una variante del CIRP. Per questo problema viene proposto un algoritmo esatto di branch-and-cut capace di risolvere istanze fino a 50 clienti.

Nel Capitolo 5 vengono proposte delle classi di math-euristiche (matheuristic) per risolvere il CIRP mediante la risoluzione di una formulazione basata su variabili di rotta. Le rotte sono definite a priori risolvendo dei VRPs. Per le rotte generate sono definiti dei rapporti di prestazione. L'approccio proposto viene comparato con le soluzioni e i lower bounds forniti da un algoritmo esatto di branch-and-cut che risolve una formulazione basata su variabili di flusso. I risultati mostrano che l'approccio basato sulle classi di math-eurisitche è in grado di fornire soluzioni di alta qualità, molto spesso migliori di quelle fornite dall'algoritmo di branch-and-cut, con tempi computazoinali contenuti.

Chapter 1

Introduction

1.1 On Logistic and Supply Chain decision-making problems

Logistics can be considered as the backbone of processes and activities in companies and organizations operating in different fields. According to Christopher (2016), a proper definition of the term is the following:

> *Logistics is the process of strategically managing the procurement, movement and storage of materials, parts and finished inventory (and the related information flows) through the organisation and its marketing channels in such a way that current and future profitability are maximised through the cost-effective fulfilment of orders.*

From this definitions it follows that the mission of logistics is to serve customers (in a broad sense), ensuring a certain service level, in the most cost-effective way. The spectrum of activities is large and involves different actors, facilities and transportation services.

Facilities are all the sites in which materials are processed (stored, manufactured, sorted, combined, sold, consumed), like warehouses, manufacturing and assembly centres, distribution centres, transshipment points, transportation terminals (Ghiani et al. (2013)).

Transportation services are related to flows of materials among different facilities.

Goods can be moved by using different transportation modes (ship, rail, truck, air and pipeline) that can also be combined in several ways. All these components are part of a logistics network, defined by a set of interconnected facilities.

The concept of *Supply Chain*, arises in the context of complex logistics networks and it is defined in Christopher (2016) as:

> *A network of connected and interdependent organisations mutually and co-operatively working together to control, manage and improve the flow of materials and information from suppliers to end users.*

The discipline that deals with decision-making activities related to the supply chain, is named *supply chain management*. The concept is relatively new and the definition provided by Christopher (2016) is the following:

The management of upstream and downstream relationships with suppliers and customers in order to deliver superior customer value at less cost to the supply chain as a whole.

In this field, different types of activities like purchasing, inventory control, production, sales and distribution must be coordinated (see Andersson et al. (2010), Stadtler (2015), Christopher (2016), Simchi-Levi et al. (2008)).

The following of this dissertation, is focused on optimization problems related to distribution and inventory activities occurring in the supply chain management.

One of the most studied and well known problem in the field of Operations Research that is related to the transportation and distribution of goods is the *Vehicle Routing Problem* (VRP). The problem states as follows: several customer locations request a service that can be performed by a fleet of capacitated vehicles starting from a single depot. A routing cost is defined for traveling from a customer location to another one. The problem consists in designing vehicle routes such that each customer is served by a single vehicle, the vehicles capacity is not violated in any routes and the total routing cost is minimized. Since the VRP has been introduced, many challenging variants of the problem have been defined, considering new features and side constraints to increase the degree of realism. These problems are generally know as *rich vehicle routing problems*. A feature that has gained more attention because of its relevance in real life applications, is the possibility of using each vehicle of the fleet more than once within the planning horizon. Vehicle routing problems in which each vehicle can perform more than one trip are known as *Multi Trip Vehicle Routing Problems* (MTVRPs). The problem was introduced for the first time by Fleischmann (1990) and since then, different solution approaches and richer variants of the problem have been proposed.

The VRP belongs to the class of operational problems, in which, a decision-maker plans the distribution considering the customers demand and the set of customers to serve as given input data. This is the case when decisions related to different facilities are taken independently. However logistics cost can be reduced by taking decisions related to different processes of the supply chain in an integrated manner (Andersson et al. (2010)).

An example of integrated decision-making policy is the vendor-managed inventory (VMI) (Chopra & Meindl (2007)). This business practice aims at reducing logistics cost by integrating inventory and transportation decisions. The vendor is responsible for all the decisions regarding customers inventory level. In this way, the vendor is free to plan the distribution, ensuring that customers are never out of stock. This policy is beneficial for vendors and for customers: vendors can save on transportation costs because shipments to different customers can be coordinated with inventory decisions and customers can outsource inventory control activities.

Inspired by the VMI policy, the *Inventory Routing Problem* (IRP) has been formalized and studied for the first time by Bell et al. (1983). However, the the first paper that used the term "Inventory Routing Problem" is the one of Golden (1984). Since then, the IRP has attracted a lot of attention and many versions of the problem have been intensively studied. The general version of the problem can be defined as follows. A single facility must supply a set of customers with a single commodity over a given planning horizon. Each customer consumes the product at a known rate and local inventory, that does not exceed a storage capacity, can be maintained at each customer by paying a unit inventory cost per period. A fleet of capacitated homogeneous vehicles is available for the distribution. The problem consists in minimizing

the distribution cost and total inventory cost during the planning period avoiding stock-out at any customer. We remind to Coelho et al. (2013*b*) for a complete overview on the problem. A version of the IRP that has not been extensively studied is the *Cyclic Inventory Routing Problem* (CIRP). The CIRP is a variant of the IRP in which the plan must be cyclically applied over an infinite planning horizon. This type of policy finds applications in industrial systems characterized by lean production principles as standardization of the workload and level production planning.

The purpose of the present dissertation is to study variants and extensions of two of the aforementioned classes of problems: the *Multi Trip Vehicle Routing Problem* and the *Cyclic Inventory Routing Problem*.

1.2 Research Motivations

The MTVRP and the CIRP are challenging classes of problems (both $\mathcal{NP}$-$hard$, for reduction to the VRP) with a wide range of real life applications. The MTVRP finds applications in last mile delivery and city logistics contexts. As pointed out in Cattaruzza et al. (2017), carriers and cities have strong incentives for generalizing the use of small environmentally friendly vehicles in urban areas. These vehicles are usually characterized by a limited capacity or a limited autonomy. Moreover, factors as the development of the e-commerce, higher service levels required by customers, entail the tendency to introduce intermediate facilities. The effect is a greater interest in vehicle routing problems in which multiple trips are allowed: in fact, carriers are interested in reducing the size of their fleet while increasing the service level offered to customers. Considering that the in 2007, 85 % of the European GDP was generated in urban areas and that the 25 % of the CO_2 emission of the whole transport sector comes from urban transport, it is easy to understand the importance of last mile and urban logistics (Cattaruzza et al. (2017)), making this class of problem worthy of further studies.

For what concerns the CIRP, real life applications can be founded in several industrial sectors. The cyclic aspect is related to the lean philosophy introduced by the Toyota Production System, in particular to the notion of *heijunka* (workload leveling). Ideally, a lean production system consists in a one-piece flow with minimal transportation and inventory costs (Ohlmann et al. (2008)). However, this cannot be the case when facilities are geographically dispersed. Applying a cyclic plan over an infinite planning horizon aims at facilitating *heijunka*, by fixing the frequency of visits and by keeping a levelled inventory level at each facility. Generally, frequent visits lead to smooth flow of materials with low inventory levels but with increasing transportation costs. On the other hands, infrequent deliveries with higher loads increase transportation costs but reducing inventory costs. Solving the CIRP aims at defining a cyclic plan to achieve the optimal trade-off.

1.3 Contents of the Thesis

In this thesis we deal with the following variants and extensions of the MTVRP and of the CIRP.

- The Capacitated Multi Trip Vehicle Routing Problem with Time Windows.
- The Installation Planning of Offshore Wind Farms Problem.
- The Cyclic Inbound Inventory Routing Problem.
- The Cyclic Inventory Routing Problem with Split Deliveries.

1.3.1 The Capacitated Multi Trip Vehicle Routing Problem with Time Windows

Despite different versions of MTVRPs have been introduced, it is possible to identify a common underlying problem, the *Capacitated Multi Trip Vehicle Routing Problem with Time Windows*, which consists in a MTVRP in which (a) each vehicle has a limited capacity and (b) each customer must be served within a given time window.

However, in the literature, also additional side constrains are considered, as loading times to reload vehicles at the depot (Hernandez et al. (2014, 2016)), limited trip duration (Hernandez et al. (2014)), release dates on the availability of goods to deliver (Cattaruzza et al. (2016*a*)) and drone battery capacity with load-dependent consumption (Cheng et al. (2018)). These four side constraints give rise to four different problems, namely, the *CMTVRPTW with Loading Times* (CMTVRPTW-LT), the *CMTVRPTW with Limited Trip Duration* (CMTVRPTW-LD), the *CMTVRPTW with Release Dates* (CMTVRPTW-R) and the *Drone Routing Problem* (DRP), respectively.

In Chapter 2 we present a new, standalone, exact framework to solve all the four variants of the problem proposed in the literature. The exact algorithm is based on a novel formulation characterized by exponential number of variables and constraints and relies on column generation, column enumeration and cutting plane. A computational study shows how the method outperforms the state-of-the-art algorithms present in the literature, solving instances up to 50 customers.

1.3.2 The Installation Planning of Offshore Wind Farms Problem

The growing of energy demand and challenges posed by pollution and global warming motivate an increasing interest in the field of the green sources energy. In this context, the utilization of offshore wind farms is going to increase in the coming years.

Increasing the efficiency of offshore wind farms does not only call for technical improvements in terms of turbines components, but also for improvements regarding the logistic activities supporting all the life-cycle activities of an offshore wind park, since the complexity of these processes represents the main drawback of this technology. Typically, these activities can be grouped in three main processes: design, installation and maintenance.

In Chapter 3 an optimization problem related to the installation phase of offshore wind parks is studied and introduced. The problem, defined as the Installation Planning of Offshore Wind Farms Problem (IPOWF), consists of scheduling the activities to build the farm, according to the specifications defined during the design phase, while minimizing the construction costs. The activities are carried out by using a fleet of different type of vessels. Each type of vessel is able to perform only a certain type of activity. Moreover, the routes of the vessels can be characterized by multiple visits to the port (where vessels are located at the beginning and where they need to return at the end of the operations) to load resources to construct the assigned turbines. Therefore, the underling routing problem is an extension of the MTVRP. Another degree of complexity is added by weather requirements. Each activity can be completed only under specific weather conditions. This results in a different amount of available working time for carrying out operations during the planning horizon, according to the weather forecast for that period.

This problem is currently faced by Vattenfall, one of the major companies operating in the field, that has contributed to this study.

In Chapter 3 we formulate the problem as a Mixed Integer Linear Programming (MILP) and then we derive other MILPs that can provide lower and upper bounds for the problem on real life instances.

1.3.3 The Cyclic Inbound Inventory Routing Problem

The Cyclic Inbound Inventory Routing Problem is a problem motivated by lean production policies applied to component collection rather than more common perspective of end product distribution. The problem considers a single manufacturing plant and a set of geographically dispersed suppliers. The aim is to find the minimal-cost inbound logistic plan that collects inventory components from a set of suppliers to supply manufacturing at the plant.

To incorporate level production planning (*heijunka*), beside the cyclic planning, an *inventory clearing policy* is enforced, such that, if a supplier is visited in a period, the collected quantity must correspond to the entire inventory amount, so that the inventory at the supplier is zero after the pick-up.

In Chapter 4 the problem is formally introduced and an exact branch-and-cut algorithm is designed to solve the problem.

1.3.4 The Cyclic Inventory Routing Problem with Split Deliveries

The Cyclic Inventory Routing Problem with Split Deliveries is a problem in which a supplier needs to supply a set of geographical dispersed customers over an infinite planning horizon cyclically applying the distribution plan.

In Chapter 5, we propose a worst-case analysis for the problem with respect to the solutions that can be obtained by using set of routes generated by solving VRPs.

Inspired by the worst-case analysis, a class of matheuristics is designed. The solutions provided by the proposed approach are benchmarked with upper and lower bounds obtained by solving a flow based formulation with a branch-and-cut algorithm. The proposed approach is able to obtain high quality solutions by considering a relatively small set of routes embedded in a route based formulation.

Chapter 2

An Exact Solution Framework for Multi-Trip Vehicle Routing Problems with Time Windows*

Abstract

Multi-Trip Vehicle Routing Problems (MTVRP) generalize the well-known VRP by allowing vehicles to perform multiple trips per day. MTVRPs have received a lot of attention lately because of their relevance in real-life applications, e.g., in city logistics and last-mile delivery. Several variants of the MTVRP have been investigated in the literature, and a number of exact methods have been proposed. Nevertheless, the computational results currently available suggest that MTVRPs with different side-constraints require ad-hoc formulations and solution methods to be solved. Moreover, solving instances with just 25 customers can be out of reach for such solution methods. In this paper, we proposed an exact solution framework to address four different MTVRPs proposed in the literature. The exact solution framework is based on a novel formulation that has an exponential number of variables and constraints. It relies on column generation, column enumeration, and cutting plane. We show that this solution framework can solve instances with up to 50 customers of four MTVRP variants and outperforms the state-of-the-art methods from the literature.

***Key words**: multi-trip vehicle routing, time windows, column generation, exact methods, dynamic programming*

2.1 Introduction

Most of the literature on the *Vehicle Routing Problem* (VRP) addresses problems where each vehicle is limited to perform at most one trip per day. The first attempt to investigate VRPs where vehicles are allowed to perform multiple trips dates back to Fleischmann (1990). Since then, many contributions on *Multi-Trip Vehicle Routing Problems* (MTVRP) have been published, especially in the last decade, as observed in the recent survey of Cattaruzza et al. (2016*b*). Such an increasing interest in MTVRPs is due, for example, to the need of new practices in city logistics and last-mile delivery. The demand of limiting noise and pollution in city centers requires the usage of small vans, electric vehicles, and/or unmanned

*This chapter is based on Paradiso et al. (2019).

aerial vehicles (UAVs, commonly known as drones) and forbids heavy large trucks from entering city centers. The limited capacity and autonomy of these small vehicles force them to perform multiple trips and to return to the depot to reload multiple times over the day.

In the literature, MTVRPs with different features are addressed in different papers, and a wide range of solution methods (both exact and heuristic) have been proposed. Nevertheless, we can identify a common underlying problem, the *Capacitated MTVRP with Time Windows* (CMTVRPTW), that is a special case of the problems investigated in many papers, such as Hernandez et al. (2014, 2016), Cattaruzza et al. (2016*a*), Cheng et al. (2018). The features of this CMTVRPTW are the following: (a) the goal is to minimize the routing costs, (b) all customers must be served, (c) multiple homogeneous vehicles are available, (d) vehicles are capacitated, and (e) time window constraints are imposed on the customer visits. Yet different papers consider additional side constraints on top of the CMTVRPTW, such as loading times to reload the vehicles at the depot (Hernandez et al. (2014, 2016)), limited trip duration (Hernandez et al. (2014)), release date on the availability of the goods to deliver (Cattaruzza et al. (2016*a*)), drone battery capacity and load-dependent battery consumption (Cheng et al. (2018)). These four side constraints give rise to four different problems, namely, the *CMTVRPTW with Loading Times* (CMTVRPTW-LT), the *CMTVRPTW with Limited Trip Duration* (CMTVRPTW-LD), the *CMTVRPTW with Release Dates* (CMTVRPTW-R), and the *Drone Routing Problem* (DRP), respectively. For the sake of brevity, in the following, we will refer to these four problems as "the four variants of the CMTVRPTW".

The exact methods currently available for these four variants of the CMTVRPTW are based on several different mathematical formulations, which makes it unclear which formulation is most suitable to solve a CMTVRPTW. Moreover, in spite of the effort devoted to develop exact solution methods for CMTVRPTWs, the literature indicates that small instances with 25 customers cannot be consistently solved, and it could take a few hours of computing time to solve them.

With this paper, we aim at closing part of this research gap. We propose an *Exact Solution Framework* (hereafter referred to as ESF) based on a novel mathematical model that can solve medium-size instances with up to 50 customers of the four variants of the CMTVRPTW, significantly outperforming the state-of-the-art exact methods tailored for the single variants. We describe the ESF by focusing on the CMTVRPTW, and we show later how it can be tailored to solve the four variants by simply adapting one of its steps. The main contributions of this paper are the following: (a) we propose a novel mathematical model with an exponential number of variables and constraints that is valid for the CMTVRPTW and its four variants; (b) we describe two relaxations of this mathematical model that provide good lower bounds and can be efficiently computed; (c) we describe a seven-step ESF for the CMTVRPTW based on these two lower bounds and on a branch-and-cut algorithm, with an embedded branch-and-price to separate violated inequalities, to close the gap; (d) we illustrate how the ESF can be applied to solve each of the four variants of the CMTVRPTW by simply specializing one of the seven steps; (e) we provide a computational proof that the ESF can solve instances with up to 50 customers and outperforms the state-of-the-art exact methods for the four individual variants of the CMTVRPTW.

The paper is organized as follows. Section 2.2 reviews the main contributions from the literature on exact methods for MTVRP with Time Windows. Section 2.3 formally introduces the CMTVRPTW and presents two column-generation-based formulations from the literature. Section 2.4 describes the novel formulation for the CMTVRPTW and its variants. Section 2.5 presents two lower bounds derived from the novel formulation. Section 2.6 provides an outline of the ESF and describes how to apply it to solve

the CMTVRPTW. Section 2.7 describes the steps of the ESF in detail. Section 2.8 illustrates how the ESF can be tailored to solve each of the four variants of the CMTVRPTW. A computational analysis to show the effectiveness of the ESF and a comparison of its performance with the literature is provided in Section 2.9. Finally, conclusions are drawn in Section 2.10.

2.2 Literature Review

This section reviews the main exact methods proposed for MTVRPs with time windows. For the sake of brevity, we omit contributions on exact methods for MTVRP without time windows and on heuristic methods. For a recent overview of the literature on these two topics, the reader is referred to Cattaruzza et al. (2016*b*).

In the remainder of the paper, we refer to a *structure* as a sequence of customers that can be visited consecutively by a vehicle between two visits at the depot, such that capacity constraints (and possibly other side constraints) are fulfilled and a departure time from the depot can be scheduled in such a way that all time windows are satisfied. Moreover, following the convention of Cattaruzza et al. (2016*b*), we refer to a *trip* as a structure with a fixed departure time from the depot, and we refer to a *journey* as a sequence of non-overlapping trips assigned to the same vehicle. It is worth mentioning that, as stated in Cattaruzza et al. (2016*b*), different authors use different terms to refer to trips and journeys.

Azi et al. (2007) study a single-vehicle variant of the CMTVRPTW where it is not mandatory to serve all customers and the objective function is first to maximize the number of customers served and second to minimize the routing cost. Moreover, a limited trip duration constraint (called deadline constraint) on the maximum time that the goods can stay on board before they are delivered at customers and a setup time to load the vehicle are considered. They propose a two-phase exact algorithm. In the first phase, all feasible non-dominated trips are generated by complete enumeration, which is possible when time windows and limited trip duration constraints are tight. In the second phase, feasible routes are generated by combining the trips generated in the first phase. The algorithm is tested on a set of instances adapted from the well-known Solomon instances for the VRPTW (Solomon (1987)) with up to 100 customers. The results show that the algorithm is very sensitive to the limited trip duration constraint. Indeed, if this constraint is not tight, it is impossible to enumerate all feasible trips in the first phase.

Azi et al. (2010) investigate the multi-vehicle version of the problem considered in Azi et al. (2007). The authors propose a branch-and-price algorithm based on a set packing formulation where each column represents a feasible journey. The pricing problem corresponds to an *Elementary Shortest Path Problem with Resource Constraints* (ESPPRC). The algorithm is tested on instances with 25 and 40 customers derived from the Solomon VRPTW instances. The results show that instances with 25 customers can be routinely solved, but the complexity of the problem strongly depends on the tightness of limited trip duration constraints.

Macedo et al. (2011) study the same problem of Azi et al. (2010). They propose an iterative two-phase algorithm. In the first phase, all feasible trips are generated. The second phase corresponds to a pseudo-polynomial network flow model where nodes represent time instants and arcs represent feasible vehicle trips and is solved using a commercial solver. To create the network, a time discretization is needed. The model is iteratively solved until the current discretization provides a feasible solution. Computational results show that the proposed algorithm outperforms the two-phase method of Azi et al. (2010).

Hernandez et al. (2014) address the CMTVRPTW-LD, which differs from the problem considered in Azi et al. (2010) and Macedo et al. (2011) in that all customers must be served and the objective function aims at minimizing the total routing cost. They propose a two-phase exact algorithm. In the first phase, all feasible structures are enumerated by considering resource constraints. The second phase is a branch-and-price algorithm based on a set covering formulation with side constraints, where columns represent trips and side constraints guarantee that each vehicle is assigned non-overlapping trips. The pricing problem can be solved in pseudo-polynomial time. The algorithm is tested on instances with 25 and 40 customers as in Azi et al. (2010) and Macedo et al. (2011). The computational results show that the algorithm of Hernandez et al. (2014) performs, on average, better than the method of Azi et al. (2010) and, on some instances, better than the method of Macedo et al. (2011).

The problem studied in Hernandez et al. (2016) differs from the problem of Hernandez et al. (2014) in that limited trip duration is not considered. Two branch-and-price exact algorithms are proposed both based on a set covering formulation. In the first formulation, each column represents a journey, and the pricing problem is an ESPPRC, which can be solved by dynamic programming. In the second formulation, each column represents a trip, and additional side constraints guarantee the feasibility of the solutions; the pricing problem can again be solved by dynamic programming, where the concepts of group of labels and representative labels are exploited. Computational results on 25-customer instances show that the second branch-and-price, based on the trip-formulation, performs better than the first.

Recent contributions in the literature investigate the usage of truck-drone tandem systems and drone-only systems to deliver parcels. In particular, drones can perform multiple trips to deliver parcels by flying from/to trucks and depots where they can recharge their battery and pick up packages to deliver. The first significant contribution devoted to exact methods for drone routing problems is owed to Cheng et al. (2018). They propose two mathematical formulations to solve the (multi-trip) drone routing problem. The objective function also takes energy consumption into account into the routing cost. The first formulation is based on drone-index variables, whereas the second formulation does not use such drone index. Both formulations are strengthened by valid inequalities. The authors develop two branch-and-cut algorithms that are able to solve new benchmark instances with up to 50 customers. Their computational study shows that the formulation without drone index outperforms the one with drone index in terms of number of instances solved to optimality and computing time.

2.3 Description of the CMTVRPTW and Formulations from the Literature

In this section, we formally introduce the CMTVRPTW, which is used in the following sections to describe the ESF, and describe the two column-generation-based formulations from the literature.

The CMTVRPTW can be represented on a directed graph $G = (V, A)$. The vertex set $V = \{0\} \cup N$ consists of $n + 1$ vertexes, where 0 represents the depot and $N = \{1, 2, \ldots, n\}$ represents a set of n customers to serve. For each customer $i \in N$, the demand q_i, the service time st_i, and a (hard) time window $[a_i, b_i]$ are given. A time window $[a_0, b_0]$ is associated with the depot. For the depot, we assume that $q_0 = 0$ and $st_0 = 0$. The arc set A is defined as $A = \{(i, j) \,|\, i, j \in V \,:\, a_i + t_{ij} + st_i \leq b_j\}$, where t_{ij} is the travel time from vertex i to vertex j - without loss of generality, we assume that $t_{ij} \leq t_{ik} + st_k + t_{kj}$, for all $i, j, k \in V$ such that $i \neq j \neq k$. A travel cost c_{ij} is also associated with each arc $(i, j) \in A$.

Customers can be served by using a fleet of m vehicles, each one of capacity $\overline{q}$, that are located at the depot.

A trip h is represented as $h = (0, i_1, i_2, \ldots, i_{\mu_h}, 0)$, where μ_h is the number of visited customers. A journey r is represented as a sequence of non-overlapping trips $r = (0, i_1, i_2, \ldots, i_{\mu_{r_1}}, 0, i_1, i_2, \ldots, i_{\mu_{r_2}}, 0, \ldots, 0)$.

The goal of the CMTVRPTW is to find a set of at most m journeys of minimum total cost such that each customer is visited exactly once.

In the literature, two formulations with an exponential number of variables have been proposed: a trip-based formulation (see Hernandez et al. (2016)) and a journey-based formulation (see Hernandez et al. (2014, 2016)).

Let $\mathcal{H}$ be the set of all feasible trips, and let c_h be the cost of trip $h \in \mathcal{H}$ given by the sum of the traversed arcs. A trip $h \in \mathcal{H}$ is described by coefficients α_{ih} indicating the number of times trip h visits customer $i \in N$ and by binary coefficients τ_{th} that are equal to 1 if trip h is *active* at time $t \in [a_0, b_0]$, where *active* means that the vehicle performing trip h is either traveling between two vertices or serving/waiting at a customer. Let $x_h \in \{0, 1\}$ be a binary variable equal to 1 if trip $h \in \mathcal{H}$ is selected (0 otherwise). The trip-based formulation, $\mathrm{F}_{\mathcal{H}}$, is the following

$$z(\mathrm{F}_{\mathcal{H}}) = \min \sum_{h \in \mathcal{H}} c_h x_h \tag{2.1a}$$

$$\text{s.t.} \sum_{h \in \mathcal{H}} \alpha_{ih} x_h = 1 \qquad i \in N \tag{2.1b}$$

$$\sum_{h \in \mathcal{H}} \tau_{th} x_h \leq m \quad t \in [a_0, b_0] \tag{2.1c}$$

$$x_h \in \{0, 1\} \qquad h \in \mathcal{H} \tag{2.1d}$$

The objective function (2.1a) aims at minimizing the cost of the selected trips. Constraints (2.1b) ensure that each customer is visited exactly once. Constraints (2.1c) guarantee that at most m trips are active at any point in time. Constraints (2.1d) are integrality constraints.

Let $\mathcal{R}$ be the set of all feasible journeys. Moreover, let c_r be the cost of journey $r \in \mathcal{R}$ given by the sum of the costs of its individual trips, and let α_{ir} be a coefficient indicating the number of times journey $r \in \mathcal{R}$ visits customer $i \in N$. Let $x_r \in \{0, 1\}$ be a binary variable equal to 1 if journey $r \in \mathcal{R}$ is selected (0 otherwise). The journey-based formulation, $\mathrm{F}_{\mathcal{R}}$, is the following

$$z(\mathrm{F}_{\mathcal{R}}) = \min \sum_{r \in \mathcal{R}} c_r x_r \tag{2.2a}$$

$$\text{s.t.} \sum_{r \in \mathcal{R}} \alpha_{ir} x_r = 1 \quad i \in N \tag{2.2b}$$

$$\sum_{r \in \mathcal{R}} x_r \leq m \tag{2.2c}$$

$$x_r \in \{0, 1\} \qquad r \in \mathcal{R} \tag{2.2d}$$

The objective function (2.2a) aims at minimizing the cost of the selected journeys. Constraints (2.2b) ensure that each customer is visited exactly once. Constraint (2.2c) guarantees that at most m journeys are selected. Constraints (2.2d) are integrality constraints.

Both formulations $F_{\mathcal{H}}$ and $F_{\mathcal{R}}$ have an exponential number of variables that cannot be enumerated a-priori even for small instances. Therefore, both formulations can be used to solve the CMTVRPTW only if a column generation framework is applied. Earlier contributions from the literature (see, e.g., Azi et al. (2010), Hernandez et al. (2014, 2016)) show that the lower bound provided by the linear relaxation of both formulations is of high-quality. Nevertheless, solving the pricing problem of both formulations is particularly challenging, so it could take hours of computing time to find an optimal solution even for instances with only 25 customers.

2.4 The Novel Structure-Based Formulation

Due to the computational complexity of using formulations $F_{\mathcal{H}}$ and $F_{\mathcal{R}}$, we introduce a novel structure-based formulation that has an exponential number of variables and constraints, but involves much fewer variables than both $F_{\mathcal{H}}$ and $F_{\mathcal{R}}$.

Let a *structure* $s = (0, i_1, i_2, \ldots, i_{\mu_s}, 0)$ be an ordered set of μ_s customers that can be visited by a vehicle in between two visits at the depot and can start from the depot within a time interval $[e_s, \ell_s]$, such that: (i) capacity constraints are satisfied, (ii) the duration d_s and the cost c_s are constant for each departure time from the depot within $[e_s, \ell_s]$, and (iii) the duration d_s is the minimum duration to serve the set of customers in the given order. In comparison with a trip that is defined on the basis of a given departure time from the depot in the interval $[e_s, \ell_s]$, a structure represents the family of all the trips visiting the same sequence of customers between two visits at the depot, satisfying all the constraints, and having the same minimum time duration d_s. Each trip of this family has a different departure time from the depot within $[e_s, \ell_s]$. Let $\mathcal{S}$ be the set of all feasible structures, and let α_{is} be a coefficient equal to the number of times customer $i \in N$ is served by structure $s \in \mathcal{S}$. Let $x_s \in \{0, 1\}$ be a binary variable equal to 1 if structure $s \in \mathcal{S}$ is selected (0 otherwise). The structure-based formulation, $F_{\mathcal{S}}$, is the following

$$z(F_{\mathcal{S}}) = \min \sum_{s \in \mathcal{S}} c_s x_s \tag{2.3a}$$

$$\text{s.t.} \sum_{s \in \mathcal{S}} \alpha_{is} x_s = 1 \qquad i \in N \tag{2.3b}$$

$$\sum_{s \in \widehat{\mathcal{S}}} x_s \leq \eta_m(\widehat{\mathcal{S}}) \quad \widehat{\mathcal{S}} \subseteq \mathcal{S} \tag{2.3c}$$

$$x_s \in \{0, 1\} \qquad s \in \mathcal{S} \tag{2.3d}$$

where $\eta_m(\widehat{\mathcal{S}})$ is the maximum number of structures of the set $\widehat{\mathcal{S}}$ that can be simultaneously in a solution given the number of vehicles m.

The objective function (2.3a) aims at minimizing the cost of the selected structures. Constraints (2.3b) ensure that each customer is visited exactly once. Constraints (2.3c) (hereafter called *Structure Feasibility Constraints*, SFC) guarantee that the set of selected structures can be performed by the m vehicles. Constraints (2.3d) are integrality constraints.

Notice that the number of variables of $F_{\mathcal{S}}$ can be significantly lower than the number of variables of $F_{\mathcal{H}}$ and, in turn, of $F_{\mathcal{R}}$. Indeed, formulation $F_{\mathcal{H}}$ has a binary variable for each structure $s \in \mathcal{S}$ and each instant of time $t \in [e_s, \ell_s]$. Moreover, each feasible solution of $F_{\mathcal{S}}$ can correspond to multiple (possibly an infinite number of) solutions of formulation $F_{\mathcal{H}}$. Indeed, given the set of structures of a feasible solution

of $F_{\mathcal{S}}$, it is possible to obtain a feasible solution of $F_{\mathcal{H}}$ by fixing the departure time from the depot of each of these structures within the corresponding time interval $[e_s, \ell_s]$.

2.5 Lower Bounds Based on the Structure-Based Formulation

In this section, we describe two lower bounds derived from formulation $F_{\mathcal{S}}$ that are used by the ESF and are described in the following sections. The first lower bound is based on a continuous relaxation of $F_{\mathcal{S}}$ that ignores SFC and therefore provides a lower bound for the CVRPTW without the multi-trip feature. The second lower bound is based on a relaxed version of SFC that can be enumerated by inspection.

2.5.1 First Lower Bound: CVRPTW Lower Bound

The first lower bound, which we call CVRPTW-LB in the following, corresponds to the optimal solution of the linear relaxation of formulation $F_{\mathcal{S}}$ without SFC, i.e., it corresponds to the optimal value $z(P1_{\mathcal{S}})$ of the following problem $P1_{\mathcal{S}}$

$$z(P1_{\mathcal{S}}) = \min \sum_{s \in \mathcal{S}} c_s x_s \tag{2.4a}$$

$$\text{s.t.} \sum_{s \in \mathcal{S}} \alpha_{is} x_s = 1 \quad i \in N \tag{2.4b}$$

$$x_s \in \mathbb{R}_+ \qquad s \in \mathcal{S} \tag{2.4c}$$

As $P1_{\mathcal{S}}$ is the continuous relaxation of a set partitioning problem that does not consider SFC, it provides a lower bound to a VRP with capacity and time window constraints.

2.5.2 Second Lower Bound: Relaxed SFC Lower Bound

The second lower bound, which we call RSFC-LB in the following, is obtained from $F_{\mathcal{S}}$ by replacing SFC with a relaxed version and by adding a set of valid inequalities as follows.

Let $\tau_{hs} \in \{0, 1\}$ be a binary coefficient defined for each structure $s \in \mathcal{S}$ and each instant of time $h \in [a_0, b_0]$ as follows

$$\tau_{hs} = \begin{cases} 1 & \text{if } \ell_s < h < e_s + d_s \\ 0 & \text{otherwise} \end{cases}$$

that is, τ_{hs} is equal to 1 if structure $s \in \mathcal{S}$ is active at time h for any possible departure time from the depot. A relaxed version of SFC (hereafter referred to as *Relaxed SFC*, or in short RSFC) is therefore given by

$$\text{(RSFC)} \qquad \sum_{s \in \mathcal{S}} \tau_{hs} x_s \leq m \quad h \in [a_0, b_0] \tag{2.5}$$

which indicate that, at any point in time, at most m structures (and thus vehicles) can be active.

To better explain constraints (2.5), let us consider two structures $s_1, s_2 \in \mathcal{S}$ with the following features: $[e_{s_1}, \ell_{s_1}] = [10, 20]$, $d_{s_1} = 15$, $[e_{s_2}, \ell_{s_2}] = [18, 22]$, $d_{s_2} = 10$. The only coefficients τ_{hs} equal to 1, for structure s_1 are τ_{21,s_1}, τ_{22,s_1}, τ_{23,s_1}, τ_{24,s_1}, and, for structure s_2 τ_{23,s_2}, τ_{24,s_2}, τ_{25,s_2}, τ_{26,s_2}, τ_{27,s_2}.

Therefore, of all constraints (2.5) with $h \in [21, 27]$, the tightest constraints are those with $h = 23, 24$ that are $x_{s_1} + x_{s_2} \leq m$.

RSFC-LB also uses a subset of the well-known *Subset-Row* (SR) inequalities, introduced by Jepsen et al. (2008) for the VRPTW. In particular, for each triplet of customers $\{i, j, k\} \in N$ and for each structure $s \in \mathcal{S}$, let β_{ijks} be a binary coefficient defined as

$$\beta_{ijks} = \begin{cases} 1 & \text{if } \alpha_{is} + \alpha_{js} + \alpha_{ks} \geq 2 \\ 0 & \text{otherwise} \end{cases}$$

which is, β_{ijks} is equal to 1 if at least two of the customers of the set $\{i, j, k\}$ are visited by structure s (0 otherwise). SR inequalities are defined as

$$\sum_{s \in \mathcal{S}} \beta_{ijks} x_s \leq 1 \quad \{i, j, k\} \in N \tag{2.6}$$

RSFC-LB corresponds to the optimal value $z(\text{P2}_{\mathcal{S}})$ of the following problem $\text{P2}_{\mathcal{S}}$

$$z(\text{P2}_{\mathcal{S}}) = \min \sum_{s \in \mathcal{S}} c_s x_s \tag{2.7a}$$

$$\text{s.t.} \sum_{s \in \mathcal{S}} \alpha_{is} x_s = 1 \qquad i \in N \tag{2.7b}$$

$$\sum_{s \in \mathcal{S}} \tau_{hs} x_s \leq m \qquad h \in [a_0, b_0] \tag{2.7c}$$

$$\sum_{s \in \mathcal{S}} \beta_{ijks} x_s \leq 1 \quad \{i, j, k\} \in N \tag{2.7d}$$

$$x_s \in \mathbb{R}_+ \qquad s \in \mathcal{S} \tag{2.7e}$$

2.6 Outline of the ESF

This section provides an outline of the ESF described for the CMTVRPTW. We assume that a feasible solution of the CMTVRPTW exists. The algorithm uses the following notation:

- ub^* is the best upper bound (if any) found;
- ub_{guess} is a guess upper bound on the value of the optimal solution cost;
- gap_{guess} is a guess on the gap in percentage between the optimal CMTVRPTW solution cost and CVRPTW-LB;
- gap^0_{guess} is the initial value of gap_{guess};
- gap_{step} is the increase, at each iteration, of gap_{guess};
- $status$ is the status of the solution process, which can take one of the following three values: $optimal$ (i.e., an optimal solution of cost ub^* was found), $feasible$ (i.e., a feasible solution of cost ub^* was found, but it may not be optimal), nil (i.e., no solution has been found yet).

The algorithm consists of the following seven steps:

1. *Initialization*: Set $status = nil$ and $gap_{guess} = gap^0_{guess}$;

2. *CVRPTW-LB Computation* (see Section 2.5.1 for an overview and Section 2.7.1 for the details): solve problem $\text{P1}_{\mathcal{S}}$ by column generation, and compute its optimal solution cost $z(\text{P1}_{\mathcal{S}})$; set $ub_{guess} = z(\text{P1}_{\mathcal{S}}) * (1 + gap_{guess})$;

3. *Structure Enumeration* (see Section 2.7.2 for the details): enumerate the set $\mathcal{S}_1$ of all the structures having reduced costs not greater than $ub_{guess} - z(\text{P1}_{\mathcal{S}})$ with respect to the dual solution corresponding to lower bound $z(\text{P1}_{\mathcal{S}})$;

4. *RSFC-LB Computation* (see Section 2.5.2 for an overview and Section 2.7.3 for the details): Solve problem $\text{P2}_{\mathcal{S}_1}$ obtained from $\text{P2}_{\mathcal{S}}$ by replacing the set of structures $\mathcal{S}$ with its subset $\mathcal{S}_1$. Let $z(\text{P2}_{\mathcal{S}_1})$ be the optimal cost of $\text{P2}_{\mathcal{S}}$.

5. *Structure Reduction* (see Section 2.7.4): Compute the set of structures $\mathcal{S}_2 \subseteq \mathcal{S}_1$ obtained from $\mathcal{S}_1$ by removing all the structures having reduced cost greater than $ub_{guess} - z(\text{P2}_{\mathcal{S}_1})$ with respect to the dual solution of cost $z(\text{P2}_{\mathcal{S}_1})$ of problem $\text{P2}_{\mathcal{S}_1}$.

6. *Branch-and-Cut to Close the Gap* (see Section 2.7.5): By using a branch-and-cut method, solve problem $\text{F}_{\mathcal{S}_2}$ obtained from $\text{F}_{\mathcal{S}}$ by replacing the set of structures $\mathcal{S}$ with its subset $\mathcal{S}_2$. If $\text{F}_{\mathcal{S}_2}$ contains feasible solutions, let $z(\text{F}_{\mathcal{S}_2})$ be the cost of an optimal solution.

7. *Iterative Step*: there are three possible cases

   (a) A feasible solution of $\text{F}_{\mathcal{S}_2}$ exists

   (a1) If $z(\text{F}_{\mathcal{S}_2}) \leq ub_{guess}$: such a solution is an optimal CMTVRPTW solution; set $status = optimal$, $ub^* = z(\text{F}_{\mathcal{S}_2})$, and terminate;

   (a2) If $z(\text{F}_{\mathcal{S}_2}) > ub_{guess}$: such a solution is a valid upper bound to the CMTVRPTW; set $status = feasible$, $ub^* = z(\text{F}_{\mathcal{S}_2})$, $ub_{guess} = z(\text{F}_{\mathcal{S}_2})$, and go to Step 3;

   (b) $\text{F}_{\mathcal{S}_2}$ does not have any feasible solution: set $gap_{guess} = gap_{guess} + gap_{step}$, $ub_{guess} = z(\text{P1}_{\mathcal{S}}) * (1 + gap_{guess})$, and go to Step 3.

2.7 Detailed Description of Steps 2-6 of the ESF

2.7.1 Step2: CVRPTW-LB Computation

Step 2 of the ESF requires solving problem $\text{P1}_{\mathcal{S}}$ to compute its optimal value $z(\text{P1}_{\mathcal{S}})$. As the set of structures $\mathcal{S}$ increases exponentially with the number of customers, problem $\text{P1}_{\mathcal{S}}$ must be solved via column generation. The column generation procedure we propose initializes the master problem with the single-customer structures and solves it via a general purpose solver. At each iteration, at most col_iter (where col_iter is a parameter) negative reduced cost structures are added to the master problem. Such negative reduced cost structures are priced out by using dynamic programming as follows.

Let $u_i \in \mathbb{R}$ be the dual variable associated with constraint (2.4b) of customer $i \in N$. Let $\tilde{c}_{ij}$ be the reduced cost of arc $(i, j) \in A$ with respect to $\boldsymbol{u} \in \mathbb{R}^n$ defined as

$$\tilde{c}_{ij} = \begin{cases} c_{ij} - \frac{1}{2}(u_i + u_j) & \text{if } i \neq 0 \text{ and } j \neq 0 \\ c_{ij} - \frac{1}{2}u_j & \text{if } i = 0 \\ c_{ij} - \frac{1}{2}u_i & \text{otherwise (i.e., if } j = 0) \end{cases} \tag{2.8}$$

Let $f = (0, i_1, \ldots, i_{\mu_f} = i^f)$ be an elementary forward path that starts from the depot at time a_0, visits the set of customers $N^f = \{i_1, \ldots, i^f\}$ each within its time window, ends at customer i^f at time t^f, such that the total demand q^f of the visited customers does not exceed the vehicle capacity. Let $\tilde{c}^f$ be the reduced cost of path f defined as the sum of the reduced costs of the traversed arcs.

We associate a label $L^f = (N^f, q^f, t^f, i^f, \tilde{c}^f)$ with each path $f = (0, i_1, \ldots, i^f)$. Due to capacity and time window constraints, a label $L^f = (N^f, q^f, t^f, i^f, \tilde{c}^f)$ is feasible only if $N^f \subseteq N$, $q^f \leq \overline{q}$, $t^f \in [a_{if}, b_{if}]$, and $i^f \in N^f$. Clearly, each path $f = (0, i_1, \ldots, i^f)$ such that $i^f = 0$ and $\tilde{c}^f < 0$ corresponds to a negative reduced cost structure.

To generate such structures, the following initialization and extension are needed:

Initialization: A single label $L^f = (N^f, q^f, t^f, i^f, \tilde{c}^f)$ is generated, where $N^f = \varnothing$, $q^f = 0$, $t^f = a_0$, $i^f = 0$, $\tilde{c}^f = 0$.

Extension: Extend each label $L^f = (N^f, q^f, t^f, i^f, \tilde{c}^f)$ such that either (a) $i^f \in N$ or (b) $i^f = 0$ and $N^f = \varnothing$ toward any vertex $j \in V \setminus N^f$ and generate label $L^{f'} = (N^{f'}, q^{f'}, t^{f'}, i^{f'}, \tilde{c}^{f'})$, where

$$\begin{cases} N^{f'} = N^f \cup \{j\} \text{ (if } j \in N) \text{ or } N^{f'} = N^f \text{ (if } j = 0) \\ q^{f'} = q^f + q_j \\ t^{f'} = \max\{a_j, t^f + st_{if} + t_{if\,j}\} \\ i^{f'} = j \\ \tilde{c}^{f'} = \tilde{c}^f + \tilde{c}_{if\,j} \end{cases}$$

To speed up the solution process and limit the number of labels to generate, the following dominance rule can be applied: given two labels $L^{f_1} = (N^{f_1}, q^{f_1}, t^{f_1}, i^{f_1}, \tilde{c}^{f_1})$ and $L^{f_2} = (N^{f_2}, q^{f_2}, t^{f_2}, i^{f_2}, \tilde{c}^{f_2})$, L^{f_2} is dominated if $N^{f_1} \subseteq N^{f_2}$, $q^{f_1} \leq q^{f_2}$, $t^{f_1} \leq t^{f_2}$, $i^{f_1} = i^{f_2}$, and $\tilde{c}^{f_1} \leq \tilde{c}^{f_2}$. Moreover, we also apply two other well-known techniques, i.e., ng-path relaxation (see Baldacci et al. (2011)) and completion bounds (see Baldacci et al. (2012)), to further limit the number of labels to generate. For the sake of brevity, we omit the details on this matter.

Let $\boldsymbol{u}^1 = (u_1^1, u_2^1, \ldots, u_n^1)$ be the optimal dual solution of cost $z(\text{P1}_{\mathcal{S}})$ achieved by Step 2.

2.7.2 Step 3: Structure Enumeration

Step 3 requires enumerating the set $\mathcal{S}_1$ all the structures having reduced costs not greater than $ub_{guess} - z(\text{P1}_{\mathcal{S}})$ with respect to the dual solution $\boldsymbol{u}^1$ of cost $z(\text{P1}_{\mathcal{S}})$. The set $\mathcal{S}_1$ can be generated with the following dynamic programming recursion, which is similar to the recursions proposed by Hernandez et al. (2014, 2016), and Tilk & Irnich (2016).

Let $b = (0, i_1, \ldots, i_{\mu_b} = i^b)$ be an elementary backward path that can start from the depot not earlier than g^b, visits the set of customers N^b each within its time window, ends at customer i^b not later than ℓ^b, has a duration equal to d^b, and such that the total demand of the visited customers is equal to q^b. Let $\tilde{c}^b$ be

the reduced cost of path b with respect of the dual solution $\boldsymbol{u}^1$, given by the sum of the reduced costs of the traversed arcs computed as in (2.8).

With each path $b = (0, i_1, \ldots, i^b)$, we associate a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$. A label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is feasible if the following conditions hold

$$\begin{cases} N^b \subseteq N \\ q^b \leq \overline{q} \\ a_{i^b} \leq \ell^b \leq b_{i^b} \\ a_0 \leq g^b \leq b_0 \\ i^b \in N^b \end{cases} \tag{2.9}$$

A structure $s \in \mathcal{S}$ corresponds to an elementary backward path $b = (0, i_1, \ldots, i^b)$ (and the corresponding label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$), such that $i^b = 0$, and where $\ell_s = \ell^b$, $e_s = g^b - d^b$, $d_s = d^b$, $c_s = \tilde{c}^b + \sum_{i \in N^b} u_i^1$, and $\alpha_{is} = 1, \forall i \in N^b$ (0 if $i \in N \setminus N^b$).

To generate the set of structures $\mathcal{S}_1$, the following initialization and extension are needed:

Initialization: A single label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is generated, where $N^b = \varnothing$, $q^b = 0$, $\ell^b = b_0$, $g^b = a_0$, $d^b = 0$, $i^b = 0$, and $\tilde{c}^b = 0$.

Extension: Extend each label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ such that either (a) $i^b \in N$ or (b) $i^b = 0$ and $N^b = \varnothing$ toward any vertex $j \in V \setminus N^b$ to generate label $L^{b'} = (N^{b'}, q^{b'}, \ell^{b'}, g^{b'}, d^{b'}, i^{b'}, \tilde{c}^{b'})$ where

$$\begin{cases} N^{b'} = N^b \cup \{j\} \text{ (if } j \in N) \text{ or } N^{b'} = N^b \text{ (if } j = 0) \\ q^{b'} = q^b + q_j \\ \ell^{b'} = \min\{b_j, \ell^b - t_{ji^b} - st_j\} \\ g^{b'} = \max\{a_j + d^b + t_{ji^b} + st_j,\ g^b\} \\ d^{b'} = \max\{d^b + t_{ji^b} + st_j,\ g^b - b_j\} \\ i^{b'} = j \\ \tilde{c}^{b'} = \tilde{c}^b + \tilde{c}_{ji^b} \end{cases} \tag{2.10}$$

To speed up the enumeration phase and limit the number of generated labels, the following dominance rule can be applied: given two labels $L^{b_1} = (N^{b_1}, q^{b_1}, \ell^{b_1}, g^{b_1}, d^{b_1}, i^{b_1}, \tilde{c}^{b_1})$ and $L^{b_2} = (N^{b_2}, q^{b_2}, \ell^{b_2}, g^{b_2}, d^{b_2}, i^{b_2}, \tilde{c}^{b_2})$, L^{b_2} is dominated if $N^{b_1} = N^{b_2}$ (and thus $q^{b_1} = q^{b_2}$), $\ell^{b_1} \geq \ell^{b_2}$, $g^{b_1} \leq g^{b_2}$, $d^{b_1} \leq d^{b_2}$, $i^{b_1} = i^{b_2}$, and $\tilde{c}^{b_1} \leq \tilde{c}^{b_2}$.

The number of labels can further be limited by using completion bounds based on the ng-path relaxation, as described in Baldacci et al. (2011, 2012).

2.7.3 Step 4: RSFC-LB Computation

In Step 4, problem $\text{P2}_{\mathcal{S}_1}$ is solved to compute its optimal solution value $z(\text{P2}_{\mathcal{S}_1})$. Problem $\text{P2}_{\mathcal{S}_1}$ is obtained from $\text{P2}_{\mathcal{S}}$ by replacing the set of structures $\mathcal{S}$ with its subset $\mathcal{S}_1$ generated in Step 3. As the set of columns is known, no column generation is needed. At the beginning of Step 4, problem $\text{P2}_{\mathcal{S}_1}$ is solved with an LP-solver without imposing constraints (2.7c) and (2.7d). These constraints are subsequently iteratively added in a cutting plane fashion.

The separation of both (2.7c) and (2.7d) can easily be done by inspection given the current optimal solution $\tilde{\boldsymbol{x}}$. Let $\tilde{\mathcal{S}} = \{s \in \mathcal{S} \mid \tilde{x}_s > 0\}$ be the set of structures in the solution $\tilde{\boldsymbol{x}}$. The violation of (2.7c) is

checked only for instants of times $h \in [a_0, b_0]$ for which it exists $s \in \tilde{\mathcal{S}}$ such that $h = \ell_s$, whereas the violation of constraints (2.7d) is checked simply for all triplets of customers $\{i, j, k\} \in N$.

At the end of Step 4, an optimal dual solution $(\boldsymbol{u}^2, \boldsymbol{v}^2, \boldsymbol{w}^2)$ of $\text{P2}_{\mathcal{S}_1}$ of cost $z(\text{P2}_{\mathcal{S}_1})$ is available, where $\boldsymbol{u}^2$, $\boldsymbol{v}^2$, and $\boldsymbol{w}^2$ are the vectors of dual variables associated with constraints (2.7b), (2.7c), and (2.7d), respectively.

2.7.4 Step 5: Structure Reduction

Step 5 aims at deriving the set of structures $\mathcal{S}_2 \subseteq \mathcal{S}_1$ that is obtained from $\mathcal{S}_1$ by removing all structures with a reduced cost greater than $ub_{guess} - z(\text{P2}_{\mathcal{S}_1})$ with respect to the dual solution $(\boldsymbol{u}^2, \boldsymbol{v}^2, \boldsymbol{w}^2)$ of cost $z(\text{P2}_{\mathcal{S}_1})$ of problem $\text{P2}_{\mathcal{S}_1}$ achieved at Step 4. The set $\mathcal{S}_2$ is easily derived by inspection of the set $\mathcal{S}_1$.

2.7.5 Step 6: Branch-and-Cut to Close the Gap

Step 6 attempts to find an optimal CMTVRPTW solution by applying a branch-and-cut method. In Steps 4 and 5, a limited set of structures $\mathcal{S}_2$ has been generated. This set contains optimal CMTVRPTW solutions under the assumption that ub_{guess} is a valid upper bound to the CMTVRPTW. Step 6 solves problem $\text{F}_{\mathcal{S}_2}$ obtained from $\text{F}_{\mathcal{S}}$ by replacing the set of structures $\mathcal{S}$ with its subset $\mathcal{S}_2$. It applies a branch-and-cut algorithm because all columns of $\text{F}_{\mathcal{S}_2}$ are known, so only SFC constraints (2.3c) may be missing. The SFC constraints are added in a cutting plane fashion as follows. Notice that as the separation problem is NP-hard, it is performed on integer solutions only.

Let us assume that we have an integer solution $\tilde{\boldsymbol{x}}$ of $\text{F}_{\mathcal{S}_2}$, and we want to check its feasibility by separating violated SFC constraints. Let $\tilde{\mathcal{S}} \subseteq \mathcal{S}_2$ be the subset of structures in a solution (i.e., $\tilde{\mathcal{S}} = \{s \in \mathcal{S}_2 \,|\, \tilde{x}_s = 1\}$). The right-hand side of constraint (2.7c) corresponding to the set $\tilde{\mathcal{S}}$ is equal to the maximum number of structures of the set $\tilde{\mathcal{S}}$ that can be simultaneously in solution. The separation problem of determining if the structures of the set $\tilde{\mathcal{S}}$ represent a feasible CMTVRPTW solution can be then formulated as a *Team Orienteering Problem with Time Windows*, TOPTW (see, e.g., Vansteenwegen et al. (2009), Archetti et al. (2014)), where (a) each node is a structure to assign to a vehicle, (b) m vehicles are available, (c) each structure has a unit profit and a time window $[e_s, \ell_s]$, and (d) the goal is to maximize the number of structures that can be assigned to the m vehicles, which is achieved by maximizing the profit of the TOPTW.

Therefore, the separation problem of SFC on the set $\tilde{\mathcal{S}}$ can be formulated on a support graph $\tilde{G} = (\tilde{V}, \tilde{A})$ defined as follows. The vertex set $\tilde{V}$ contains a node for each structure of $\tilde{\mathcal{S}}$ plus a dummy node o (i.e., $\tilde{V} = \{o\} \cup \tilde{\mathcal{S}}$). The arc set $\tilde{A}$ is defined as $\tilde{A} = \{(o, s) \,|\, s \in \tilde{\mathcal{S}}\} \cup \{(s_1, s_2) \,|\, s_1, s_2 \in \tilde{\mathcal{S}}, s_1 \neq s_2, e_{s_1} + d_{s_1} \leq \ell_{s_2}\} \cup \{(s, o) \,|\, s \in \tilde{\mathcal{S}}\}$. With each arc $(s_1, s_2) \in \tilde{A}$, we associate a travel time $\tilde{t}_{s_1 s_2}$ defined as $\tilde{t}_{s_1 s_2} = d_{s_1}$ if $s_1 \neq o$, and $\tilde{t}_{s_1, s_2} = 0$ if $s_1 = o$. For example, let us assume that $\tilde{S} = \{s_1, s_2, s_3, s_4, s_5, s_5, s_6\}$ and $e_{s_1} = \ell_{s_1} = 119.7$, $d_{s_1} = 145.6$; $e_{s_2} = 264$, $\ell_{s_2} = 276$, $d_{s_2} = 136.3$; $e_{s_3} = \ell_{s_3} = 415.5$, $d_{s_3} = 166.5$; $e_{s_4} = \ell_{s_4} = 607.4$, $d_{s_4} = 240.8$; $e_{s_5} = \ell_{s_5} = 142.1$, $d_{s_5} = 257.1$; $e_{s_6} = \ell_{s_6} = 437.9$, $d_{s_6} = 298.1$. The corresponding support graph $\tilde{G}$ is represented in Figure 2.1.

Let $\tilde{\mathcal{R}}$ be the set of all elementary journeys of graph $\tilde{G}$ that start and end at node o and such that the sequence of visited nodes/structures can be assigned to the same vehicle. For each journey $r = (s_1, s_2, \ldots, s_{\mu_r}) \in \tilde{\mathcal{R}}$, let α_{sr} be a binary coefficient equal to 1 if structure $s \in \tilde{\mathcal{S}}$ is assigned to

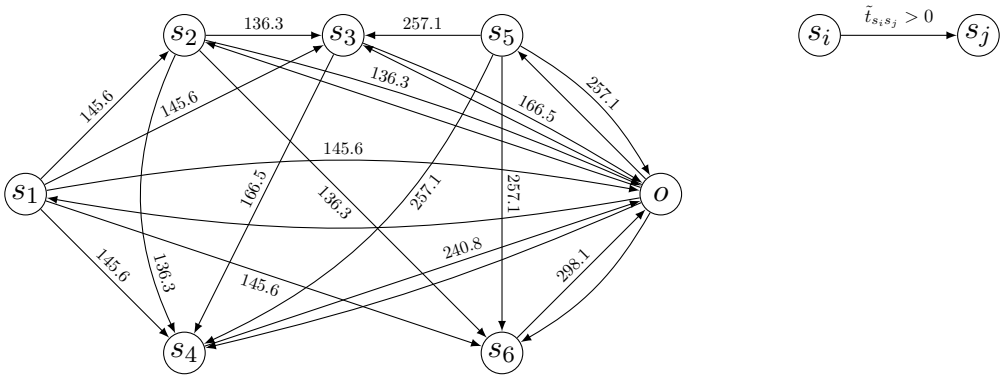


Figure 2.1: Example of a support graph $\tilde{G} = (\tilde{V}, \tilde{A})$ with six structures

journey r. Moreover, let $y_r \in \{0, 1\}$ be a binary variable equal to 1 if journey $r \in \tilde{\mathcal{R}}$ is selected (0 otherwise). The separation problem of SFC can be formulated as

$$\eta_m(\tilde{\mathcal{S}}) = \max \sum_{r \in \tilde{\mathcal{R}}} \Big(\sum_{s \in \tilde{\mathcal{S}}} \alpha_{sr} \Big) y_r \tag{2.11a}$$

$$\text{s.t.} \sum_{r \in \tilde{\mathcal{R}}} \alpha_{sr} y_r \leq 1 \qquad s \in \tilde{\mathcal{S}} \tag{2.11b}$$

$$\sum_{r \in \tilde{\mathcal{R}}} y_r \leq m \tag{2.11c}$$

$$y_r \in \{0, 1\} \qquad r \in \tilde{\mathcal{R}} \tag{2.11d}$$

The objective function (2.11a) aims at maximizing the number of structures used in the selected journeys. Constraints (2.11b) ensure that each structure is not assigned more than once. Constraint (2.11c) guarantees that at most m journeys are selected. Constraints (2.11d) are integrality constraints.

The cardinality of the set $\tilde{\mathcal{R}}$ clearly increases exponentially with the number of structures of the set $\tilde{\mathcal{S}}$, so problem (2.11a)-(2.11d) can be solved by using column generation. The master problem corresponds to the linear relaxation of (2.11a)-(2.11d). The pricing problem corresponds to finding journeys of the set $\tilde{\mathcal{R}}$ having negative reduced cost with respect to the dual solution $(\boldsymbol{u}, v)$ of the master problem, where $\boldsymbol{u} \in \mathbb{R}_+^{|\tilde{\mathcal{S}}|}$ and $v \in \mathbb{R}_+$ are the dual variables associated with constraints (2.11b) and (2.11c), respectively.

Solving the Pricing Problem.

The pricing problem can be solved by using dynamic programming as follows. Let $f = (o, s_1, \ldots, s_{\mu_f} = s^f)$ be an elementary forward path of graph $\tilde{G}$ that starts from node o, visits the set of nodes $\tilde{S}^f = \{s_1, \ldots, s^f\}$, and ends at node s^f at time $\tilde{e}^f$. Let $\tilde{c}^f$ be the reduced cost of path f with respect to the dual solution $(\boldsymbol{u}, v)$. We associate a label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ with each path $f = (o, s_1, \ldots, s^f)$. A label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ is feasible if $\tilde{S}^f \subseteq \tilde{V}$, $s^f \in \tilde{S}^f$, and $\tilde{e}^f \in [e_{s^f}, \ell_{s^f}]$. Each label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ with $s^f = o$ and $\tilde{c}^f < 0$ corresponds to a negative reduced cost journey.

To generate such negative reduced cost journeys, the following initialization and extension are needed:

Initialization: A label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ is generated, where $\tilde{S}^f = \varnothing$, $s^f = o$, $\tilde{e}^f = a_0$, $\tilde{c}^f = -v$.

Extension: Each label $L^f = (\tilde{S}^f, s^f, \tilde{e}^f, \tilde{c}^f)$ for which either (a) $s^f \in \tilde{\mathcal{S}}$ or (b) $s^f = o$ and $\tilde{S}^f = \varnothing$ is extended toward any node $s \in \tilde{V} \setminus \tilde{S}^f$ and generates label $L^{f'} = (\tilde{S}^{f'}, s^{f'}, \tilde{e}^{f'}, \tilde{c}^{f'})$ where

- $\tilde{S}^{f'} = \tilde{S}^f \cup \{s\}$,
- $s^{f'} = s$,
- $\tilde{e}^{f'} = \max\{\tilde{e}^f + \tilde{t}_{s^f s}, e_s\}$,
- $\tilde{c}^{f'} = \tilde{c}^f + 1 - u_s$ (if $s \in \tilde{\mathcal{S}}$) or $\tilde{c}^{f'} = \tilde{c}^f$ (if $s = o$).

To speed up the solution process and limit the number of labels to generate, the following the dominance rule can be applied: given two labels $L^{f_1} = (\tilde{S}^{f_1}, s^{f_1}, \tilde{e}^{f_1}, \tilde{c}^{f_1})$ and $L^{f_2} = (\tilde{S}^{f_2}, s^{f_2}, \tilde{e}^{f_2}, \tilde{c}^{f_2})$, L^{f_2} is dominated if $\tilde{S}^{f_1} = \tilde{S}^{f_2}$, $s^{f_1} = s^{f_2}$, $\tilde{e}^{f_1} \leq \tilde{e}^{f_1}$ and $\tilde{c}^{f_1} \leq \tilde{c}^{f_1}$.

Branching Scheme.

Given a fractional solution $\tilde{\boldsymbol{y}}$ of problem (2.11a)-(2.11d) three types of branching are performed in a hierarchical way to find an integer solution.

Branching on Structures: a binary branching is performed on the structure s' that is closest to be used 0.5 times, i.e., $s' = \arg\min_{s \in \tilde{\mathcal{S}}(\tilde{\boldsymbol{y}})}\{|\sum_{r \in \tilde{\mathcal{R}}} \alpha_{sr}\tilde{y}_r - 0.5|\}$, where $\tilde{\mathcal{S}}(\tilde{\boldsymbol{y}}) = \{s \in \tilde{\mathcal{S}} \,|\, 0 < \sum_{r \in \tilde{\mathcal{R}}} \alpha_{sr}\tilde{y}_r < 1\}$. In the first branch, structure s' must be used, i.e., $\sum_{r \in \tilde{\mathcal{R}}} \alpha_{s'r}\tilde{y}_r = 1$. In the second branch, structure s' cannot used, i.e., $\sum_{r \in \tilde{\mathcal{R}}} \alpha_{s'r}\tilde{y}_r = 0$;

Branching on Arcs: a binary branching is performed on the arc $(s, s') \in \tilde{A}$ that is closest to be traversed 0.5 times by the journeys in solution $\tilde{\boldsymbol{y}}$, i.e., $(s, s') = \arg\min_{(s_1,s_2) \in \tilde{A}(\tilde{\boldsymbol{y}})}\{|\sum_{r \in \tilde{\mathcal{R}}} \gamma_{s_1 s_2 r}\tilde{y}_r - 0.5|\}$, where $\tilde{A}(\tilde{\boldsymbol{y}}) = \{(s_1, s_2) \in \tilde{A} \,|\, 0 < \sum_{r \in \tilde{\mathcal{R}}} \gamma_{s_1 s_2 r}\tilde{y}_r < 1\}$ and $\gamma_{s_1 s_2 r}$ is the number of times arc $(s_1, s_2) \in \tilde{A}$ is traversed by journey $r \in \tilde{\mathcal{R}}$. In the first branch, arc (s, s') is removed from the arc set $\tilde{A}$. In the second branch, arc (s, s') is forced to be in solution if structures s and s' are visited, which is achieved by removing (a) all arcs $\{(s_1, s') \in \tilde{A} \,|\, s_1 \neq s\}$ and $\{(s, s_2) \in \tilde{A} \,|\, s_2 \neq s'\}$ if $s, s' \in \tilde{\mathcal{S}}$, (b) all arcs $\{(s_1, s') \in \tilde{A} \,|\, s_1 \neq o\}$ if $s = o$, and (c) all arcs $\{(s, s_2) \in \tilde{A} \,|\, s_2 \neq o\}$ if $s' = o$.

Branching on the Number of Vehicles: a binary branching is performed on the number of selected journeys, i.e., on $\sum_{r \in \tilde{\mathcal{R}}} \tilde{y}_r$. In the first branch, constraint $\sum_{r \in \tilde{\mathcal{R}}} y_r \leq \lfloor \sum_{r \in \tilde{\mathcal{R}}} \tilde{y}_r \rfloor$ is added. In the second branch, constraint $\sum_{r \in \tilde{\mathcal{R}}} y_r \geq \lceil \sum_{r \in \tilde{\mathcal{R}}} \tilde{y}_r \rceil$ is added.

In the search tree, the best-bound-first policy is applied to identify the next node to branch on.

Acceleration Techniques.

To speed up the solution process, we embed three more features in the branch-and-price to solve problem (2.11a)-(2.11d):

- Relaxed SFC inequalities (2.7c) are added in a cutting plane fashion and separated both on integer and fractional solutions as described in Section 2.7.3.
- The pricing problem is relaxed by dropping resource $\tilde{S}^f$, so non-elementary journeys are allowed.

- The computation of $\eta_m(\tilde{\mathcal{S}})$ is stopped as soon as the highest upper bound (let us call this upper bound $\overline{\eta}_m(\tilde{\mathcal{S}})$) of all unexplored nodes in the search tree is strictly less than $|\tilde{\mathcal{S}}|$. Indeed, this condition is sufficient to add constraint $\sum_{s\in\tilde{\mathcal{S}}} x_s \leq \lfloor \overline{\eta}_m(\tilde{\mathcal{S}}) \rfloor$, which cuts off the integer solution $\tilde{\boldsymbol{x}}$.

2.8 Tailoring the ESF to Solve the Four Variants of the CMTVRPTW

As mentioned in Section 2.1, the CMTVRPTW can be specialized by adding side constraints that determine the feasibility of the structures and therefore define the set $\mathcal{S}$. Notice that CVRPTW-LB remains a valid lower bound even if these side constraints are added. Notice also that, once the set of structures $\mathcal{S}_1$ is enumerated in Step 3, the remaining steps are not affected by the constraints that define the feasibility of a structure but just rely on having the set of feasible structures $\mathcal{S}_1$. Therefore, the ESF can be tailored to solve the four variants of the CMTVRPTW by simply modifying the dynamic programming recursion of Step 3 to enumerate the structures in the gap $ub_{guess} - z(\text{P1}_{\mathcal{S}})$. In the following sections, we describe how Step 3 (described in Section 2.7.2 for the CMTVRPTW) can be modified to tackle each of the four variants.

2.8.1 Structure Enumeration for the CMTVRPTW-LT

The CMTVRPTW-LT proposed by Hernandez et al. (2016) considers the time to load the vehicle before it can depart from the depot. In particular, given the loading time lt_i for the goods requested by customer $i \in N$, the total loading time of a vehicle performing a given structure is equal to the sum of the individual loading times of the customers in the structure. For each label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$, let us define its total loading time lt^b as $lt^b = \sum_{i\in N^b} lt_i$.

The structure enumeration for the CMTVRPTW-LT requires just two changes with respect to the one for the CMTVRPTW. The first change is in the feasibility of a label, and the second change is in the extension of a label to the depot.

A label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ with $i^b \neq 0$ is feasible if conditions (2.9) are satisfied and the following condition holds

$$\ell^b - t_{0i^b} - lt^b \geq a_0 \tag{2.12}$$

Indeed, condition (2.12) ensures that there is enough time to close the path by returning to the depot and to reload the vehicle.

When a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is extended to a customer, extension functions (2.10) are still valid. Whereas when a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is extended to the depot to generate label $L^{b'} = (N^{b'}, q^{b'}, \ell^{b'}, g^{b'}, d^{b'}, i^{b'}, \tilde{c}^{b'})$, functions (2.10) are replaced with the following functions

$$\begin{cases} N^{b'} = N^b \\ q^{b'} = q^b \\ \ell^{b'} = \ell^b - t_{0i^b} - lt^b \\ g^{b'} = \max\{a_0 + d^b + t_{0i^b} + lt^b,\ g^b\} \\ d^{b'} = \max\{d^b + t_{0i^b} + lt^b,\ g^b - b_0\} \\ i^{b'} = 0 \\ \tilde{c}^{b'} = \tilde{c}^b + \tilde{c}_{0i^b} \end{cases}$$

that take into account the total loading time of a structure in computing the time to leave from the depot, return to the depot, and in the total duration.

2.8.2 Structure enumeration for the CMTVRPTW-LD

In the CMTVRPTW-LD studied by Hernandez et al. (2014), a limited trip duration $\overline{d}$ is considered, which represents a maximum limit on the time that goods can be on board of the vehicle before being delivered. This does not include loading times, nor the service time of the last customer of a structure, nor the travel time to return to the depot. We accept the definition of limited trip duration given by Hernandez et al. (2014) even though we believe it misleading and it should be called differently, for example, *maximum goods travel time*; indeed, we think it is more intuitive to call *limited trip duration* the maximum amount of time that the vehicle can be traveling in each trip from the departure from the depot to the return to it.

The structure enumeration of the CMTVRPTW-LD differs from the one for the CMTVRPTW because an additional resource is necessary, and feasibility conditions and dominance rules must take the limited trip duration into account.

With each backward path $b = (0, i_1, \ldots, i^b)$, we associate a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, i_1^b, \tilde{c}^b)$, where i_1^b represents the first customer visited by the backward path. Let dg^b be the goods travel time computed as $dg^b = d^b - t_{0i_1} - st_{i_1} - lt^b$. A label is feasible if it satisfies conditions (2.9), (2.12), and the additional condition $dg^b \leq \overline{d}$.

Moreover, dominance rules need to be adjusted as follows. Given two labels $L^{b_1} = (N^{b_1}, q^{b_1}, \ell^{b_1}, g^{b_1}, d^{b_1}, i^{b_1}, i_1^{b_1}, \tilde{c}^{b_1})$ and $L^{b_2} = (N^{b_2}, q^{b_2}, \ell^{b_2}, g^{b_2}, d^{b_2}, i^{b_2}, i_1^{b_2}, \tilde{c}^{b_2})$, L^{b_2} is dominated if $N^{b_1} = N^{b_2}$ (and thus $q^{b_1} = q^{b_2}$), $\ell^{b_1} \geq \ell^{b_2}$, $g^{b_1} \leq g^{b_2}$, $d^{b_1} \leq d^{b_2}$, $i^{b_1} = i^{b_2}$, $i_1^{b_1} = i_1^{b_2}$, and $\tilde{c}^{b_1} \leq \tilde{c}^{b_2}$.

2.8.3 Structure Enumeration for the CMTVRPTW-R

Cattaruzza et al. (2016*a*) introduce the concept of *release dates* in the CMTVRPTW and study the CMTVRPTW-R. Each customer $i \in N$ has an associated release date rd_i that indicates the instant of time when the goods to deliver to customer i become available at the depot; therefore, the vehicle that serves customer i cannot depart from the depot before rd_i.

Similar to the case of the CMTVRPTW-LT, the structure enumeration for the CMTVRPTW-R requires two changes with respect to the one for the CMTVRPTW: the first in the feasibility conditions of a label, and the second change in the extension of a label to the depot.

Let us indicate with rd^b the maximum release date of the customers serve by a backward path $b = (0, i_1, \ldots, i^b)$, i.e., $rd^b = \max_{i \in N^b}\{rd_i\}$. The label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ associated with path $b = (0, i_1, \ldots, i^b)$ is feasible if it satisfies the following conditions

$$\begin{cases} N^b \subseteq N \\ q^b \leq \overline{q} \\ a_{i^b} \leq \ell^b \leq b_{i^b} \\ rd^b \leq g^b \leq b_0 \\ i^b \in N^b \end{cases}$$

When a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ is extended to the depot to generate label $L^{b'} = (N^{b'}, q^{b'}, \ell^{b'}, g^{b'}, d^{b'}, i^{b'}, \tilde{c}^{b'})$, functions (2.10) are replaced with the following extension functions

$$\begin{cases} N^{b'} = N^b \\ q^{b'} = q^b \\ \ell^{b'} = \ell^b - t_{0i^b} \\ g^{b'} = \max\{\max\{rd^{b'}, a_0\} + d^b + t_{0i^b},\ g^b\} \\ d^{b'} = \max\{d^b + t_{0i^b},\ g^b - b_0\} \\ i^{b'} = 0 \\ \tilde{c}^{b'} = \tilde{c}^b + \tilde{c}_{0i^b} \end{cases}$$

2.8.4 Structure enumeration for the DRP

The Drone Routing Problem studied by Cheng et al. (2018) can be seen as a generalization of the CMTVRPTW, where vehicles are drones and specific features of the drones must be taken into account. In particular, unlike standard vehicles, drones have a limited battery capacity whose consumption depends on the load carried. Therefore, energy consumption and energy cost should be considered when optimizing drone-based distribution systems.

The energy consumed by a drone to traverse arc $(i, j) \in A$ depends both on the distance traveled and the weight transported while traversing the arc. Let en_{ijq} and ce_{ijq} be the energy consumption and the energy costs, respectively, associated with $(i, j) \in A$ when the drone is carrying weight q. Each drone has to return to the depot before running out of battery. Let us call $\overline{en}$ the drone battery capacity. Moreover, following Cheng et al. (2018), let us also assume the each drone starts each trip equipped with a fully charged battery.

Let us associate with each backward path $b = (0, i_1, \ldots, i^b)$ a label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, en^b, \tilde{c}^b)$, where en^b is the energy consumption of path b. A label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, en^b, \tilde{c}^b)$ is feasible if the following conditions hold

$$\begin{cases} N^b \subseteq N \\ q^b \leq \overline{q} \\ a_{i^b} \leq \ell^b \leq b_{i^b} \\ a_0 \leq g^b \leq b_0 \\ en^b \leq \overline{en} \\ i^b \in N^b \end{cases}$$

The enumeration is initialized with a single label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, en^b, \tilde{c}^b)$ where $N^b = \varnothing$, $q^b = 0$, $\ell^b = b_0$, $g^b = a_0$, $d^b = 0$, $en^b = 0$, $\tilde{c}^b = 0$. Each label $L^b = (N^b, q^b, \ell^b, g^b, d^b, i^b, \tilde{c}^b)$ for which either (a) $i^b \in N$ or (b) $i^b = 0$ and $N^b = \varnothing$ is extended toward any vertex $j \in V \setminus N^b$ to generate label $L^{b'} = (N^{b'}, q^{b'}, \ell^{b'}, g^{b'}, d^{b'}, i^{b'}, \tilde{c}^{b'})$ where

$$\begin{cases} N^{b'} = N^b \cup \{j\} \text{ (if } j \in N) \text{ or } N^{b'} = N^b \text{ (if } j = 0) \\ q^{b'} = q^b + q_j \\ \ell^{b'} = \min\{b_j, \ell^b - t_{ji^b} - st_j\} \\ g^{b'} = \max\{a_j + d^b + t_{ji^b} + st_j,\ g^b\} \\ d^{b'} = \max\{d^b + t_{ji^b} + st_j,\ g^b - b_j\} \\ en^{b'} = en^b + en_{ji^bq^b} \\ i^{b'} = j \\ \tilde{c}^{b'} = \tilde{c}^b + c_{ji^b} + ce_{ji^bq^b} - u_j^1 \text{ (if } j \in N) \text{ or } \tilde{c}^{b'} = \tilde{c}^b + c_{ji^b} + ce_{ji^bq^b} \text{ (if } j = 0) \end{cases}$$

Finally, dominance rules should be adjusted as follows. Given two labels $L^{b_1} = (N^{b_1}, q^{b_1}, \ell^{b_1}, g^{b_1}, d^{b_1}, i^{b_1}, en^{b_1}, \tilde{c}^{b_1})$ and $L^{b_2} = (N^{b_2}, q^{b_2}, \ell^{b_2}, g^{b_2}, d^{b_2}, i^{b_2}, en^{b_2}, \tilde{c}^{b_2})$, L^{b_2} is dominated if $N^{b_1} = N^{b_2}$, $i^{b_1} = i^{b_2}$, $\ell^{b_1} \geq \ell^{b_2}$, $g^{b_1} \leq g^{b_2}$, $d^{b_1} \leq d^{b_2}$, $en^{b_1} \leq en^{b_2}$, $\tilde{c}^{b_1} \leq \tilde{c}^{b_2}$.

2.9 Computational Results

In this section, we report a summary of the computational results achieved by the ESF on the CMTVRPTW and its four variants, and, when available, we compare its performance with the state-of-the-art exact methods. Detailed computational results are reported in the e-companion of this paper. The ESF is coded in C and compiled with Microsoft Visual C++ compiler. Cplex 12.8 is used to solve the master problem in Step 2, in Step 4 and for the branch-and-cut in Step 6. For Step 2 and Step 4, we set Cplex to use the primal and the dual simplex method, respectively; for all the other parameters, we use the default setting. For Step 6, we disable the presolve phase and the cutting planes generation is embedded in the callback of the solver; all the others parameter are set to the default setting of Cplex. The tests were performed, in single thread mode, on a Windows server equipped with six Virtual CPU running at 2.59 GHz and with 16 GBs of RAM. A time limit of three hours per instance was set. All computing times are in seconds. The same parameter setting was used in all tests, namely, $col\_iter = 100$, $gap^0_{guess} = 0.05$, and $gap_{step} = 0.05$.

The first five subsections are devoted to the CMTVRPTW, CMTVRPTW-LD, CMTVRPTW-LT, CMTVRPTW-R, and DRP, respectively. In each subsection, we describe the instances first, and then provide the results achieved by our solution method. The following subsections are devoted to the comparison with the literature. Finally, the impact of time windows' width on the performance of ESF is investigated in Section 2.9.7.

Tables 2.1-2.6 summarize the results by group of instances, and the following information is indicated: group of instances (Group), number of customers (n), number of instances (Inst), number of instances solved (Solved), average gap between $z(\text{P1}_{\mathcal{S}})$ and the optimal solution cost ($\text{P1}_{\mathcal{S}}$%), average time to compute $z(\text{P1}_{\mathcal{S}})$ (T_{P1}), average cardinality of the set $\mathcal{S}_1$ ($|\mathcal{S}_1|$), average gap between $z(\text{P2}_{\mathcal{S}_1})$ and the optimal solution cost ($\text{P2}_{\mathcal{S}_1}$%), average cardinality of the set $\mathcal{S}_2$ ($|\mathcal{S}_2|$), average gap between the root node relaxation of $z(\text{F}_{\mathcal{S}_2})$ and the optimal solution cost ($\text{LF}_{\mathcal{S}}$%), average number of SFC added (nSFC), average number of nodes explored (Nds), average time spent to execute Step 6 ($\text{T}_{\text{B\&C}}$), and average total computing time (T_{tot}). Notice that, as Steps 3-7 may be iterated because of the conditions in Step 7, all information about Steps 3-7 refer to the last iteration. Table 2.3 has an additional column ($\bar{d}$) reporting the maximum trip duration, and Table 2.4 has an additional column (κ) indicating the type of release date (κ).

The last row of each table indicates the total number of instances, the total number of instances solved, and average values of each column computed over the instances solved only.

In the literature, different authors use different ways to compute travel times and costs even when solving the same set of instances. Indeed, travel times and costs are either truncated or rounded to the first, second, or to another decimal digit. The computational results in the literature show that this does not affect the complexity of the problem. Therefore, in the experiments reported in Tables 2.1-2.6, travel time and costs are truncated to one decimal digit as done in Solomon (1987), who introduced the test instances used (or extended) by most papers on vehicle routing.

All test instances used in this section are available as supplementary material to this paper.

2.9.1 Computational Results on the CMTVRPTW

The benchmark set consists of 81 instances, derived from the Type 2 Solomon instances by using a procedure similar to Hernandez et al. (2014, 2016). We consider Type 2 instances only because the short planning horizon and the tight time windows of instances of Type 1 prevent the vehicles from performing multiple trips. There are three groups of instances (C, R, RC) differing in the customer distribution (i.e., clustered, random, and randomly-clustered, respectively). We consider instances with 25, 40, and 50 customers by selecting the first customers from the original Solomon instances. Instances with 25 customers feature two vehicles, whereas instances with 40 and 50 customers feature four vehicles. The vehicle capacity is set equal to 100.

Table 2.1: Summary of the computational results for the CMTVRPTW

Group	n	Inst	Solved	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	8	8	2.37	11.1	75,662	0.94	6,075	0.69	0	88	1.0	16.1
C	40	8	6	2.71	22.0	1,054,466	1.73	207,049	1.71	0	22,257	1,589.5	2,589.9
C	50	8	2	3.50	21.0	1,820,115	0.95	75,645	0.99	0	1,190	90.3	1,601.1
R	25	11	11	7.76	9.4	1,338,633	1.24	5,018	0.77	1	35	1.3	83.1
R	40	11	10	2.53	21.1	2,777,514	0.36	31,561	0.42	0	17	2.7	377.7
R	50	11	0										
RC	25	8	8	12.88	4.0	124,067	2.14	4,143	1.75	5,695	4,036	70.9	175.7
RC	40	8	8	11.95	9.4	3,935,229	0.45	97,816	0.66	304	4,241	479.7	1,223.7
RC	50	8	7	5.95	23.3	1,966,892	0.61	14,038	0.61	1,011	9,886	279.8	647.6
All		81	60	6.55	14.1	1,655,251	1.03	45,449	0.90	918	4,543	268.8	654.9

Table 2.1 summarizes computational results on the CMTVRPTW. It shows that the ESF can solve all 25-customer instances and all but three 40-customer instances. Instances with 50 customers represent the limit of the ESF as only nine out of 27 instances are solved. Overall, 60 of the 81 instances are solved in an average computing time of 11 minutes.

2.9.2 Computational Results on the CMTVRPTW-LT

Hernandez et al. (2016) introduce 27 instances of the CMTVRPTW-LT, generated by considering the first 25 customers of the Solomon instances of Type 2. The fleet size is equal to two, and vehicle capacity equals to 100. The loading time of each customer $i \in N$ is computed as $lt_i = 0.2st_i$. Following the

procedure of Hernandez et al. (2016), we introduce 54 additional instances with 40 and 50 customers. We set capacity and loading times as in Hernandez et al. (2016), but we set the fleet size equal to four.

Table 2.2: Summary of the computational results for the CMTVRPTW-LT

Group	n	Inst	Solved	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	8	8	3.08	13.3	73,298	1.35	5,575	0.73	1	71	1.0	19.0
C	40	8	7	2.71	29.4	1,323,350	1.55	190,787	1.51	0	10,877	933.9	2,170.0
C	50	8	3	3.99	35.0	2,429,725	1.35	78,585	1.41	3	5,980	494.7	3,576.6
R	25	11	11	8.00	11.2	1,483,675	0.91	2,992	0.78	2	27	0.8	114.8
R	40	11	10	2.74	22.5	2,821,178	0.31	27,430	0.41	0	7	2.6	417.9
R	50	11	0										
RC	25	8	8	13.39	4.7	117,308	2.30	4,658	1.91	7,614	10,379	733.9	879.7
RC	40	8	8	12.04	9.5	3,677,170	0.46	73,755	0.83	299	2,745	186.3	871.9
RC	50	8	7	6.37	17.6	1,965,379	0.60	7,269	0.59	843	2,936	26.5	311.7
All		81	62	6.76	16.2	1,706,204	1.05	41,956	0.96	1,117	3,557	251.8	769.5

Table 2.2 summarizes the results on the 81 CMTVRPTW-LT instances. The ESF can solve all original 25-customer instances and all but two 40-customer instances. Ten of the 27 50-customer instances can be solved. All in all, the computational behaviour of the ESF is similar to that observed on the CMTVRPTW in Table 2.1.

2.9.3 Computational Results on the CMTVRPTW-LD

For the CMTVRPTW-LD, we use a set of instances based on the Solomon instances, with 25, 40, and 50 customers of groups C2, R2 and RC2, considering a shorter limited trip duration ($\overline{d} = \overline{d}_{short}$) and a longer limited trip duration ($\overline{d} = \overline{d}_{long}$), for a total of 162 instances. As done in Azi et al. (2010), we set $\overline{d}_{short} = 75$ and $\overline{d}_{long} = 100$ for instances of type R2 and RC2, and we set $\overline{d}_{short} = 220$ and $\overline{d}_{long} = 250$ for instances type C2. Travel time, distances, fleet size and loading times are set as in the CMTVRPTW-LT instances of Table 2.2.

Table 2.3 shows that the ESF can solve all 162 instances of the dataset with an average computing time of 16.8 seconds.

2.9.4 Computational Results on the CMTVRPTW-R

The only CMTVRPTW-R instances available in the literature are provided by Cattaruzza et al. (2016*a*), who generated a test bed based on the Solomon instances. In particular, for each instance, they consider four types of release date (for details see Cattaruzza et al. (2016*a*)) and set the capacity equal to half of the original value. Nonetheless, all instances of this test bed feature 100 customers, which is out of reach for our method. Therefore, we generate a set of 210 instances starting from the instances of Cattaruzza et al. (2016*a*) by considering the first 25, 40 and 50 customers only and by considering three types of tighter and tighter release dates (0.25, 0.5, 0.75). As for the CMTVRPTW, instances with 25 customers feature two vehicles, and instances with 40 and 50 customers four vehicles.

Table 2.4 summarizes the results by instance group, number of customers, and type of release date. The ESF can solve 154 of the 210 instances. The complexity of the problem clearly increases with the number of customers. Instances of Type R are more challenging than instances of Type C and RC. The

Table 2.3: Summary of the computational results for the CMTVRPTW-LD

Group	n	$\overline{d}$	Inst	Solved	P1$_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	P2$_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	LF$_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	220	8	8	5.43	11.2	454	1.47	179	0.11	0	0	0.1	11.4
C	25	250	8	8	2.06	16.3	1,842	0.42	528	0.65	1	45	0.2	16.6
C	40	220	8	8	4.16	25.6	1,161	0.15	397	0.02	0	1	2.1	27.9
C	40	250	8	8	1.24	41.0	5,325	0.50	3,042	0.36	0	1,422	2.8	44.1
C	50	220	8	8	4.38	40.0	1,859	0.51	1,120	0.07	0	0	9.8	50.0
C	50	250	8	8	1.27	65.7	8,977	0.50	5,632	0.37	0	1,224	6.9	73.1
R	25	75	11	11	3.15	1.7	609	0.10	91	0.32	0	0	0.0	1.8
R	25	100	11	11	6.11	2.7	2,886	0.23	273	0.18	0	0	0.0	2.9
R	40	75	11	11	4.70	4.0	3,412	0.50	794	0.16	0	1	0.2	4.3
R	40	100	11	11	5.94	8.1	26,682	0.36	2,972	0.35	0	36	0.1	8.8
R	50	75	11	11	5.06	6.0	5,393	0.44	1,900	0.20	4	42	1.8	8.1
R	50	100	11	11	6.22	12.7	52,811	0.24	7,940	0.24	335	1,507	16.0	31.8
RC	25	75	8	8	15.70	1.4	507	4.52	222	0.22	16	31	0.1	1.8
RC	25	100	8	8	18.93	2.5	2,733	0.69	553	0.75	15	16	0.1	3.0
RC	40	75	8	8	11.86	2.6	631	3.28	356	0.42	0	1	0.7	3.6
RC	40	100	8	8	18.02	6.5	3,614	0.65	1,263	0.18	1	8	0.1	7.1
RC	50	75	8	8	11.04	4.2	1,295	2.64	774	0.47	163	406	5.7	10.1
RC	50	100	8	8	18.20	10.4	10,795	1.74	5,074	0.08	0	1	0.4	12.3
All			162	162	7.66	13.6	8,168	0.97	1,894	0.28	33	263	2.67	16.8

bottleneck of the ESF is clearly Step 3, i.e., the structure enumeration. Indeed, instances cannot be solved when it is impossible to enumerate all structures, whereas instances can be solved, on average, in less than two minutes when all structures can be enumerated.

2.9.5 Computational Results on the DRP

We tested the ESF on the two sets (i.e., A and B) of DRP instances proposed by Cheng et al. (2018). Set A is created according to the framework presented in Solomon (1987) and Dorling et al. (2017); set B extend the Solomon instances to take into account the features of the drones. Type A instances consists of 85 instances divided in two groups (A1 and A2), which differ in the depot location; each group contains five instances with 10, 15, etc up to 50 customers. Type B instances are generated by considering the first 25 and 40 customers of the Type 2 (C2, R2, RC2) Solomon instances, for a total of 54 instances. For further details about the instances, we refer the reader to Cheng et al. (2018).

Table 2.5 summarizes the results on DRP instances of Type A. Results are grouped by instance group and number of customers. Table 2.5 shows that all 85 instances can be solved with an average computing time of 8.4 seconds.

Table 2.6 summarizes the results on the DRP instances of Type B. Results are group by customer distribution (C, R, RC) and number of customers (25, 40). Table 2.6 shows that all 54 can be solved with an average computing time of 2.1 seconds.

2.9.6 Comparison with the Literature

In this section, we compare the performance of the ESF with the state-of-the-art exact methods from the literature. In particular, we compare the ESF with the methods of Hernandez et al. (2016) (hereafter Hern16) on the CMTVRPTW-LT, of Hernandez et al. (2014) (hereafter Hern14) on the CMTVRPTW-

Table 2.4: Summary of the computational results for CMTVRPTW-R

Group	n	κ	Inst	Solved	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	0.25	8	8	9.24	8.6	740,684	2.69	6,722	0.64	16	216	3.6	30.4
C	25	0.5	8	8	9.25	9.0	752,587	2.70	6,515	0.64	25	463	7.3	35.1
C	25	0.75	8	8	9.35	8.9	677,542	2.74	6,666	0.67	20	629	9.6	39.3
C	40	0.25	8	6	4.33	20.3	586,585	1.00	23,165	0.89	0	921	13.9	75.3
C	40	0.5	8	6	5.39	20.3	541,465	1.12	21,346	0.96	0	1,417	13.3	72.3
C	40	0.75	8	6	7.05	18.4	595,428	0.92	27,052	0.64	0	593	11.3	73.9
C	50	0.25	8	5	4.49	34.8	459,779	0.72	3,024	0.56	0	975	1.8	59.7
C	50	0.5	8	5	4.86	35.9	534,103	0.61	13,925	0.44	0	361	1.8	104.3
C	50	0.75	8	5	7.13	35.1	729,090	0.26	2,129	0.22	0	4	0.2	86.2
R	25	0.25	11	9	10.77	7.0	1,379,650	0.15	96	0.19	0	0	0.0	18.4
R	25	0.5	11	9	11.42	6.7	1,602,670	0.13	518	0.33	0	0	0.0	20.8
R	25	0.75	11	8	13.82	5.6	1,042,126	0.14	443	0.06	0	0	0.0	25.5
R	40	0.25	11	4	7.64	12.8	5,238,404	0.02	2,761	0.14	0	0	0.1	105.1
R	40	0.5	11	4	7.87	13.4	3,924,628	0.04	1,049	0.21	0	0	0.0	78.1
R	40	0.75	11	2	11.37	10.3	2,280,014	0.00	1,400	0.04	0	0	0.0	48.0
R	50	0.25	11	0										
R	50	0.5	11	0										
R	50	0.75	11	0										
RC	25	0.25	8	8	16.12	3.3	120,983	1.28	3,915	0.98	496	192	1.0	12.1
RC	25	0.5	8	8	16.19	3.1	118,613	1.18	4,287	0.91	443	258	0.9	10.8
RC	25	0.75	8	8	19.92	2.7	112,202	0.41	1,148	0.25	4	4	0.1	5.7
RC	40	0.25	8	7	16.34	8.2	1,750,114	0.44	19,208	0.65	13	9,115	108.1	217.5
RC	40	0.5	8	7	16.41	7.3	1,547,847	0.21	13,625	0.43	336	2,118	42.4	141.6
RC	40	0.75	8	7	21.24	6.9	1,096,531	0.46	2,898	0.47	0	935	1.4	59.7
RC	50	0.25	8	6	10.65	18.3	1,720,962	0.95	9,708	0.64	6,983	16,660	1,038.5	1,221.4
RC	50	0.5	8	5	13.25	17.0	1,768,256	0.43	9,591	0.20	1	5	0.6	173.4
RC	50	0.75	8	5	16.53	16.0	1,875,569	0.59	15,468	0.32	34	155	6.1	271.4
All			210	154	11.21	13.4	1,409,377	0.74	7,581	0.45	319	1334	48.1	118.0

LD, and of Cheng et al. (2018) (hereafter Cheng18) on the DRP. Notice that these three methods have been designed and tailored on the problem at hand and have not been generalized to several CMTVRPTW as our proposed ESF.

Comparison with Hernandez et al. (2016) on the CMTVRPTW-LT

Table 2.7 compares the performance of EFS and Hern16 on the 27 25-customer CMTVRPTW-LT instances proposed by Hernandez et al. (2016). In particular, as Hernandez et al. (2016) propose two exact branch-and-price algorithms, the comparison is done with the better of the two, which relies on a trip-based formulation. The branch-and-price of Hernandez et al. (2016) is run on a Intel Core i7 2670QM with 8 GBs of RAM. A dash indicates that the instance was not solved.

Instances are grouped by customer distribution (C2, R2, RC2). For each instance, we report the instance number (Inst); the best-known upper bound (UB); for ESF, the gap (Gap) between $LF_{\mathcal{S}}$ and the optimal solution cost, and the total computing time (Time), for Hern16, the gap (Gap) between the linear relaxation of the corresponding trip-based formulation and the optimal solution cost, and the total computing time (Time).

Table 2.7 shows the ESF can solve all 27 instances, whereas Hern16 cannot solve instances RC204 and RC208. Moreover, the ESF is on average several times faster than Hern16, mainly because of the

Table 2.5: Summary of the computational results for the DRP (Type A instances)

Group	n	Inst	Solved	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
A1	10	5	5	11.07	0.2	105	3.43	52	0.59	6	4	0.1	0.5
A1	15	5	5	13.06	0.5	887	2.17	175	1.26	3	110	0.1	0.9
A1	20	5	5	8.69	0.5	930	1.05	229	0.60	58	163	1.2	1.9
A1	25	5	5	9.32	1.2	3,355	1.53	413	1.02	92	333	1.7	3.8
A1	30	5	5	8.25	1.3	9,884	0.83	1,091	0.77	140	323	1.5	4.4
A1	35	5	5	8.53	2.5	19,224	0.59	6,725	0.57	7	1,231	15.4	18.7
A1	40	5	5	8.07	2.8	16,630	0.35	1,525	0.26	71	174	4.3	7.5
A1	45	5	5	8.29	5.6	58,018	0.52	4,139	0.43	90	421	10.1	18.9
A2	10	5	5	11.15	0.1	54	1.92	22	0.21	3	0	0.0	0.4
A2	15	5	5	13.22	0.2	120	2.02	50	1.30	42	25	0.5	1.6
A2	20	5	5	11.28	0.4	432	1.72	125	1.12	73	120	1.2	2.0
A2	25	5	5	9.74	0.7	498	0.91	110	1.06	63	123	1.6	2.5
A2	30	5	5	10.48	1.6	1,798	0.73	578	0.26	25	45	1.0	2.9
A2	35	5	5	10.18	2.3	2,759	0.65	1,089	0.32	5	5	1.4	4.1
A2	40	5	5	9.17	3.3	3,648	0.51	1,279	0.25	89	163	5.8	9.4
A2	45	5	5	9.23	3.3	6,654	0.45	787	0.29	40	85	4.2	7.8
A2	50	5	5	8.82	5.8	6,706	0.53	1,905	0.34	239	573	48.8	54.9
All		85	85	9.91	1.9	7,747	1.17	1,194	0.63	62	229	5.8	8.4

Table 2.6: Summary of the computational results for DRP (Type B instances)

Group	n	Inst	Solved	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C	25	8	8	8.97	0.5	10,377	1.98	577	1.32	0	568	0.4	1.2
C	40	8	8	7.54	1.4	63,146	0.77	6,989	0.44	3	203	1.7	4.9
R	25	11	11	7.11	0.3	5,305	0.02	200	0.13	0	0	0.0	0.5
R	40	11	11	7.04	0.8	43,020	0.03	2,889	0.03	0	0	0.5	1.9
RC	25	8	8	12.46	0.1	1,232	2.20	356	0.07	7	17	0.1	0.5
RC	40	8	8	10.51	0.5	2,105	1.39	1,503	0.14	200	538	3.4	4.2
All		54	54	8.73	0.6	21,231	0.95	2,026	0.32	31	196	0.9	2.1

smaller gaps provided by the lower bounding procedure.

Comparison with Hernandez et al. (2014) on the CMTVRPTW-LD

Tables 2.8 and 2.9 compare the results achieved by the ESF and by the branch-and-price of Hernandez et al. (2014) on the instances having a feasible solution used by Hernandez et al. (2014) and introduced in Azi et al. (2010). The branch-and-price of Hernandez et al. (2014) was run on an Intel Core 2 Duo 2.10 GHz with 2 GBs of RAM. Table 2.8 reports the results on 27 instances with 25 customers, each one solved with a shorter and a long limited trip duration. Table 2.9 reports the results on 18 instances with 40 customers, each one solved with a shorter and a long limited trip duration. Columns of these two tables have the same meaning of the columns of Table 2.7.

Table 2.8 shows that the ESF can solve all 25-customer instances, whereas Hern14 cannot solve five of them. Moreover, the ESF is, on average, much faster than Hern14 on both instances with short and long limited duration, in particular on the latter ones.

Table 2.9 indicates that the ESF can solve all 35 instances, which is ten more than Hern14. The ESF

Table 2.7: Comparison with Hernandez et al. (2016) on the CMTVRPTW-LT

C2						R2						RC2					
		ESF		Hern16				ESF		Hern16				ESF		Hern16	
Inst	UB	Gap	Time	Gap	Time	Inst	UB	Gap	Time	Gap	Time	Inst	UB	Gap	Time	Gap	Time
1	380.8	0.45	8.7	3.46	7.2	1	554.6	0.71	154.0	3.19	13.8	1	660.0	0.19	11.0	3.52	5.9
2	368.6	0.11	19.2	1.71	245.7	2	485.0	0.81	482.1	2.49	53.2	2	596.8	9.91	6,891.4	8.98	19,737.7
3	361.7	0.48	23.7	0.55	37.6	3	444.2	0.10	108.9	2.00	198.2	3	530.1	1.82	66.0	7.16	11,904.5
4	358.8	0.11	45.0	0.76	180.7	4	407.5	2.02	158.9	4.70	2,242.8	4	518.0	1.34	12.0	-	-
5	377.2	2.84	11.2	4.60	65.4	5	448.4	0.00	37.1	1.19	13.3	5	605.3	0.02	22.7	3.84	86.3
6	367.2	1.53	13.1	2.87	56.7	6	413.9	0.00	43.3	0.44	69.7	6	575.1	0.78	13.7	5.41	526.4
7	359.1	0.04	17.6	1.76	154.4	7	400.1	2.62	18.4	2.53	842.4	7	528.2	1.13	12.3	3.54	7,764.0
8	360.9	0.27	13.7	2.20	112.6	8	394.3	0.09	21.5	3.47	1,049.0	8	506.4	0.07	8.2	-	-
						9	418.3	1.26	48.6	1.54	62.0						
						10	448.3	0.07	124.2	1.83	79.2						
						11	400.1	0.94	65.6	3.14	2,481.7						
Avg		0.73	19.0	2.24	107.5			0.78	114.8	2.41	645.9			1.91	1,169.5	5.41	6,670.8
8 inst.		Solved: 8		Solved: 8		11 inst.		Solved: 11		Solved: 11		8 inst.		Solved: 8		Solved: 6	

is more efficient in particular on instances with longer limited duration. Also in terms of computing time, the ESF is clearly superior.

Comparison with Cheng et al. (2018) on the DRP

Table 2.10 compares the results achieved by the ESF and by the branch–and–cut of Cheng et al. (2018) based on the mathematical formulation without a drone index, referred to as $(R+E)_e\,|nk$ in the paper of Cheng et al. (2018). The branch–and–cut of Cheng et al. (2018) is run on a cluster of Intel Xeon X5650 CPUs with 2.67 GHz and 24 GB of RAM under Linux 6.3.

For each instance type (Type), we report the instance group (Group), the number of customers (n) and the number of instances (Inst). Columns under heading ESF report the number of instances solved (Solved), the average gap (Gap) between $\mathsf{LF}_{\mathcal{S}}$% and the optimal solution cost, and the average computing time (Time) of the ESF. Columns under heading Cheng18 report the number of instances solved (Solved), the gap (Gap) between the root node relaxation of $(R+E)_e\,|nk$ and the optimal solution cost, and the average computing time (Time) of Cheng18. The average values reported in columns Gap and Time are computed over the instances solved only.

Table 2.10 shows that the ESF can solve all 139 instances, whereas Cheng18 can solve 45 instances only. Moreover, the ESF is on average much faster than Cheng18, thanks to the smaller gaps provided by the lower bounding procedure.

2.9.7 Impact of the Width of the Time Windows

In this section, we investigate the impact of time windows width on the performance of ESF. We test ESF on the CMTVRPTW instances studied in Section 2.9.1 by enlarging the original time windows. For each of the 81 instances, we consider three additional instances: the first featuring time windows that are 50%

Table 2.8: Comparison with Hernandez et al. (2014) on the CMTVRPTW-LD (instances with 25 customers)

	Short Limited Duration					Long Limited Duration				
		ESF		Hern14			ESF		Hern14	
Inst	UB	Gap	Time	Gap	Time	UB	Gap	Time	Gap	Time
C201	659.02	0.00	5.3	1.90	1.3	540.90	0.00	3.6	0.00	0.1
C202	653.37	0.00	14.2	2.85	49.3	533.43	0.00	18.6	1.54	51.4
C203	646.40	0.22	20.1	3.15	265	532.77	0.21	25.6	1.70	335.7
C204	602.46	0.01	28.7	1.73	248	525.46	0.86	35.6	2.29	4734.4
C205	636.39	0.00	7.5	4.48	38.1	529.94	1.49	9.4	1.49	0.9
C206	636.39	0.00	8.3	5.19	692.4	527.84	0.04	12.7	2.21	123.9
C207	603.22	0.00	10.2	2.39	104.7	525.46	1.86	18.4	1.86	31.1
C208	613.20	0.00	9.6	2.59	41.4	525.46	1.86	14.6	1.86	4.7
R201	762.43	0.61	1.2	0.61	0.1	698.18	0.00	1.3	0.99	0.8
R202	645.78	0.00	1.6	0.00	0.6	617.53	0.00	2.9	0.14	4.1
R203	621.97	0.29	2.4	0.16	2.0	577.74	0.00	4.6	0.11	11.6
R204	579.68	0.32	2.9	0.70	4.9	483.3	0.00	4.0	0.54	33.6
R205	634.09	1.12	1.2	1.20	1.0	559.14	0.03	2.2	0.64	3.7
R206	596.74	0.00	1.9	0.00	0.8	523.64	0.00	3.3	0.00	5.7
R207	585.74	0.00	2.7	0.34	3.5	512	0.73	4.1	2.85	418.9
R208	579.68	0.32	3.0	0.70	7.2	483.3	0.49	4.2	1.30	97.8
R209	602.39	0.62	1.7	0.71	1.7	517.69	0.00	2.9	1.11	14.1
R210	636.15	0.00	1.6	2.49	8.5	547.23	0.00	2.8	0.00	2.6
R211	575.91	0.23	2.0	1.28	27.6	474.49	0.00	4.1	0.93	80.4
RC201	988.05	0.15	1.2	0.37	0.9	849.33	2.79	14.3	4.15	3.6
RC202	881.49	0.00	2.2	4.98	24.5	679.86	0.00	3.0	0.00	3.6
RC203	749.15	0.00	2.3	5.87	62.3	593.56	0.07	4.5	0.10	13.1
RC204	744.72	0.00	3.2	-	-	587.22	0.01	5.0	-	-
RC205	840.35	0.45	2.0	3.78	3.7	702.49	0.54	2.5	0.54	2.6
RC206	761.03	0.08	1.5	4.59	35.7	604.12	0.00	1.9	0.65	2.9
RC207	738.87	0.35	2.1	-	-	514.81	0.00	3.2	0.84	45.5
RC208	727.99	5.85	2.8	-	-	502.18	0.00	4.6	-	-
Avg		0.39	5.3	2.17	67.7		0.41	7.9	1.11	241.1
	27 inst.	Solved: 27		Solved: 24		27 inst.	Solved: 27		Solved: 25	

larger, the second featuring time windows twice as large as the initial one, and the third without time windows. In particular, time windows $[a'_i, b'_i]$, $i \in N$, of these new instances are defined as

$$a'_i = \lceil \max\{a_0 + t_{0i}, a_i - (b_i - a_i)\delta\} \rceil \qquad i \in N$$
$$b'_i = \lfloor \min\{b_0 - t_{i0} - st_i, b_i + (b_i - a_i)\delta\} \rfloor \qquad i \in N$$

and parameter δ is set equal to 0.25, 0.5, and ∞ to obtain the three instances. Notice that the time window of the depot remains unchanged.

Table 2.11 compares the results of ESF on the original instances (i.e., with $\delta = 0$) with the results on the instances obtained by setting $\delta = 0.25$, 0.50, and ∞. We report the instances group (Group), the number of customers (n), and the number of instances (Inst). For each value of δ, we indicate the number of instances solved (Solved), the average gaps between the lower bounds and the optimal solution costs ($P1_{\mathcal{S}}$%, $P2_{\mathcal{S}_1}$%, $LF_{\mathcal{S}}$%), and the average total computing time (T_{tot}).

Table 2.9: Comparison with Hernandez et al. (2014) on the CMTVRPTW-LD (instances with 40 customers)

	Short Limited Duration					Long Limited Duration				
		ESF		Hern14			ESF		Hern14	
Inst	UB	Gap	Time	Gap	Time	UB	Gap	Time	Gap	Time
C201	1,168.83	1.95	1,068.5	3.79	31.3	966.70	2.38	194.9	3.32	90.4
C202	1,111.15	0.10	218.1	1.19	67.4	919.85	0.00	50.3	0.63	84.2
C203	1,088.55	0.00	43.0	1.04	186.5	915.04	1.27	81.1	-	-
C204	1,039.16	0.00	59.9	0.45	145.3	908.24	1.13	115.9	-	-
C205	1,083.81	0.00	14.0	0.64	34.1	921.19	0.00	19.6	1.02	66.3
C206	1,081.37	0.00	16.2	0.67	184	919.05	0.25	131.3	0.87	1,539.1
C207	1,055.04	0.12	307.3	1.16	1,491.5	910.43	0.54	46.3	-	-
C208	1,071.99	0.00	22.3	0.75	52.6	915.41	0.00	31.0	0.73	2,673.7
R202			infeasible			961.33	0.38	17.0	-	-
R203	962.22	0.49	7.0	-	-	816.51	0.34	67.9	1.07	15.5
R204	858.22	0.00	6.7	2.14	4,049.2	708.98	0.76	13.9	-	-
R205	1,017.84	2.98	684.3	2.48	1,193.4	873.22	3.68	2313	2.25	2,429.0
R206	927.22	1.05	12.3	0.28	171.5	812.31	0.46	14.5	1.11	926.4
R207	886.22	0.00	5.6	0.39	68.9	764.39	0.27	11.5	-	-
R208	858.22	0.66	6.7	2.14	4,954.8	708.01	0.95	14.5	-	-
R209	935.81	0.02	4.0	1.00	198.2	768.84	0.44	15.3	1.12	1,511.4
R210	952.92	0.83	272.3	0.98	246.5	822.78	1.50	115.5	-	-
R211	869.75	2.32	54.5	2.18	5,093.9	728.94	3.06	549.0	-	-
RC204	1,362.34	0.00	8.2	-	-	985.98	1.34	13.0	-	-
Avg		0.58	156.2	1.33	1,135.6		0.99	200.8	1.35	1,037.3
	17 inst.	Solved: 17		Solved: 16		18 inst.	Solved: 18		Solved: 9	

Table 2.10: Comparison with Cheng et al. (2018) on the DRP

				ESF			Cheng18		
Type	Group	n	Inst	Solved	Gap	Time	Solved	Gap	Time
A	1	10	5	5	0.59	0.5	5	4.07	0.6
A	1	15	5	5	1.26	0.9	1	7.92	22.5
A	1	20	5	5	0.60	1.9	2	1.28	13.6
A	1	25	5	5	1.02	3.8	0	-	-
A	1	30	5	5	0.77	4.4	0	-	-
A	1	35	5	5	0.57	18.7	0	-	-
A	1	40	5	5	0.26	7.5	0	-	-
A	1	45	5	5	0.43	18.9	0	-	-
A	2	10	5	5	0.21	0.4	5	4.92	0.3
A	2	15	5	5	1.30	1.6	3	4.65	4.2
A	2	20	5	5	1.12	2.0	3	5.29	21.5
A	2	25	5	5	1.06	2.5	0	-	-
A	2	30	5	5	0.26	2.9	0	-	-
A	2	35	5	5	0.32	4.1	0	-	-
A	2	40	5	5	0.25	9.4	0	-	-
A	2	45	5	5	0.29	7.8	0	-	-
A	2	50	5	5	0.34	54.9	0	-	-
B	C	25	8	8	1.32	1.2	8	1.85	35.5
B	C	40	8	8	0.44	4.9	1	1.64	43.8
B	R	25	11	11	0.13	0.5	11	2.82	66.6
B	R	40	11	11	0.03	1.9	3	4.56	5,543.3
B	RC	25	8	8	0.07	0.5	3	4.42	95.2
B	RC	40	8	8	0.14	4.2	0	-	-
Avg					0.51	5.9		3.55	402.4
Solved			139	139			45		

Table 2.11: Summary of the computational results with time windows of increasing width

			$\delta = 0$					$\delta = 0.25$					$\delta = 0.50$					$\delta = \infty$				
Group	n	Inst	Solved	P1$_{\mathcal{S}}$%	P2$_{\mathcal{S}_1}$%	LF$_{\mathcal{S}}$%	T$_{\text{tot}}$	Solved	P1$_{\mathcal{S}}$%	P2$_{\mathcal{S}_1}$%	LF$_{\mathcal{S}}$%	T$_{\text{tot}}$	Solved	P1$_{\mathcal{S}}$%	P2$_{\mathcal{S}_1}$%	LF$_{\mathcal{S}}$%	T$_{\text{tot}}$	Solved	P1$_{\mathcal{S}}$%	P2$_{\mathcal{S}_1}$%	LF$_{\mathcal{S}}$%	T$_{\text{tot}}$
C	25	8	8	2.37	0.94	0.69	16.1	8	2.36	1.16	0.38	13.8	8	1.93	1.00	0.28	14.1	8	1.81	1.15	0.89	13.9
C	40	8	6	2.71	1.73	1.71	2589.9	8	2.59	1.65	1.28	428.7	8	2.49	1.65	1.41	289.0	8	2.39	1.48	1.44	174.0
C	50	8	2	3.50	0.95	0.99	1601.1	7	2.76	0.97	0.93	1362.5	8	2.70	1.02	0.97	1242.8	8	2.67	0.88	0.84	1507.6
R	25	11	11	7.76	1.24	0.77	83.1	11	5.38	1.27	0.35	28.5	11	5.49	1.53	0.90	78.0	11	5.16	1.88	0.00	16.8
R	40	11	10	2.53	0.36	0.42	377.7	11	2.07	0.09	0.06	69.9	11	2.28	0.23	0.16	83.0	11	1.65	0.00	0.00	39.4
R	50	11	0					4	3.11	0.94	0.92	480.0	2	2.86	0.86	0.86	279.3	0				
RC	25	8	8	12.88	2.14	1.75	175.7	8	9.79	1.11	0.93	57.0	8	8.37	0.63	0.52	11.5	8	6.87	0.00	0.00	4.1
RC	40	8	8	11.95	0.45	0.66	1223.7	8	10.79	0.13	0.06	164.2	8	10.24	0.07	0.00	165.9	8	9.98	0.00	0.00	44.4
RC	50	8	7	5.95	0.61	0.61	647.6	8	4.76	0.43	0.24	256.3	7	3.07	0.09	0.05	29.9	8	2.64	0.00	0.00	24.7
Avg				6.55	1.03	0.90	654.9		4.88	0.84	0.52	272.6		4.48	0.80	0.55	229.9		4.08	0.70	0.36	211.0
Solved		81	60					73					71					70				

Table 2.11 shows that the performance of ESF does not deteriorate when time windows are enlarged. Indeed, ESF can solve 73 instances with $\delta = 0.25$, 71 with $\delta = 0.5$, and 70 instances with $\delta = \infty$, whereas 60 instances were solved with $\delta = 0$. Therefore, we cannot conclude that the width of the time windows has a consistent impact on the performance of ESF.

Nevertheless, we can observe a general trend in the quality of the three lower bounds: by increasing the width of the time windows, the three lower bounds are often on average better. This is probably why instances with larger time windows are not always more difficult to solve in spite of the larger the number of feasible structures involved, and the higher complexity of the pricing problems and enumeration phase.

2.10 Conclusions

We have presented an *Exact Solution Framework* (ESF) for Multi-Trip Vehicle Routing Problems with Time Windows (MTVRPTW). The ESF relies on a novel mathematical model with an exponential number of variables and constraints. Unlike the other mathematical models from the literature, the proposed model is based on the concept of *structure*. The novel formulation is used to derive two lower bounds that can be efficiently computed. These lower bounds are exploited in the ESF to generate a reduced set of columns containing any optimal MTVRP solution, which is then found by using a branch-and-cut. The computational results show that the ESF can efficiently solve instances with up to 50 customers of the CMTVRPTW significantly outperforming the state-of-the-art exact methods. We have also showed that the ESF can easily be adapted to solve four generalizations of the CMTVRPTW, and the corresponding computational results are significantly better than those achieved by the best exact methods from the literature.

As the bottleneck of the ESF is represented by the enumeration of all structures in the gap given the CVRPTW lower bound, future research can be directed toward studying valid inequalities to tighten such lower bound without affecting the complexity of the pricing problem.

Moreover, given the performance of the ESF on a well-studied class of vehicle routing problems, we consider it promising to explore the usage of similar mathematical models with an exponential number of variables and constraints for other combinatorial optimization problems.

Appendix

2.A Detailed Computational Results

Tables 2.A.1-2.A.15 report detailed computational results summarized in Tables 2.1-2.6 in Section 2.9. The following information is indicated:

- group of instances (Group);
- number of customers (n);
- the optimal solution cost (ub^*) if the instance is solved to optimality; otherwise, we report TL if the instance is not solved because the time limit is reached and ML if the instance is not solved because of memory limit during the structure enumeration at Step 3;
- gap between $z(\mathrm{P1}_{\mathcal{S}})$ and the optimal solution cost ($\mathrm{P1}_{\mathcal{S}}$%);
- time to compute $z(\mathrm{P1}_{\mathcal{S}})$ (T_{P1});
- cardinality of the set $\mathcal{S}_1$ ($|\mathcal{S}_1|$);
- gap between $z(\mathrm{P2}_{\mathcal{S}_1})$ and the optimal solution cost ($\mathrm{P2}_{\mathcal{S}_1}$%);
- cardinality of the set $\mathcal{S}_2$ ($|\mathcal{S}_2|$);
- gap between the root node relaxation of $z(\mathrm{F}_{\mathcal{S}_2})$ and the optimal solution cost ($\mathrm{LF}_{\mathcal{S}}$%);
- number of SFC added (nSFC);
- number of nodes explored (Nds);
- time spent to execute Step 6 ($\mathrm{T}_{\mathrm{B\&C}}$);
- total computing time ($\mathrm{T}_{\mathrm{tot}}$).

Notice that, as Steps 3-7 may be iterated because of the conditions in Step 7, all information about Steps 3-7 refer to the last iteration.

Tables 2.A.5-2.A.7 have an additional column ($\overline{d}$) reporting the maximum trip duration. Tables 2.A.8-2.A.12 have an additional column (κ) indicating the type of release date (κ). Tables 2.A.13-2.A.14 have and additional column (Num) to indicate the instance number.

Detailed computational results on the CMTVRPTW-R instances of type R with 50 customers are not reported because none of the instances was solved due to memory limit.

Table 2.A.1: Computational results on the CMTVRPTW: instances with 25 and 40 customers

Group	n	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	25	378.6	4.53	4.11	22,571	2.12	1,299	2.29	2	328	0.6	5.6
C202	25	363.0	1.67	12.88	74,759	0.00	2,782	0.00	1	0	0.2	14.5
C203	25	358.8	0.74	17.05	127,667	0.00	9,731	0.00	0	0	0.6	24.7
C204	25	358.8	1.15	21.77	234,785	0.32	11,460	0.00	0	0	0.5	38.2
C205	25	368.3	3.57	5.92	32,250	1.00	1,428	0.54	0	0	0.3	7.2
C206	25	367.2	3.70	7.58	30,270	2.13	4,184	1.46	0	212	2.0	10.8
C207	25	358.8	1.81	11.51	49,322	1.15	9,218	1.20	0	165	2.0	15.8
C208	25	359.1	1.76	8.29	33,671	0.81	8,497	0.00	0	0	1.5	12.0
R201	25	546.8	15.27	2.48	2,106,695	0.80	2,718	0.77	1	14	0.3	82.3
R202	25	482.8	14.04	7.29	5,137,202	2.35	814	1.64	0	90	0.2	212.2
R203	25	442.6	8.20	10.19	1,009,098	1.25	1,415	1.09	3	34	0.3	77.2
R204	25	404.9	6.49	12.68	1,264,793	2.08	29,615	2.09	0	187	11.1	257.3
R205	25	448.4	7.93	5.33	419,946	0.35	225	0.00	0	0	0.0	24.7
R206	25	413.9	5.76	8.07	882,241	0.00	2,774	0.00	0	0	0.1	32.9
R207	25	399.8	4.32	12.24	114,559	1.38	275	0.33	3	4	0.1	14.5
R208	25	394.3	4.21	15.63	142,971	1.88	458	0.00	3	0	0.1	18.8
R209	25	418.3	5.32	9.23	813,099	1.51	7,240	0.65	2	2	0.4	58.4
R210	25	448.3	7.43	8.70	1,474,542	0.62	1,744	0.50	0	21	0.3	67.4
R211	25	399.9	6.40	11.62	1,359,822	1.45	7,921	1.38	0	28	1.6	68.7
RC201	25	660.0	22.41	1.70	91,312	2.45	1,619	1.21	22	16	0.5	10.1
RC202	25	576.1	13.66	3.52	126,410	6.79	5,660	6.67	40,954	27,210	541.7	1,299.9
RC203	25	528.8	9.29	5.43	64,325	2.21	751	1.82	4,019	4,093	19.8	26.6
RC204	25	516.5	8.11	4.95	90,740	1.18	1,783	0.77	130	403	0.7	7.4
RC205	25	603.0	17.99	3.15	310,630	1.01	16,287	0.19	2	0	2.0	33.6
RC206	25	575.1	14.87	3.43	130,332	2.14	5,326	1.92	34	59	1.3	12.5
RC207	25	526.4	9.69	4.11	69,675	1.18	552	1.34	297	315	0.5	7.8
RC208	25	506.4	7.00	5.81	109,108	0.14	1,165	0.09	103	191	0.4	7.8
C201	40	625.6	3.13	9.07	550,331	1.92	99,651	1.92	0	20,302	990.3	1,281.3
C202	40	616.1	2.62	39.65	1,365,158	1.73	279,149	1.67	1	18,385	1,939.1	3,359.4
C203	40	TL										
C204	40	TL										
C205	40	621.0	3.63	11.27	724,811	2.22	75,327	2.22	0	43,178	1,354.7	1,597.5
C206	40	612.4	2.46	20.23	965,910	1.66	247,584	1.64	0	6,224	538.1	1,653.8
C207	40	609.9	2.30	26.17	1,523,940	1.48	277,402	1.46	0	36,824	3,910.4	6,281.7
C208	40	609.9	2.13	25.90	1,196,646	1.38	263,180	1.33	1	8,631	804.4	1,365.7
R201	40	ML										
R202	40	638.7	4.17	16.66	1,200,193	0.00	207	0.00	0	0	0.0	59.1
R203	40	604.7	4.25	22.78	1,709,463	0.89	3,126	0.89	0	72	1.2	122.4
R204	40	572.6	2.01	28.61	5,175,888	0.15	59,005	0.17	0	0	4.6	623.1
R205	40	632.9	3.59	11.61	1,097,579	0.65	3,654	0.65	0	39	0.7	55.8
R206	40	598.8	3.43	18.15	1,719,461	0.71	5,349	0.71	2	13	0.9	82.6
R207	40	577.9	2.18	27.30	3,169,326	0.44	70,633	0.45	0	7	7.5	1,021.0
R208	40	569.4	1.65	25.34	6,504,543	0.00	66,790	0.04	0	0	3.4	944.8
R209	40	587.0	0.75	19.31	1,618,310	0.00	34,716	0.53	0	0	0.3	255.9
R210	40	591.5	0.50	17.98	1,560,500	0.00	36,715	0.00	0	0	0.3	138.9
R211	40	576.1	2.74	23.01	4,019,873	0.78	35,410	0.73	0	40	8.0	473.9
RC201	40	1,014.7	17.42	5.27	1,089,795	1.32	787	0.53	1	11	0.2	82.1
RC202	40	907.3	10.25	8.63	1,846,497	0.18	17,323	0.16	10	8	2.1	195.9
RC203	40	887.7	10.39	10.28	5,052,645	0.70	128,836	0.70	2,325	3,805	265.0	1,152.3
RC204	40	865.4	9.66	11.73	6,462,921	0.00	136,611	0.26	0	0	13.7	973.2
RC205	40	940.6	13.26	8.03	1,326,179	0.34	381	0.23	57	80	0.2	82.6
RC206	40	937.9	13.92	7.04	6,051,753	1.10	309,904	2.02	40	26,299	3,242.4	5,001.3
RC207	40	881.8	10.71	11.64	2,451,305	0.00	30,206	0.08	1	0	2.4	405.9
RC208	40	864.4	9.98	12.96	7,200,736	0.00	158,483	1.30	0	3,726	311.8	1,896.7

Table 2.A.2: Computational results on the CMTVRPTW: instances with 50 customers

Group	n	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	50	704.4	3.22	15.66	1,431,060	0.88	88,755	0.89	0	226	45.7	1,254.9
C202	50	TL										
C203	50	ML										
C204	50	ML										
C205	50	698.5	3.77	26.28	2,209,169	1.01	62,534	1.09	0	2,153	134.9	1,947.3
C206	50	TL										
C207	50	TL										
C208	50	TL										
R201	50	ML										
R202	50	ML										
R203	50	ML										
R204	50	ML										
R205	50	ML										
R206	50	ML										
R207	50	ML										
R208	50	ML										
R209	50	ML										
R210	50	ML										
R211	50	ML										
RC201	50	1,086.2	11.78	11.14	3,792,194	0.00	1,693	0.00	0	0	0.1	484.7
RC202	50	981.2	5.83	20.20	2,390,290	0.08	9,971	0.49	0	0	0.1	384.5
RC203	50	941.2	3.98	25.43	589,850	0.58	5,497	0.51	8	17	0.3	54.4
RC204	50	915.9	2.55	32.66	893,681	0.00	9,802	0.00	0	0	0.5	66.0
RC205	50	ML										
RC206	50	1,027.4	10.39	17.17	4,485,983	2.66	57,015	2.33	0	56,537	1,823.0	3,273.4
RC207	50	941.7	4.18	25.45	504,678	0.59	2,090	0.58	2,315	2,707	15.5	61.1
RC208	50	915.0	2.97	31.30	1,111,570	0.34	12,195	0.34	4,753	9,942	119.4	209.3

Table 2.A.3: Computational results on the CMTVRPTW-LT: instances with 25 and 40 customers

Group	n	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	25	380.8	5.08	5.4	27,752	2.65	1,148	0.45	0	0	0.1	8.7
C202	25	368.6	3.16	16.1	49,950	0.75	3,498	0.11	2	0	0.8	19.2
C203	25	361.7	1.53	20.0	124,419	0.00	7,172	0.48	0	0	0.2	23.7
C204	25	358.8	1.15	25.7	234,785	0.23	11,471	0.11	0	0	1.2	45.0
C205	25	377.2	5.84	6.0	35,354	3.14	1,620	2.84	1	272	0.8	11.2
C206	25	367.2	3.70	8.8	31,191	1.86	3,102	1.53	0	294	2.7	13.1
C207	25	359.1	1.90	14.1	49,258	1.12	9,619	0.04	0	0	1.0	17.6
C208	25	360.9	2.25	10.3	33,671	1.03	6,970	0.27	2	0	0.9	13.7
R201	25	554.6	16.46	3.2	2,106,695	0.74	728	0.71	9	17	0.3	154.0
R202	25	485.0	14.43	8.8	6,429,418	1.25	689	0.81	2	95	0.4	482.1
R203	25	444.2	8.53	13.5	1,202,841	0.41	432	0.10	0	0	0.1	108.9
R204	25	407.5	7.08	15.1	1,352,523	2.18	12,475	2.02	0	149	5.6	158.9
R205	25	448.4	7.93	5.9	419,946	0.07	158	0.00	0	0	0.0	37.1
R206	25	413.9	5.76	9.8	882,262	0.00	2,949	0.00	0	0	0.2	43.3
R207	25	400.1	4.39	15.1	140,796	1.45	158	2.62	4	3	0.0	18.4
R208	25	394.3	4.21	18.0	138,476	1.88	666	0.09	2	2	0.1	21.5
R209	25	418.3	5.32	10.9	813,099	0.53	6,336	1.26	0	0	0.1	48.6
R210	25	448.3	7.43	9.3	1,474,542	0.22	1,464	0.07	0	0	0.2	124.2
R211	25	400.1	6.44	13.2	1,359,822	1.30	6,855	0.94	0	26	1.3	65.6
RC201	25	660.0	22.41	2.2	73,332	1.98	193	0.19	0	0	0.0	11.0
RC202	25	596.8	16.66	4.5	182,791	10.02	23,871	9.91	51,847	76,042	5,836.5	6,891.4
RC203	25	530.1	9.51	6.6	68,726	2.09	914	1.82	8,339	5,027	28.7	66.0
RC204	25	518.0	8.37	6.4	91,796	1.34	1,628	1.34	589	1,770	3.5	12.0
RC205	25	605.3	18.30	3.7	171,107	0.73	247	0.02	1	0	0.1	22.7
RC206	25	575.1	14.87	3.5	126,074	0.94	4,674	0.78	13	20	1.1	13.7
RC207	25	528.2	9.99	4.7	121,058	1.17	4,648	1.13	119	170	1.2	12.3
RC208	25	506.4	7.00	6.2	103,578	0.14	1,088	0.07	4	0	0.1	8.2
C201	40	626.6	3.29	13.2	511,637	1.60	68,945	1.51	0	7,846	427.7	668.7
C202	40	617.9	2.91	48.9	1,334,402	1.37	90,391	1.30	0	6,036	428.3	1,005.9
C203	40	611.9	2.16	58.9	3,190,776	1.10	340,263	1.06	0	5,493	1,333.1	3,769.3
C204	40	TL										
C205	40	621.0	3.63	14.5	715,459	2.16	90,659	2.16	1	25,382	1,064.1	1,356.6
C206	40	613.0	2.55	21.8	953,651	1.71	209,249	1.70	0	6,803	635.7	1,969.1
C207	40	609.9	2.30	25.7	1,336,260	1.48	276,916	1.44	0	17,076	1,758.7	4,238.9
C208	40	610.1	2.16	22.5	1,221,262	1.41	259,089	1.40	2	7,502	889.8	2,181.8
R201	40	ML										
R202	40	639.8	4.34	16.5	1,251,623	0.00	80	0.36	0	0	0.0	65.3
R203	40	604.7	4.25	24.1	1,805,535	0.64	788	0.59	0	5	0.1	120.4
R204	40	572.6	2.01	32.2	5,091,722	0.15	58,506	0.13	0	0	3.6	965.4
R205	40	632.9	3.59	13.8	1,097,579	0.24	2,038	0.22	0	6	0.3	60.0
R206	40	598.8	3.43	19.1	1,719,645	0.71	5,769	0.71	0	38	0.9	86.8
R207	40	577.9	2.18	26.0	3,379,735	0.39	56,248	0.37	0	3	7.8	983.3
R208	40	569.4	1.65	30.9	6,667,257	0.00	65,712	0.19	0	0	2.8	985.5
R209	40	587.0	0.75	20.6	1,618,310	0.00	37,754	0.52	0	0	0.3	234.7
R210	40	603.4	2.46	18.9	1,560,500	0.21	16,530	0.28	0	0	2.5	119.5
R211	40	576.1	2.74	23.1	4,019,873	0.75	30,871	0.69	1	17	8.1	558.1
RC201	40	1,018.7	17.74	5.8	1,173,328	1.58	388	0.67	1	1	0.1	137.1
RC202	40	908.1	10.33	8.9	1,832,297	0.26	18,348	0.18	25	27	2.8	209.8
RC203	40	887.7	10.39	12.1	5,022,013	0.70	123,664	0.70	2,328	3,765	413.5	1,410.1
RC204	40	865.4	9.66	12.8	6,536,683	0.00	158,010	0.09	0	0	9.4	1,184.9
RC205	40	942.1	13.40	7.9	1,324,762	0.00	37	2.07	0	0	0.0	105.9
RC206	40	939.6	14.07	7.3	3,876,234	1.11	112,583	1.19	16	708	54.7	1,127.3
RC207	40	881.8	10.71	10.1	2,451,305	0.00	32,267	0.00	0	0	2.1	312.7
RC208	40	864.4	9.98	11.5	7,200,736	0.00	144,739	1.71	18	17,456	1,007.8	2,487.7

Table 2.A.4: Computational results on the CMTVRPTW-LT: instances with 50 customers

Group	n	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	50	714.2	4.55	16.1	1,313,798	1.65	13,088	1.56	2	5,421	70.6	559.6
C202	50	700.1	3.57	58.9	3,810,994	1.59	189,322	1.82	0	11,398	1,334.7	8,147.6
C203	50	ML										
C204	50	ML										
C205	50	699.1	3.86	30.0	2,164,384	0.82	33,346	0.85	6	1,121	78.7	2,022.6
C206	50	TL										
C207	50	TL										
C208	50	TL										
R201	50	ML										
R202	50	ML										
R203	50	ML										
R204	50	ML										
R205	50	ML										
R206	50	ML										
R207	50	ML										
R208	50	ML										
R209	50	ML										
R210	50	ML										
R211	50	ML										
RC201	50	1,096.6	12.62	8.5	3,770,301	0.47	1,279	0.45	0	0	0.1	435.4
RC202	50	1,001.6	7.75	13.9	2,548,537	0.00	2,954	0.01	0	0	0.2	301.4
RC203	50	941.2	3.98	21.1	564,827	0.46	4,477	0.38	8	5	0.3	42.0
RC204	50	915.9	2.55	25.6	893,679	0.00	9,795	0.63	0	0	0.4	53.3
RC205	50	ML										
RC206	50	1,027.4	10.39	13.2	4,485,967	2.51	21,307	1.98	0	4,789	66.1	1,131.8
RC207	50	941.7	4.18	18.0	421,819	0.44	1,089	0.35	24	67	0.2	32.2
RC208	50	916.8	3.16	23.1	1,072,523	0.33	9,980	0.33	5,870	15,689	118.4	186.1

Table 2.A.5: Computational results on the CMTVRPTW-LD: instances with 25 customers

Group	n	$\overline{d}$	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	25	220	657.8	1.89	3.5	99	1.54	69	0.36	0	0	0.1	3.7
C201	25	250	539.8	0.87	3.7	147	0.00	70	0.00	0	0	0.0	3.8
C202	25	220	652.1	2.87	12.5	457	1.56	197	0.01	0	0	0.2	12.8
C202	25	250	532.5	1.57	18.9	1,466	0.75	563	0.00	0	0	0.2	19.2
C203	25	220	645.2	8.93	17.9	747	1.85	192	0.04	0	0	0.1	18.1
C203	25	250	531.8	3.63	25.5	3,404	0.75	446	0.22	0	0	0.1	25.7
C204	25	220	601.3	5.88	24.1	1,005	1.08	479	0.45	0	3	0.3	24.6
C204	25	250	524.6	2.30	35.5	4,978	1.39	1,833	1.02	0	357	0.5	36.2
C205	25	220	635.2	6.43	5.6	215	2.85	132	0.00	0	0	0.2	6.0
C205	25	250	528.9	2.12	7.9	478	0.00	159	1.49	0	0	0.2	8.2
C206	25	220	635.2	6.84	8.1	288	2.85	152	0.00	0	0	0.0	8.3
C206	25	250	526.9	2.29	10.3	843	0.44	272	0.02	8	2	0.2	10.6
C207	25	220	602.1	7.30	9.6	460	0.00	148	0.00	0	0	0.1	9.8
C207	25	250	524.6	1.86	16.5	2,154	0.00	526	0.55	0	0	0.2	16.8
C208	25	220	610.3	3.33	7.9	358	0.00	66	0.00	0	0	0.0	8.1
C208	25	250	524.6	1.86	12.2	1,269	0.00	357	1.86	0	0	0.1	12.4
R201	25	75	761.4	2.92	0.6	145	0.00	56	0.61	0	0	0.0	0.8
R201	25	100	697.2	8.40	0.8	428	0.58	67	0.00	0	0	0.0	1.0
R202	25	75	644.7	0.94	1.3	320	0.00	25	0.00	0	0	0.0	1.4
R202	25	100	616.6	10.34	2.1	2,611	0.00	279	0.14	0	0	0.0	2.3
R203	25	75	620.9	3.19	1.9	699	0.00	25	0.00	0	0	0.0	2.0
R203	25	100	576.6	11.86	3.3	4,387	0.00	131	0.00	0	0	0.0	3.5
R204	25	75	576.6	2.94	2.5	826	0.00	133	0.00	0	0	0.0	2.7
R204	25	100	482.1	2.85	3.5	2,710	0.00	104	0.73	0	0	0.0	3.6
R205	25	75	632.9	3.77	1.0	333	0.00	40	1.12	0	0	0.0	1.1
R205	25	100	557.9	5.05	1.7	1,442	0.13	272	0.02	1	0	0.0	1.9
R206	25	75	595.7	2.53	1.6	599	0.00	32	0.00	0	0	0.0	1.7
R206	25	100	522.5	4.15	2.6	2,269	0.00	15	0.00	0	0	0.0	2.7
R207	25	75	584.7	2.26	2.3	732	0.00	72	0.00	0	0	0.0	2.4
R207	25	100	510.9	5.26	3.8	4,889	1.37	1,483	0.17	0	0	0.1	4.2
R208	25	75	576.6	2.94	2.4	842	0.00	116	0.93	0	0	0.0	2.5
R208	25	100	482.1	2.97	3.7	2,919	0.00	114	0.00	0	0	0.0	3.9
R209	25	75	601.2	3.78	1.5	563	0.00	62	0.61	0	0	0.0	1.5
R209	25	100	516.5	7.66	2.3	2,868	0.48	174	0.00	0	0	0.0	2.5
R210	25	75	634.9	6.71	1.3	664	0.58	160	0.00	0	0	0.0	1.4
R210	25	100	546.2	6.93	2.4	3,496	0.00	131	0.00	0	0	0.0	2.5
R211	25	75	574.8	2.64	1.8	979	0.55	276	0.27	0	0	0.1	2.0
R211	25	100	473.4	1.74	3.3	3,731	0.00	236	0.94	0	0	0.0	3.5
RC201	25	75	986.5	11.96	0.6	109	0.68	33	0.17	12	12	0.1	0.9
RC201	25	100	824.9	24.84	1.0	276	4.30	80	3.53	110	122	0.5	2.0
RC202	25	75	880.4	20.74	1.3	360	3.70	157	0.00	0	0	0.0	1.7
RC202	25	100	678.8	23.93	2.2	1,476	0.00	170	0.00	0	0	0.0	2.6
RC203	25	75	748.1	14.44	2.1	585	4.36	198	0.00	0	0	0.0	2.4
RC203	25	100	592.7	21.21	3.7	2,717	0.00	98	1.72	0	0	0.0	4.0
RC204	25	75	743.7	17.58	2.1	771	9.08	377	0.00	2	0	0.4	2.9
RC204	25	100	586.2	23.77	4.0	4,638	0.80	1,200	0.00	0	0	0.0	4.5
RC205	25	75	839.2	16.57	1.0	336	0.56	91	0.32	19	30	0.1	1.4
RC205	25	100	701.3	20.34	1.7	1,232	0.02	50	0.53	0	0	0.0	2.0
RC206	25	75	759.7	9.39	1.0	278	0.08	121	0.08	0	0	0.0	1.3
RC206	25	100	602.9	12.44	1.5	882	0.00	42	0.00	0	0	0.0	1.7
RC207	25	75	737.8	14.78	1.3	637	4.84	318	0.35	25	26	0.1	1.8
RC207	25	100	513.9	10.25	2.8	3,284	0.39	719	0.18	5	3	0.0	3.0
RC208	25	75	727.2	20.16	1.6	977	12.84	478	0.83	66	178	0.3	2.3
RC208	25	100	501.4	14.64	3.4	7,357	0.00	2,063	0.00	1	0	0.1	4.0

Table 2.A.6: Computational results on the CMTVRPTW-LD: instances with 40 customers

Group	n	$\overline{d}$	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	40	220	1,121.8	1.19	6.9	229	0.00	183	0.00	0	0	0.1	7.0
C201	40	250	922.2	0.09	9.4	376	0.00	307	0.00	0	0	0.0	9.5
C202	40	220	1,108.2	4.95	27.6	1,100	0.00	705	0.00	0	0	7.3	35.0
C202	40	250	918.3	2.07	41.8	3,738	0.00	1,795	0.00	0	0	0.1	42.1
C203	40	220	1,086.8	7.24	42.0	1,836	0.27	801	0.17	0	5	0.3	42.5
C203	40	250	913.5	2.22	71.6	9,330	1.36	5,720	1.33	0	3,699	5.4	77.3
C204	40	220	1,037.5	4.52	61.4	2,553	0.00	85	0.00	0	0	0.4	61.9
C204	40	250	906.7	1.62	84.5	16,259	1.17	9,254	1.12	0	7,584	15.7	101.1
C205	40	220	1,082.1	1.83	11.4	539	0.44	332	0.00	0	0	0.2	11.7
C205	40	250	919.5	1.01	18.3	1,242	0.45	732	0.00	0	0	0.1	18.5
C206	40	220	1,079.6	4.40	16.5	785	0.32	124	0.00	0	0	0.0	16.6
C206	40	250	915.2	0.73	27.2	2,401	0.17	1,417	0.00	0	0	0.2	27.5
C207	40	220	1,052.0	5.54	19.4	1,264	0.15	765	0.00	0	0	8.3	27.9
C207	40	250	908.7	1.47	41.3	5,641	0.62	3,159	0.45	0	93	0.8	42.3
C208	40	220	1,068.5	3.57	20.1	978	0.00	184	0.00	0	0	0.1	20.3
C208	40	250	913.9	0.71	34.2	3,613	0.27	1,953	0.00	0	0	0.3	34.6
R201	40	75	1,159.3	3.26	1.4	532	0.00	174	0.10	0	0	0.0	1.5
R201	40	100	1,013.8	6.93	2.4	2,069	0.13	754	0.32	0	0	0.1	2.7
R202	40	75	991.5	3.65	3.4	2,428	0.00	137	0.00	0	0	0.0	3.5
R202	40	100	898.7	9.66	7.7	17,740	0.00	3,424	0.00	0	0	0.1	8.2
R203	40	75	953.6	7.80	5.3	3,808	1.00	1,086	0.34	0	3	0.2	5.6
R203	40	100	803.8	8.31	9.6	31,425	0.00	336	0.05	0	0	0.0	10.0
R204	40	75	856.3	3.63	5.5	5,169	0.97	999	0.22	0	0	0.3	5.9
R204	40	100	706.0	4.64	11.8	38,841	1.25	794	1.13	0	153	0.2	12.7
R205	40	75	971.8	4.80	2.4	1,458	0.12	1,054	0.10	0	0	0.8	3.3
R205	40	100	836.7	4.95	5.4	8,473	0.00	3,642	0.02	0	0	0.2	5.9
R206	40	75	911.3	4.87	4.0	3,463	0.37	2,153	0.31	0	0	0.1	4.3
R206	40	100	806.0	7.56	7.1	28,582	0.70	3,031	0.70	0	20	0.2	7.7
R207	40	75	884.2	4.12	4.7	4,080	0.11	128	0.00	0	0	0.0	4.8
R207	40	100	759.7	6.24	9.4	38,549	0.47	7,499	0.07	0	0	0.3	10.5
R208	40	75	856.3	3.63	5.3	5,190	0.97	1,027	0.04	0	0	0.1	5.5
R208	40	100	706.0	4.75	12.6	40,263	1.29	931	1.20	0	218	0.3	13.5
R209	40	75	933.7	4.33	3.6	2,443	0.28	160	0.05	0	0	0.0	3.7
R209	40	100	762.5	4.92	7.3	18,610	0.00	5,783	0.00	0	0	0.1	8.0
R210	40	75	942.7	8.85	3.9	3,455	1.03	353	0.00	0	0	0.0	4.1
R210	40	100	799.1	6.72	8.1	29,170	0.14	2,886	0.34	0	0	0.1	8.7
R211	40	75	840.6	2.80	4.6	5,502	0.69	1,458	0.64	2	12	0.2	4.9
R211	40	100	671.2	0.63	7.8	39,785	0.00	3,610	0.00	0	0	0.0	8.8
RC201	40	75	1,786.4	8.65	0.9	157	2.42	75	2.42	0	0	0.0	1.0
RC201	40	100	1,369.8	14.44	2.0	464	0.00	265	0.00	0	0	0.0	2.3
RC202	40	75	1,593.0	15.41	2.6	451	2.05	274	0.00	0	0	0.3	3.2
RC202	40	100	1,237.1	22.90	5.5	2,152	0.00	899	0.00	0	0	0.0	6.0
RC203	40	75	1,449.8	11.56	3.8	688	2.25	360	0.00	0	0	0.5	4.5
RC203	40	100	1,084.6	19.26	9.1	3,860	0.00	1,122	0.00	0	0	0.1	9.6
RC204	40	75	1,360.9	11.40	4.6	988	4.96	593	0.18	0	0	0.7	6.0
RC204	40	100	984.6	19.53	9.9	6,122	1.58	2,606	1.41	0	63	0.3	10.8
RC205	40	75	1,549.4	12.13	2.0	406	0.00	125	0.00	0	0	0.0	2.3
RC205	40	100	1,298.3	22.41	5.2	1,670	0.06	807	0.00	0	0	0.1	5.7
RC206	40	75	1,551.6	11.74	2.0	352	2.53	227	0.00	0	0	1.2	3.4
RC206	40	100	1,125.8	17.26	4.4	1,233	0.20	289	0.00	0	0	0.0	4.8
RC207	40	75	1,444.4	11.37	2.3	815	5.21	504	0.00	0	0	1.9	4.5
RC207	40	100	976.1	14.31	7.7	4,172	1.68	2,420	0.00	2	0	0.0	8.2
RC208	40	75	1,342.5	12.59	2.9	1,192	6.81	693	0.74	0	7	0.8	4.0
RC208	40	100	899.8	14.08	8.2	9,241	1.65	1,693	0.00	2	0	0.1	9.3

Table 2.A.7: Computational results on the CMTVRPTW-LD: instances with 50 customers

Group	n	$\overline{d}$	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	50	220	1,322.1	1.58	10.2	352	0.00	255	0.00	0	0	0.6	10.9
C201	50	250	1,098.6	0.90	12.6	590	0.00	465	0.00	0	0	0.1	12.8
C202	50	220	1,308.4	5.63	36.1	1,691	0.00	1,130	0.00	0	0	6.4	42.7
C202	50	250	1,076.3	1.28	57.8	6,418	0.00	4,169	0.00	0	0	0.2	58.3
C203	50	220	1,245.5	6.74	62.8	3,109	0.23	1,586	0.00	0	0	0.5	63.5
C203	50	250	1,071.3	1.82	107.1	17,111	0.93	11,535	0.92	0	764	4.1	111.9
C204	50	220	1,193.3	4.78	87.2	4,268	0.66	2,810	0.00	0	0	58.4	145.9
C204	50	250	1,060.4	1.84	134.6	29,395	1.08	17,782	0.98	0	8,583	45.4	182.2
C205	50	220	1,263.4	2.62	17.6	787	1.13	494	0.00	0	0	1.2	18.9
C205	50	250	1,095.9	1.67	27.4	1,816	0.86	1,181	0.45	0	169	0.5	28.0
C206	50	220	1,260.2	4.84	29.0	1,231	1.26	873	0.56	0	0	3.0	32.3
C206	50	250	1,089.3	1.26	53.2	3,502	0.69	2,178	0.60	0	273	4.3	57.7
C207	50	220	1,214.0	5.04	43.5	1,910	0.13	1,271	0.00	0	0	8.0	51.6
C207	50	250	1,074.3	0.74	72.9	7,761	0.23	4,714	0.00	0	0	0.4	73.6
C208	50	220	1,244.9	3.82	33.6	1,525	0.67	541	0.00	0	0	0.6	34.3
C208	50	250	1,074.3	0.67	60.2	5,225	0.23	3,033	0.00	0	0	0.2	60.6
R201	50	75	1,399.0	2.42	1.8	751	0.00	475	0.00	0	0	0.0	1.9
R201	50	100	1,223.2	6.93	5.0	3,160	0.00	1,101	0.16	0	0	0.1	5.3
R202	50	75	1,227.6	4.66	5.2	3,986	0.61	188	0.05	0	0	0.1	5.4
R202	50	100	1,102.8	10.25	10.7	32,397	0.02	6,184	0.00	0	0	0.2	12.0
R203	50	75	1,101.9	5.52	7.6	5,911	0.20	3,639	0.26	0	0	0.1	7.9
R203	50	100	974.1	10.98	15.3	58,152	0.62	11,433	0.54	0	101	1.3	20.0
R204	50	75	1,029.3	4.74	8.7	8,190	0.84	308	0.30	0	2	0.3	9.1
R204	50	100	821.6	2.28	16.8	82,059	0.00	7,180	0.00	0	0	0.2	18.6
R205	50	75	1,228.0	4.83	3.5	2,192	0.00	1,662	0.00	0	0	16.7	20.4
R205	50	100	1,041.5	7.10	8.5	14,366	0.00	2,670	0.00	0	0	0.1	9.0
R206	50	75	1,152.1	5.26	6.2	5,396	0.00	3,653	0.00	0	0	0.1	6.5
R206	50	100	966.3	5.38	13.1	52,846	0.00	14,560	0.00	0	0	0.1	15.1
R207	50	75	1,092.2	5.31	7.9	6,589	0.45	4,419	0.26	0	0	0.4	8.6
R207	50	100	904.0	5.77	14.3	71,509	0.00	12,548	0.00	0	0	0.2	23.6
R208	50	75	1,029.3	4.74	9.1	8,217	0.84	316	0.27	0	12	1.0	10.3
R208	50	100	821.6	2.28	17.2	82,789	0.00	6,306	0.00	0	0	0.2	19.3
R209	50	75	1,133.3	5.67	5.1	4,008	0.07	2,770	0.00	0	0	0.2	5.5
R209	50	100	922.3	5.49	12.7	35,452	0.04	12,465	0.00	0	0	0.9	16.0
R210	50	75	1,159.4	7.21	5.8	5,432	0.34	2,028	0.00	0	0	0.1	6.1
R210	50	100	976.3	8.50	13.6	54,327	0.00	378	0.00	0	0	0.0	14.6
R211	50	75	1,031.1	5.32	5.5	8,654	1.45	1,445	1.08	46	448	0.8	7.2
R211	50	100	831.3	3.52	12.6	93,865	1.91	12,513	1.90	3,687	16,479	173.1	196.3
RC201	50	75	1,873.9	10.15	1.4	263	0.00	198	0.00	0	0	0.0	1.7
RC201	50	100	1,461.5	17.27	3.6	758	0.00	155	0.00	0	0	0.0	3.9
RC202	50	75	1,761.7	15.16	3.5	825	1.55	512	0.00	0	0	2.3	6.2
RC202	50	100	1,360.2	23.66	9.3	4,479	0.00	1,207	0.00	0	0	0.1	10.0
RC203	50	75	1,593.5	11.18	5.6	1,446	2.17	748	0.00	0	0	2.4	8.2
RC203	50	100	1,242.6	23.76	13.4	10,195	3.06	3,565	0.62	0	6	1.6	15.8
RC204	50	75	1,455.9	7.39	7.2	2,263	2.15	1,368	0.14	0	0	2.6	10.0
RC204	50	100	1,069.2	16.83	15.0	22,182	3.79	11,310	0.00	0	0	0.6	18.4
RC205	50	75	1,777.9	13.23	3.3	749	2.89	274	0.00	0	0	0.0	3.5
RC205	50	100	1,470.0	26.55	7.2	3,713	0.33	1,632	0.00	0	0	0.1	7.9
RC206	50	75	1,712.1	10.02	3.0	700	2.95	382	1.07	0	0	0.1	3.4
RC206	50	100	1,258.4	16.56	8.5	2,976	0.22	915	0.00	0	0	0.0	8.9
RC207	50	75	1,620.7	13.16	3.7	1,595	5.89	1,106	0.00	1	0	8.2	12.2
RC207	50	100	1,049.4	8.41	14.1	12,055	2.48	5,679	0.00	0	0	0.6	15.2
RC208	50	75	1,438.5	8.01	5.9	2,522	3.52	1,601	2.54	1,303	3,246	29.6	35.6
RC208	50	100	991.0	12.56	12.3	30,001	4.01	16,129	0.00	0	0	0.5	18.7

Table 2.A.8: Computational results on the CMTVRPTW-R: type C and RC instances with 25 customers

Group	n	κ	ub*	P1$_{\mathcal{S}}$%	T$_{P1}$	$\lvert\mathcal{S}_1\rvert$	P2$_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	LF$_{\mathcal{S}}$%	nSFC	Nds	T$_{B\&C}$	T$_{tot}$
C201	25	0.25	380.8	5.08	4.6	9,637	2.68	3,537	0.52	0	3	0.3	6.0
C201	25	0.5	380.8	5.08	4.2	7,104	2.68	1,048	0.19	0	0	0.1	4.8
C201	25	0.75	380.8	5.08	4.1	6,052	2.68	1,045	0.00	0	0	0.1	4.7
C202	25	0.25	412.6	13.49	8.1	251,175	1.05	393	0.64	0	26	0.2	14.6
C202	25	0.5	412.6	13.49	7.9	289,678	1.05	411	0.54	0	3	0.1	15.3
C202	25	0.75	412.6	13.49	8.4	262,940	1.02	328	0.48	2	35	0.2	15.8
C203	25	0.25	431.3	16.72	9.6	2,812,513	2.08	11,940	1.97	9	1,128	24.4	118.6
C203	25	0.5	431.3	16.72	11.9	2,741,875	2.08	11,372	1.94	32	2,974	51.2	153.6
C203	25	0.75	435.5	17.53	10.1	2,741,701	2.81	12,872	2.66	44	4,433	72.1	178.2
C204	25	0.25	407.8	12.97	14.0	2,753,906	1.89	2,294	1.78	117	567	2.0	53.7
C204	25	0.5	407.8	12.97	14.3	2,890,702	1.89	2,505	1.72	162	723	3.2	56.8
C204	25	0.75	407.8	12.97	14.9	2,321,763	1.48	3,449	1.28	115	562	3.0	68.5
C205	25	0.25	378.2	6.09	5.5	15,490	2.61	4,842	0.00	0	0	0.2	6.7
C205	25	0.5	378.2	6.09	5.5	12,429	2.61	4,826	0.19	2	0	0.4	6.8
C205	25	0.75	378.2	6.09	5.5	12,636	2.61	4,826	0.77	0	0	0.2	6.7
C206	25	0.25	378.2	6.50	7.3	17,543	3.80	6,775	0.17	0	0	0.6	10.3
C206	25	0.5	378.2	6.50	7.4	15,811	3.80	7,910	0.17	0	0	0.5	10.1
C206	25	0.75	378.2	6.50	7.8	14,126	3.80	6,869	0.17	0	0	0.6	10.4
C207	25	0.25	377.4	6.65	11.5	41,235	3.76	14,658	0.00	0	0	0.7	18.6
C207	25	0.5	377.4	6.65	11.8	39,771	3.76	14,673	0.00	0	0	0.6	16.0
C207	25	0.75	377.4	6.65	11.6	39,510	3.76	14,720	0.00	0	0	0.6	18.0
C208	25	0.25	377.1	6.45	8.3	23,969	3.68	9,333	0.00	0	0	0.5	14.2
C208	25	0.5	377.4	6.52	8.7	23,323	3.75	9,372	0.39	0	0	2.5	17.7
C208	25	0.75	377.4	6.52	8.4	21,604	3.75	9,222	0.00	0	0	0.4	12.3
RC201	25	0.25	660.0	22.41	1.9	23,277	1.84	752	1.51	10	13	0.1	3.4
RC201	25	0.5	660.0	22.41	1.7	17,057	0.93	558	0.07	3	0	0.1	3.0
RC201	25	0.75	660.0	22.41	1.9	8,064	0.78	231	0.00	1	0	0.0	2.7
RC202	25	0.25	611.4	17.75	2.8	46,246	4.87	711	4.58	3,688	1,295	5.0	21.1
RC202	25	0.5	611.4	17.75	2.7	46,297	4.87	616	4.66	3,176	1,567	4.6	16.4
RC202	25	0.75	643.9	21.25	2.5	44,556	1.34	66	0.00	0	0	0.0	3.6
RC203	25	0.25	594.0	18.19	3.2	302,472	0.93	19,554	0.00	1	0	1.0	36.2
RC203	25	0.5	594.0	18.19	3.3	302,455	0.93	20,570	0.41	1	0	0.7	32.4
RC203	25	0.75	639.6	22.89	3.2	282,140	0.00	117	0.00	1	0	0.0	7.9
RC204	25	0.25	595.7	18.76	3.3	362,367	0.34	3,688	0.32	14	17	0.4	12.4
RC204	25	0.5	595.7	18.76	3.3	362,367	0.34	3,688	0.32	14	17	0.4	12.1
RC204	25	0.75	672.0	27.45	2.9	376,442	0.24	3,084	0.59	0	0	0.3	14.1
RC205	25	0.25	603.0	17.99	3.2	85,563	0.95	3,859	0.38	1	0	0.2	8.0
RC205	25	0.5	603.0	17.99	3.0	83,747	0.95	7,667	0.29	5	0	0.5	8.0
RC205	25	0.75	642.4	22.10	2.4	44,583	0.00	963	0.00	0	0	0.0	4.4
RC206	25	0.25	575.2	14.89	3.3	41,712	0.56	2,011	0.43	130	51	0.6	5.1
RC206	25	0.5	575.2	14.89	3.4	33,810	0.10	550	0.38	117	39	0.4	4.8
RC206	25	0.75	588.7	16.49	2.5	13,820	0.00	77	0.85	0	0	0.0	3.0
RC207	25	0.25	538.8	11.16	3.5	38,757	0.00	385	0.00	0	0	0.0	4.0
RC207	25	0.5	538.8	11.16	3.3	36,718	0.00	305	0.00	0	0	0.0	3.8
RC207	25	0.75	575.5	16.60	2.9	29,613	0.35	227	0.00	0	0	0.0	3.6
RC208	25	0.25	510.8	7.80	5.2	67,468	0.74	358	0.65	125	161	0.4	6.5
RC208	25	0.5	513.8	8.34	4.3	66,456	1.32	338	1.12	224	439	0.6	6.0
RC208	25	0.75	525.2	10.14	3.2	98,395	0.57	4,417	0.57	30	34	0.4	6.4

Table 2.A.9: Computational results on the CMTVRPTW-R: type R instances with 25 customers

Group	n	κ	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
R201	25	0.25	549.4	15.67	2.9	222,339	0.00	506	0.14	0	0	0.0	8.0
R201	25	0.5	555.8	16.64	2.9	151,037	0.44	164	0.74	0	0	0.0	6.8
R201	25	0.75	570.0	18.72	2.7	51,597	0.18	585	0.18	2	3	0.0	4.4
R202	25	0.25	524.2	12.86	4.6	356,859	0.00	5	0.00	0	0	0.0	8.6
R202	25	0.5	524.2	12.86	4.4	356,222	0.00	5	0.00	0	0	0.0	8.5
R202	25	0.75	533.3	14.34	4.5	567,175	0.00	1,975	0.00	0	0	0.1	13.4
R203	25	0.25	518.4	13.17	6.6	1,890,266	0.00	32	0.51	0	0	0.0	25.9
R203	25	0.5	518.4	13.17	6.4	1,890,211	0.00	40	0.51	0	0	0.0	26.0
R203	25	0.75	518.5	13.19	5.7	1,877,607	0.00	10	0.00	0	0	0.0	19.7
R204	25	0.25	ML										
R204	25	0.5	ML										
R204	25	0.75	ML										
R205	25	0.25	453.9	9.05	5.4	108,401	0.00	6	0.00	0	0	0.0	6.4
R205	25	0.5	461.7	10.58	5.5	265,527	0.00	165	0.00	0	0	0.0	8.0
R205	25	0.75	471.6	11.78	5.5	60,173	0.00	13	0.00	0	0	0.0	6.2
R206	25	0.25	442.3	7.09	8.7	443,102	0.00	140	0.19	0	0	0.0	16.2
R206	25	0.5	459.0	10.47	8.3	1,462,204	0.33	1,482	0.04	0	0	0.0	29.2
R206	25	0.75	475.0	12.54	7.2	1,193,351	0.68	131	0.29	0	0	0.0	25.7
R207	25	0.25	ML										
R207	25	0.5	ML										
R207	25	0.75	ML										
R208	25	0.25	474.3	12.94	9.2	8,122,796	0.00	15	0.00	0	0	0.0	55.2
R208	25	0.5	474.3	12.94	9.0	8,122,796	0.00	15	0.00	0	0	0.0	54.5
R208	25	0.75	ML										
R209	25	0.25	452.6	11.95	6.8	587,355	0.00	111	0.59	0	0	0.0	17.6
R209	25	0.5	452.6	11.95	7.6	583,164	0.00	122	1.19	1	0	0.0	18.4
R209	25	0.75	467.6	14.18	6.7	234,462	0.00	219	0.00	0	0	0.0	9.1
R210	25	0.25	469.8	9.83	7.3	643,750	1.31	44	0.32	0	1	0.0	15.9
R210	25	0.5	472.3	9.80	6.4	1,551,508	0.42	2,668	0.46	0	0	0.1	25.5
R210	25	0.75	542.0	20.05	6.1	4,068,450	0.29	359	0.00	1	0	0.1	116.6
R211	25	0.25	400.1	4.40	12.0	41,986	0.00	5	0.00	0	0	0.0	12.3
R211	25	0.5	400.1	4.40	10.1	41,364	0.00	4	0.00	0	0	0.0	10.5
R211	25	0.75	406.0	5.79	6.8	284,195	0.00	255	0.00	0	0	0.0	8.7

Table 2.A.10: Computational results on the CMTVRPTW-R: type C and RC instances with 40 customers

Group	n	κ	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	40	0.25	628.6	3.59	9.6	72,827	1.50	11,657	1.50	0	1,821	11.8	37.6
C201	40	0.5	636.2	4.74	9.3	52,878	0.91	1,753	0.86	0	374	0.9	11.8
C201	40	0.75	651.5	6.98	8.9	68,996	0.40	15,658	0.39	0	19	1.2	20.8
C202	40	0.25	682.6	11.00	27.6	2,438,024	0.60	19,231	0.34	0	25	5.4	107.1
C202	40	0.5	684.3	11.22	23.9	2,424,262	0.44	19,558	0.02	0	2	4.3	143.4
C202	40	0.75	689.2	11.86	22.5	2,431,698	1.05	16,183	0.63	0	225	4.7	102.4
C203	40	0.25	ML										
C203	40	0.5	ML										
C203	40	0.75	ML										
C204	40	0.25	ML										
C204	40	0.5	ML										
C204	40	0.75	ML										
C205	40	0.25	621.0	3.63	12.5	136,055	0.98	9,392	0.97	0	521	2.6	18.6
C205	40	0.5	634.0	5.61	11.7	206,785	0.98	58,106	0.89	0	438	19.6	125.9
C205	40	0.75	643.1	6.95	11.1	183,688	0.38	26,410	0.33	0	19	2.9	31.7
C206	40	0.25	613.0	2.55	20.4	184,347	0.93	25,625	0.76	0	505	12.3	51.2
C206	40	0.5	622.0	3.96	20.6	130,366	1.48	8,298	1.07	0	406	3.1	26.6
C206	40	0.75	641.7	6.91	20.9	276,606	1.44	39,012	1.32	0	2,796	39.3	86.7
C207	40	0.25	611.7	2.59	25.6	371,387	0.88	36,358	0.75	0	709	21.5	116.4
C207	40	0.5	615.8	3.24	30.9	260,147	1.33	26,267	1.32	0	4,016	32.9	74.9
C207	40	0.75	621.6	4.01	26.2	241,410	1.07	4,348	0.11	0	1	0.9	30.7
C208	40	0.25	613.0	2.62	26.3	316,869	1.12	36,725	1.02	0	1,945	30.1	121.2
C208	40	0.5	619.0	3.57	25.8	174,354	1.61	14,094	1.60	0	3,263	18.8	51.5
C208	40	0.75	632.4	5.61	21.0	370,169	1.18	60,702	1.07	0	497	18.4	171.2
RC201	40	0.25	1,038.1	19.28	5.4	301,763	1.32	7,119	1.22	1	73	0.9	17.9
RC201	40	0.5	1,038.1	19.28	5.3	169,441	0.00	1,955	0.26	1	0	0.1	10.4
RC201	40	0.75	1,048.6	20.09	5.3	95,225	0.00	175	0.00	0	0	0.0	7.1
RC202	40	0.25	972.2	15.00	6.4	402,424	0.00	1,117	0.00	0	0	0.0	11.1
RC202	40	0.5	972.2	15.00	7.0	391,901	0.00	1,098	0.00	2	0	0.0	11.6
RC202	40	0.75	1,092.9	24.01	5.9	426,700	0.29	1,547	0.02	0	0	0.1	16.4
RC203	40	0.25	1,041.9	22.60	6.4	6,222,602	0.70	31,438	2.20	0	63,362	727.6	1,011.4
RC203	40	0.5	1,041.9	22.60	6.3	6,220,996	0.70	30,198	2.10	1	10,418	163.6	463.2
RC203	40	0.75	1,145.7	28.98	6.4	4,240,524	1.41	2,932	2.27	0	6,545	8.3	233.7
RC204	40	0.25	ML										
RC204	40	0.5	ML										
RC204	40	0.75	ML										
RC205	40	0.25	972.2	16.08	7.7	642,077	0.32	1,875	0.24	18	27	0.2	19.5
RC205	40	0.5	973.6	16.20	6.4	366,037	0.25	1,188	0.14	0	0	0.0	11.5
RC205	40	0.75	1,071.7	22.90	6.3	204,157	0.86	5,160	0.61	0	3	0.3	13.3
RC206	40	0.25	984.7	18.01	7.2	1,198,757	0.64	46,282	0.84	2	275	17.5	103.6
RC206	40	0.5	984.7	18.01	7.0	626,830	0.00	9,444	0.00	0	0	1.4	27.2
RC206	40	0.75	1,007.9	19.69	8.1	317,735	0.00	1,748	0.00	0	0	0.1	14.3
RC207	40	0.25	908.6	12.98	10.4	680,191	0.04	275	0.00	0	0	0.0	28.0
RC207	40	0.5	908.6	12.98	8.8	579,656	0.04	189	0.00	0	0	0.0	22.6
RC207	40	0.75	1,006.4	20.98	8.8	1,008,959	0.44	1,194	0.19	0	0	0.1	55.5
RC208	40	0.25	868.5	10.41	14.2	2,802,983	0.06	46,353	0.06	67	67	10.4	330.7
RC208	40	0.5	872.1	10.78	10.3	2,480,067	0.47	51,303	0.47	2,346	4,410	131.9	444.9
RC208	40	0.75	885.5	12.01	7.5	1,382,420	0.24	7,527	0.19	0	0	0.7	77.7

Table 2.A.11: Computational results on the CMTVRPTW-R: type R instances with 40 customers

Group	n	κ	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
R201	40	0.25	766.4	10.24	6.0	6,422,777	0.00	9,801	0.03	0	0	0.3	143.1
R201	40	0.5	767.4	10.36	6.1	3,434,211	0.00	3,727	0.00	0	0	0.0	54.9
R201	40	0.75	779.6	11.76	5.9	1,533,262	0.00	1,064	0.00	0	0	0.0	25.2
R202	40	0.25	ML										
R202	40	0.5	ML										
R202	40	0.75	ML										
R203	40	0.25	ML										
R203	40	0.5	ML										
R203	40	0.75	ML										
R204	40	0.25	ML										
R204	40	0.5	ML										
R204	40	0.75	ML										
R205	40	0.25	658.2	7.29	11.6	5,014,279	0.00	1,099	0.00	0	0	0.0	96.1
R205	40	0.5	668.0	8.65	11.8	2,475,194	0.00	26	0.00	0	0	0.0	47.1
R205	40	0.75	686.3	10.97	14.7	3,026,765	0.00	1,736	0.07	0	0	0.0	70.8
R206	40	0.25	664.0	8.88	15.6	8,149,868	0.09	44	0.00	0	0	0.0	139.3
R206	40	0.5	ML										
R206	40	0.75	ML										
R207	40	0.25	ML										
R207	40	0.5	ML										
R207	40	0.75	ML										
R208	40	0.25	ML										
R208	40	0.5	ML										
R208	40	0.75	ML										
R209	40	0.25	ML										
R209	40	0.5	636.1	8.29	17.6	8,506,644	0.17	332	0.00	0	0	0.0	169.8
R209	40	0.75	ML										
R210	40	0.25	ML										
R210	40	0.5	ML										
R210	40	0.75	ML										
R211	40	0.25	588.7	4.15	18.1	1,366,692	0.00	100	0.52	0	0	0.0	42.0
R211	40	0.5	588.7	4.15	18.0	1,282,463	0.00	110	0.83	0	0	0.0	40.4
R211	40	0.75	ML										

Table 2.A.12: Computational results on CMTVRPTW-R: type C and RC instances with 50 customers

Group	n	κ	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	50	0.25	711.9	4.24	16.4	147,637	0.56	3,881	0.39	0	148	1.6	24.2
C201	50	0.5	716.2	4.81	15.6	134,290	0.55	3,824	0.34	0	99	0.8	25.5
C201	50	0.75	747.3	8.77	15.3	316,965	0.21	930	0.16	0	0	0.1	27.5
C202	50	0.25	ML										
C202	50	0.5	ML										
C202	50	0.75	ML										
C203	50	0.25	ML										
C203	50	0.5	ML										
C203	50	0.75	ML										
C204	50	0.25	ML										
C204	50	0.5	ML										
C204	50	0.75	ML										
C205	50	0.25	705.6	4.74	25.2	270,920	0.16	121	0.00	0	0	0.0	34.7
C205	50	0.5	709.5	5.27	25.0	1,103,524	0.11	59,730	0.05	1	0	5.6	256.1
C205	50	0.75	738.1	8.94	26.9	825,799	0.07	484	0.00	0	0	0.0	69.2
C206	50	0.25	703.3	4.47	42.9	411,186	1.05	3,604	0.86	0	1,653	2.7	61.7
C206	50	0.5	705.1	4.71	43.3	326,616	0.82	2,458	0.62	0	461	1.4	57.7
C206	50	0.75	734.0	8.46	39.3	1,457,240	0.51	8,810	0.48	0	22	0.9	171.4
C207	50	0.25	700.8	4.39	48.1	880,401	0.77	4,182	0.61	0	792	1.9	105.2
C207	50	0.5	703.6	4.77	54.5	688,963	0.82	2,037	0.48	0	304	0.5	126.3
C207	50	0.75	703.7	4.77	52.5	669,477	0.24	213	0.38	0	0	0.0	111.5
C208	50	0.25	702.8	4.61	41.5	588,751	1.05	3,333	0.94	0	2,281	2.6	72.5
C208	50	0.5	703.7	4.73	41.3	417,124	0.73	1,574	0.71	0	942	1.0	55.9
C208	50	0.75	703.7	4.73	41.6	375,971	0.27	206	0.10	0	0	0.0	51.4
RC201	50	0.25	1,096.6	12.62	10.9	604,229	0.00	213	0.00	0	0	0.0	31.2
RC201	50	0.5	1,111.3	13.77	11.3	706,072	0.00	8,469	0.00	1	0	0.5	45.1
RC201	50	0.75	1,151.1	16.76	11.1	383,562	0.33	1,112	0.00	0	0	0.1	25.5
RC202	50	0.25	1,140.9	17.83	16.2	2,341,129	1.57	1,903	1.48	2	60	0.3	187.4
RC202	50	0.5	1,142.7	17.96	16.4	2,275,325	1.27	1,055	0.54	2	3	0.1	187.7
RC202	50	0.75	1,201.7	21.78	15.5	1,911,310	0.87	312	0.00	2	0	0.0	129.4
RC203	50	0.25	ML										
RC203	50	0.5	ML										
RC203	50	0.75	ML										
RC204	50	0.25	ML										
RC204	50	0.5	ML										
RC204	50	0.75	ML										
RC205	50	0.25	1,059.4	12.56	17.7	1,515,179	2.44	5,908	0.90	314	371	4.3	64.6
RC205	50	0.5	1,126.3	17.66	18.6	2,027,428	0.00	19	0.00	0	0	0.0	96.4
RC205	50	0.75	1,212.2	22.88	17.6	1,139,310	0.00	7,221	0.00	0	0	0.2	89.3
RC206	50	0.25	1,036.3	11.16	18.3	2,845,972	0.92	21,466	0.71	0	148	4.0	370.2
RC206	50	0.5	1,036.3	11.16	17.3	1,811,941	0.86	15,970	0.44	0	22	1.5	159.4
RC206	50	0.75	1,096.2	16.02	19.0	1,886,079	1.73	8,151	1.55	0	631	3.2	206.7
RC207	50	0.25	965.1	5.71	20.0	2,274,605	0.04	22,985	0.02	0	0	1.2	399.4
RC207	50	0.5	965.1	5.71	21.5	2,020,514	0.04	22,443	0.00	0	0	0.8	378.6
RC207	50	0.75	ML										
RC208	50	0.25	925.1	4.03	26.9	744,659	0.71	5,772	0.71	41,580	99,383	6,221.2	6,275.8
RC208	50	0.5	TL										
RC208	50	0.75	941.8	5.22	16.7	4,057,582	0.03	60,545	0.03	169	145	27.3	906.1

Table 2.A.13: Computational results on the DRP: type A1 instances

Group	n	Num	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
A1	10	1	3,132.0	12.08	0.3	82	5.93	40	2.42	21	18	0.1	0.8
A1	10	2	4,738.9	6.62	0.3	45	0.00	16	0.00	0	0	0.0	0.4
A1	10	3	4,556.3	13.22	0.1	115	2.85	43	0.00	0	0	0.0	0.4
A1	10	4	4,391.5	15.98	0.1	203	8.19	136	0.51	7	0	0.1	0.6
A1	10	5	4,524.2	7.44	0.1	78	0.19	24	0.00	1	0	0.0	0.3
A1	15	1	7,072.0	11.25	0.4	389	0.70	142	0.26	14	8	0.1	0.8
A1	15	2	4,397.8	13.33	0.8	2,123	0.29	26	1.26	0	0	0.0	1.1
A1	15	3	5,968.2	16.71	0.7	1,034	5.99	457	4.65	1	542	0.3	1.4
A1	15	4	5,491.0	8.77	0.3	440	0.81	54	0.00	0	0	0.0	0.4
A1	15	5	7,383.4	15.22	0.6	451	3.09	198	0.13	0	0	0.0	1.0
A1	20	1	8,284.9	6.91	0.3	691	0.00	80	0.00	0	0	0.0	0.5
A1	20	2	9,548.0	8.49	0.4	636	0.81	129	0.02	1	0	0.1	0.6
A1	20	3	8,816.1	8.92	0.3	1,030	1.71	231	0.40	9	44	0.2	0.7
A1	20	4	6,693.8	7.80	0.7	1,230	0.00	92	0.00	0	0	0.1	1.0
A1	20	5	7,782.1	11.35	0.6	1,064	2.73	615	2.61	280	771	5.6	6.8
A1	25	1	10,680.0	9.80	1.0	2,040	2.31	566	1.96	2	760	0.3	2.3
A1	25	2	8,636.2	10.16	2.2	7,680	1.10	163	1.00	0	0	0.1	2.6
A1	25	3	10,094.5	10.42	0.9	2,627	2.21	705	1.84	459	898	7.9	11.5
A1	25	4	10,146.6	7.45	1.1	1,542	0.65	259	0.00	0	0	0.1	1.3
A1	25	5	11,166.0	8.76	0.9	2,886	1.40	374	0.29	0	7	0.1	1.2
A1	30	1	9,831.6	8.46	1.3	12,705	0.00	215	0.82	0	0	0.0	1.7
A1	30	2	12,665.0	7.18	1.4	8,417	0.19	1,966	0.19	0	0	0.1	1.7
A1	30	3	12,359.4	9.39	1.2	5,094	2.08	798	1.74	594	1,291	4.7	12.5
A1	30	4	12,512.8	8.22	1.1	10,145	0.97	1,101	0.41	108	101	2.4	3.8
A1	30	5	12,086.1	7.97	1.6	13,061	0.89	1,373	0.71	0	224	0.2	2.1
A1	35	1	12,434.0	10.53	2.6	36,006	1.61	28,939	1.56	34	5,747	75.4	80.6
A1	35	2	13,020.8	8.80	3.0	20,915	0.84	1,207	0.86	0	359	0.7	4.1
A1	35	3	13,230.4	7.66	3.1	17,011	0.00	893	0.00	0	0	0.1	3.5
A1	35	4	13,863.5	7.86	2.5	10,215	0.00	1,143	0.00	0	0	0.1	2.9
A1	35	5	13,281.6	7.81	1.6	11,974	0.51	1,442	0.43	0	50	0.7	2.6
A1	40	1	15,540.1	9.25	3.3	13,985	0.49	327	0.38	0	213	0.6	4.4
A1	40	2	16,881.3	7.56	2.3	14,346	0.35	2,256	0.19	106	176	6.9	9.5
A1	40	3	14,178.4	7.99	3.6	29,633	0.22	1,622	0.20	8	16	1.0	5.3
A1	40	4	16,286.8	7.50	2.3	17,214	0.21	2,170	0.17	11	22	1.1	3.7
A1	40	5	15,620.2	8.07	2.4	7,970	0.48	1,251	0.37	231	442	11.8	14.4
A1	45	1	14,569.0	8.00	7.7	167,407	0.38	8,185	0.33	35	219	10.1	31.3
A1	45	2	19,727.5	8.68	4.4	33,975	0.91	2,537	0.67	224	717	24.9	30.1
A1	45	3	18,825.4	7.83	5.0	28,606	0.21	4,593	0.19	39	191	4.3	9.9
A1	45	4	16,298.5	9.08	5.1	33,504	0.68	1,161	0.60	153	887	9.9	15.6
A1	45	5	18,727.9	7.88	5.8	26,600	0.44	4,221	0.35	0	89	1.4	7.7

Table 2.A.14: Computational results on the DRP: type A2 instances

Group	n	Num	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
A2	10	1	4,999.0	10.49	0.1	84	3.98	33	0.02	7	0	0.0	0.4
A2	10	2	5,825.5	6.51	0.1	31	0.00	12	0.00	0	0	0.0	0.3
A2	10	3	5,269.9	10.05	0.2	65	0.00	14	0.00	0	0	0.0	0.5
A2	10	4	6,157.2	16.25	0.1	51	4.85	35	0.26	7	0	0.0	0.5
A2	10	5	5,534.0	12.46	0.1	38	0.78	14	0.78	1	0	0.0	0.3
A2	15	1	6,869.6	10.37	0.1	166	0.97	45	0.00	2	0	0.0	0.4
A2	15	2	8,535.0	15.81	0.2	119	1.43	59	0.83	40	41	0.4	0.9
A2	15	3	6,612.0	9.53	0.2	87	0.00	32	0.78	0	0	0.0	0.5
A2	15	4	8,777.9	18.26	0.4	127	7.69	66	4.88	169	82	1.9	5.8
A2	15	5	8,672.1	12.15	0.2	101	0.00	47	0.00	0	0	0.0	0.4
A2	20	1	11,422.7	14.77	0.2	153	3.35	96	2.23	181	275	1.8	2.9
A2	20	2	9,730.0	12.30	0.4	486	3.50	237	3.35	181	323	3.9	4.8
A2	20	3	10,093.7	7.73	0.7	220	0.00	58	0.00	0	0	0.0	1.0
A2	20	4	9,492.4	8.98	0.2	214	0.00	19	0.00	0	0	0.0	0.4
A2	20	5	8,299.5	12.62	0.5	1,088	1.73	216	0.03	3	0	0.1	0.9
A2	25	1	11,436.3	8.91	0.7	570	0.01	35	0.00	1	0	0.1	0.9
A2	25	2	12,426.4	7.07	0.8	612	0.00	126	0.13	0	0	0.0	1.0
A2	25	3	10,973.4	13.20	0.8	480	3.71	128	2.53	314	615	7.3	8.4
A2	25	4	12,275.4	9.09	0.7	394	0.69	50	2.65	0	0	0.1	1.0
A2	25	5	11,788.0	10.44	0.6	436	0.15	211	0.00	0	0	0.2	1.2
A2	30	1	14,997.4	9.44	1.0	839	0.45	79	0.00	0	0	0.1	1.5
A2	30	2	12,794.3	11.94	1.4	1,578	0.96	494	0.28	1	11	0.3	2.0
A2	30	3	12,234.4	11.17	1.3	2,365	0.81	1,223	0.22	88	136	2.3	3.9
A2	30	4	11,587.3	8.09	2.1	1,715	0.19	128	0.00	0	0	0.4	2.7
A2	30	5	13,261.5	11.75	2.3	2,494	1.23	964	0.81	35	77	1.8	4.4
A2	35	1	14,282.9	9.98	2.8	4,539	0.42	2,730	0.13	11	1	0.7	3.8
A2	35	2	17,443.4	9.69	1.5	691	0.27	547	0.46	8	6	5.5	7.4
A2	35	3	14,691.3	10.05	3.0	3,419	0.92	226	0.33	4	8	0.3	3.8
A2	35	4	17,689.3	9.99	2.3	1,554	1.20	354	0.22	0	0	0.1	2.8
A2	35	5	16,812.4	11.19	2.1	3,592	0.45	1,590	0.45	2	8	0.4	2.8
A2	40	1	17,002.9	9.44	3.5	3,623	0.23	3,015	0.00	5	0	1.1	4.9
A2	40	2	17,949.0	9.22	3.6	3,086	1.07	371	0.54	0	65	0.4	4.3
A2	40	3	18,078.8	7.85	1.8	2,402	0.16	466	0.00	3	0	0.5	2.5
A2	40	4	18,559.6	8.57	3.9	2,642	0.59	452	0.37	262	349	21.4	25.5
A2	40	5	13,798.5	10.77	3.9	6,485	0.51	2,092	0.32	175	400	5.4	9.7
A2	45	1	18,654.7	7.41	2.6	4,965	0.07	936	0.01	0	0	0.1	2.9
A2	45	2	19,590.6	12.09	3.6	11,937	0.21	1,313	0.83	72	86	5.3	9.4
A2	45	3	20,207.2	8.35	2.2	3,331	0.73	1,078	0.47	22	199	3.6	6.0
A2	45	4	17,306.9	9.14	5.8	11,519	0.52	351	0.13	107	140	11.3	17.4
A2	45	5	24,311.7	9.14	2.2	1,518	0.72	255	0.00	1	0	0.8	3.3
A2	50	1	24,698.5	8.78	4.1	11,373	0.79	2,155	0.67	1,012	2,553	185.7	190.3
A2	50	2	21,939.4	11.06	6.2	4,760	0.94	3,829	0.39	61	136	30.6	37.2
A2	50	3	19,700.0	7.82	6.5	7,130	0.19	1,917	0.18	0	3	0.6	7.4
A2	50	4	19,841.8	8.39	7.5	6,613	0.12	352	0.09	119	151	25.4	33.2
A2	50	5	23,721.5	8.06	4.5	3,655	0.59	1,271	0.38	5	24	1.8	6.5

Table 2.A.15: Computational results on the DRP: type B instances

Group	n	ub*	$P1_{\mathcal{S}}$%	T_{P1}	$\lvert\mathcal{S}_1\rvert$	$P2_{\mathcal{S}_1}$%	$\lvert\mathcal{S}_2\rvert$	$LF_{\mathcal{S}}$%	nSFC	Nds	$T_{B\&C}$	T_{tot}
C201	25	644.7	8.77	0.38	3,226	0.90	177	0.33	0	0	0.1	0.7
C202	25	644.1	9.03	0.47	10,536	2.14	590	1.80	0	457	0.4	1.1
C203	25	643.9	9.00	0.74	20,786	2.16	795	1.78	0	708	0.6	1.9
C204	25	643.9	9.00	0.87	25,543	2.17	974	1.81	0	1,880	1.1	2.5
C205	25	644.4	9.04	0.29	4,841	2.12	406	0.86	1	56	0.1	0.6
C206	25	643.8	8.96	0.33	5,494	2.10	509	1.24	0	263	0.2	0.8
C207	25	643.6	8.96	0.45	6,733	2.14	612	1.79	0	1,059	0.5	1.2
C208	25	643.6	8.96	0.34	5,856	2.09	556	0.93	0	122	0.1	0.7
R201	25	752.8	8.16	0.10	2,228	0.22	74	0.00	0	0	0.0	0.3
R202	25	730.4	7.11	0.20	4,651	0.00	193	0.00	1	0	0.0	0.4
R203	25	721.6	6.99	0.28	5,274	0.00	240	0.00	0	0	0.0	0.5
R204	25	721.4	6.97	0.41	8,170	0.00	240	0.69	0	0	0.0	0.6
R205	25	731.8	6.96	0.18	4,086	0.00	210	0.00	0	0	0.0	0.4
R206	25	725.5	7.05	0.23	5,272	0.00	179	0.07	0	0	0.0	0.4
R207	25	721.6	6.99	0.45	5,780	0.00	225	0.67	0	0	0.1	0.7
R208	25	721.4	6.97	0.53	7,205	0.00	248	0.00	0	0	0.1	0.8
R209	25	721.7	7.00	0.27	4,167	0.00	196	0.00	0	0	0.0	0.5
R210	25	725.5	7.05	0.27	4,732	0.00	161	0.05	0	0	0.0	0.5
R211	25	721.4	6.97	0.39	6,790	0.00	232	0.00	0	0	0.0	0.6
RC201	25	1,022.3	13.11	0.09	597	2.05	158	0.00	0	0	0.0	0.4
RC202	25	1,007.1	12.50	0.11	1,031	2.16	300	0.00	0	0	0.0	0.4
RC203	25	1,000.0	11.87	0.15	1,584	1.96	479	0.00	0	0	0.0	0.4
RC204	25	1,000.0	11.87	0.16	1,644	1.96	488	0.00	0	0	0.1	0.5
RC205	25	1,013.4	13.04	0.10	954	2.74	303	0.44	47	136	0.3	0.7
RC206	25	1,013.8	13.07	0.10	907	2.43	249	0.12	6	1	0.1	0.4
RC207	25	1,005.5	12.36	0.11	1,337	2.37	395	0.00	0	0	0.0	0.4
RC208	25	999.8	11.86	0.16	1,802	1.96	479	0.00	0	0	0.1	0.5
C201	40	1,103.5	7.57	0.51	19,307	0.52	3,430	0.26	0	12	0.3	1.4
C202	40	1,102.3	7.48	1.47	52,210	0.80	6,287	0.39	0	296	1.3	4.2
C203	40	1,101.7	7.55	2.58	108,538	0.87	10,008	0.45	0	134	1.9	8.9
C204	40	1,101.2	7.59	2.20	168,749	0.89	12,447	0.44	0	522	5.4	11.4
C205	40	1,101.5	7.61	0.59	29,387	0.70	4,248	0.61	14	176	0.8	1.9
C206	40	1,101.2	7.50	0.98	38,434	0.77	5,536	0.35	6	66	1.0	3.2
C207	40	1,101.1	7.50	1.32	46,887	0.89	7,876	0.67	0	188	1.4	3.9
C208	40	1,101.1	7.50	1.23	41,657	0.77	6,080	0.36	7	231	1.9	4.0
R201	40	1,200.1	10.14	0.25	10,516	0.31	4,309	0.25	0	1	0.3	1.1
R202	40	1,145.3	7.23	0.51	31,782	0.00	901	0.00	0	0	0.1	1.2
R203	40	1,142.4	6.63	0.81	46,580	0.00	3,010	0.00	0	0	0.3	1.8
R204	40	1,140.6	6.48	1.06	61,886	0.00	4,428	0.00	0	0	0.6	2.5
R205	40	1,159.7	7.29	0.44	20,690	0.00	913	0.00	0	0	0.0	0.8
R206	40	1,141.8	6.54	0.73	40,075	0.00	2,615	0.04	0	0	0.2	1.5
R207	40	1,141.8	6.58	0.89	51,739	0.00	2,902	0.00	0	0	0.4	2.0
R208	40	1,140.6	6.48	1.35	62,423	0.00	4,135	0.00	0	0	2.0	4.2
R209	40	1,149.1	6.97	0.93	36,510	0.00	2,220	0.00	0	0	0.4	1.9
R210	40	1,142.4	6.59	0.79	41,025	0.00	2,678	0.00	0	0	0.2	1.6
R211	40	1,140.6	6.48	1.06	69,996	0.00	3,667	0.00	0	0	0.5	2.5
RC201	40	1,692.1	12.45	0.25	1,097	1.72	570	0.67	1,520	3,998	17.2	17.8
RC202	40	1,642.4	10.24	0.44	1,772	1.32	1,366	0.00	0	0	0.6	1.3
RC203	40	1,635.3	9.85	0.62	2,670	1.20	2,028	0.00	0	0	0.9	1.9
RC204	40	1,635.3	9.86	0.59	2,684	1.20	1,918	0.00	0	0	3.2	4.1
RC205	40	1,650.6	10.67	0.28	1,653	1.57	1,245	0.31	77	301	2.6	3.1
RC206	40	1,658.9	11.12	0.36	1,623	1.57	1,126	0.15	1	1	0.3	1.0
RC207	40	1,639.0	10.06	0.45	2,252	1.37	1,724	0.00	1	0	1.4	2.1
RC208	40	1,635.1	9.85	0.67	3,085	1.20	2,046	0.00	0	0	1.0	2.0

Chapter 3

Optimization Models for the Installation Planning of Offshore Wind Farms*

Abstract

Challenges posed by global warming motivate the increasing interest in green sources of energy, such as wind energy. To make wind energy attractive from an economical viewpoint, the decision-making problems faced in designing, installing, and maintaining wind farms have to be optimized. In this paper, we focus on the problem of optimally planning the installation process of offshore wind farms, as faced by Vattenfall, a leading European energy company. We formulate this Installation Planning of Offshore Wind Farms (IPOWF) problem as a Mixed Integer Linear Programming (MILP) model. From this model, we then derive other MILPs that can provide lower and upper bounds to the problem. Such bounds are tested on real-life instances corresponding to two wind farms recently built in Denmark and Germany by Vattenfall. Computational results show that primal solutions of the IPOWF that are within 2% from optimality can be computed with the proposed models.
***Keywords**: wind farms, wind energy, installation, mixed integer linear programming*

3.1 Introduction

Worldwide energy demand is constantly growing, due to, for example, population grow and technological development. As global warming and pollution are main challenges of our century (Saidur et al. (2011)), finding alternative sources of energy is more and more important. In this context, the wind energy field has attracted a lot of attention and investments over the past decades. In particular, construction and utilization of offshore wind farms is going to increase in the coming years.

Not only do technical challenges related to turbine components require major improvements, but also the logistics process supporting the construction and utilization of offshore wind farms needs major improvements. As a matter of fact, even if offshore turbines are relatively efficient from an energetic point of view, the life-cycle operations related to offshore wind farms are complex and represent the main drawback of this technology. For example, offshore turbines generally produce more electricity than the

*This chapter is based on Fischetti et al. (2019).

onshore counterparts (Irawan, Ouelhadj, Jones, Stålhane & Sperstad (2017)), but onshore wind farms can be installed and maintained using less resources (Snyder & Kaiser (2009)).

The life-cycle activities of offshore wind farms can be grouped in three main processes or phases, namely, *design*, *installation*, and *maintenance*.

The *design* process aims at defining the specifications of the wind farm. Decision makers involved in this phase are to decide, for example, the number and types of turbines to install and the layout of the farm. A crucial decision is about turbines locations, which has a major impact on the power production of the wind farm. Indeed, the production of a turbine depends on its location and on the interaction with other turbines caused by the wind flow generated by upstream turbines that can affect the productivity of downstream turbines (this phenomenon is known as *wake effect*). Once the turbine positioning is defined, it must be decided how to connect them, i.e., a *cable routing problem* is faced. This task consists of finding an optimal connection among offshore turbines and some collection points at sea, i.e. the so-called substations. Different cables, with different capacities and costs, can be used. The goal is to minimize the overall costs while avoiding crossing between cables. The decisions made in the design process affect the productivity of the farm until the end of its life-cycle, so massive savings can be achieved by applying optimization techniques to support the decision maker (see, e.g., Fischetti & Pisinger (2019) for a complete overview on the design process).

The *installation* process is about constructing the wind farm and receives the layout of the farm as input from the design phase. In this phase, the main logistic activities are related to the supply of the required material and resources, and to the scheduling of the activities to built the farm. The installation phase is affected by many disturbance variables, such as delayed in supply and weather conditions (Scholz-Reiter, Heger, Lütjen & Schweizer (2011)).

The *maintenance* process is continuously repeated over the life-cycle of the farm. According to Irawan, Ouelhadj, Jones, Stålhane & Sperstad (2017), this phase contributes to a quarter of the life-cycle costs. Maintenance activities are categorized into three different types: *corrective*, *preventive*, and *condition-based*. Corrective maintenance is performed to restore the normal status of the system when a failure is detected. Preventive maintenance aims at reducing the risk of failures and is conducted based on criteria such the age of the equipment. Condition-based maintenance is based on the condition monitoring of the equipment. When a specific indicator suggests that the system is deteriorating over a given threshold, maintenance is performed (see, e.g., Irawan, Ouelhadj, Jones, Stålhane & Sperstad (2017) for more details on this process).

In this paper, we focus on the installation process. An offshore wind farm must be installed, following the specifications defined in the design process, and construction costs have to be minimized. The installation process mainly deals with turbines, cables, and substations, as shown in Figure 3.1. Turbines are connected by *inter-array cables*. These cables connect the turbines to an *offshore substation*, where the voltage is increased to improve transmission over long distances. From the substation, energy is transmitted to an onshore control centre through *export cables* (high voltage cables). From the control centre, energy is finally transmitted to the onshore grid.

In the construction phase, substations are usually already present on site, so they are involved in the cabling activity only, but there is no need to construct them. On the other hand, turbines need to be constructed in the locations defined in the design phase. The tasks to construct any type of turbine can be grouped in the following macro activities: (1) *installation of foundations*, (2) *installation of transition*

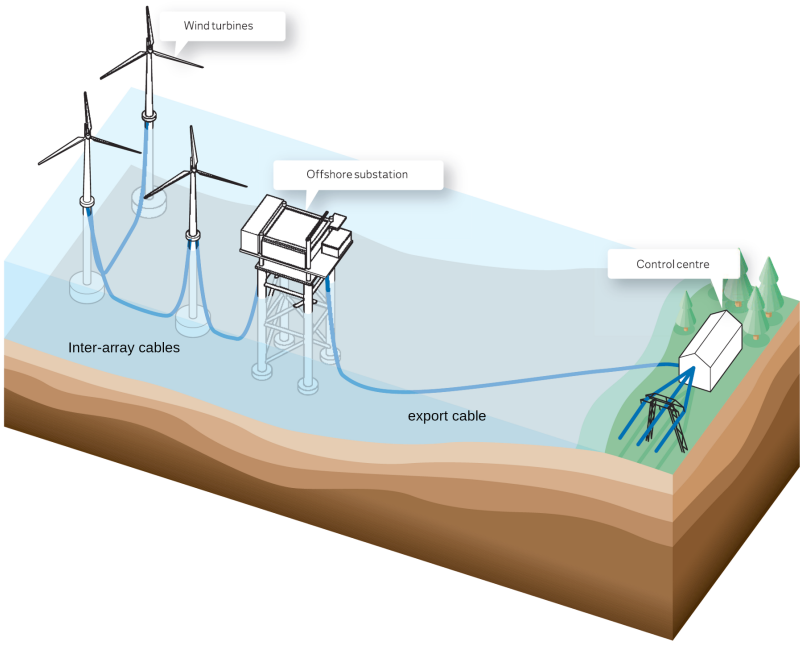


Figure 3.1.1: Graphical representation of the key components of an offshore wind farm (provided by Vattenfall)

pieces, (3) *installation of cables*, and (4) *installation of turbine itself* (i.e., turbine tower, nacelle, and blades).

The installation of foundations is the first activity to be carried out when constructing a turbine. Foundations are needed to attach the entire turbine structure to the ground. Figure 3.1 shows the foundations at the bottom of each turbines. There can be different types of foundations, depending mainly on the sea-bottom conditions and water depth of the specific site, but the most common foundations are *monopile* foundations. For this reason, following the naming used by our industrial partner, we refer to the installation of foundations as MP (for monopile).

Once foundation is ready, it is possible to install the transition pieces (TP), which connect the foundation to the turbine tower.

The next activity is the installation of cables that deals with the connection of turbines to the substation. In order for the energy to be transferred from the turbines to the substation, turbines are often connected in strings. In this context, a *string* is a set of turbines directly connected by inter-array cables. Due to technical reasons, the cabling operation must be carried out *string-by-string*, meaning that the energy generated by each turbine can be transmitted only when all the turbines of that string are completed. In Figure 3.1, all the turbines belong to the same string. By definition of string, only one turbine of the string is connected to the substation, we will refer to it as the *first* turbine of a string. On the other hand, the *last* turbine of a string is the turbine that is directly connected to a single turbine only. Our industrial partner specified that the cabling operation must be carried out from the first turbine to the last turbine of a string; therefore, in the following, we assume that the cabling operations is carried out string-by-string meaning that cables are installed by string, starting from the first turbine all the way until the last turbine.

Once the cabling operation is completed, it is possible to finalize the turbine by installing the turbine itself (i.e., tower, nacelle, and blades). For this activity, it is not required to proceed string-by-string.

However, as energy transmission is associated to whole strings rather than single turbines, the turbines of a string can start producing energy only once all the turbines of the string have been completed.

An example of the string-by-string approach can be given looking at Figure 3.1.2. Figure 3.1.2 represents a real-word wind farm, namely, Horns Rev 3, a wind farm recently installed by Vattenfall off the Danish Jutland coast. The black numbered dots represent the turbines positioned in the area, and the black lines represent inter-array cables connecting turbines of the same string and strings to the only substation (the red square). The turbines are arranged in twelve strings. If we consider, for example, the string composed by turbines 15, 16, 24 and 35, when the vessel is performing the cabling operations it must proceed following the order 35, 24, 16, 15. Once the cabling operation is completed for turbine 15, the vessel can move to the first turbine of another string.

The aforementioned four activities involved in the installation process are carried out by using a fleet of vessels located in a port. Each activity requires different materials, crews, and tools (all located at the port), and a specific type of vessel. Therefore, each type of vessel can be used to complete a single activity, so we can refer to a specific vessel type referring to the corresponding activity. To perform each activity, a single vessel is usually available. The capacity of a vessel to transport materials, crews, and tools is limited by the weights of these elements. Therefore, vessels usually go back and forth from/to the port multiple times to transport materials, crews, and tools from the port to the turbine locations.

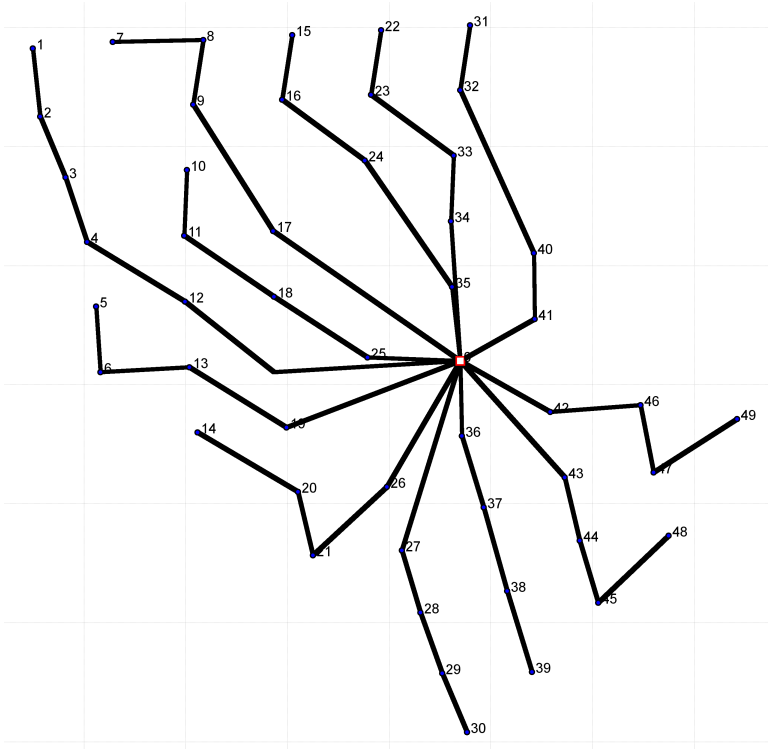


Figure 3.1.2: Graphical representation of Horns Rev 3 wind farm. Turbines (blue dots) are connected to the substation (red square) through cables (black lines). The connection has a string structure, with 12 strings connecting to the substation. The port is located South-East of the farm, at about 50 km from it.

In this paper, we address the problem of optimally scheduling the activities of the vessels to install a given offshore wind farm as the Installation Planning of Offshore Wind Farms problem (IPOWF). The main contributions of this paper are: (a) we formally define and formulate, via *Mixed Integer Linear Programming* (MILP), the IPOWF by including several real-life features validated by our industrial patner

Vattenfall, the second largest offshore wind power developer in the world with major offshore wind farms in operation and under development; (b) we derive, from the MILP, a tight lower bound to the optimal solution cost of the IPOWF; (c) we present several models, again derived from the MILP, that can be used to find high-quality solutions to the problem; and (d) we provide managerial insights based on computational experiments on real-life IPOWF instances provided by Vattenfall.

The paper is organized as follows. Section 3.2 reviews the main contributions from the literature on decision-making problems related to the IPOWF. In Section 3.3, we formally define the IPOWF and present a MILP to formulate it. In Section 3.4, we introduce a MILP that provides tight lower bounds to the IPOWF. Section 3.5 presents several mathematical models to compute upper bounds. Section 3.6 documents the computational experiments conducted on real-life instances provided by Vattenfall. Finally, conclusions are drawn in Section 3.7.

3.2 Literature Review

Section 3.2.1 reviews only contributions related to the installation process of offshore wind farms. Section 3.2.2 is devoted to routing problems related to the IPOWF.

3.2.1 Literature on Installation of Offshore Wind Farms

The literature on installation of Offshore Wind Farms is not so rich. To the best of our knowledge it is limited to Scholz-Reiter, Heger, Lütjen & Schweizer (2011), Scholz-Reiter, Karimi, Lütjen, Heger & Schweizer (2011), Ait-Alla et al. (2013), and Irawan, Jones & Ouelhadj (2017).

Scholz-Reiter, Heger, Lütjen & Schweizer (2011) present a MILP to deal with the problem of minimizing the completion time to install offshore wind farms by using a single vessel. The activities that the vessel can complete are grouped in building sub-structures and building top-structures. Weather variability is handled by defining and embedding in the formulation three possible weather conditions: good, medium, and bad weather conditions. Under good weather conditions, all activities can be carried out, so both sub-structures and top-structures can be built. Under medium weather conditions, sub-structures only can be build. Under bad weather conditions, no activities can be carried out.

The same authors of Scholz-Reiter, Heger, Lütjen & Schweizer (2011) propose a heuristic algorithm to solve the same problem but considering larger instances (longer planning horizon, multiple vessel, and more involved weather conditions) in Scholz-Reiter, Karimi, Lütjen, Heger & Schweizer (2011).

Ait-Alla et al. (2013) study the aggregated installation planning of offshore wind farms. They propose an optimization model where the objective is to minimizes the vessels chartering cost. Constraints related to weather conditions, operations type, and vessels availability are taken into account.

Irawan, Jones & Ouelhadj (2017) formulate an installation scheduling problem as a bi-objective optimization problem. The two conflicting objectives are the installation cost and the completion period. Weather conditions are modelled as in Scholz-Reiter, Heger, Lütjen & Schweizer (2011). A MILP is proposed to model the problem, and two different approaches are used to solve it: an exact method based on compromise programming, and a metaheuristic approach mainly based on Variable Neighbourhood Search and Simulated Annealing. The approaches are tested on two datasets generated by the authors.

3.2.2 Literature on Vehicle Routing Problems related to the IPOWF

The IPOWF has many features of some well-known Vehicle Routing Problems (VRP) encountered in road transportation, namely, Multi-Trip VRP with Time Windows (MT-VRPTW). This section reviews first exact methods for MT-VRPTWs and then heuristic methods for MT-VRPTWs.

Exact Methods for MT-VRPTWs

Azi et al. (2007) investigate a single-vehicle MT-VRPTW with a hierarchical objective function that, first, maximizes the number of customers visited and, second, minimizes the travel costs. They consider a deadline constraint that limits the time goods can stay on board. They propose a two-phase exact algorithm and test it on instances adapted from the well-known Solomon instances for the VRPTW (Solomon (1987)) with up to 100 customers. The results show that the algorithm is sensitive to deadline constraints, and, when this is not tight, instances cannot be solved.

The multi-vehicle version of the problem studied in Azi et al. (2007) is addressed in Azi et al. (2010). They present a branch-and-price algorithm able to routinely solve instances with 25 customers to optimality. The performance of the algorithm strongly depends on the tightness of the deadline constraint.

Macedo et al. (2011) propose an iterative two-phase algorithm for the problem of Azi et al. (2010). The method relies on an iterative discretization of the planning horizon. Computational results show that this method outperforms the solution method of Azi et al. (2010).

Hernandez et al. (2014) address an MT-VRPTW where all customers must be served and the goal is to minimize the routing cost. A two-phase exact method based on column generation is proposed. The method can solve instances with 25 and 40 customers.

Hernandez et al. (2016) study the problem of Hernandez et al. (2014) but removing the deadline constraint. Two exact branch-and-price methods are described and tested on instances with 25 customers.

An MT-VRPTW applied to drone distribution is investigated in Cheng et al. (2018). Constraints related to drone battery capacity are considered. They develop two branch-and-cut algorithms that can solve instances with up to 50 customers.

Paradiso et al. (2019) propose an exact solution framework to address four different MT-VRPTWs. The framework is based on a formulation featuring an exponential number of variables and constraints. A solution method based on column generation, column enumeration, and cutting plane is presented. The proposed framework can solve instances with up to 50 customers.

Heuristic Methods for MT-VRPTWs

Battarra et al. (2009) study an MT-VRPTW where the goal is to first minimize the number of trips and then routing costs. They propose an iterative solution approach based on decomposing the problem into simpler ones. Computational experiments are carried out on real-life instances faced in a multi-regional scale distribution problem.

Azi et al. (2014) describe an adaptive large neighborhood serach to solve the problem studied by Azi et al. (2010). The algorithm is tested on instances with up to 100 customers based on the VRPTW instances proposed in Solomon (1987) and instances with up to 1000 customers based on the VRPTW instances introduced by Gehring & Homberger (1999).

Wang et al. (2014) address the same problem addressed by Azi et al. (2014). A metaheuristic based on Adaptive Memory Procedure (AMP) is proposed and tested on the same instances used by Azi et al. (2014). The results show that the metaheuristic based on AMP provides better solutions than the algorithm of Azi et al. (2014) on all instances where all customers are served.

Cattaruzza et al. (2014) present an iterated local search for the Multi-Commodity MT-VRPTW, where different commodities are transported and some of them cannot be transported together. The goal is to minimize the number of vehicles. Benefits of optimally designing the vehicle fleet are showed.

Cattaruzza et al. (2016*a*) introduce the MT-VRPTW and Release Dates and propose an hybrid genetic algorithm to solve it. Instances with up to 100 customers are generated and tested.

Heuristic methods for other MT-VRPTWs encountered in real-life applications are proposed, among others, by Martínez & Amaya (2013), Nguyen et al. (2013), Qin et al. (2015), Anaya-Arenas et al. (2016), Lim et al. (2017), and Nguyen et al. (2017).

3.3 Problem Description and Mathematical Model

In this section, we formally describe the IPOWF, and we provide a compact MILP model to formulate the IPOWF.

3.3.1 Problem Description

An offshore wind farm is to be constructed over a given planning horizon by using a fleet of vessels located at a port. The wind farm consists of a set of $n_{\mathcal{T}}$ turbines $\mathcal{T} = \{1, \ldots, n_{\mathcal{T}}\}$ that have to be constructed at pre-specified locations.

To build each turbine, a set of $n_{\mathcal{O}}$ operations $\mathcal{O} = \{1, \ldots, n_{\mathcal{O}}\}$ must be carried out at the turbine's location. In particular, four operations must be carried out: (1) installation of foundations (MP), (2) installation of transition pieces (TP), (3) installation of cables (cabling), and (4) installation of turbine (TT). For each turbine, the operations must be carried out in the sequence MP-TP-cabling-TT, so each operation can be conducted only if the previous ones have already been completed. To simplify the notation, in the following, we will also refer to MP as operation 1, to TP as operation 2, to cabling as operation 3, and to TT as operation 4.

Operation $o \in \mathcal{O}$ at turbine $t \in \mathcal{T}$ takes d_{ot} hours to be completed and requires some material whose total weight is q_{ot}. Vessels located at the port can transport the required material from the port to the turbine's location. There is a single vessel available for each operation, and each vessel can be used to carry out an operation only, so the set $\mathcal{O}$ can refer both to the operations and to the vessels. Each vessel $o \in \mathcal{O}$ has a maximum load capacity Q_o. Vessels are rented. The weekly rental cost (expressed in euros) of vessel $o \in \mathcal{O}$ is indicated as c_o.

We call *voyage* a sequence of turbines that can be visited by a vessel in between two stops at the port. For each voyage of each vessel, the total weight of the material required to complete the corresponding operation at the turbines of the voyage cannot exceed the vessel's maximum load capacity. Each vessel can perform multiple voyages in the planning horizon. We call *route* the set of voyages performed by a vessel; such voyages cannot overlap in time. Because of this feature, the routing sub-problem related to each vessel is a Multi-Trip VRP with Time Windows (MT-VRPTW). The activity of loading, onto vessel $o \in \mathcal{O}$, the material required in a voyage takes a loading time l_o, which is spent at the port.

The cabling operation has additional requirements compared to the other operations. Due to technical reasons (see Section 3.1), the cabling operations must indeed be carried out string-by-string by the corresponding vessel; this means that, once the vessel completes the cabling operation of the first turbine of a string, it must carry out the cabling operations of the other turbines of the same string (in the order defined by the string itself) before moving to another string. We refer to the set of string as $\mathcal{S} = \{1, \ldots, n_{\mathcal{S}}\}$ and to the turbines of string $s \in \mathcal{S}$ as $\mathcal{T}_s \subseteq \mathcal{T}$. The string of turbine $t \in \mathcal{T}$ is indicated as s_t. The location of turbine $t \in \mathcal{T}$ in the corresponding string is pos_t. The weekly revenue (expressed in euros) derived from selling the energy produced by the turbines of string $s \in \mathcal{S}$ is indicated as r_s. Moreover, for each string $s \in \mathcal{S}$, we denote the first and the last turbine of the string with $f_s \in \mathcal{T}_s$ and $\ell_s \in \mathcal{T}_s$, respectively.

The operations needed to construct the wind farm must be scheduled over a planning horizon consisting of a set of $n_{\mathcal{W}}$ weeks $\mathcal{W} = \{1, \ldots, n_{\mathcal{W}}\}$. The number of available working hours per week depend on the weather forecast and on the operation considered. In particular, operations 1, 2, and 4 can be performed under the same weather conditions, so the number of working hours available in week $w \in \mathcal{W}$ for each of these three operations is indicated as h_w. On the other hand, operation 3 (i.e., cabling) can be performed under different weather conditions, so the working hours available in week $w \in \mathcal{W}$ for cabling is indicated as h'_w, which could differ from h_w. In the following, we use $\underline{h}_w$ ($\underline{h}'_w$, resp.) to refer to the total working hours available for operations 1/2/4 (3, resp.) up to week w, week w excluded (i.e., $\underline{h}_w = \sum_{w'=1}^{w-1} h_{w'}$ and $\underline{h}'_w = \sum_{w'=1}^{w-1} h'_{w'}$). Similarly, we use $\overline{h}_w$ ($\overline{h}'_w$, resp.) to refer to the total working hours available for operations 1/2/4 (3, resp.) up to week w, but including week w (i.e., $\overline{h}_w = \sum_{w'=1}^{w} h_{w'} = \underline{h}_w + h_w$ and $\overline{h}'_w = \sum_{w'=1}^{w} h'_{w'} = \underline{h}'_w + h'_w$).

The time it takes to vessel $o \in \mathcal{O}$ to sail from location i to location j (where i and j can be a turbine location or the port) is indicated as t^o_{ij}, which is computed by taking into account the distance between i and j and the average sailing speed of the vessel.

The goal of the IPOWF is to optimally schedule the operations to complete the wind farm by defining the feasible route of each vessel that minimizes the total rental cost and the missed revenue due to missed production of energy of the turbines before they are completed in such a way that: (1) all operations of all turbines are carried out within the planning horizon; (2) each operation at each turbine is conducted only if the previous operations at the same turbine are completed; (3) the load on the vessels upon leaving the port does not exceed the vessel load capacity; (4) the weekly working hours for each operation are not exceeded; and (5) the cabling operation is scheduled string-by-string.

3.3.2 Mathematical Model

The IPOWF can be represented on a complete directed graph $G = (\mathcal{V}, \mathcal{A})$. The vertex set $\mathcal{V}$ is defined as $\mathcal{V} = \mathcal{T} \cup \{0\} \cup \{0'\}$, where both 0 and $0'$ represent the port, which we duplicate for notational convenience; in particular, 0 ($0'$, resp.) represents the starting (ending, resp.) point of the vessels' routes. The arc set $\mathcal{A}$ is defined as $\mathcal{A} = \{(0, j) \,|\, j \in \mathcal{T}\} \cup \{(i, 0') \,|\, i \in \mathcal{T}\} \cup \mathcal{A}_{\mathcal{T}}$, where $\mathcal{A}_{\mathcal{T}} = \{(i, j) \,|\, i, j \in \mathcal{T} : i \neq j\}$ is the set of arcs linking pairs of turbines.

Let $\mathcal{A}_{\mathcal{S}} \subset \mathcal{A}_{\mathcal{T}}$ be the set of arcs $(i, j) \in \mathcal{A}_{\mathcal{T}}$ such that turbines i and j belong to the same string and i immediately precedes j in the sequence defined by the corresponding string, i.e., $\mathcal{A}_{\mathcal{S}} = \{(i, j) \in \mathcal{A}_{\mathcal{T}} \,|\, s_i = s_j,\, pos_i + 1 = pos_j\}$. Moreover, let $\mathcal{A}' = \{(\ell_{s_1}, f_{s_2}) \,|\, s_1, s_2 \in \mathcal{S}, s_1 \neq s_2\}$ be the set of arcs connecting the last turbine of a string to the first turbine of another string.

To formulate the IPOWF, we introduce the following sets of decision variables:

$x^o_{ij} \in \{0, 1\}$ binary variable equal to 1 arc $(i, j) \in \mathcal{A}$ is traversed by vessel $o \in \mathcal{O}$ (0 otherwise);

$y^o_{ij} \in \{0, 1\}$ binary variable equal to 1 if vessel $o \in \mathcal{O}$ visits turbine $i \in \mathcal{T}$ at the end of a voyage before returning to the port to reload and then leaves the port to visit turbine $j \in \mathcal{T}$ at the beginning of the next voyage (0 otherwise) - in other words, y^o_{ij} is equal to 1 if vessel o traverses arcs $(i, 0')$ and $(0, j)$ one after the other;

$\tau_{ot} \in \mathbb{R}_+$ non-negative real variable indicating the starting time of operation $o \in \mathcal{O}$ at turbine $t \in \mathcal{T}$, expressed in terms of working hours from the start of the planning horizon;

$\lambda_{ot} \in \mathbb{R}_+$ non-negative real variable indicating the total weight of the materials needed by vessel $o \in \mathcal{O}$ up until visiting turbine $i \in \mathcal{T}$ in the corresponding voyage;

$\alpha_{ow} \in \{0, 1\}$ binary variable equal to 1 if the rent of vessel $o \in \mathcal{O}$ starts in week $w \in \mathcal{W}$ (0 otherwise);

$\beta_{ow} \in \{0, 1\}$ binary variable equal to 1 if the rent of vessel $o \in \mathcal{O}$ ends in week $w \in \mathcal{W}$ (0 otherwise);

$\varphi_{sw} \in \{0, 1\}$ binary variable equal to 1 if string $s \in \mathcal{S}$ starts producing energy in week $w \in \mathcal{W}$ (0 otherwise);

$z_{tw} \in \{0, 1\}$ binary variable equal to 1 if the cabling operation at turbine $t \in \mathcal{T}$ starts in week $w \in \mathcal{W}$ (0 otherwise);

$z'_{tw} \in \{0, 1\}$ binary variable equal to 1 if the cabling operation at turbine $t \in \mathcal{T}$ ends in week $w \in \mathcal{W}$ (0 otherwise).

The IPOWF can then be formulated as follows:

$$\min \sum_{o \in \mathcal{O}} c_o \Big(1 + \sum_{w \in \mathcal{W}} w(\beta_{ow} - \alpha_{ow})\Big) + \sum_{s \in \mathcal{S}} r_s \sum_{w \in \mathcal{W}} w \varphi_{sw} \tag{3.1a}$$

The objective function (3.1a) aims at minimizing the total rental cost of the vessels (determined by the first term) and the missed revenue due to strings that are not producing energy until they are completed

(determined by the second term).

$$\text{s.t.} \sum_{(0,j)\in\mathcal{A}} x^o_{0j} = 1 \qquad o \in \mathcal{O} \tag{3.1b}$$

$$\sum_{(j,0')\in\mathcal{A}} x^o_{j0'} = 1 \qquad o \in \mathcal{O} \tag{3.1c}$$

$$\sum_{(i,j)\in\mathcal{A}} x^o_{ij} + \sum_{(i,j)\in\mathcal{A}_\mathcal{T}} y^o_{ij} = 1 o \in \mathcal{O}, \quad i \in \mathcal{T} \tag{3.1d}$$

$$\sum_{(j,i)\in\mathcal{A}} x^o_{ji} + \sum_{(j,i)\in\mathcal{A}_\mathcal{T}} y^o_{ji} = 1 o \in \mathcal{O}, \quad i \in \mathcal{T} \tag{3.1e}$$

Constraints (3.1b) and (3.1c) force each vessel to start and end its route at the port. Constraints (3.1d) and (3.1e) ensure that each operation at each turbine is carried out.

$$\tau_{oi} + d_{oi} + (t^o_{ij} + M)x^o_{ij} + \\ + (l_o + t^o_{i0'} + t^o_{0j} + M)y^o_{ij} \le \tau_{oj} + M \qquad (i,j) \in \mathcal{A}_\mathcal{T}, \quad o \in \mathcal{O} \tag{3.1f}$$

$$\tau_{o0} + (l_o + t^o_{0j} + M)x^o_{0j} \le \tau_{oj} + M \qquad j \in \mathcal{T}, \quad o \in \mathcal{O} \tag{3.1g}$$

$$\tau_{oi} + d_{oi} + (t^o_{i0'} + M)x^o_{i0'} \le \tau_{o0'} + M \qquad i \in \mathcal{T}, \quad o \in \mathcal{O} \tag{3.1h}$$

$$\tau_{oi} + d_{oi} + (t^o_{ij} - M)x^o_{ij} \ge \tau_{oj} - M \qquad (i,j) \in \mathcal{A}_\mathcal{T}, \quad o \in \mathcal{O} \tag{3.1i}$$

$$\tau_{o0} + (l_o + t^o_{0j} - M)x^o_{0j} \ge \tau_{oj} - M \qquad j \in \mathcal{T}, \quad o \in \mathcal{O} \tag{3.1j}$$

$$q_{ot} \le \lambda_{ot} \le Q_o \qquad t \in \mathcal{T}, \quad o \in \mathcal{O} \tag{3.1k}$$

$$\lambda_{oi} + (Q_o + q_{oj})x^o_{ij} + (Q_o - q_{oi})x^o_{ji} \le \lambda_{oj} + Q_o \qquad (i,j) \in \mathcal{A}_\mathcal{T}, \quad o \in \mathcal{O} \tag{3.1l}$$

$$\lambda_{oj} \le q_{oj} + Q_o(1 - x^o_{0j} - \sum_{(i,j)\in\mathcal{A}_\mathcal{T}} y^o_{ij}) \qquad j \in \mathcal{T}, \quad o \in \mathcal{O} \tag{3.1m}$$

Constraints (3.1f)-(3.1j) set the τ-variables to take into account sailing times, jobs duration, and loading times. Constraints (3.1k)-(3.1m) set the λ-variables and act as capacity constraints.

$$\sum_{w\in\mathcal{W}} \underline{h}_w \alpha_{ow} \le \tau_{o0} \qquad o \in \mathcal{O} \setminus \{3\} \tag{3.1n}$$

$$\sum_{w\in\mathcal{W}} \overline{h}_w \beta_{ow} \ge \tau_{o0'} \qquad o \in \mathcal{O} \setminus \{3\} \tag{3.1o}$$

$$\sum_{w\in\mathcal{W}} \underline{h}'_w \alpha_{3w} \le \tau_{30} \tag{3.1p}$$

$$\sum_{w\in\mathcal{W}} \overline{h}'_w \beta_{3w} \ge \tau_{30'} \tag{3.1q}$$

$$\sum_{w\in\mathcal{W}} \alpha_{ow} = \sum_{w\in\mathcal{W}} \beta_{ow} = 1 \qquad o \in \mathcal{O} \tag{3.1r}$$

Constraints (3.1n)-(3.1q) link the starting and the ending time of the routes of the vessels with the variables indicating when the rentals start and end. Constraints (3.1r) ensure that each vessel is rented and dismissed

exactly once.

$$\sum_{w\in\mathcal{W}} \underline{h}'_w z_{tw} \le \tau_{3t} \le \sum_{w\in\mathcal{W}} \overline{h}'_w z_{tw} \qquad t\in\mathcal{T} \tag{3.1s}$$

$$\sum_{w\in\mathcal{W}} \underline{h}'_w z'_{tw} \le \tau_{3t} + d_{t3} \le \sum_{w\in\mathcal{W}} \overline{h}'_w z'_{tw} \qquad t\in\mathcal{T} \tag{3.1t}$$

$$\sum_{w\in\mathcal{W}} z_{tw} = \sum_{w\in\mathcal{W}} z'_{tw} = 1 \qquad t\in\mathcal{T} \tag{3.1u}$$

$$\tau_{1t} + d_{1t} \le \tau_{2t} \qquad t\in\mathcal{T} \tag{3.1v}$$

$$\tau_{2t} + d_{2t} \le \sum_{w\in\mathcal{W}} \underline{h}_w z_{wt} \qquad t\in\mathcal{T} \tag{3.1w}$$

$$\sum_{w\in\mathcal{W}} \overline{h}_w z'_{tw} \le \tau_{4t} \qquad t\in\mathcal{T} \tag{3.1x}$$

$$\tau_{n_{\mathcal{O}}\ell_s} + d_{n_{\mathcal{O}}\ell_s} \le \sum_{w\in\mathcal{W}} \overline{h}_w \varphi_{sw} \qquad s\in\mathcal{S} \tag{3.1y}$$

$$\sum_{w\in\mathcal{W}} \varphi_{ws} = 1 \qquad s\in\mathcal{S} \tag{3.1z}$$

$$x^3_{ij} + y^3_{ij} = 1 \qquad (i,j)\in\mathcal{A}_{\mathcal{S}} \tag{3.1aa}$$

Constraints (3.1s)-(3.1u) allow to compute the initial and ending week where the cabling operation is carried out at each turbine. Constraints (3.1v)-(3.1x) are precedence constraints that ensure that, at each turbine, the four operations are performed in the right order and do not overlap. Constraints (3.1y)-(3.1z) compute the week in which each string can start producing energy. Constraints (3.1aa) guarantee that cabling operations are performed string by string.

$$x^o_{ij} \in \{0,1\} \qquad (i,j)\in\mathcal{A},\quad o\in\mathcal{O} \tag{3.1ab}$$

$$y^o_{ij} \in \{0,1\} \qquad (i,j)\in\mathcal{A}_{\mathcal{T}},\quad o\in\mathcal{O} \tag{3.1ac}$$

$$\tau_{ot} \in \mathbb{R}_+ \qquad t\in\mathcal{V},\quad o\in\mathcal{O} \tag{3.1ad}$$

$$\lambda_{ot} \in \mathbb{R}_+ \qquad t\in\mathcal{T},\quad o\in\mathcal{O} \tag{3.1ae}$$

$$\alpha_{ow} \in \{0,1\} \qquad \beta_{ow} \in \{0,1\} \qquad o\in\mathcal{O},\quad w\in\mathcal{W} \tag{3.1af}$$

$$\varphi_{sw} \in \{0,1\} \qquad s\in\mathcal{S},\quad w\in\mathcal{W} \tag{3.1ag}$$

$$z_{tw} \in \{0,1\} \qquad z'_{tw} \in \{0,1\} \qquad t\in\mathcal{T},\quad w\in\mathcal{W} \tag{3.1ah}$$

Constraints (3.1ab)-(3.1ah) define the domain of the decision variables. In the following, we shortly refer to model (3.1) as $\mathrm{M_G}$.

In principle, model $\mathrm{M_G}$ could be solved with a general-purpose MILP solver to achieve an optimal solution to the IPOWF. Unfortunately, it is too complex to solve even for the most advanced MILP solvers currently available, especially when it comes to solving real-life instances. Therefore, in order to achieve high-quality upper bounds by using $\mathrm{M_G}$, some simplifications are needed. In Section 3.5, we will show how $\mathrm{M_G}$ can be simplified to obtain easier MILPs that can be efficiently handled by MILP solvers and provide high-quality upper bounds. We would also like to assess the quality of such upper bounds by computing lower bounds to the IPOWF. In the next section, we propose a MILP derived from $\mathrm{M_G}$ that provides high-quality lower bounds in short amounts of computing time. We present the model to achieve lower bounds first because some modeling decisions made to derive such a model are instrumental to derive the upper bounds described in the following section.

3.4 A Lower Bound to the IPOWF

The idea behind the model to compute lower bounds to the optimal solution of a generic IPOWF instance $\mathcal{I}$ is to solve a IPOWF instance $\mathcal{I}'$, derived from $\mathcal{I}$, with a simplified MILP model derived from $\mathrm{M_G}$. In the following, we first describe how instance $\mathcal{I}'$ is derived from $\mathcal{I}$, and we then describe the simplified model to solve $\mathcal{I}'$.

The simplified IPOWF instance $\mathcal{I}'$ is obtained from $\mathcal{I}$ by applying the following changes (all input parameters that are not mentioned remain the same):

- *Turbines Overlap*: all turbines in $\mathcal{I}'$ are located at the same point, namely, at the position of the turbines of $\mathcal{I}$ that is closest to the port; under this condition, travel times of $\mathcal{I}'$, $\tilde{t}^o_{ij}$, representing the time it takes for vessel $o \in \mathcal{O}$ to traverse arc $(i,j) \in \mathcal{A}$ are defined as follows:

$$\tilde{t}^o_{ij} = 0 \qquad o \in \mathcal{O}, \quad (i,j) \in \mathcal{A}_{\mathcal{T}} \tag{3.2a}$$

$$\tilde{t}^o_{0j} = \min_{i \in \mathcal{T}} \{t^o_{0i}\} \qquad o \in \mathcal{O}, \quad j \in \mathcal{T} \tag{3.2b}$$

$$\tilde{t}^o_{j0'} = \min_{i \in \mathcal{T}} \{t^o_{i0'}\} \qquad o \in \mathcal{O}, \quad j \in \mathcal{T} \tag{3.2c}$$

  Notice that because of (3.2b), for a given operation $o \in \mathcal{O}$, $\tilde{t}^o_{0j}$ is equal for any turbine $j \in \mathcal{T}$, so we can redefine $\tilde{t}^o_{0j}$ as $t'^o_0 = \tilde{t}^o_{0j}$. Similarly, we can define $t'^o_{0'} = \tilde{t}^o_{0'j}$

- *Homogeneous Turbines*: all turbines of $\mathcal{I}'$ are of the same type, which implies that, for each operation, all turbines require the same material and the same amount of time to be completed; under this condition, the weight, q'_o, of the materials required by any turbine for operation $o \in \mathcal{O}$ and the duration, d'_o, of operation $o \in \mathcal{O}$ for any turbine of $\mathcal{I}'$ are defined as follows:

$$q'_o = \min_{i \in \mathcal{T}} \{q_{oi}\} \quad o \in \mathcal{O} \tag{3.3}$$

$$d'_o = \min_{i \in \mathcal{T}} \{d_{oi}\} \quad o \in \mathcal{O} \tag{3.4}$$

- *Homogeneous Strings for Missed Revenue*: the last change affects the strings of $\mathcal{I}'$, $\mathcal{S}'$, and the corresponding turbines $\mathcal{T}'_{s'}$, for each $s' \in \mathcal{S}'$; the idea is that all strings that contribute to the missed revenue in the objective function have the same number of turbines (equal to the lowest number of turbines of any string in the set $\mathcal{S}$) and any additional turbine is removed from the strings of the set $\mathcal{S}$ and represents a new single-turbine string with no revenue; given the string with the minimum number of turbines in the original set $\mathcal{S}$, $s_t = \min_{s \in \mathcal{S}} |\mathcal{T}_s|$, the set of strings $\mathcal{S}'$ of $\mathcal{I}'$, where each $s' \in \mathcal{S}'$ consists of a set of turbines $\mathcal{T}'_{s'}$, is defined as follows:

  1. For each $s \in \mathcal{S}$ of $\mathcal{I}$, define a string s' in the set $\mathcal{S}'$ such that $|\mathcal{T}_{s'}| = s_t$ and the corresponding set of turbines $\mathcal{T}'_{s'}$ contains the first s_t strings of $\mathcal{T}_s$; moreover, set the weekly revenue of string s' as $r_{s'} = r_s$;
  2. Let $\tilde{\mathcal{T}}$ be the set of the turbines $t \in \mathcal{T}$ not associated to any string $s \in \mathcal{S}'$; for each turbine $\tilde{t} \in \tilde{\mathcal{T}}$, add a new string $\tilde{s}$ consisting of a single turbine $\tilde{t}$ and characterized by no revenue, i.e., $r_{\tilde{s}} = 0$.

For example, suppose that there are nine turbines in $\mathcal{I}$, i.e., $\mathcal{T} = \{1,2,3,4,5,6,7,8,9\}$, and the set $\mathcal{S}$ contains three strings s_1, s_2, and s_3 such that $\mathcal{T}_{s_1} = \{1,2\}$, $\mathcal{T}_{s_2} = \{3,4,5\}$, and $\mathcal{T}_{s_3} = \{6,7,8,9\}$. Then, the minimum number of turbines in a string is two, i.e., $s_t = |\mathcal{T}_{s_1}| = 2$. In the first step of the procedure to define the set $\mathcal{S}'$ of $\mathcal{I}'$, three strings s'_1, s'_2, and s'_3 are generated, where $\mathcal{T}'_{s'_1} = \{1,2\}$, $\mathcal{T}'_{s'_2} = \{3,4\}$, $\mathcal{T}'_{s'_3} = \{6,7\}$, and $r_{s'_1} = r_{s_1}$, $r_{s'_2} = r_{s_2}$, $r_{s'_3} = r_{s_3}$. In the second step, the set $\tilde{\mathcal{T}}$ contains the three turbines 5, 8, and 9; therefore, three strings $\mathcal{T}'_{s'_4} = 5$, $\mathcal{T}'_{s'_5} = 8$, $\mathcal{T}'_{s'_6} = 9$ are additionally defined and $r_{s'_4} = r_{s'_5} = r_{s'_6} = 0$. In the end, the set $\mathcal{S}'$ contains six strings $\mathcal{S}' = s'_1, s'_2, s'_3, s'_4, s'_5, s'_6$.

It is quite easy to observe that, because of the way instance $\mathcal{I}'$ is constructed, the cost of any optimal solution of $\mathcal{I}'$ is a valid lower bound to the cost of any optimal solution of $\mathcal{I}$.

In order to solve $\mathcal{I}'$, model $\mathrm{M_G}$ can be adapted as follows:

$$\min \sum_{o\in\mathcal{O}} c_o\Big(1 + \sum_{w\in\mathcal{W}} w(\beta_{ow} - \alpha_{ow})\Big) + \sum_{s\in\mathcal{S}'} r_s \sum_{w\in\mathcal{W}} w\varphi_{sw} \tag{3.5a}$$

$$\text{s.t. } \tau_{oi} + d'_o + Mx^o_{ij} + (l_o + t'^o_{0'} + t'^o_0 + M)y^o_{ij} \le \tau_{oj} + M \qquad (i,j)\in\mathcal{A}_\mathcal{T},\quad o\in\mathcal{O} \tag{3.5b}$$

$$\tau_{o0} + (l_o + t'^o_0 + M)x^o_{0j} \le \tau_{oj} + M \qquad j\in\mathcal{T},\quad o\in\mathcal{O} \tag{3.5c}$$

$$\tau_{oi} + d'_o + (t'^o_{0'} + M)x^o_{i0'} \le \tau_{o0'} + M \qquad i\in\mathcal{T},\quad o\in\mathcal{O} \tag{3.5d}$$

$$\tau_{oi} + d'_o - Mx^o_{ij} \ge \tau_{oj} - M \qquad (i,j)\in\mathcal{A}_\mathcal{T},\quad o\in\mathcal{O} \tag{3.5e}$$

$$\tau_{o0} + (l_o + t'^o_0 - M)x^o_{0j} \ge \tau_{oj} - M \qquad j\in\mathcal{T},\quad o\in\mathcal{O} \tag{3.5f}$$

$$q'_o \le \lambda_{ot} \le Q_o \qquad t\in\mathcal{T},\quad o\in\mathcal{O} \tag{3.5g}$$

$$\lambda_{oi} + (Q_o + q'_o)x^o_{ij} + (Q_o - q'_o)x^o_{ji} \le \lambda_{oj} + Q_o \qquad (i,j)\in\mathcal{A}_\mathcal{T},\quad o\in\mathcal{O} \tag{3.5h}$$

$$\lambda_{oj} \le q'_o + Q_o(1 - x^o_{0j} - \sum_{(i,j)\in\mathcal{A}_\mathcal{T}} y^o_{ij}) \qquad j\in\mathcal{T},\quad o\in\mathcal{O} \tag{3.5i}$$

$$\sum_{w\in\mathcal{W}} \underline{h}'_w z'_{tw} \le \tau_{3t} + d'_3 \le \sum_{w\in\mathcal{W}} \overline{h}'_w z'_{tw} \qquad t\in\mathcal{T} \tag{3.5j}$$

$$\tau_{1t} + d'_1 \le \tau_{2t} \qquad t\in\mathcal{T} \tag{3.5k}$$

$$\tau_{2t} + d'_2 \le \sum_{w\in\mathcal{W}} \underline{h}_w z_{wt} \qquad t\in\mathcal{T} \tag{3.5l}$$

$$\tau_{n_\mathcal{O}\ell_s} + d'_{n_\mathcal{O}} \le \sum_{w\in\mathcal{W}} \overline{h}_w \varphi_{sw} \qquad s\in\mathcal{S} \tag{3.5m}$$

$$+ (3.1b) - (3.1e), (3.1n) - (3.1s), (3.1u), (3.1x), (3.1z) - (3.1ah)$$

where constraints (3.5b)-(3.5f), (3.5g)-(3.5i), (3.5j), (3.5k)-(3.5l), (3.5m) replace constraints (3.1f)-(3.1j), (3.1k)-(3.1m), (3.1t), (3.1v)-(3.1w), (3.1y), respectively. Hereafter, we refer to model (3.5) as $\mathrm{M}^{\mathrm{lb}}_{\mathrm{G}}$.

$\mathrm{M}^{\mathrm{lb}}_{\mathrm{G}}$ can still be too difficult to solve with general-purpose MILP solvers. However, it is possible to derive some features of some of the optimal solutions of $\mathcal{I}'$ that can be used to add additional constraints to restrict the solution space of $\mathrm{M}^{\mathrm{lb}}_{\mathrm{G}}$ without cutting off all the optimal solutions of $\mathcal{I}'$. Adding such constraints allows to help the solution of $\mathrm{M}^{\mathrm{lb}}_{\mathrm{G}}$.

Theorem 3.1. *There always exists an optimal solution of model M_G^{lb} such that*

all vessels carry out the corresponding operation in the same sequence of turbines, (3.6a)

the sequence of turbines visited by each vessel follows a string-by-string policy, (3.6b)

and the sequence of strings followed by each vessel is by non-increasing value of productivity of the corresponding strings. (3.6c)

Proof. Let $\mathcal{X} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\tau}, \boldsymbol{\lambda}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\varphi}, \boldsymbol{z}, \boldsymbol{z'})$ be a feasible solution of model $\mathrm{M}_{\mathrm{G}}^{\mathrm{lb}}$. Let us assume that $\mathcal{X}$ does not have all the three properties (3.6). Let now $\mathcal{X}'$ be the solution of $\mathrm{M}_{\mathrm{G}}^{\mathrm{lb}}$ obtained by reordering, in the route of each vessel, the turbines of $\mathcal{X}$ in such a way that properties (3.6) are satisfied. We can prove that solution $\mathcal{X}'$ is feasible, has the same rental cost of $\mathcal{X}$, and has the same or a lower missed revenue than $\mathcal{X}$.

First, we prove that $\mathcal{X}'$ is feasible. Then, we prove that $\mathcal{X}'$ has the same rental cost of $\mathcal{X}$. Finally, we prove that $\mathcal{X}'$ has the same or a lower missed revenue of $\mathcal{X}$.

Feasibility. To prove that $\mathcal{X}'$ is feasible, we prove that (i) it satisfies capacity constraints, (ii) operation 3 is conducted string-by-string, (iii) for each vessel, the start time of the corresponding operation of the p^{th} turbine in the corresponding route of $\mathcal{X}$ and $\mathcal{X}'$ is the same, and (iv) it satisfies precedence constraints.

With regard to capacity constraints (i.e., constraints (3.5g)-(3.5i)), we can observe that, given the feasible route of a given vessel $o \in \mathcal{O}$ of $\mathcal{X}$, any permutation of the turbines in the route maintains the feasibility of the route (in terms of capacity) because q'_o does not depend on the turbine.

As to the constraint that operation 3 must be conducted string-by-string, $\mathcal{X}'$ satisfies such a condition by definition (as all operations are conducted string-by-string, see condition (3.6b)).

The condition that, for each vessel, the start time of the corresponding operation of the p^{th} turbine in the corresponding route of $\mathcal{X}$ and $\mathcal{X}'$ is the same holds because for any operation $o \in \mathcal{O}$ and for any pair of arcs $(i,j),(k,\ell) \in \mathcal{A}_{\mathcal{T}}$, $(i,j) \neq (k,\ell)$, we have $\tilde{t}^o_{ij} = \tilde{t}^o_{k\ell}$, so, in constraints (3.5b) and (3.5e), variables x^o_{ij} and $x^o_{k\ell}$ have the same coefficients. Similarly, for any operation $o \in \mathcal{O}$ and for any pair of turbines $i, j \in \mathcal{T}$, $i \neq j$, we have $\tilde{t}^o_{0j} = \tilde{t}^o_{0j}$ and $\tilde{t}^o_{i0'} = \tilde{t}^o_{j0'}$, so, in constraints (3.5b)-(3.5f), variable x^o_{0i} has the same coefficients of variable x^o_{0j}, and variable $x^o_{i0'}$ has the same coefficients of variable $x^o_{j0'}$. Consequently, in any permutation of the turbines in the routes of the vessels, the value of the τ_{oi_p} variable of the p^{th} turbine of the route of vessel $o \in \mathcal{O}$ is the same, which implies that as $\mathcal{X}$ is feasible, so is $\mathcal{X}'$.

As to precedence constraints, because $\mathcal{X}$ is feasible, we can observe that, when operation $o \in \mathcal{O} \setminus \{1\}$ at the p^{th} turbine of the route of vessel $o \in \mathcal{O}$ starts, operation $o-1$ must be completed for at least the first p turbines of the route of vessel $o \in \mathcal{O}$. As solution $\mathcal{X}'$ has property (3.6a), for any $p = 1, \ldots, n_{\mathcal{T}}$, the p^{th} turbine of any of the routes is the same. Therefore, precedence constraints are satisfied for $\mathcal{X}'$.

Rental cost. As we have shown that, for each vessel, the start time of the corresponding operation of the p^{th} turbine in the corresponding route of $\mathcal{X}$ and $\mathcal{X}'$ is the same, we can also conclude that, for each operation $o \in \mathcal{O}$, the values of variables τ_{o0} and $\tau_{o0'}$ is the same in both solutions $\mathcal{X}$ and $\mathcal{X}'$. Therefore, the rental cost of both solutions is the same.

Missed revenue. Because of condition (3.6c), all single-turbine strings with zero revenue are scheduled last in all routes of $\mathcal{X}'$, in particular in route of vessel 4. This is clearly an optimal decision in terms of missed revenue. As to the turbines belonging to the other strings (i.e., those with a strictly positive revenue), they are scheduled by string and in non-increasing order of missed revenue in $\mathcal{X}'$ because of

(3.6b)-(3.6c). It is easy to see that, as the start time of operation 4 of the p^{th} turbine is the same in both $\mathcal{X}$ and $\mathcal{X}'$ and all strings with a strictly positive revenue have the same number of turbines, it is always cheaper (in terms of missed revenue) to complete one string at a time in non-increasing order of missed revenue.

As $\mathcal{X}'$ is feasible, has the same rental cost of $\mathcal{X}$, and has a missed revenue which is not higher than $\mathcal{X}$, then there will always exists an optimal solution of $\mathcal{I}'$ that satisfies properties (3.6). □

Lemma 3.1. *The optimal solution cost of M_G^{lb} does not increase if all feasible solutions of $\mathcal{I}'$ that do not satisfy* (3.6) *are cut off.*

Proof. Lemma 3.1 immediately follows from Theorem 3.1. □

Let us assume w.l.o.g. that the strings of the set $\mathcal{S}'$ are indexed by non-increasing value of productivity, and let us define a set of arcs $\mathcal{B}$ as follows $\mathcal{B} = \{(i,j) \,|\, i,j \in \mathcal{S}', s_i = s_j, pos_i + 1 = pos_j\} \cup \{(0, f_1), (\ell_{|\mathcal{S}'|}, 0')\} \cup \{(\ell_{s-1}, f_s) \,|\, s = 2, \ldots, |\mathcal{S}'|\}$. Because of Theorem 3.1 and Lemma 3.1, it is possible to find an optimal solution of $\mathcal{I}'$, without increasing its optimal cost, by adding, to $\text{M}_\text{G}^{\text{lb}}$, the following set of constraints

$$x_{ij}^o + y_{ij}^o = 1 \quad o \in \mathcal{O}, \quad (i,j) \in \mathcal{B} \tag{3.7}$$

which impose on the routes of any solution of $\text{M}_\text{G}^{\text{lb}}$ to satisfy conditions (3.6). Hereafter, we refer to model $\text{M}_\text{G}^{\text{lb}}$ with constraints (3.7) as $\text{M}_\text{FS}^{\text{lb}}$.

3.5 Upper Bounds to the IPOWF

Model M_G cannot be used to solve to optimality IPOWF instances of size comparable to problems faced in real-life applications. Therefore, in this section, we present four MILPs derived from M_G that provide upper bounds to the IPOWF.

3.5.1 Model 1: Same-Sequence Model

The first model we present is the Same-Sequence Model. The idea is to add, to M_G, the constraint that all vessels have to visit the turbines in the same sequence. This can be imposed by adding the following constraints:

$$x_{ij}^{o-1} + y_{ij}^{o-1} = x_{ij}^o + y_{ij}^o \qquad (i,j) \in \mathcal{A}_\mathcal{T}, \quad o \in \mathcal{O} \setminus \{1\} \tag{3.8}$$

As the vessel of the cabling operation must visit the turbines in a string-by-string fashion, then all other vessels are forced to follow this policy as a result of adding constraints (3.8). Therefore, the following constraints can also be added:

$$x_{ij}^o = 0 \qquad (i,j) \in \mathcal{A}_\mathcal{T} \setminus \{\mathcal{A}_\mathcal{S} \cup \mathcal{A}'\}, \quad o \in \mathcal{O} \tag{3.9a}$$
$$y_{ij}^o = 0 \qquad (i,j) \in \mathcal{A}_\mathcal{T} \setminus \{\mathcal{A}_\mathcal{S} \cup \mathcal{A}'\}, \quad o \in \mathcal{O} \tag{3.9b}$$

Constraints (3.9) set equal to 0 all variables that do not adhere to the string-by-string policy.

In the following, we shortly refer to the model obtained by adding to M_G constraints (3.8) and (3.9) as M_{SS}^{ub}.

3.5.2 Model 3: Fixed Sequence Model

The second model, we call Fixed Sequence Model, is inspired by Theorem 3.1. Indeed, the second model is derived from M_{SS}^{ub} by adding the constraint that the sequence of turbines of each vessel is fixed a-priori. In particular, this sequence follows a string-by-string policy and orders strings by non-increasing value of weekly string productivity.

Let us assume w.l.o.g. that the strings of the set $\mathcal{S}$ are indexed by non-increasing value of productivity, and let us define a set of arcs $\mathcal{B}$ as follows $\mathcal{B} = \{(i,j) \,|\, i,j \in \mathcal{S}, s_i = s_j, pos_i + 1 = pos_j\} \cup \{(0,f_1),(\ell_{|\mathcal{S}|},0')\} \cup \{(\ell_{s-1},f_s) \,|\, s = 2,\dots,|\mathcal{S}|\}$. Then, the Fixed Sequence Model is obtained by adding, to M_{SS}^{ub}, the following set of constraints:

$$x_{ij}^o + y_{ij}^o = 1 \qquad o \in \mathcal{O}, \quad (i,j) \in \mathcal{B}$$

In the following, we shortly refer to the Fixed Sequence Model as M_{FS}^{ub}.

3.5.3 Models 4 and 5: Missed Revenue Model and Rental Cost Model

The last two models we consider are based on the idea of solving the IPOWF by considering one of the two cost components at a time. The two derived models represent upper bounds of M_G, but give also interesting insights to practitioners.

When the missed revenue is the only cost component considered in the objective function, model M_G can be simplified by removing variables α_{ow} and β_{ow} ($o \in \mathcal{O}$, $w \in \mathcal{W}$) and constraints (3.1n)-(3.1r). Moreover, the objective function (3.1a) is simplified to

$$\min \sum_{s \in \mathcal{S}} r_s \sum_{w \in \mathcal{W}} w \varphi_{sw} \tag{3.10}$$

We call the resulting formulation Missed Revenue Model, and we shortly refer to it as M_{MR}^{ub}.

On the other hand, when only rental costs are taken into account and missed revenues are ignored, variables φ_{sw} ($s \in \mathcal{S}$, $w \in \mathcal{W}$) and constraints (3.1y)-(3.1z) can be removed, and the objective function (3.1a) changes to

$$\min \sum_{o \in \mathcal{O}} c_o \Big(1 + \sum_{w \in \mathcal{W}} w(\beta_{ow} - \alpha_{ow})\Big)$$

We call the resulting formulation Rental Cost Model, and we shortly refer to it as M_{RC}^{ub}.

Note that, in order to provide comparable upper bounds for M_G, once a (sub)optimal solution of the resulting models is achieved, the cost of the cost component ignored can be computed a-posteriori to assess the overall cost of the solution.

3.6 Computational Experiments

In this section, we report the computational results achieved by the mathematical models described in Sections 3.5 (i.e., models M_{SS}^{ub}, M_{FS}^{ub}, M_{MR}^{ub}, and M_{RC}^{ub}) on real-life instances provided by Vattenfall. We

also compare the upper bounds achieved by these four models with the lower bounds provided by solving M_{FS}^{lb}.

All models are implemented in OPL and solved with the commercial solver IBM ILOG CPLEX Optimization Studio 12.8. Tests are conducted in single-thread mode on a machine equipped with a Intel Core i7-6700k 4.00 GHz and 24 GB of RAM with a time limit of 3600 seconds.

3.6.1 Description of the Test Instances

The test instances of this study are based on two offshore wind farms owned by Vattenfall, namely, Horn Rev 3 and Sanbank. Horns Rev 3 is located in the North Sea, 25-40 km off the Danish cost and is, as of today, Denmark's largest offshore wind farm. Its installation has been completed this year (2019). The farm will provide enough power to satisfy the annual electricity consumption of 425,000 Danish households. The farm consists of 12 strings, each one made up of four or five turbines, for a total of 49 turbines. Sandnbank instead has 72 smaller (4MW) wind turbines, organized in 8 strings. With an overall capacity of 288 megawatts, the wind farm provides green power to supply up to 400,000 households. The wind farm extends over an area of 60 square kilometres and is located 90 kilometres off the coast of Germany. The offshore wind farm has been finalized in 2017. We used the real data as provided by our industrial partner Vattenfall, and we also generated an additional dataset of 18 instances of different sizes from them: 12 of these instances are derived from Horn Rev 3, and the remaining 6 instances are derived from Sanbank. Each instance contains a subset of strings of the corresponding original instance. For every instance, the planning horizon is a calendar year (January-December).

Table 3.6.1: Features of Horn Rev 3 instances

Inst	$n_{\mathcal{S}}$	$\lvert\mathcal{T}_s\rvert$	$n_{\mathcal{T}}$
HR3.s4.t4	4	1	4
HR3.s4.t8	4	2	8
HR3.s4.t12	4	3	12
HR3.s4.t17	4	4 or 5	17
HR3.s8.t8	8	1	8
HR3.s8.t16	8	2	16
HR3.s8.t24	8	3	24
HR3.s8.t33	8	4 or 5	33
HR3.s12.t12	12	1	12
HR3.s12.t24	12	2	24
HR3.s12.t36	12	3	36
HR3.s12.t49	12	4 or 5	49

Table 3.6.2: Features of Sanbank instances

Inst	$n_{\mathcal{S}}$	$\lvert\mathcal{T}_s\rvert$	$n_{\mathcal{T}}$
SB.s4.t12	4	3	12
SB.s4.t24	4	6	24
SB.s4.t36	4	9	36
SB.s8.t24	8	3	24
SB.s8.t48	8	6	48
SB.s8.t72	8	9	72

The features of the 18 instances are summarized in Tables 3.6.1 and 3.6.2, reporting the 12 instances derived from Horn Rev 3 and the 6 instances derived from Sanbank, respectively. For each instance, we report the instance name (Inst), the number of strings ($n_{\mathcal{S}}$), the number of turbines per string ($|\mathcal{T}_s|$), and the total number of turbines ($n_{\mathcal{T}}$). To identify the instances, we use the following naming convention TT.sXX.tYY, where TT indicates the wind farm (i.e., HR3: Horn Rev 3; SB: Sanbank), XX the number of strings, and YY the total number of turbines. Notice that the last instance of each table (i.e., HR3.s12.t49 and SB.s8.t72) corresponds to the whole wind farms built by Vattenfall.

3.6.2 Computational Results

Tables 3.6.3 and 3.6.4 show the upper bounds achieved by models M_G, M^{ub}_{SS}, M^{ub}_{FS}, M^{ub}_{MR} and M^{ub}_{RC}, and the lower bounds achieved by M^{lb}_{FS} on the Horn Rev 3 and the Sanbank instances, respectively.

In both tables, the first column reports the instance name (Instance). Then, for each of the five models M_G, M^{ub}_{SS}, M^{ub}_{FS}, M^{ub}_{MR}, and M^{ub}_{RC}, three values are reported: the total cost (TC, expressed in thousands of euros) of the best solution found (a dash indicates that no feasible solution was found), the total computing time (CPU, where tl stands for *time limit*), and the gap (gap), expressed in percentage, between the best solution found and the best available lower bound. The last column shows the value of the best lower bound found by solving model M^{lb}_{FS} (lb_{FS}). Note that when model M^{lb}_{FS} is not solved to optimality within the time limit, the reported value is the current lower bound indicated by the solver. The best upper bounds achieved on each instance are highlighted in bold. The last three rows of each table report the number of instances for which each model finds a feasible solution, the number of instances for which each model finds an optimal solution, and the average gap over all instances. In order to make a proper comparison, the cost for solutions of models M^{ub}_{MR} and M^{ub}_{RC} has been post-processed by adding the missing cost (i.e., rental costs and missed revenue, respectively).

Tables 3.6.3 and 3.6.4 show that models M^{ub}_{SS} and M^{ub}_{FS} clearly outperform the other three models in terms of number of instances where a feasible solution is provided; indeed, they are the only two models that can find a feasible solution for all instances. Model M^{ub}_{FS} can find the best solution on all but one instance (i.e., instance HR3.s4.t17), and six of these solutions are also optimal. In terms of average gap, model M^{ub}_{FS} is clearly the best as the average gap on the Horn Rev 3 instances is 1.6% and on the Sanbank instances is 0.4%, whereas the second best model (i.e., M^{ub}_{SS}) has an average gap of 3.8% and 3.3%, respectively. We can also observe that the upper bounds provided by M^{ub}_{FS} and the lower bounds provided by M^{lb}_{FS} are of high-quality as the maximum gap is 3.4% on the Horn Rev 3 instances and 1.4% on the Sanbank instances.

We can notice that even though M^{ub}_{FS} is more constrained than M_G and M^{ub}_{SS}, it is able to provide better upper bounds on all but one instance (i.e., HR3.s4.t17) - this is probably due to the computational complexity of solving models M_G and M^{ub}_{SS}. From a managerial viewpoint, this result suggests that it is often a good decision to design solutions to the IPOWF that satisfy properties (3.6). Nevertheless, such a decision may not be optimal as shown by the fact that M^{ub}_{SS} finds an upper bound to instance HR3.s4.t17 that is strictly better than the upper bound provided by M^{ub}_{FS}.

By looking at the low quality of the upper bounds provided by M^{ub}_{MR} and M^{ub}_{RC}, even on small instances, we can draw the conclusion that both cost components (rental costs and missed revenue) should be taken into account when solving the IPOWF.

3.6.3 Analysis on the Starting Period of Operations

The tests summarized in Section 3.6.2 are conducted by assuming that the planning horizon is a calendar year, which corresponds to the decision-making problem faced by Vattenfall. This means that we assume that all materials, vessels, and crew members are available from the 1st of January onward, and it is Vattenfall's choice to decide when to use them (i.e., start renting vessels and installing turbines). Scheduling the installation operations at the beginning of the year has the advantage of producing and selling energy from the wind farm as soon as possible, but winter months have also the worst weather conditions, so the operations may last longer, resulting in higher rental costs. For this reason, it may be

Table 3.6.3: Computational results on the Horn Rev 3 instances

	M_G			M^{ub}_{SS}			M^{ub}_{FS}			M^{ub}_{MR}			M^{ub}_{RC}			M^{lb}_{FS}
Instance	TC	CPU	gap	TC	CPU	gap	TC	CPU	gap	TC	CPU	gap	TC	CPU	gap	lb_{FS}
HR3.s4.t4	**11,032**	tl	0.0	**11,032**	tl	0.0	**11,032**	4	0.0	14,958	tl	26.2	14,901	tl	26.0	11,032
HR3.s4.t8	21,033	tl	5.8	**20,006**	tl	1.0	**20,006**	48	1.0	26,060	tl	24.0	25,381	tl	21.9	19,811
HR3.s4.t12	30,487	tl	10.7	**28,206**	tl	3.4	**28,206**	17	3.4	33,575	tl	18.9	38,362	tl	29.0	27,239
HR3.s4.t17	43,891	tl	12.2	**39,545**	tl	2.5	39,784	13	3.1	45,277	tl	14.9	57,749	tl	33.3	38,541
HR3.s8.t8	20,375	tl	3.2	20,061	tl	1.6	**19,926**	22	1.0	28,510	tl	30.8	25,864	tl	23.7	19,731
HR3.s8.t16	60,778	tl	41.2	37,170	tl	3.9	**35,732**	13	0.0	42,253	tl	15.4	52,053	tl	31.4	35,732
HR3.s8.t24	-	tl		54,996	tl	3.1	**54,099**	35	1.5	79,300	tl	32.8	85,369	tl	37.6	53,297
HR3.s8.t33	-	tl		75,417	tl	2.9	**74,514**	85	1.8	-	tl		115,274	tl	36.5	73,204
HR3.s12.t12	34,856	tl	22.3	30,130	tl	10.1	**28,045**	49	3.4	37,873	tl	28.5	43,011	tl	37.0	27,078
HR3.s12.t24	-	tl		55,993	tl	6.0	**53,425**	60	1.4	74,050	tl	28.9	79,441	tl	33.7	52,652
HR3.s12.t36	-	tl		82,426	tl	1.7	**80,988**	128	0.0	-	tl		-	tl		80,988
HR3.s12.t49	-	tl		119,683	tl	8.7	**111,398**	19	1.9	-	tl		-	tl		109,267
Feas. sol. found			7			12			12			9			10	
Opt. sol. found			1			1			3			0			0	
Avg gap			13.6			3.8			1.6			24.5			31.0	

Table 3.6.4: Computational results on the Sanbank instances

	M_G			M^{ub}_{SS}			M^{ub}_{FS}			M^{ub}_{MR}			M^{ub}_{RC}			M^{lb}_{FS}
Instance	TC	CPU	gap	TC	CPU	gap	TC	CPU	gap	TC	CPU	gap	TC	CPU	gap	lb_{FS}
SB.s4.t12	24,734	tl	17.7	**20,641**	tl	1.4	**20,641**	17	1.4	37,207	tl	45.3	31,630	tl	35.7	20,348
SB.s4.t24	53,579	tl	23.7	41,550	tl	1.6	**40,874**	71	0.0	-	tl		69,998	tl	41.6	40,874
SB.s4.t36	-	tl		64,380	tl	4.4	**62,074**	183	0.9	-	tl		102,665	tl	40.1	61,540
SB.s8.t24	-	tl		44,427	tl	8.5	**40,632**	70	0.0	-	tl		57,248	tl	29.0	40,632
SB.s8.t48	-	tl		82,795	tl	1.8	**81,283**	259	0.0	-	tl		-	tl		81,283
SB.s8.t72	-	tl		131,150	tl	2.3	**128,405**	329	0.3	-	tl		-	tl		128,081
Feas. sol. found			2			6			6			1			4	
Opt. sol. found			0			0			3			0			0	
Avg gap			20.5			3.3			0.4			45.3			36.6	

intuitive to plan the installation of wind farms in the summer, when weather conditions are expected to be better than in the other seasons. In this section, we want to assess the impact of different starting periods on the overall costs. We consider the two test instances corresponding to the complete wind farms (i.e., instances HR3.s12.t49 and SB.s8.t72), and we compute the upper bounds returned by M^{ub}_{FS} and the lower bounds provided by M^{lb}_{FS} by considering a different month as the earliest possible starting time of the installation operations.

Tables 3.6.5 and 3.6.6 report the computational results achieved on HR3.s12.t49 and SB.s8.t72, respectively, by setting the earliest starting time of the operations at the beginning of January, then February, March, and so on, until the last month where a feasible solution exists. The tables report, for each month, details about the cost of the best upper bound found by M^{ub}_{FS} (i.e., total costs (TC), rental cost (RC), and missed revenue from January 1st (MR)), the gap between the upper bound returned by M^{ub}_{FS} and the lower bound returned by M^{lb}_{FS} (gap), and the best lower bound found by M^{lb}_{FS} (lb_{FS}).

In Tables 3.6.5 and 3.6.6, we can observe that the upper bounds returned by M^{ub}_{FS} are within 3.4% from optimality for HR3.s12.t49 and 2.1% from optimality for SB.s8.t72, so reliable conclusions can be drawn by comparing the cost of the solutions found for the different months. Results show that our

Table 3.6.5: Results on Horn Rev 3 when changing the earliest starting time

Starting Month	TC	RC	MR	gap	lb_{FS}
January	111,398	55,209	56,189	1.9	109,267
February	114,118	49,658	64,460	3.4	110,184
March	119,046	46,718	72,328	2.2	116,435
April	127,076	43,267	83,809	2.8	123,463
May	141,849	43,267	98,582	1.9	139,199
June	158,869	43,267	115,602	2.5	154,935
July	177,767	45,787	131,980	2.2	173,857
Augustus	201,864	50,617	151,247	2.8	196,262

Table 3.6.6: Results on Sanbank when changing the earliest starting time

Starting Month	TC	RC	MR	gap	lb_{FS}
January	128,405	72,912	55,493	0.3	128,081
February	129,054	71,022	58,032	0.0	129,054
March	130,032	65,471	64,561	0.0	130,032
April	135,247	63,070	72,177	0.0	135,247
May	147,388	64,330	83,058	0.0	147,388
June	163,328	66,850	96,478	0.0	163,328

model is a valuable support also in deciding when is the best moment to start operation, depending on the wind-farm specific input: according to our tests, postponing the beginning of the installation operations has a negative impact on the total costs, which is due to the higher and higher missed revenues. However, we can also observe a different trend in the rental cost, which gradually decreases (up to April) and then starts increasing in July for HR3.s12.t49 and in May for SB.s8.t72.

3.7 Conclusions and Future Research

We have dealt with the decision-making problem of planning the installation of offshore wind farms as faced by our industrial partner, Vattenfall, a leading European energy company. We have defined the problem of Installation Planning of Offshore Wind Farms (IPOWF) and formulated it as a Mixed Integer Linear Problem (MILP). Given the complexity of the IPOWF, we have then derived some simpler MILPs that can provide lower and upper bounds to the problem. The quality of these bounds has been documented by testing the MILPs on real-life instances corresponding to two offshore wind farms recently built by Vattenfall.

This study offers Vattenfall and other energy companies facing the same decision-making problem some important managerial insights. First, the paper offers mathematical models that can provide, in less than an hour of computing time, high-quality installation plans that are provably close to an optimal min-cost plan. Second, the computational results show that the policy of building the turbines string-by-string by starting from the string with the highest revenue is efficient, at least for the real-world dataset we received. Third, the computational results also show that there are two main important cost components to take into account when designing installation plans, namely the vessel's rental cost and the missed revenue for delaying the completion of the installation, and both components must be taken into account

in the planning.

Even though we have investigated an IPOWF that is as close as possible to the problem encountered by our industrial partner, we had to make some assumptions. In particular, we have assumed that the IPOWF is a deterministic problem, where input data is not affected by uncertainty. For example, the number of working days per month was assumed to be known upfront, which is clearly not the case in practice. As we consider this assumption the main limitation of our study, we are planning to study stochastic versions of the IPOWF in our future research. Moreover, additional cost components, such as harbour costs, crew costs, and management costs, could have an impact on optimally planning the installation of an offshore wind farm, so it might be worth taking them into account in the planning.

Acknowledgements

We would like to thank Anne Lina Holy, from the Contract Management and Planning department in Vattenfall, and Georgios Katsouris, from System Design in Vattenfall. Their support to the project, in particular in helping us define the problem and in providing us with the data and with feedback on the results is highly appreciated.

Chapter 4

An Exact Approach for Cyclic Inbound Inventory Routing in a Level Production System*

Abstract

We study an inbound inventory routing problem concerned with the minimal-cost collection of distinct components from a network of suppliers and subsequent delivery to a manufacturing plant. We assume known and constant production of end products at the plant that generates a synchronized production of components at each supplier. The lean production philosophy motivates two distinctive features of our formulation. To facilitate standardized work, we consider inventory collection plans that are cyclic and repeatable into the near future. To support the notion of level production planning, we consider inventory collection plans such that the pickup amount at each supplier is a multiple of the daily demand and in exact proportion to the number of days since the last pickup. We study the polyhedron of the convex hull of our mathematical formulation and define new valid inequalities that we implement within our branch-and-cut algorithm for the problem. As our computational experiments confirm, our cyclic formulation is significantly more difficult to solve to optimality than the standard non-cyclic formulation. Regardless, our three-phase approach obtains competitive results for one-, two-, and three-vehicle instances over three- and six-period planning horizons.

Keywords: *routing, branch-and-cut, inventory clearing policy, lean production, cyclic planning horizon*

4.1 Introduction

Popularized by the success of the Toyota Production System (TPS), many companies strive to incorporate the principles of lean manufacturing into the management of their supply chains (Liker 2005). Explicitly incorporating lean principles further complicates the underlying optimization problems involving the routing and inventory decisions of the supply chain. To examine the implications of the lean philosophy on supply chain optimization, we consider a lean production system consisting of a single manufacturing plant and a set of geographically-dispersed suppliers. We seek to identify a minimal-cost inbound logistics

*This chapter is based on Bertazzi, Laganà, Ohlmann & Paradiso (2019).

plan that collects component inventory from a network of suppliers to supply the manufacturing plant at minimal cost and in a manner that supports the lean principles of standard work and level production planning. In the context of inbound inventory routing, standardization refers to the establishment of consistent, repeatable patterns in the pickup quantities and pickup times at the suppliers, and in the corresponding delivery quantities and delivery times at the plant. This standardization is facilitated by a level production plan resulting from the smoothing of demand (in both volume and variety) over a specified period.

The problem we consider is a formulation variant within the broad class of inventory routing problems (IRPs). The standard IRP formulation is motivated by the outbound logistics problem of determining routes to distribute one or more products from a manufacturing plant to a set of customers in a manner that minimizes the transportation costs and inventory holding costs at the plant over a planning horizon. While our formulation's identifying characteristics are motivated by lean production philosophies applied to component collection rather than the more common perspective of end product distribution, we show how to translate our model into an equivalent outbound logistics in 4.A.

To explain the distinctive features of our problem, we formally introduce some notation. Let $\mathcal{I} = \{1, 2, \ldots, I\}$ be the set of suppliers and denote the plant with the node 0. We denote the set of edges as $\mathcal{E} = \{(i, j) : i, j \in \mathcal{I} \cup \{0\}, i < j\}$. Then, the complete undirected graph $G = (\mathcal{I} \cup \{0\}, \mathcal{E})$ represents the transportation network in which a transportation cost, c_{ij}, is associated with each edge $(i, j) \in \mathcal{E}$. Each supplier i provides a distinct component to the plant, so we may reference the terms *supplier* i and *component* i interchangeably. The plant adheres to a level production plan that generates a known and constant daily demand, d_i, for each component $i \in \mathcal{I}$. Furthermore, we assume that each supplier $i \in \mathcal{I}$ produces its component at a matching daily rate d_i. The unit inventory holding cost for supplier i's product at the plant is h_0^i, and the unit inventory holding cost at supplier i is h_i. Our objective is to minimize the sum of transportation cost and system-wide inventory holding cost. Unlike the traditional outbound IRP formulation, we consider inventory holding costs at both the plant and the suppliers to reflect the philosophy that a lean production system should reduce system-wide inventory rather than just shift the cost burden from the plant to the suppliers.

To facilitate the notion of standard work, we design routes over a T-day planning horizon and assume that this routing plan is repeated every T days. We denote this horizon with the cyclic set, $\mathcal{T} = \{1, 2, \ldots, T\}$ such that $next(T) = 1$ and $prev(1) = T$. In practice, such a T-day routing plan is executed repeatedly over a production period (which may be as long as one or two months) during which the plant's manufacturing schedule is effectively fixed. As such, the component inventory level at a supplier at the beginning of the T-day horizon must be the same as the component inventory level at the end of the T-day horizon. In contrast, standard IRP formulations typically consider a non-cyclical planning horizon in which the projected demand in future periods is used to determine optimal or near-optimal inventory and routing decisions for the immediate period; the immediate period's actions are then executed and the problem is re-solved using updated demand data in a rolling horizon manner.

To incorporate level production planning (*heijunka*), we enforce an *inventory clearing policy* requiring that, if supplier i is visited on day t, the quantity collected must correspond to the entire on-hand inventory amount so that the inventory level of supplier i on day t is equal to zero after the pickup. For a cyclic planning horizon, this results in pickup amounts at each supplier i that are a multiple of the daily demand d_i and in exact proportion to the number of days since the last pickup.

This paper makes the following contributions. First, we introduce a new problem motivated by industry practice and present its corresponding mathematical formulation. Second, we study the polyhedron of the convex hull of the problem and define valid inequalities that we leverage in our exact method to solve the problem. Owing to the particular structure of the problem, we provide an extended mathematical formulation of a relaxation of the problem. We solve the relaxation to generate a lower bound for the original problem whenever the solution is optimal, or use the best feasible solution of this relaxation to construct a heuristic solution for the original problem. As a final step, whenever the solution of the extended lower bound formulation is not optimal, we use a branch-and-cut algorithm to solve the problem. This solution approach represents the first exact method for an IRP considering a cyclic planning period and an inventory clearing policy. Third, although our computational experiments establish that our cyclic IRP formulation is significantly more difficult than the standard non-cyclic IRP formulation, our three-phase exact approach obtains competitive results. Specifically, our exact method consistently obtains optimal solutions to one-vehicle instances with up to 50 suppliers over a three-period planning horizon, and up to 30 suppliers over a six-period planning horizon. Although scalability is a challenge even for the much simpler non-cyclic IRP, we obtain optimal or near-optimal solutions to the cyclic IRP for two-vehicle instances with up to 20 suppliers over a three-period planning horizon, and with up to 10 suppliers over a six-period planning horizon. Furthermore, our exact method obtains optimal solutions for three-vehicle instances over a three-period planning horizon for instances with up to 10 suppliers. Finally, we demonstrate that our solution approach is also capable of solving more a general cyclic IRP formulation by removing the inventory clearing policy and allowing more than one vehicle to visit a supplier per day (thereby splitting the collection of supplier demand).

We begin our study with a review of related research in §4.2. We present the formulation of our problem in §4.3. In §4.4, we describe our exact and heuristic solution algorithms. In §4.5, we present our computational experiments on a set of cyclic instances derived from benchmark non-cyclic instances. We provide concluding remarks in §4.6.

4.2 Related Literature

The research literature contains a wide range of IRP formulations varying by objective function, inventory replenishment strategy, and treatment of demand. In this section, we discuss the most relevant studies to our work. Coelho et al. (2013*b*) provide a comprehensive survey of the IRP literature.

4.2.1 Exact Methods for the Inventory Routing Problem

To our knowledge, there is no exact algorithm in the literature that addresses our inbound IRP formulation with a cyclic planning horizon and inventory clearing policy. Exact algorithms for the IRP include branch-and-cut, branch-and-price, and branch-price-and-cut. Generally speaking, branch-and-cut algorithms have proved to be the most successful to date, but implementations of branch-price-and-cut have shown promise for large instances. In the remainder of this section, we document literature on these three algorithmic approaches for the IRP.

Coelho & Laporte (2013*b*) and Adulyasak et al. (2013) propose branch-and-cut algorithms to solve the multi-vehicle extensions of the Archetti et al. (2007) one-vehicle IRP formulation with the order-up-to-level and maximum-level inventory policies. Using the CPLEX mixed-integer programming solver,

instances with up to 45 customers, three vehicles, and three periods have been solved. Coelho & Laporte (2013*a*) further extend the branch-and-cut approach to accommodate a multi-product IRP and solve instances with up to five products, five vehicles, three periods, and 30 customers. Avella et al. (2015) present a branch-and-cut algorithm for the one-vehicle, single-product IRP based on new classes of valid inequalities strictly related to the nature of the instances of Archetti et al. (2007); their polyhedral analysis is based on a unit-stock reformulation of the single-item lot-sizing problem to derive more effective inequalities valid for the inventory problem. For the order-up-to level inventory policy, the branch-and-cut of Avella et al. (2015) significantly improves the upper bounds of Archetti et al. (2007) for the instances with 50 customers and 6 periods. For the maximum-level policy, the branch-and-cut of Avella et al. (2015) and the branch-and-cut of Coelho & Laporte (2014) are able to solve all the instances of Archetti et al. (2007) to optimality. Avella et al. (2018) propose a new generic family of valid inequalities combining both inventory and routing elements of the IRP with the maximum-level policy. Some of these new inequalities are effective in a branch-and-cut algorithm, allowing Avella et al. (2018) to consistently obtain optimal solutions to two-vehicle instances, 50 customers, and three periods. While unable to completely close the optimality gap, Avella et al. (2018) do achieve near-optimal solutions for instances with up to five vehicles, 20 customers and six periods.

There is little evidence in the literature supporting the effectiveness of branch-and-price for the IRP. Bard & Nananukul (2010) use branch-and-price to solve the production routing problem (PRP), which reduces to the IRP when the production schedule is fixed (rather than a decision variable). Bard & Nananukul (2010) solve instances with up to 10 customers and two periods. In comparison, Adulyasak et al. (2013) solve PRP instances with 45 customers, three vehicles, and three periods.

Desaulniers et al. (2016) introduce a branch-price-and-cut algorithm for an extended IRP formulation based on route delivery patterns and the maximum-level inventory policy. Desaulniers et al. (2016) tighten this formulation with known and new valid inequalities, and develop a labeling algorithm to solve the subproblems in the column generation routine. For problems with four and five vehicles, the branch-price-and-cut algorithm of Desaulniers et al. (2016) outperforms the branch-and-cut algorithm of Coelho & Laporte (2014). For instances with fewer than four vehicles, however, the branch-and-cut algorithm is superior.

4.2.2 Cyclic Inventory Routing Problems

Past work considering cyclic schedules for the IRP have been based on continuous-time models or discrete-time models. We review both of these research streams in this section.

There is a body of research which considers a continuous-time IRP with an infinite planning horizon and constant demand rate in which the delivery cycle time for each customer must be identified. Aghezzaf et al. (2006) address this long-term planning problem with a column generation based heuristic to construct policies akin to economic order quantities. Raa & Aghezzaf (2009) explicitly add the practical considerations of cargo handling times, customer inventory capacities, and maximum driving limits for vehicles. Chitsaz et al. (2016) develop a two-phase iterative heuristic and benchmark their performance against Raa & Aghezzaf (2009), demonstrating they can improve the best-known solution for 60 of 80 instances.

Zhao et al. (2007) consider a fixed partition policy in which the replenishment interval is based on a power-of-two (as often used in the economic order quantity literature). Ekici et al. (2015) also utilize a

fixed partition policy, and develop a two-stage heuristic that achieves solutions with an average optimality gap of 3.76 percent for real-life instances with 26 to 80 customers and a planning horizon of 14 days. Motivated by applications in liner shipping and facility logistics, Zenker et al. (2016) consider a cyclic IRP in which all customers are located along a line.

Aghezzaf et al. (2012) study a one-vehicle cyclic IRP in which not all customers need to be visited and for each customer selected for replenishments, the supplier collects a corresponding fixed reward. Zhong & Aghezzaf (2011) and Zhong & Aghezzaf (2012) examine various heuristic approaches to solve this one-vehicle cyclic IRP. Similarly, Vansteenwegen & Mateo (2014) cast a one-vehicle cyclic inventory routing problem as a variant of an orienteering problem and design an iterated local search heuristic.

Our work shares more similarities with the discrete-time models of cyclic IRPs than the aforementioned continuous-time models. Similar to our formulation, Aksen et al. (2012) consider a cyclic planning horizon and employ an inventory clearing policy for the collection of a single item (used vegetable oil) from suppliers. Aksen et al. (2012) present two different mixed-integer programming formulations, which include some additional features such as supplier selection, and obtain solutions with an average optimality gap of 3.28 percent for instances with 25 suppliers and a planning horizon of seven days.

Francis & Smilowitz (2006) present a variant of the period vehicle routing problem in which the choice of visit schedule over the planning horizon is a decision variable. The choice of visit schedule controls service benefit, which corresponds to inventory holding cost in the context of an IRP. To facilitate the solution of their model, Francis & Smilowitz (2006) implement a pair of restrictions to reduce the solution space: (i) they consider a restricted set of schedules, and (ii) while their model supports unequal service amounts, they always consider the largest possible service amount corresponding to the selected schedule. Our model does not restrict the choice of schedules for better cost management and explicitly considers unequal service amounts in routing to make full utilization of vehicle capacity. Francis & Smilowitz (2006) present an exact Lagrangian-based branch-and-bound procedure and a related heuristic approach. Similar to Francis & Smilowitz (2006), Rusdiansyah & Tsao (2005) present a two-phase heuristic (initialization and tabu-search based improvement) for the period vehicle routing problem with time windows as the basis for a new model in which customer visit frequencies are a decision variable.

4.2.3 Rich Vehicle Routing Problems

Rich vehicle routing problems consider complications that arise in real-world routing that typically are ignored in stylized vehicle routing models (Schmid et al. 2013). Of particular relevance to this paper is the related body of rich vehicle routing problems focused specifically on lean production systems; these studies are devoted to model development and heuristic solution methods.

Chuah & Yingling (2005) incorporate aspects of a just-in-time system into a single-day vehicle routing problem with time windows in which the number of times that a supplier is visited is a decision variable, but require that the pickup amounts at a supplier are equally-sized. Chuah & Yingling (2005) restrict the problem by requiring all suppliers on the same route to have the same visit frequency; this restriction is called common frequency routing. Chuah & Yingling (2005) solve small instances of their integer programming model with a set partitioning column generation approach and develop a tabu search heuristic for larger instances. Ohlmann et al. (2008) extend the work of Chuah & Yingling (2005) by relaxing the restriction of common frequency routing and allowing each supplier to have its own distinct visit frequency (although still requiring that the pickup amounts at a supplier are equally-sized). Ohlmann et al.

(2008) design a nested tabu search algorithm that iterates between setting the suppliers' visit frequencies and routing the resulting visits; they apply their algorithm to data sets with up to 109 customers. Stacey et al. (2007) present an inbound inventory routing problem similar to Chuah & Yingling (2005) with the exception that time window constraints are not considered. Stacey et al. (2007) present two heuristics based on common frequency routing, one which attempts to explicitly account for crossdock capacity in the route selection process, and perform computational testing on 14-customer data sets. Natarajarathinam et al. (2012) consider the same problem and present two new heuristics that provide lower cost solutions on the 14-customer data sets. Dong & Turnquist (2015) use a "cluster first, route second" approach for a single-period inbound inventory routing problem. Integrating the choice of visit frequency with supplier location, Dong & Turnquist (2015) identify sets of customers to be routed together by solving a single-source capacitated facility location problem (SSCFLP). A traveling salesman algorithm then executes the common frequency routing for each set of customers. Dong & Turnquist (2015) solve small instances of the SSCFLP with the commercial solver CPLEX and large instances with the very large scale neighborhood search heuristic from Ahuja et al. (2004). Vaidyanathan et al. (1999) introduce a nonlinear capacity constraint to a vehicle routing problem with the goal of constructing routes that deliver loads to customers that exactly meet the accumulated demand between visits. Vaidyanathan et al. (1999) generate a lower bound from a relaxation of their integer programming formulation and outline a heuristic procedure which performs well when vehicle capacities are relatively large. Satoglu & Sahin (2013) formulate the supply of various stations along a television assembly line as a vehicle routing problem with simultaneous pickup and delivery to capture the dynamics of picking up empty kanban kits and delivering full kanban kits. Satoglu & Sahin (2013) devise a heuristic based on common frequency routing to solve their nonlinear integer programming model that results from modeling a station's delivery quantity as a function of the delivery route to account for the amount of demand accumulating between visits. More recently, the problem of integrating inbound supply routing with production and inventory decisions has been gaining attention. For both a supply chain in which component inventory is allowed to be stored at the production plant and a supply chain in which component inventory must be supplied in a JIT manner, Hein & Almeder (2016) conduct computational experiments on small instances (4 to 6 suppliers, 8 to 12 distinct components, 3 to 4 end products, and 5 to 10 periods) to estimate the savings achievable from solving the integrated routing and production problem versus decomposing the problem by first establishing a production plan and then developing the supply routes. Chitsaz et al. (2019) introduce a matheuristic for an assembly routing problem which integrates the optimization of production, inventory, and inbound transportation decisions. While Hein & Almeder (2016) and Chitsaz et al. (2019) allow production planning to be decision variables and we assume that production at the plant to be known and constant, our model offers the unique features of a cyclic planning horizon incorporating an inventory clearing policy.

4.3 Formulation of the Cyclic Inbound Inventory Routing Problem

We adapt the integer programming formulation without vehicle indexing from Adulyasak et al. (2013) to formally describe our cyclic inventory routing problem. With a goal of minimizing the sum of routing cost and inventory cost over a T-day horizon, our formulation seeks to determine the quantity to send from each supplier $i \in \mathcal{I}$ to the plant at each time $t \in \mathcal{T}$, to sequence routes that visit specified suppliers at each time $t \in \mathcal{T}$, and to set the initial inventory levels at the suppliers and at the plant. Because the problem

is cyclic, the inventory levels at the end of the period T must be equal to the initial inventory levels of all the suppliers and the plant. As the initial inventory levels at all locations are decision variables, the model assumes that there is negligible cost of establishing the initial inventory levels to facilitate the cyclic inventory collection plan. If this is not the case, a cost term that reflects the cost of establishing the requisite initial inventory at each location can be added.

A homogeneous fleet of vehicles, $\mathcal{K} = \{1, \ldots, K\}$, with integer capacity Q, collects the components from the set of suppliers. We assume that at most one vehicle can visit each supplier per day, i.e., split collections are not allowed. Without loss of generality, we assume $d_i \leq Q$; if $d_i > Q$, one can dedicate $\lfloor \frac{d_i}{Q} \rfloor$ vehicles to supplier i and then consider $d'_i = d_i - \lfloor \frac{d_i}{Q} \rfloor Q$ in the routing problem.

We introduce the following decision variables:

- q_{ti}: quantity to pick up from supplier $i \in \mathcal{I}$ on day $t \in \mathcal{T}$
- I_{ti}: inventory level at supplier $i \in \mathcal{I}$ at end of day $t \in \mathcal{T}$
- I^i_{t0}: inventory level of component i at the plant at end of day $t \in \mathcal{T}$
- z_{t0}: integer variable indicating the number of vehicles that leave the plant on day $t \in \mathcal{T}$
- z_{ti}: binary variable equal to 1 if node $i \in \mathcal{I}$ is visited on day $t \in \mathcal{T}$ and 0 otherwise
- x_{tij}: binary variable equal to 1 if edge $(i, j) \in \mathcal{E}$ is part of a route on day $t \in \mathcal{T}$ and 0 otherwise.

To facilitate the formulation, we define $\delta(S)$ as the set of edges $(i, i') \in \mathcal{E}$ incident to the nodes $i \in S \subseteq \mathcal{I} \cup \{0\}$; for notational convenience, if $S = \{i\}$, we denote the corresponding edge cutset as $\delta(i)$. We let $\mathcal{E}(\mathcal{U})$ be the set of edges $(i, j) \in \mathcal{E}$ such that $i, j \in \mathcal{U}$, where $\mathcal{U} \subseteq \mathcal{I}$ is a given subset of suppliers. We describe our optimization problem, (**P**), via (4.1)–(4.14):

$$\min \quad \sum_{t\in\mathcal{T}} \sum_{(i,j)\in\mathcal{E}} c_{ij} x_{tij} + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} h_i I_{ti} + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} h^i_0 I^i_{t0} \tag{4.1}$$

$$\text{s.t.} \quad I_{ti} = I_{prev(t),i} + d_i - q_{ti} \qquad \forall t \in \mathcal{T}, \forall i \in \mathcal{I} \tag{4.2}$$

$$I^i_{t0} = I^i_{prev(t),0} + q_{ti} - d_i \qquad \forall t \in \mathcal{T}, \forall i \in \mathcal{I} \tag{4.3}$$

$$I_{ti} \leq (T-1) d_i \, (1 - z_{ti}) \qquad \forall t \in \mathcal{T}, \forall i \in \mathcal{I} \tag{4.4}$$

$$z_{t0} \leq K \qquad \forall t \in \mathcal{T} \tag{4.5}$$

$$q_{ti} \leq \min\{Q, T d_i\} \, z_{ti} \qquad \forall t \in \mathcal{T}, \forall i \in \mathcal{I} \tag{4.6}$$

$$\sum_{(j,j')\in\delta(i)} x_{tjj'} = 2 z_{ti} \qquad \forall t \in \mathcal{T}, \forall i \in \mathcal{I} \cup \{0\} \tag{4.7}$$

$$Q \sum_{(i,j)\in\mathcal{E}(\mathcal{U})} x_{tij} \leq \sum_{i\in\mathcal{U}} (Q z_{ti} - q_{ti}) \qquad \forall t \in \mathcal{T}, \forall \mathcal{U} \subseteq \mathcal{I}, |\mathcal{U}| \geq 2, \forall u \in \mathcal{U} \tag{4.8}$$

$$q_{ti} \geq 0 \qquad \forall t \in \mathcal{T}, \forall i \in \mathcal{I} \tag{4.9}$$

$$I_{ti}, I^i_{t0} \geq 0 \qquad \forall t \in \mathcal{T}, \forall i \in \mathcal{I} \tag{4.10}$$

$$z_{t0} \in \mathcal{Z}_+ \qquad \forall t \in \mathcal{T} \tag{4.11}$$

$$z_{ti} \in \{0, 1\} \qquad \forall t \in \mathcal{T}, \forall i \in \mathcal{I} \tag{4.12}$$

$$x_{tij} \in \{0, 1\} \qquad \forall t \in \mathcal{T}, \forall (i, j) \in \mathcal{E} : i \notin \{0\} \tag{4.13}$$

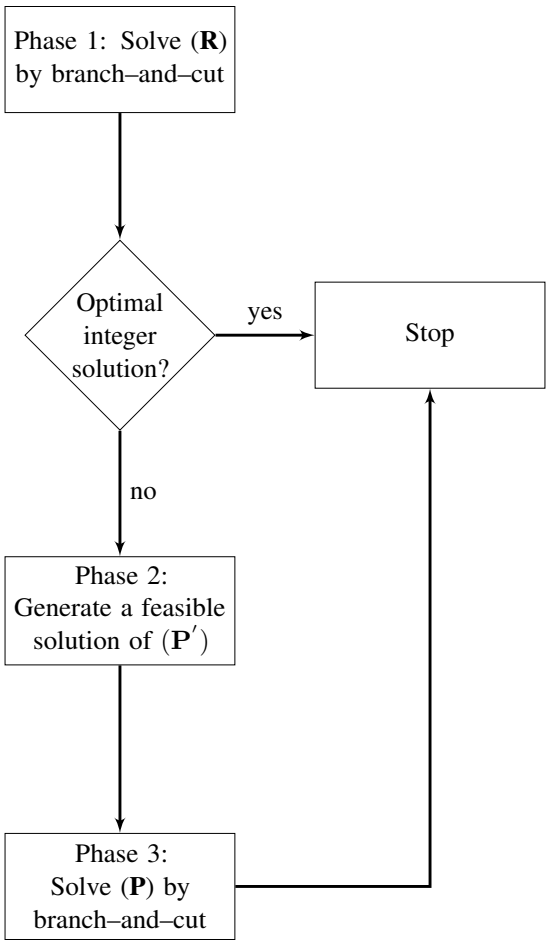
$$x_{t0j}, x_{tj0} \in \{0, 1\} \qquad \forall t \in \mathcal{T}, \forall j \in I. \tag{4.14}$$

The objective function (4.1) minimizes the sum of routing and inventory costs at the suppliers and at the plant. Constraints (4.2) and (4.3) balance the inventory flow at the suppliers and at the plant, respectively, over the T-day horizon. We assume, similar to Adulyasak et al. (2013), that the day t production at a supplier can be delivered to the plant to satisfy its demand on the same day. Constraints (4.2) and (4.3) imply that the inventory level at the end of the horizon is equal the inventory level at the beginning of the horizon, the total service amount transported over the cyclic horizon is equal to Td_i and the total inventory level of supplier i's product is constant over time and equal to $I^i_{t0} + I_{ti}$. Constraints (4.4) enforce the *inventory clearing policy* at the suppliers. Constraints (4.5) enforce the fleet size. Constraints (4.6) limit the amount of the pickup from supplier i on day t. In the underlying graph G, constraints (4.7) control the degree of the nodes, while constraints (4.8) prohibit subtours and ensure that the vehicle capacity is not violated. These constraints are identical to those used by Adulyasak et al. (2013), and they will be referred to in the following as capacitated subtour elimination (CSE) constraints. Finally, constraints (4.9) imply non-negative pick-up quantities, while (4.10) disallow the possibility of a stockout at either the suppliers or plant. Constraints (4.11)–(4.14) define the integer and binary decision variables. We denote the optimal cost of this model as z^*.

4.4 Exact Solution Approach

As Figure 4.4.1 illustrates, we develop a three-phase exact solution approach to solve (**P**), described by (4.1)–(4.14). Our approach leverages a vehicle-indexed reformulation of (**P**) which we refer to as $(\mathbf{P}')$, and an extended relaxation of $(\mathbf{P}')$ which we refer to as (**R**). We formally describe these formulations in the following sections. In the first phase, we employ a branch-and-cut algorithm with a two-hour time limit to solve (**R**) in order to obtain an extended lower bound on the optimal cost z^* of (**P**). We add valid inequalities to the linear programming relaxation at the root of the branch-and-bound tree; we provide details about these inequalities in the following sections. In this phase, if the branch-and-cut algorithm is able to solve the extended lower bound formulation to optimality, we can provide a lower bound on the optimal cost and we can identify a cut pool of violated valid inequalities to use in the third phase. If we obtain an optimal integer solution within the time limit, this solution is optimal for (**P**) (and our solution approach terminates). If we do not obtain an optimal integer solution of the extended lower bound formulation within the time limit, we execute the second phase by applying a heuristic, based on the solution obtained in the first phase, to find an upper bound on the optimal cost z^*. Then, in the third phase, we apply a branch-and-cut algorithm to (**P**), using the heuristic solution from the second phase as the initial upper bound. We first add the cut pool of violated valid inequalities identified in the first phase and any new valid inequalities to the initial linear program at the root of the branch-and-bound tree. Further, we dynamically add some of these inequalities. The following sections describe each phase in detail.

Figure 4.4.1: Three-Phase Solution Procedure



4.4.1 First Phase: Extended Lower Bound

We obtain the extended lower bound on the optimal cost z^* of (**P**) on the basis of a vehicle-indexed integer programming formulation that is an adaptation from Adulyasak et al. (2013). This vehicle-indexed formulation, $(\mathbf{P}')$, which is equivalent to (**P**), relies on the following decision variables that use an explicit indexing of the vehicles in the fleet:

- q_{tki}: quantity to pick up from supplier $i \in \mathcal{I}$ by vehicle $k \in \mathcal{K}$ on day $t \in \mathcal{T}$
- I_{ti}: inventory level at supplier $i \in \mathcal{I}$ at end of day $t \in \mathcal{T}$
- I^i_{t0}: inventory level of component i at the plant at end of day $t \in \mathcal{T}$
- z_{tki}: binary variable equal to 1 if node $i \in \mathcal{I} \cup \{0\}$ is visited by vehicle $k \in \mathcal{K}$ on day $t \in \mathcal{T}$ and 0 otherwise
- x_{tkij}: binary variable equal to 1 if vehicle $k \in \mathcal{K}$ travels directly from node $i \in \mathcal{I} \cup \{0\}$ to node $j \in \mathcal{I} \cup \{0\}$ on day $t \in \mathcal{T}$ and 0 otherwise.

We formally describe the vehicle-indexed formulation, $(\mathbf{P}')$, via (4.15)–(4.27):

$$\min \quad \sum_{t\in\mathcal{T}}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{E}} c_{ij}x_{tkij} + \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}} h_i I_{ti} + \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}} h_0^i I_{t0}^i \tag{4.15}$$

$$\text{s.t.} \quad I_{ti} = I_{prev(t),i} + d_i - \sum_{k\in\mathcal{K}} q_{tki} \qquad \forall t\in\mathcal{T}, \forall i\in\mathcal{I} \tag{4.16}$$

$$I_{t0}^i = I_{prev(t),0}^i + \sum_{k\in\mathcal{K}} q_{tki} - d_i \qquad \forall t\in\mathcal{T}, \forall i\in\mathcal{I} \tag{4.17}$$

$$I_{ti} \leq (T-1)d_i \left(1 - \sum_{k\in\mathcal{K}} z_{tki}\right) \qquad \forall t\in\mathcal{T}, \forall i\in\mathcal{I} \tag{4.18}$$

$$\sum_{i\in\mathcal{I}} q_{tki} \leq Q z_{tk0} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K} \tag{4.19}$$

$$q_{tki} \leq \min\{Q, Td_i\} z_{tki} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall i\in\mathcal{I} \tag{4.20}$$

$$\sum_{(j,j')\in\delta(i)} x_{tkjj'} = 2z_{tki} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall i\in\mathcal{I}\cup\{0\} \tag{4.21}$$

$$\sum_{(i,j)\in\mathcal{E}(\mathcal{U})} x_{tkij} \leq \sum_{i\in\mathcal{U}} z_{tki} - z_{tku} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K},$$

$$\forall \mathcal{U}\subseteq\mathcal{I}, |\mathcal{U}|\geq 2, \forall u\in\mathcal{U} \tag{4.22}$$

$$q_{tki} \geq 0 \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall i\in\mathcal{I} \tag{4.23}$$

$$I_{ti}, I_{t0}^i \geq 0 \qquad \forall t\in\mathcal{T}, \forall i\in\mathcal{I} \tag{4.24}$$

$$z_{tki} \in \{0,1\} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall i\in\mathcal{I}\cup\{0\} \tag{4.25}$$

$$x_{tkij} \in \{0,1\} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall (i,j)\in\mathcal{E} : i\notin\{0\} \tag{4.26}$$

$$x_{tk0j}, x_{tkj0} \in \{0,1\} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall j\in I. \tag{4.27}$$

The objective function (4.15) and constraints (4.16), (4.17), (4.18), (4.20) and (4.21) are analogous to (4.1), (4.2), (4.3), (4.4), (4.6) and (4.7), respectively. Constraints (4.19) enforce the total amount serviced by a vehicle to be within the effective vehicle capacity. Constraints (4.22) prohibit subtours. Constraints (4.23)–(4.27) mirror the variable definitions in (4.9)–(4.14). We observe that constraints (4.18) and (4.24) prohibit split collections, i.e., multiple visits to a supplier in a single day. We introduce (**R**), the extended relaxation of ($\mathbf{P}'$), as the formulation that relaxes both the subtour elimination constraints (4.22) and the binary nature of the z_{tki} and x_{tkij} variables. An observation regarding the q_{tki} variables motivates a change of variables in our relaxation of ($\mathbf{P}'$). Because Q is integer, the variables q_{tki} are integer in the extreme points of the convex hull of ($\mathbf{P}'$). Therefore, due to constraints (4.18), if the quantity q_{tki} is greater than zero, it is a multiple of d_i, i.e., $q_{tki} = \alpha d_i$, where $\alpha = 1, \ldots, T$. We introduce the binary variable $w_{tk\alpha i}$ equal to 1 if the quantity αd_i is picked-up at the supplier i by vehicle k on day t. We replace the variable q_{tki} with $\sum_{\alpha=1}^{T} \alpha d_i w_{tk\alpha i}$ everywhere in the formulation. Then, we can formulate (**R**) with (4.28)–(4.41):

$$\min \quad \sum_{t\in\mathcal{T}}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{E}} c_{ij}x_{tkij} + \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}} h_i I_{ti} + \sum_{t\in\mathcal{T}}\sum_{i\in\mathcal{I}} h_0^i I_{t0}^i \tag{4.28}$$

$$\text{s.t.} \quad I_{ti} = I_{prev(t),i} + d_i - \sum_{k\in\mathcal{K}}\sum_{\alpha=1}^{T} \alpha d_i w_{tk\alpha i} \qquad \forall t\in\mathcal{T}, \forall i\in\mathcal{I} \tag{4.29}$$

$$I_{t0}^i = I_{prev(t),0}^i + \sum_{k\in\mathcal{K}}\sum_{\alpha=1}^{T} \alpha d_i w_{tk\alpha i} - d_i \qquad \forall t\in\mathcal{T}, \forall i\in\mathcal{I} \tag{4.30}$$

$$I_{ti} \le (T-1)d_i \left(1 - \sum_{k\in\mathcal{K}} z_{tki}\right) \qquad \forall t\in\mathcal{T}, \forall i\in\mathcal{I} \tag{4.31}$$

$$\sum_{i\in\mathcal{I}}\sum_{\alpha=1}^{T} \alpha d_i w_{tk\alpha i} \le Q z_{tk0} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K} \tag{4.32}$$

$$z_{tki} = \sum_{\alpha=1}^{T} w_{tk\alpha i} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall i\in\mathcal{I} \tag{4.33}$$

$$\sum_{\alpha=1}^{T} \alpha d_i w_{tk\alpha i} \le \min\{Q, Td_i\} z_{tki} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall i\in\mathcal{I} \tag{4.34}$$

$$\sum_{(j,j')\in\delta(i)} x_{tkjj'} = 2z_{tki} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall i\in\mathcal{I}\cup\{0\} \tag{4.35}$$

$$I_{it}, I_{0t}^i \ge 0 \qquad \forall t\in\mathcal{T}, \forall i\in\mathcal{I} \tag{4.36}$$

$$w_{tk\alpha i} \in \{0,1\} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall \alpha = 1,\dots,T, \forall i\in\mathcal{I} \tag{4.37}$$

$$z_{tk0} \in \{0,1\} \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K} \tag{4.38}$$

$$0 \le z_{tki} \le 1 \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall i\in\mathcal{I} \tag{4.39}$$

$$0 \le x_{tkij} \le 1 \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall (i,j)\in\mathcal{E} : i \notin \{0\} \tag{4.40}$$

$$0 \le x_{tk0j}, x_{tkj0} \le 1 \qquad \forall t\in\mathcal{T}, \forall k\in\mathcal{K}, \forall j\in\mathcal{I}. \tag{4.41}$$

We solve (**R**) by applying a branch-and-cut algorithm with a time limit of two hours. We add the classical symmetry breaking inequalities described in 4.C to the initial linear program at the root of the branch-and-bound tree for each $i \in \mathcal{I}$. Then, we dynamically add the subtour elimination constraints (4.22) to the current linear program at each node of the branch-and-bound tree for the vehicle k, day t combination for which the violation is checked, as in Adulyasak et al. (2013). If we obtain an optimal solution of (**R**) within the time limit which has integer values for all x_{tikj} and z_{tki} variables, this solution is also optimal for (**P**) and we terminate, otherwise we continue on to the second phase.

4.4.2 Second Phase: Heuristic Solution

If the first phase terminates with an optimal solution of (**R**) with fractional values of x_{tikj} and z_{tki} variables or before a provably optimal solution of (**R**) is achieved, we generate a heuristic solution to $(\mathbf{P}')$ from the solution of (**R**) in hand. In any feasible solution of (**R**), the quantity picked-up from each supplier by each vehicle on each day is also feasible for the original formulation $(\mathbf{P}')$. Therefore, the

values of the variables I_{ti} and I^i_{t0} in any feasible solution of (**R**) are also feasible for $(\mathbf{P}')$. Moreover, the value of variables z_{tki} are either equal to 0 or 1 in any feasible solution of (**R**) due to constraints (4.33); thus, these variables are feasible for $(\mathbf{P}')$. So, in a feasible solution to (**R**), the x_{tkij} variables are the only ones that may not be feasible in $(\mathbf{P}')$ as they may be fractional. As Algorithm 1 displays, we generate the heuristic solution by solving a traveling salesman problem (TSP) defined by the set of visited suppliers for each vehicle-day combination. We solve these TSP instances with the Concorde TSP Solver.

Algorithm 1: Heuristic

```
for each day t ∈ 𝒯 do
    for each vehicle k ∈ 𝒦 do
        Compute the subset of visited suppliers Z_tk = {i ∈ ℐ : z_tki = 1} ⊆ ℐ.
        Find an optimal TSP tour on the complete undirected subgraph induced by Z_tk ∪ {0}.
    end for
end for
```

We remark that basing an extended lower bound formulation directly on (**P**) (instead of $(\mathbf{P}')$) is not efficient because solutions obtained from such an extended lower bound formulation would be expressed in terms of decision variables without an explicit vehicle index. Therefore, a solution of an extended lower bound formulation that is fractional in terms of variables x_{tij} would require solving a capacitated vehicle routing problem instead of a TSP in each period of the planning horizon. Hence, a heuristic based on an extended lower bound formulation with no vehicle index generally does not produce high quality feasible solutions.

4.4.3 Third Phase: Branch-and-Cut on Original Formulation

In the third phase, we apply a branch-and-cut algorithm (with a time limit of four hours) to (**P**). We set the initial upper bound equal to the objective value of the heuristic solution from the second phase. To the initial linear program at the root of the branch-and-bound tree, we add the cut pool of violated constraints (4.8). More specifically, for each subset $\overline{\mathcal{U}}$ such that a violation of constraints (4.22) is detected, we define and add constraints (4.8) to the initial linear program, along with the classical valid inequalities described in 4.C and the single-item lot-sizing valid inequalities in §4.4.3. Then, we dynamically add the parity inequalities described in 4.C. We also dynamically add the CSE constraints to the current linear program at each node of the branch-and-bound tree. We separate these constraints using the procedure of Picard & Ratliff (1973) that polynomially solves a min-cut problem.

subsectionValid Inequalities

As previously described, our solution approach leverages valid inequalities in its branch-and-cut algorithm. In this section, we describe the single-item lot-sizing valid inequalities that we employ. In 4.C, we describe the additional valid inequalities from the literature that we incorporate.

We introduce new valid inequalities based on a polyhedral analysis of the subproblem concerning a single supplier (item) i. Let P_{CIRP} be the convex hull of all the vectors $\left(q, I, I_0, z_0, z, x_{E\setminus E_0}, x_{E_0}\right) \in \mathcal{R}_+^{|\mathcal{T}||\mathcal{I}|} \times \mathcal{R}_+^{|\mathcal{T}||\mathcal{I}|} \times \mathcal{R}_+^{|\mathcal{T}||\mathcal{I}|} \times \mathcal{Z}_+^{|\mathcal{T}|} \times \{0,1\}^{|\mathcal{T}||\mathcal{I}|} \times \{0,1\}^{|\mathcal{T}||E\setminus E_0|} \times \{0,1\}^{2(|\mathcal{T}||E_0|)}$ satisfying the constraints of (**P**), where E_0 is the set of all the edges that are incident to the plant, node 0. Because Q is integer, the values of q, I, I_0 are integer in the extreme points of P_{CIRP}. Any feasible solution of (4.1)–(4.14) belongs to the space $\{0, \ldots, Td_i\}^{|\mathcal{T}||\mathcal{I}|} \times \{0, \ldots, (T-1)\, d_i\}^{2(|\mathcal{T}||\mathcal{I}|)} \times \mathcal{Z}_+^{|\mathcal{T}|} \times \{0,1\}^{|\mathcal{T}||\mathcal{I}|} \times \{0,1\}^{|\mathcal{T}||E\setminus E_0|} \times \{0,1\}^{2(|\mathcal{T}||E_0|)}$ because the variables q_{ti} to pick-up from supplier i on day t is an integer multiple of the plant demand d_i, which in turn implies that the inventory levels I_{ti} and I^i_{t0} are also all integer multiples of the plant demand d_i.

In this section, we study the polyhedron of the single-item lot-sizing problem (with an inventory clearing policy and no split collection) defined by (4.42)–(4.49) to derive inequalities that are also valid for the P_{CIRP}. As we deal with a single supplier in this section, we drop the index i for notational brevity and, without loss of generality, assume $d_i = 1$. Because each supplier provides a unique component to the plant, the polyhedron P_{CIRP} is equal to $\cup_{i\in\mathcal{I}} P^i_{SILSP} \cap P_{PCVRP}$, where P^i_{SILSP} is the convex hull of all the vectors satisfying (4.43)-(4.49), and where P_{PCVRP} is the polyhedron of the periodic capacitated vehicle routing problem defined by all the vectors satisfying constraints (4.5), (4.7), (4.8) and (4.11)–(4.14).

$$
\begin{aligned}
(\textbf{SILS})\ \min \quad & \sum_{t\in\mathcal{T}} hI_t + \sum_{t\in\mathcal{T}} h_0 I_{t0} && && (4.42)\\
\text{s.t.} \quad & I_t = I_{prev(t)} + 1 - q_t && \forall t \in \mathcal{T} && (4.43)\\
& I_{t0} = I_{prev(t),0} + q_t - 1 && \forall t \in \mathcal{T} && (4.44)\\
& I_t \le (T-1)\,(1 - z_t) && \forall t \in \mathcal{T} && (4.45)\\
& q_t \le \min\{Q, T\}\, z_t && \forall t \in \mathcal{T} && (4.46)\\
& q_t \ge 0 && \forall t \in \mathcal{T} && (4.47)\\
& I_t, I_{t0} \ge 0 && \forall t \in \mathcal{T} && (4.48)\\
& z_t \in \{0,1\} && \forall t \in \mathcal{T} && (4.49)
\end{aligned}
$$

Constraints (4.43) and (4.44) imply that the total inventory level of item i is constant over time, $I_t + I_{t0} = I_{prev(t)} + I_{prev(t)0}$. Moreover, the maximum total inventory level occurs when the total quantity T needed by the plant is picked-up at the supplier on a single day over the time horizon. In this case, the total inventory level is equal to $T-1$ and because it is constant over time, we know $I_t + I_{t0} \le T-1$ for all $t \in \mathcal{T}$. We leverage the fact that $I_t + I_{t0} \le T-1$ for all $t \in \mathcal{T}$ in an extended reformulation of $(\textbf{SILS})$ which we define by (4.50)–(4.61) and refer to as the *extended unit stock formulation*.

Constraints (4.51) and (4.52) link the inventory variables I_t with the stock variables $u_{\alpha t}$. The binary variable $u_{\alpha t}$ is equal to 1 if $I_t = \alpha$ (for $\alpha \in \{0, 1, \ldots, T-1\}$), and 0 otherwise. Constraints (4.53) and (4.54) link the inventory variables I_{t0} with the stock variables $y_{\beta t}$. The binary variable $y_{\beta t}$ is equal to 1 if $I_{t0} = \beta$ (for $\beta \in \{0, 1, \ldots, T-1\}$), and 0 otherwise. Constraints (4.55) imply that if the supplier is visited on day t, then its inventory level is equal to 0 (otherwise it is at least equal to one). Constraints (4.56) relate the value of the inventory level at the supplier to the inventory level at the production plant: when $I_{t0} = \beta$, I_t must be a value in set $\{0, 1, \ldots, T-1-\beta\}$. Constraints (4.57) and (4.58) state that the decision variables $u_{\alpha t}$ and $y_{\beta t}$ are non-negative. Constraints (4.59)–(4.61) define $u_{\alpha t}$, $y_{\beta t}$ and z_t as binary variables. Although constraints (4.57) and (4.58) are redundant, we include them to facilitate the definition of properties of the continuous polyhedron associated with $(\textbf{EUS})$.

$$
\begin{aligned}
(\mathbf{EUS})\ \min \quad & \sum_{t\in\mathcal{T}} hI_t + \sum_{t\in\mathcal{T}} h_0 I_{t0} && && (4.50)\\
\text{s.t.} \quad & I_t = \sum_{\alpha=0}^{T-1} \alpha u_{\alpha t} && \forall t \in \mathcal{T} && (4.51)\\
& \sum_{\alpha=0}^{T-1} u_{\alpha t} = 1 && \forall t \in \mathcal{T} && (4.52)\\
& I_{t0} = \sum_{\beta=0}^{T-1} \beta y_{\beta t} && \forall t \in \mathcal{T} && (4.53)\\
& \sum_{\beta=0}^{T-1} y_{\beta t} = 1 && \forall t \in \mathcal{T} && (4.54)\\
& z_t = u_{0t} && \forall t \in \mathcal{T} && (4.55)\\
& y_{\beta t} \leq \sum_{\alpha=0}^{T-1-\beta} u_{\alpha t} && \forall \beta \in \{0,\ldots,T-1\}, \forall t \in \mathcal{T} && (4.56)\\
& u_{\alpha t} \geq 0 && \forall \alpha \in \{0,\ldots,T-1\}, \forall t \in \mathcal{T} && (4.57)\\
& y_{\beta t} \geq 0 && \forall \beta \in \{0,\ldots,T-1\}, \forall t \in \mathcal{T} && (4.58)\\
& u_{\alpha t} \in \{0,1\} && \forall \alpha \in \{0,\ldots,T-1\}, \forall t \in \mathcal{T} && (4.59)\\
& y_{\beta t} \in \{0,1\} && \forall \beta \in \{0,\ldots,T-1\}, \forall t \in \mathcal{T} && (4.60)\\
& z_t \in \{0,1\} && \forall t \in \mathcal{T}. && (4.61)
\end{aligned}
$$

To show that the polyhedron corresponding to the *continuous* extended unit stock formulation defined by (4.51)–(4.58) is an integral polyhedron, we first introduce a new extended formulation of $(\mathbf{SILS})$, which we refer to as the *extended unit flow formulation*, $(\mathbf{EUF})$. Defining the binary variables $f_t^{\alpha,\beta}$ to be equal to 1 if $I_t = \alpha$ and $I_{t0} = \beta$ (for $\alpha, \beta \in \{0, 1, \ldots, T-1\}$), and 0 otherwise, the extended unit flow

formulation $(\mathbf{EUF})$ is:

$$\min \quad \sum_{t\in\mathcal{T}}\sum_{\alpha=0}^{T-1} h\alpha u_{\alpha t} + \sum_{t\in\mathcal{T}}\sum_{\beta=0}^{T-1} h_0\beta y_{\beta t} \tag{4.62}$$

$$\text{s.t.} \quad u_{\alpha t} = \sum_{\beta=0}^{T-1} f_t^{\alpha,\beta} \qquad \forall \alpha \in \{0,\dots,T-1\}, \forall t \in \mathcal{T} \tag{4.63}$$

$$y_{\beta t} = \sum_{\alpha=0}^{T-1} f_t^{\alpha,\beta} \qquad \forall \beta \in \{0,\dots,T-1\}, \forall t \in \mathcal{T} \tag{4.64}$$

$$f_t^{\alpha,\beta} = f_{t+1}^{\alpha+1,\beta-1} + f_{t+1}^{0,\alpha+\beta} \qquad \forall \alpha,\beta \in \{0,\dots,T-1\}, \alpha+1 \le T-1, \beta-1 \ge 0, \alpha+\beta \le T-1, \forall t \in \mathcal{T} \tag{4.65}$$

$$f_t^{\alpha,\beta} = 0 \qquad \forall \alpha,\beta \in \{0,\dots,T-1\}, \alpha+\beta > T-1, \forall t \in \mathcal{T} \tag{4.66}$$

$$\sum_{\alpha=0}^{T-1}\sum_{\beta=0}^{T-1} f_t^{\alpha,\beta} = 1 \qquad \forall t \in \mathcal{T} \tag{4.67}$$

$$u_{\alpha t} \ge 0 \qquad \forall \alpha \in \{0,\dots,T-1\}, \forall t \in \mathcal{T} \tag{4.68}$$

$$y_{\beta t} \ge 0 \qquad \forall \beta \in \{0,\dots,T-1\}, \forall t \in \mathcal{T} \tag{4.69}$$

$$f_t^{\alpha,\beta} \ge 0 \qquad \forall \alpha,\beta \in \{0,\dots,T-1\}, \forall t \in \mathcal{T} \tag{4.70}$$

$$f_t^{\alpha,\beta} \in \{0,1\} \qquad \forall \alpha,\beta \in \{0,\dots,T-1\}, \forall t \in \mathcal{T}. \tag{4.71}$$

Constraints (4.63) and (4.64) link the unit flow variables with the unit stock variables. Constraints (4.65) represent flow conservation equations: when $I_t = \alpha$ and $I_{t0} = \beta$, if no pick-up is performed at time t, then $I_{t+1} = \alpha + 1$ and $I_{t+1,0} = \beta - 1$, and therefore $f_{t+1}^{\alpha+1,\beta-1} = 1$; if a pick-up is performed at time t, $I_{t+1} = 0$ and $I_{t+1,0} = \alpha + \beta$, and therefore $f_{t+1}^{0,\alpha+\beta} = 1$. Constraints (4.66) impose that the unit flow is equal to zero for each (α, β) combination that results in more than $T - 1$ units of total inventory. Constraints (4.67) ensure a flow of one unit on each day. Constraints (4.68)–(4.71) define the unit stock variables and the unit flow variables. Constraints (4.70) are redundant, but their inclusion allows us to easily indicate that the matrix associated with (4.63)–(4.70) is totally unimodular. Because all the entries of the matrix corresponding to (4.63)–(4.70) are -1, 0, or 1, it is totally unimodular and therefore the polyhedron of $(\mathbf{EUF})$ is integral. Theorem 4.1 states that the polyhedron of the extended unit stock formulation is also integral. We prove Theorem 4.1 in 4.B by obtaining the polyhedron of the extended unit stock formulation by projecting the extended unit flow polyhedron in the space of the variables of $(\mathbf{EUS})$.

Theorem 4.1. *The linear programming formulation* (4.50)-(4.58) *has solutions with integer-valued* I_t *and* I_{t0}, $\forall t \in \mathcal{T}$, *and* z_t, $\forall t \in \mathcal{T}$.

We now show how to obtain a description of the convex hull of the solutions of $(\mathbf{SILS})$ through the Fourier-Motkzin elimination of variables w and y in the $(\mathbf{EUS})$. We consider the case of $T = 3$. From (4.51), $u_{2t} = \frac{I_t - u_{1t}}{2}$. From (4.52) and (4.55), we have $u_{1t} = 1 - u_{0t} - u_{2t} = 1 - z_t - \frac{I_t - u_{1t}}{2}$. From (4.57), this implies:

$$2 - 2z_t - I_t \ge 0. \tag{4.72}$$

Moreover, substituting $u_{1t} = 2 - 2z_t - I_t$ into $u_{2t} = \frac{I_t - u_{1t}}{2}$ and noting (4.57) yields:

$$I_t + z_t - 1 \geq 0. \tag{4.73}$$

From (4.53), $y_{2t} = \frac{I_{t0} - y_{1t}}{2}$. Combined with (4.54), we obtain $y_{0t} = \frac{2 - I_{t0} - y_{1t}}{2}$. Substituting $u_{1t} = 2 - 2z_t - I_t$ and (4.55) into (4.56) for $\beta = 1$, we have $y_{1t} \leq z_t + 2 - 2z_t - I_t = 2 - z_t - I_t$. From (4.55) and (4.56) for $\beta = 2$, we obtain $y_{2t} \leq z_t$. Substituting $y_{1t} \leq 2 - z_{tk} - I_t$ and $y_{2t} \leq z_t$ into (4.53), we have $I_{t0} \leq 2 - z_t - I_t + 2z_t = 2 + z_t - I_t$. However, the resulting inequality, $I_t + I_{t0} \leq 2 + z_t$ is dominated by $I_t + I_{t0} \leq T - 1$ for $T = 3$. Therefore the projection of the y variables onto the original space does not produce tight inequalities, and we omit them from the linear programming reformulation of (**SILS**). Therefore, for $T = 3$, we replace the (**SILS**) formulation with the linear program defined by (4.42)–(4.44), (4.46)–(4.48), (4.72) and (4.73), and the constraints $0 \leq z_t \leq 1$ for all $t \in \mathcal{T}$. Our polyhedral analysis provides a description of the convex hull of the solutions of the (**SILS**) in which continuous variables z_t, q_t, I_t and I_{t0} assume integer values in the extreme points. We now introduce a set of valid inequalities inspired by our polyhedral analysis. We prove Proposition 4.1 in 4.B.

Proposition 4.1. *The following inequalities are valid for P_{CIRP}:*

$$\begin{aligned}
I_{ti} &\geq d_i(1 - z_{ti}) \quad \forall i \in \mathcal{I},\ \forall\, t \in \mathcal{T} && (4.74)\\
I_{ti} &\leq \left(T - 2 + z_{next(t)i}\right) d_i \quad \forall i \in \mathcal{I},\ \forall\, t \in \mathcal{T} && (4.75)\\
I_{ti} &\geq I_{prev(t)i} + (1 - T z_{ti})\, d_i \quad \forall i \in \mathcal{I},\ \forall\, t \in \mathcal{T} && (4.76)\\
I^i_{prev(t)0} &\geq (1 - z_{ti})\, d_i \quad \forall i \in \mathcal{I},\ \ \forall\, t \in \mathcal{T} && (4.77)\\
I^i_{t0} &\leq (T - 2 + z_{ti})\, d_i \quad \forall i \in \mathcal{I},\ \ \forall\, t \in \mathcal{T} && (4.78)\\
I^i_{t0} &\leq I^i_{prev(t)0} + T d_i z_{ti} - d_i \quad \forall i \in \mathcal{I},\ \ \forall\, t \in \mathcal{T} && (4.79)\\
I^i_{t0} &\leq \left(z_{prev(t)i} + 2z_{ti}\right) d_i \quad \forall i \in \mathcal{I},\ \ \forall\, t \in \mathcal{T} = \{1, 2, 3\}. && (4.80)
\end{aligned}$$

In the following, an extension of the polyhedral analysis for any $T > 1$ is provided. The aim is to project variables w and y onto the original (I, I_0, z) space of the (**SILS**) using the Fourier-Motzkin elimination procedure. We re-write in a more explicit form all the constraints defining the feasible region of the continuous relaxation of the (**SILS**):

$$\begin{aligned}
& I_t = u_{1t} + 2u_{2t} + \ldots + (T-1)\, u_{(T-1)t} && \forall t \in \mathcal{T} && (4.81)\\
& u_{0t} + u_{1t} + u_{2t} + \ldots + u_{(T-1)t} = 1 && \forall t \in \mathcal{T} && (4.82)\\
& I_{t0} = y_{1t} + 2y_{2t} + \ldots + (T-1)\, y_{(T-1)t} && \forall t \in \mathcal{T} && (4.83)\\
& y_{0t} + y_{1t} + y_{2t} + \ldots + y_{(T-1)t} = 1 && \forall t \in \mathcal{T} && (4.84)\\
& z_t = u_{0t} && \forall t \in \mathcal{T} && (4.85)\\
& y_{0t} \leq u_{0t} + u_{1t} + u_{2t} + \ldots + u_{(T-1)t} && \forall t \in \mathcal{T} && (4.86)\\
& y_{1t} \leq u_{0t} + u_{1t} + u_{2t} + \ldots + u_{(T-2)t} && \forall t \in \mathcal{T} && (4.87)\\
& y_{2t} \leq u_{0t} + u_{1t} + u_{2t} + \ldots + u_{(T-3)t} && \forall t \in \mathcal{T} && (4.88)\\
& \ldots && \ldots && (4.89)\\
& y_{(T-1)t} \leq u_{0t} && \forall t \in \mathcal{T} && (4.90)\\
& u_{0t}, u_{1t}, u_{2t}, \ldots, u_{(T-1)t} \geq 0 && \forall t \in \mathcal{T} && (4.91)\\
& y_{0t}, y_{1t}, y_{2t}, \ldots, y_{(T-1)t} \geq 0 && \forall t \in \mathcal{T} && (4.92)
\end{aligned}$$

From (4.82) and (4.85), $u_{1t} = 1 - z_t - \left(u_{2t} + \ldots + u_{(T-1)t}\right)$. From (4.81) and (4.91), $\frac{I_t - u_{1t}}{(T-1)} = \frac{2}{(T-1)} u_{2t} + \ldots + u_{(T-1)t} \leq u_{2t} + \ldots + u_{(T-1)t}$. Hence, $u_{1t} \leq 1 - z_t - \left(\frac{2}{(T-1)} u_{2t} + \ldots + u_{(T-1)t}\right)$, from which $u_{1t} \leq 1 - z_t - \frac{I_t - u_{1t}}{(T-1)}$. From (4.91), $0 \leq u_{1t} \leq 1 - z_t - \frac{I_t - u_{1t}}{(T-1)}$, that is:

$$(T-1)\, z_t + I_t \leq (T-1)\,. \tag{4.93}$$

From (4.82) and (4.91), $u_{0t} + u_{1t} + 2u_{2t} + \ldots + \ldots + (T-1)\, u_{(T-1)t} \geq 1$. From (4.81), the previous inequality becomes $u_{0t} + I_t \geq 1$. Hence, from (4.85), we obtain:

$$z_t + I_t \geq 1. \tag{4.94}$$

In a similar way, we project all the remaining variables $u_{2t}, \ldots, u_{(T-1)t}$. More precisely, to project u_{2t} we use (4.81) to obtain $u_{2t} = \frac{I_t - u_{1t} - \left(3u_{3t} + \ldots + (T-1)u_{(T-1)t}\right)}{2}$. From (4.82) and (4.91), $3u_{3t} + \ldots + (T-1)\, u_{(T-1)t} \geq 1 - u_{0t} - u_{1t} - u_{2t}$. Hence, $u_{2t} \leq \left(I_t - u_{1t} - (1 - u_{0t} - u_{1t} - u_{2t})\right)/2$. From (4.91), $0 \leq 2u_{2t} \leq I_t - 1 + u_{0t} + u_{2t}$, and we obtain again inequality (4.94). We obtain again the same valid inequality (4.94), by projecting all the remaining w variables. The same logic allows to project variables $y_{0t}, y_{1t}, y_{2t}, \ldots, y_{(T-1)t}$. More specifically, from (4.86), (4.92) and (4.82), we obtain $0 \leq y_{0t} \leq u_{0t} + u_{1t} + u_{2t} + \ldots + u_{(T-1)t} = 1$. From (4.83), $y_{1t} = I_{t0} - 2y_{2t} - \ldots - (T-1)\, y_{(T-1)t}$. From (4.84) and (4.92), $2y_{2t} + \ldots + (T-1)\, y_{(T-1)t} \geq 1 - y_{0t} - y_{1t}$. Hence, from (4.92), $0 \leq y_{1t} \leq I_{t0} - 1 + y_{0t} + y_{1t}$, that is $1 - I_{t0} \leq y_{0t}$. Since $y_{0t} \leq 1$, we obtain $1 - I_{t0} \leq y_{0t} \leq 1$, that produces the following inequality:

$$I_{t0} \geq 0. \tag{4.95}$$

In a similar way, all the remaining variables $y_{2t}, \ldots, y_{(T-1)t}$ can be projected out. In fact, to project out variable y_{2t} we use (4.83) to obtain $y_{2t} = \left(I_{t0} - y_{t1} - \left(3y_{3t} + \ldots + (T-1)\, y_{(T-1)t}\right)\right)/2$. From (4.84) and (4.92), $3y_{3t} + \ldots + (T-1)\, y_{(T-1)t} \geq 1 - y_{0t} - y_{1t} - y_{2t}$. Hence, $y_{2t} \leq \frac{I_{t0} - y_{1t} - (1 - y_{0t} - y_{1t} - y_{2t})}{2} = \frac{I_{t0} - 1 + y_{0t} + y_{2t}}{2}$. From (4.92), $0 \leq y_{2t} \leq I_{t0} - 1 + y_{0t}$. Since $y_{0t} \leq 1$, then $0 \leq y_{2t} \leq I_{t0} - 1 + y_{0t} \leq I_{t0}$, that is inequality (4.95). We obtain again the same valid inequality (4.94), by projecting out all the remaining y variables.

Therefore, for any $T > 1$, we can replace the (**SILS**) formulation with the linear program defined by (4.42)–(4.44), (4.46)–(4.48), (4.93)–(4.95), and the constraints $0 \leq z_t \leq 1$ for all $t \in \mathcal{T}$. Our polyhedral analysis provides a description of the convex hull of the solutions of the (**SILS**) in which continuous variables z_t, q_t, I_t and I_{t0} assume integer values in the extreme points.

4.5 Computational results

We code our solution approach in C++ and compile with g++ -O3. We execute our computational experiments on a PC equipped with 2 Intel Xeon E5335 CPUs running at 2.00 GHz on single thread, with 6 GB of RAM.

We derive a set of instances for our inbound cyclic IRP problem from the set of outbound non-cyclic IRP instances of Archetti et al. (2007) to evaluate the performance of our solution approach. We use the locations and the demands of the customers provided in Archetti et al. (2007) as the locations and the

production rates of our suppliers, the production rate of the factory in Archetti et al. (2007) as the demand of our assembly plant, and the inventory holding cost generated by Archetti et al. (2007) in the interval $[0.1, 0.5]$ as the inventory cost at the suppliers and at the plant. We do not use the initial inventory levels as input data, as they are decision variables in our problem, and we do not use the maximum inventory level at the customers, as we do not have the maximum inventory level at the suppliers.

Mirroring the instances in Archetti et al. (2007), we consider a total of 50 single-vehicle instances (five instances for each class with 5, 10, . . ., 50 suppliers). For these one-vehicle instances, we consider both a three-day and a six-day planning horizon. To computationally demonstrate that our solution approach is scalable to multi-vehicle instances, we consider 35 instances with two vehicles (five instances for each class with 5, 10, . . ., 35 suppliers). For these two-vehicle instances, we consider both a three-day and a six-day planning horizon. We also consider 25 instances with 3 vehicles (five instances for each class with 5, 10, . . ., 25 suppliers), for which we consider a three-day planning horizon. In the instances with two and three vehicles, we use the same vehicle capacity as in the corresponding instance of Archetti et al. (2007), while in our one-vehicle instances, we double this vehicle capacity.

To provide a frame of reference for the computational difficulty of our cyclic IRP formulation, we compare it to the classical non-cyclic IRP. As the goal of our IRP formulation is to identify an optimal stationary inventory collection policy independent of any initial conditions, the initial inventory levels at the suppliers and at the plant are decision variables in our cyclic inbound IRP rather than input data. Furthermore, the inventory levels at the end of the cyclic planning horizon must be equal to initial inventory levels, and the maximum inventory level at each supplier is a function of the length of the planning horizon. However, for the classical (non-cyclic outbound) IRP, the initial inventory levels at the factory and the customers are traditionally input data, the inventory levels at the end of the planning horizon are not required to be equal to the corresponding initial inventory levels, and the maximum inventory level at each customer is exogenously set. Thus, the feasible region of our model is larger than the one of the classical IRP. In §4.5.1, our experiments show that the cyclic IRP is significantly more difficult to solve to optimality than the classical non-cyclic IRP. The results in §4.5.1 also suggest that, although tailored to our cyclic IRP formulation, our exact approach is effective at solving a non-cyclic IRP formulation.

In §4.5.2, we conduct computational experiments to investigate the impact of the inventory clearing policy and no split collections by removing these constraints from our formulation. The results in §4.5.2 demonstrate that enforcing an inventory clearing policy has a greater impact on the optimal cost than disallowing split collections for the inbound IRP. These results also provide evidence that our exact approach is capable of solving more general versions of the cyclic IRP. To the best of our knowledge, these are the first results in the literature demonstrating the performance of an exact method for an IRP with split collection (or analogously, split deliveries for the outbound IRP).

In §4.5.3, we show that our three-phase approach consistently solves to optimality single-vehicle instances with up to 50 suppliers for a three-day planning horizon, and with up to 30 suppliers for a six-day planning horizon. For the single-vehicle instances, we examine the computational behavior of the third phase's brand-and-cut algorithm in 4.D.

In §4.5.4, we show that, although scalability is a challenge even for the much simpler non-cyclic IRP, our three-phase approach scales relatively well to instances of the cyclic IRP with two vehicles over a three-day and a six-day planning horizon, and to instances with three vehicles over a three-day planning horizon. For a three-day planning horizon, our three-phase approach solves to optimality all two-vehicle

instances with up to 15 suppliers and seven out of ten three-vehicle instances with up to 10 suppliers. For two vehicles over six periods, the proposed solution approach is able to optimally solve 5 of the 25 instances. We examine the computational behavior of the third phase's brand-and-cut algorithm for the multi-vehicle instances in 4.E.

In 4.F and 4.G, we present results from additional experiments examining the behavior of the first phase's extended lower bound formulation and of the second phase's heuristic procedure, respectively.

4.5.1 Cyclic Versus Non-Cyclic IRP

In this section, we execute computational experiments that demonstrate that our cyclic IRP formulation is more difficult to solve than the non-cyclic IRP. We modify formulation (**P**) to allow a non-cyclic planning horizon with fixed initial inventory levels. To fix the initial inventory level at supplier i to $\overline{I}_{0i}$, we replace constraints (4.2) with

$$I_{1i} = \overline{I}_{0i} + d_i - q_{1i} \quad \forall i \in \mathcal{I} \tag{4.96}$$

and

$$I_{ti} = I_{prev(t),i} + d_i - q_{ti} \quad \forall t \in \{2, \ldots, T\}, \forall i \in \mathcal{I}. \tag{4.97}$$

Similarly, to fix the initial inventory level of component i at the plant to $\overline{I}^i_{00}$, we replace (4.3) with

$$I^i_{10} = \overline{I}^i_{00} + q_{1i} - d_i \quad \forall i \in \mathcal{I} \tag{4.98}$$

and

$$I^i_{t0} = I^i_{prev(t),0} + q_{ti} - d_i \quad \forall t \in \{2, \ldots, T\}, \forall i \in \mathcal{I}. \tag{4.99}$$

We assume that $\overline{I}_{0i}$ and $\overline{I}^i_{00}$ are integer multiples of d_i for each $i \in \mathcal{I}$. Because $I_{1i} = \overline{I}_{0i} + d_i - q_{1i} \leq (T-1)d_i\,(1 - z_{1i})$, if $z_{1i} = 0$, then $I_{1i} = \overline{I}_{0i} + d_i \leq (T-1)d_i$ and consequently $\overline{I}_{0i} \leq (T-2)\,d_i$. On the other hand, because $I_{ti} + I^i_{t0} = \overline{I}_{0i} + \overline{I}^i_{00} \leq (T-1)d_i$, if $\overline{I}_{0i} = (T-2)\,d_i$ then $\overline{I}^i_{00} \leq d_i$, while $\overline{I}^i_{00} = (T-1)\,d_i$ implies $\overline{I}_{0i} = 0$. Therefore, $\overline{I}_{0i}$ can be selected in $\{0, d_i, \ldots, (T-2)\,d_i\}$, and $\overline{I}^i_{00}$ can be chosen in $\{0, d_i, \ldots, (T-1)\,d_i\}$, but in such a way that satisfies $\overline{I}_{0i} + \overline{I}^i_{00} \leq (T-1)d_i$. Recalling that the pickup quantities q_{ti} are integer multiples of d_i for each $t \in \mathcal{T}$, and $i \in \mathcal{I}$, the lot-sizing inequalities (4.74), (4.76), (4.77), and (4.78) are still valid. Furthermore, the parity inequalities and classical inequalities remain valid for each $t \in \mathcal{T}$. We modify the extended relaxation (**R**) accordingly.

The first column of Table 4.5.1 (and subsequent tables) provides the instance name in the form CIRLwtxkyiz, where w denotes the instance index $(1, 2, \ldots, 5)$, x is the length of the cyclic planning horizon, y denotes the number of vehicles, and z denotes the number of the suppliers. For the case when initial inventory levels at the suppliers are free decision variables, the second, third, and fourth columns list the gap between the extended lower bound and the best known upper bound, whether the extended lower bound corresponded to an optimal integer solution, and the computation time. The fifth, sixth, and seventh columns correspond to the analogous information for the case when the initial inventory level for component i is fixed to d_i at the supplier and plant for all $i \in \mathcal{I}$. For the case when the initial inventory levels at the suppliers are free decision variables, the eighth column shows the gap between the best upper bound found by the branch-and-cut algorithm in our third phase and the best known lower bound; the ninth column lists the corresponding computation time. The tenth and eleventh columns show the analogous information regarding the branch-and-cut for the case when initial inventory level for component i is fixed to d_i at the supplier and plant for all $i \in \mathcal{I}$.

Table 4.5.1: Cyclic IRP Versus Non-Cyclic IRP

Instance	Extended Lower Bound						Branch-and-Cut			
	Cyclic			Non-Cyclic			Cyclic		Non-Cyclic	
	% Gap	Integer solution	Time (sec)	% Gap	Integer Solution	Time (sec)	% Gap	Time (sec)	% Gap	Time (sec)
CIRL1t6k1i05	0.00	*	4.6	0.00	*	0.16	0.00	-	0.00	-
CIRL1t6k1i10	0.00	*	269.51	0.00	*	5.99	0.00	-	0.00	-
CIRL1t6k1i15	0.00	*	592.73	0.00	*	18.48	0.00	-	0.00	-
CIRL1t6k1i20	1.00		TL	0.00	*	138.09	0.00	5879.62	0.00	-
CIRL1t6k1i25	0.00	*	7170.73	0.00	*	182.23	0.00	-	0.00	-
CIRL1t6k1i30	3.00		TL	0.00	*	570.54	3.02	TL	0.00	-
CIRL1t6k1i35	3.00		TL	0.00	*	1236.57	2.24	TL	0.00	-
CIRL1t6k1i40	3.62		TL	0.00	*	2674.70	2.78	TL	0.00	-
CIRL1t6k1i45	10.36		TL	0.00	*	2641.81	8.61	TL	0.00	-
CIRL1t6k1i50	15.65		TL	0.00	*	4181.34	15.07	TL	0.00	-
Average	3.66		5123.76	0.00		1164.99	3.17	12979.94	0.00	

The computational results in Table 4.5.1 clearly show that the cyclic IRP is significantly more difficult to solve to optimality than non-cyclic IRP with fixed initial inventory levels. In these tests, our three-phase approach obtains optimal solutions for the one-vehicle instances with up to 25 customers. However, for the equivalent non-cyclic instances, just the first phase of our approach is able to find an extended lower bound that is an optimal integer solution for instances with up to 50 customers. Furthermore, our modifications to (**P**) to make it non-cyclic with fixed initial inventory levels, but we do not enforce a maximum inventory level as typically done in the standard non-cyclic IRP. Thus, our computational comparison in Table 4.5.1 is an under-estimate of the difference in computational difficulty between our cyclic IRP formulation and the non-cylic IRP formulation considered in the literature as the feasible region as defined by Archetti et al. (2007) will be even smaller than what we consider.

4.5.2 Allowing Split Collection

In this section, we demonstrate that our solution approach is capable of handling more general versions of the cyclic IRP, in particular the case without the inventory clearing policy and with split collection. Because $(\mathcal{P})$ is defined on the basis of decision variables that do not use an explicit indexing of the vehicles in the fleet, we refer to $(\mathcal{P}')$ in order to analyze the effect of allowing split collection in our problem. Constraints (4.18) implicitly prohibit split collection in any feasible solution of $(\mathbf{P}')$. We can allow split collection in a feasible solution of $(\mathbf{P}')$ by relaxing constraints (4.18). By relaxing constraints (4.18), the pickup quantities q_{tki} may no longer be integer multiples of d_i for each $t \in \mathcal{T}$, $k \in \mathcal{K}$, and $i \in \mathcal{I}$. Thus, the extended formulation is not a valid lower bound anymore, but it can still provide a feasible solution. However, we can still run the second-phase heuristic and third-phase branch-and-cut with a feasible solution to prune nodes in the tree.

This demonstrates that our approach is capable of handling more general versions of the cyclic IRP, although we have tailored our analysis to a cyclic inbound IRP with an inventory clearing policy. Table 4.5.2 shows the results of a computational experiment to gauge the impact of enforcing an inventory clearing policy and no split collection. The first column of Table 4.5.2 lists the names of 15 instances (with two vehicles and a three-day planning horizon) that were solved to optimality in Table 4.E.1. The second column provides the optimal values obtained by the branch-and-cut algorithm for solving $(\mathbf{P}')$. The third column lists the optimal values obtained by the branch-and-cut algorithm for solving $(\mathbf{P}')$ without

Table 4.5.2: Effect of Inventory Clearing Policy and Split Collection on Optimal Cost

	Optimal Cost			% Gap Resulting From		
Instance	ICP, No Split	No ICP, Split	No ICP, No Split	ICP, No Split	ICP	No Split
CIRL1t3k2i05	1801.05	1796.22	1796.22	0.27	0.27	0.00
CIRL2t3k2i05	2665.44	2163.72	2163.72	23.19	23.19	0.00
CIRL3t3k2i05	4007.30	3858.16	3866.46	3.87	3.64	0.21
CIRL4t3k2i05	2438.84	2438.84	2438.84	0.00	0.00	0.00
CIRL5t3k2i05	2129.34	2032.20	2045.39	4.78	4.10	0.64
CIRL1t3k2i10	3972.29	3963.33	3963.33	0.23	0.23	0.00
CIRL2t3k2i10	4398.27	4261.35	4267.75	3.21	3.06	0.15
CIRL3t3k2i10	3610.82	3552.78	3552.78	1.63	1.63	0.00
CIRL4t3k2i10	3781.87	3737.13	3737.13	1.20	1.20	0.00
CIRL5t3k2i10	3580.32	3484.96	3484.96	2.74	2.74	0.00
CIRL1t3k2i15	3807.30	3747.91	3747.91	1.58	1.58	0.00
CIRL2t3k2i15	4077.82	4002.76	4002.76	1.88	1.88	0.00
CIRL3t3k2i15	4820.46	4747.54	4747.54	1.54	1.54	0.00
CIRL4t3k2i15	3865.76	3714.92	3714.92	4.06	4.06	0.00
CIRL5t3k2i15	4349.72	4257.67	4257.67	2.16	2.16	0.00
Average				3.49	3.42	0.07

constraints (4.18), thus relaxing the inventory clearing policy and therefore allowing split collection. The fourth column lists the optimal values obtained by the branch-and-cut algorithm for solving $(\mathbf{P}')$ without constraints (4.18), and thus relaxing the inventory clearing policy, but prohibiting split collection. The fifth column reports the percent gap between the second column and third column values, thereby measuring the impact of enforcing an inventory clearing policy and no split collection. The sixth column reports the percent gap between the second column and fourth column values, thereby measuring the impact of enforcing an inventory clearing policy when no split collection is allowed. The seventh column reports the percent gap between the third column and fourth column values, thereby measuring the impact of prohibiting split collection when no particular inventory policy is enforced.

An average gap of 3.49 percent results from enforcing an inventory clearing policy and prohibiting split collection, but this is skewed by the gap of 23.19 percent on instance CIRL2t3k2i5. The large gap on instance CIRL2t3k2i5 results from the existence of a supplier with a daily demand that is larger than $Q/2$. Because the optimal solution of $(\mathbf{P}')$ admits only pickup quantities that are integer multiples of the demands, the routing solution significantly changes when the inventory clearing policy constraints (4.18) are relaxed. However, this increase in cost is primarily due to the inventory clearing policy. If we prohibit split collection, but consider the impact of the inventory clearing policy, an average cost increase of 3.42 percent results. As Table 4.5.2 testifies, allowing or disallowing split collection does not seem to have much of an impact due to the overwhelming impact of the inventory clearing policy.

4.5.3 Performance of Solution Approach: Single-Vehicle Case

Tables 4.5.3 and 4.5.4 show the performance of the phases of our solution approach for the instances with one vehicle and three periods, and one vehicle and six periods, respectively. The second column expresses the gap between the extended lower bound and the best known upper bound: whenever the extended lower bound formulation is solved to optimality within its two-hour time limit and the solution is integer, the optimality gap is zero. The third column contains an asterisk whenever the extended lower bound corresponds to an optimal integer solution (which is also an optimal solution of $(\mathbf{P})$). The fourth column lists the optimality gap of the second-phase heuristic with respect to the best known

lower bound. Similarly, the fifth column displays the gap between the best upper bound achieved by the third-phase branch-and-cut and the best known lower bound. Whenever the first-phase extended lower bound formulation is solved to optimality and the solution is integer, we have already obtained an optimal solution to (**P**) and do not execute the second-phase and third-phase. Table 4.5.3 shows that the extended lower bound determined in the first phase of our algorithm coincided with the optimal solution in 39 of the 50 one-vehicle, three-period instances. The heuristic based on the extended lower bound is also very effective, providing an average optimality gap of 0.33 percent on the 11 instances for which the extended lower bound does not achieve the optimal integer solution. In our third phase, the branch-and-cut algorithm solves the remaining 11 one-vehicle instances to optimality. Table 4.5.4 shows that the extended lower bound determined in the first phase of our algorithm coincided with the optimal solution in 26 of the 50 one-vehicle, six-period instances. The heuristic based on the extended lower bound remains effective, providing an average optimality gap of 5.41 percent on the 24 instances for which the extended lower bound does not achieve the optimal integer solution. The third-phase branch-and-cut algorithm is unable to solve any of the remaining 24 instances to optimality, but it does close the average gap to 4.14 percent.

4.5.4 Performance of the Solution Approach: Multi-vehicle Case

Tables 4.5.5, 4.5.6 and 4.5.7 show the performance of the phases of our solution approach for the instances with two vehicles and three periods, three vehicles and three periods, and two vehicles and six periods respectively.

Table 4.5.5 demonstrates that the two-vehicle, three-period instances are more difficult. We obtain an optimal solution in 16 of the 35 instances (each instance with 5, 10, and 15 suppliers and one instance with 20 suppliers), all via the extended lower bound in our first phase. The heuristic based on the extended lower bound obtains solutions with an average optimality gap of 6.93 percent on the 19 instances for which the extended lower bound does not achieve an optimal integer solution. The third-phase branch-and-cut algorithm achieves an average optimality gap of 4.56 percent on the 19 two-vehicle instances not solved to optimality by the extended lower bound. Table 4.5.6 shows that our algorithm is able to find an optimal solution in 7 of the 35 three-vehicle, three-period instances, all via the extended lower bound in our first phase. The heuristic based on the extended lower bound obtains solutions with an average optimality gap of 17.10 percent on the 28 instances for which the extended lower bound does not achieve an optimal integer solution. The third-phase branch-and-cut algorithm achieves an average optimality gap of 11.59 percent on the 28 two-vehicle instances not solved to optimality by the extended lower bound. Finally Table 4.5.7 shows that the extended lower bound determined in the first phase of our algorithm coincided with the optimal solution in 5 of the 25 two-vehicle, six-period instances. The heuristic based on the extended lower bound produces an average optimality gap of 20.48 percent on the 20 instances for which the extended lower bound does not achieve the optimal integer solution. The third-phase branch-and-cut algorithm is unable to solve any of the remaining 20 instances to optimality, but it does close the average gap to 11.48 percent.

4.6 Conclusion

Motivated by the lean production concepts of heijunka and standard work, we develop the first exact method for the inbound inventory routing problem with a cyclic planning horizon and inventory-clearing

Table 4.5.3: 3-Phase Performance on One-Vehicle, Three-Period Instances

Instance	Extended Lower Bound		Heuristic	Best Upper Bound
	% Gap	Integer solution	% Gap	% Gap
CIRL1t3k1i05	0.00	*		
CIRL2t3k1i05	0.00	*		
CIRL3t3k1i05	0.00	*		
CIRL4t3k1i05	0.00	*		
CIRL5t3k1i05	0.00	*		
CIRL1t3k1i10	0.00	*		
CIRL2t3k1i10	0.00	*		
CIRL3t3k1i10	0.00	*		
CIRL4t3k1i10	0.00	*		
CIRL5t3k1i10	0.00	*		
CIRL1t3k1i15	0.00	*		
CIRL2t3k1i15	0.00	*		
CIRL3t3k1i15	0.00	*		
CIRL4t3k1i15	0.00	*		
CIRL5t3k1i15	0.00	*		
CIRL1t3k1i20	0.00	*		
CIRL2t3k1i20	0.00	*		
CIRL3t3k1i20	0.00	*		
CIRL4t3k1i20	0.00	*		
CIRL5t3k1i20	0.00	*		
CIRL1t3k1i25	0.00	*		
CIRL2t3k1i25	0.00	*		
CIRL3t3k1i25	0.00	*		
CIRL4t3k1i25	0.00	*		
CIRL5t3k1i25	0.00	*		
CIRL1t3k1i30	0.00	*		
CIRL2t3k1i30	0.00	*		
CIRL3t3k1i30	0.00	*		
CIRL4t3k1i30	0.00	*		
CIRL5t3k1i30	0.00	*		
CIRL1t3k1i35	0.00	*		
CIRL2t3k1i35	0.00	*		
CIRL3t3k1i35	0.00		0.00	0.00
CIRL4t3k1i35	0.00	*		
CIRL5t3k1i35	0.00		0.29	0.00
CIRL1t3k1i40	0.00	*		
CIRL2t3k1i40	1.19		0.44	0.00
CIRL3t3k1i40	0.00	*		
CIRL4t3k1i40	0.00	*		
CIRL5t3k1i40	0.00	*		
CIRL1t3k1i45	0.00	*		
CIRL2t3k1i45	0.00		0.00	0.00
CIRL3t3k1i45	0.00		0.00	0.00
CIRL4t3k1i45	0.00	*		
CIRL5t3k1i45	0.00		0.49	0.00
CIRL1t3k1i50	1.65		0.59	0.00
CIRL2t3k1i50	4.40		1.65	0.00
CIRL3t3k1i50	0.14		0.14	0.00
CIRL4t3k1i50	0.82		0.07	0.00
CIRL5t3k1i50	0.00		0.00	0.00
Average	0.16		0.07	0.00

Table 4.5.4: 3-Phase Performance on One-Vehicle, Six-Period Instances

Instance	Extended Lower Bound		Heuristic	Best Upper Bound
	% Gap	Integer solution	% Gap	% Gap
CIRL1t6k1i05	0.00	*		
CIRL2t6k1i05	0.00	*		
CIRL3t6k1i05	0.00	*		
CIRL4t6k1i05	0.00	*		
CIRL5t6k1i05	0.00	*		
CIRL1t6k1i10	0.00	*		
CIRL2t6k1i10	0.00	*		
CIRL3t6k1i10	0.00	*		
CIRL4t6k1i10	0.00	*		
CIRL5t6k1i10	0.00	*		
CIRL1t6k1i15	0.00	*		
CIRL2t6k1i15	0.00	*		
CIRL3t6k1i15	0.00	*		
CIRL4t6k1i15	0.00	*		
CIRL5t6k1i15	0.00	*		
CIRL1t6k1i20	1.15		0.01	0.01
CIRL2t6k1i20	0.00	*		
CIRL3t6k1i20	0.00	*		
CIRL4t6k1i20	2.13		1.39	0.30
CIRL5t6k1i20	0.00	*		
CIRL1t6k1i25	0.00	*		
CIRL2t6k1i25	5.74		3.08	0.92
CIRL3t6k1i25	0.37		0.01	0.01
CIRL4t6k1i25	0.66		0.01	0.01
CIRL5t6k1i25	3.55		1.22	0.01
CIRL1t6k1i30	3.00		3.37	3.02
CIRL2t6k1i30	0.00	*		
CIRL3t6k1i30	0.00	*		
CIRL4t6k1i30	0.00	*		
CIRL5t6k1i30	0.00	*		
CIRL1t6k1i35	2.64		2.24	2.24
CIRL2t6k1i35	0.00	*		
CIRL3t6k1i35	6.34		4.57	0.01
CIRL4t6k1i35	10.66		8.38	1.35
CIRL5t6k1i35	2.53		1.60	1.08
CIRL1t6k1i40	3.62		2.78	2.78
CIRL2t6k1i40	12.36		10.31	1.07
CIRL3t6k1i40	2.96		0.82	0.82
CIRL4t6k1i40	10.05		6.05	1.97
CIRL5t6k1i40	0.00	*		
CIRL1t6k1i45	10.36		8.61	8.61
CIRL2t6k1i45	17.49		14.96	14.96
CIRL3t6k1i45	0.44		0.25	0.01
CIRL4t6k1i45	21.53		19.01	19.01
CIRL5t6k1i45	0.00	*		
CIRL1t6k1i50	15.65		15.07	15.07
CIRL2t6k1i50	10.82		9.34	9.34
CIRL3t6k1i50	3.25		3.02	3.02
CIRL4t6k1i50	11.21		11.65	11.65
CIRL5t6k1i50	2.40		2.01	2.02
Average	3.21		2.60	1.99

Table 4.5.5: 3-Phase Performance on Two-Vehicle, Three-Period Instances

Instance	Extended Lower Bound % Gap	Integer Solution	Heuristic % Gap	Best Upper Bound % Gap
CIRL1t3k2i5	0.00	*		
CIRL2t3k2i5	0.00	*		
CIRL3t3k2i5	0.00	*		
CIRL4t3k2i5	0.00	*		
CIRL5t3k2i5	0.00	*		
CIRL1t3k2i10	0.00	*		
CIRL2t3k2i10	0.00	*		
CIRL3t3k2i10	0.00	*		
CIRL4t3k2i10	0.00	*		
CIRL5t3k2i10	0.00	*		
CIRL1t3k2i15	0.00	*		
CIRL2t3k2i15	0.00	*		
CIRL3t3k2i15	0.00	*		
CIRL4t3k2i15	0.00	*		
CIRL5t3k2i15	0.00	*		
CIRL1t3k2i20	9.03		8.30	7.55
CIRL2t3k2i20	0.00	*		
CIRL3t3k2i20	5.07		4.00	2.56
CIRL4t3k2i20	7.20		7.30	6.63
CIRL5t3k2i20	3.89		3.10	3.01
CIRL1t3k2i25	2.54		2.40	2.30
CIRL2t3k2i25	9.27		4.80	2.53
CIRL3t3k2i25	5.02		5.30	1.40
CIRL4t3k2i25	2.35		2.40	2.39
CIRL5t3k2i25	13.62		8.50	4.84
CIRL1t3k2i30	6.10		3.90	1.91
CIRL2t3k2i30	11.66		7.80	2.46
CIRL3t3k2i30	2.50		2.50	2.43
CIRL4t3k2i30	13.34		8.40	7.66
CIRL5t3k2i30	12.59		11.20	10.04
CIRL1t3k2i35	9.78		7.60	7.10
CIRL2t3k2i35	14.58		10.00	2.58
CIRL3t3k2i35	20.39		17.00	8.19
CIRL4t3k2i35	20.61		18.00	14.17
CIRL5t3k2i35	7.68		6.00	1.42
Average	5.06		3.96	2.60

policy governing the pickup amounts. We demonstrate that this strategic problem is computationally more difficult than the traditional operational-level outbound IRP in which the initial and maximum inventory levels are predetermined values (not decision variables as they are in our formulation). Nevertheless, our exact method achieves provably optimal solutions on many instances and optimality gaps on other instances typifying an effective heuristic as a byproduct of our analysis. We also demonstrate that our approach is capable of solving more general IRP formulations (with non-cyclic planning horizons, split collection and a non-specified inventory policy). For a three-period planning horizon, our solution approach is able to optimally solve all one-vehicle instances with up to 50 suppliers, all two-vehicle instances with up to 15 suppliers, and seven out of ten three-vehicle instances with up to 10 suppliers (and an average optimality gap of 1.57 percent on the three instances for which a provably-optimal solution is not obtained). For two-vehicle, three-period instances with 20 to 35 suppliers, the average optimality gap is 4.56 percent. The average optimality gap for three-vehicle, three-period instances with 15 to 35 suppliers is 11.59 percent. For a six-period planning horizon, the proposed solution approach is able to optimally solve 26 of the 50 one-vehicle instances and achieves an average optimality gap of 4.14 percent on the 24 instances for which it does not optimally solve. For two-vehicle instances over six

Table 4.5.6: 3-Phase Performance on Three-Vehicle, Three-Period Instances

Instance	Extended Lower Bound % Gap	Integer Solution	Heuristic % Gap	Best Upper Bound % Gap
CIRL1t3k3i05	0.00	*		
CIRL2t3k3i05	0.00	*		
CIRL3t3k3i05	0.00	*		
CIRL4t3k3i05	0.00	*		
CIRL5t3k3i05	0.00	*		
CIRL1t3k3i10	1.56		2.16	1.59
CIRL2t3k3i10	16.07		4.21	3.12
CIRL3t3k3i10	0.01	*		
CIRL4t3k3i10	0.01	*		
CIRL5t3k3i10	2.08		0.01	0.01
CIRL1t3k3i15	18.35		11.37	4.81
CIRL2t3k3i15	6.09		4.70	3.43
CIRL3t3k3i15	12.80		7.88	5.35
CIRL4t3k3i15	9.57		5.22	3.07
CIRL5t3k3i15	20.23		8.19	2.46
CIRL1t3k3i20	25.71		18.08	12.82
CIRL2t3k3i20	18.13		10.78	4.28
CIRL3t3k3i20	21.94		10.45	5.82
CIRL4t3k3i20	27.98		15.88	12.44
CIRL5t3k3i20	32.42		22.43	18.19
CIRL1t3k3i25	30.68		24.10	14.98
CIRL2t3k3i25	28.41		14.72	12.05
CIRL3t3k3i25	25.50		15.27	13.12
CIRL4t3k3i25	19.01		11.90	4.18
CIRL5t3k3i25	32.46		20.29	15.98
CIRL1t3k3i30	33.24		29.25	22.44
CIRL2t3k3i30	27.39		22.99	15.60
CIRL3t3k3i30	22.36		18.53	11.83
CIRL4t3k3i30	36.25		32.13	23.82
CIRL5t3k3i30	27.13		19.06	16.01
CIRL1t3k3i35	38.39		43.65	29.46
CIRL2t3k3i35	30.90		24.86	5.34
CIRL3t3k3i35	33.79		26.41	20.52
CIRL4t3k3i35	36.70		34.27	25.52
CIRL5t3k3i35	25.72		20.11	16.15
Average	18.88		13.68	9.72

periods, the proposed solution approach is able to optimally solve 5 of the 25 instances and achieves an average optimality gap of 11.48 percent on the 20 instances for which it does not optimally solve. The average optimality gaps on the instances in which our approach fails to obtain provably optimal solutions reveals that our analysis on an exact approach also yields an effective heuristic for our problem. We also demonstrate that our approach is capable of solving more general IRP formulations (with non-cyclic planning horizons, split collection and a non-specified inventory policy).

Table 4.5.7: 3-Phase Performance on Two-Vehicle, Six-Period Instances

Instance	Extended Lower Bound % Gap	Integer solution	Heuristic % Gap	Best Upper Bound % Gap
CIRL1t6k2i05	0.00	*		
CIRL2t6k2i05	0.00	*		
CIRL3t6k2i05	0.00	*		
CIRL4t6k2i05	0.00	*		
CIRL5t6k2i05	0.00	*		
CIRL1t6k2i10	25.75		14.68	3.63
CIRL2t6k2i10	31.75		15.06	4.78
CIRL3t6k2i10	16.97		8.51	3.58
CIRL4t6k2i10	30.61		16.00	6.37
CIRL5t6k2i10	12.38		4.73	0.95
CIRL1t6k2i15	23.97		17.47	11.95
CIRL2t6k2i15	23.25		17.43	2.49
CIRL3t6k2i15	29.17		20.13	20.13
CIRL4t6k2i15	33.39		23.33	20.41
CIRL5t6k2i15	34.10		20.69	7.96
CIRL1t6k2i20	43.33		32.24	12.33
CIRL2t6k2i20	34.05		25.21	3.64
CIRL3t6k2i20	26.74		19.54	4.92
CIRL4t6k2i20	42.35		27.71	26.75
CIRL5t6k2i20	38.42		24.87	24.14
CIRL1t6k2i25	37.26		30.16	3.71
CIRL2t6k2i25	36.55		26.29	22.72
CIRL3t6k2i25	32.36		24.18	8.35
CIRL4t6k2i25	31.00		22.62	22.50
CIRL5t6k2i25	30.99		18.68	18.23
Average	24.58		16.38	9.18

Appendix

4.A Translation of Formulation for Outbound Logistics

It is possible to translate (**P**), the formulation for an inbound IRP with a cyclic planning horizon and inventory clearing policy, into an equivalent outbound IRP through a few modifications of constraints and coefficients. Specifically, we define the feasible region for the outbound IRP formulation (**O**) with (4.5) - (4.14) as well as:

$$I_{ti} = I_{prev(t),i} - d_i + q_{ti} \quad \forall t \in \mathcal{T}, \forall i \in \mathcal{I}, \tag{4.100}$$

$$I^i_{t0} = I^i_{prev(t),0} - q_{ti} + d_i \quad \forall t \in \mathcal{T}, \forall i \in \mathcal{I}, \tag{4.101}$$

and

$$I^i_{t0} \leq (T-1)d_i\,(1 - z_{ti}) \quad \forall t \in \mathcal{T}, \forall i \in \mathcal{I}. \tag{4.102}$$

Then, if we define the objective of (**O**) as

$$\min \sum_{t\in\mathcal{T}} \sum_{(i,j)\in\mathcal{E}} c_{ij}x_{tij} + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} h^i_0 I_{ti} + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} h_i I^i_{t0}, \tag{4.103}$$

the outbound IRP formulation (**O**) will have the same optimal solution as the inbound IRP formulation (**O**). The formulation (**O**) corresponds to an outbound IRP with known and constant demand d_i at each customer i and synchronized production of product i at the plant. The plant follows an inventory clearing policy for each product i such that when it distributes product i, it ships all on-hand inventory of product i.

We note that in the outbound IRP with only one product type, the plant can pool the customers' demands and distribute production accordingly. However, the translation of our inbound IRP formulation results in a multi-product outbound IRP in which such inventory pooling is not possible.

4.B Proofs Related to Valid Inequalities

Theorem 4.1. *The linear programming formulation* (4.50)-(4.58) *has solutions with integer-valued* I_t *and* I_{t0}, $\forall t \in \mathcal{T}$, *and* z_t, $\forall t \in \mathcal{T}$.

Proof. *We proceed by proving that the extended unit stock polyhedron* (4.51)–(4.58) *is an integral polyhedron. Using Fourier-Motzkin variable elimination, constraints* (4.51)–(4.58) *can be obtained by projecting the extended unit flow polyhedron* (4.63)–(4.70) *onto the* (w, y) *space using the Fourier-Motzkin*

elimination. Specifically, constraints (4.52) *come from* (4.63) *and* (4.67). *Similarly, constraints* (4.54) *come from* (4.64) *and* (4.67). *To obtain constraints* (4.56), *observe*

$$\begin{aligned} y_{\beta t} &= \sum_{\alpha=0}^{T-1-\beta} f_t^{\alpha,\beta} + \sum_{\alpha=T-\beta}^{T-1} f_t^{\alpha,\beta} \\ &\leq \sum_{\beta=0}^{T-1} \sum_{\alpha=0}^{T-1-\beta} f_t^{\alpha,\beta} + \sum_{\beta=1}^{T-1} \sum_{\alpha=T-\beta}^{T-1} f_t^{\alpha,\beta} \\ &= \sum_{\beta=0}^{T-1} \sum_{\alpha=0}^{T-1-\beta} f_t^{\alpha,\beta} \\ &= \sum_{\alpha=0}^{T-1-\beta} u_{\alpha t}, \end{aligned}$$

where the first line follows from (4.64), *the second line holds because we are adding non-negative quantities, the third line follows from* (4.66), *and the fourth line follows from* (4.63).

To obtain constraints (4.51), first observe from (4.63) that

$$\sum_{\alpha=0}^{T-1} \alpha u_{\alpha t} = \sum_{\alpha=0}^{T-1} \alpha \sum_{\beta=0}^{T-1} f_t^{\alpha,\beta},$$

but from constraints (4.67) and the total unimodularity of the extended unit flow polyhedron, it follows that only one of the $f_t^{\alpha,\beta}$ *terms on the right-hand side of this equation will be 1 and the others 0. This implies that* $\sum_{\alpha=0}^{T-1} \alpha \sum_{\beta=0}^{T-1} f_t^{\alpha,\beta}$ *corresponds to the inventory level of supplier* i *on day* t, *so we have* $\sum_{\alpha=0}^{T-1} \alpha u_{\alpha t} = I_t$. *Similarly, constraints (4.53) derives (4.64) and (4.67).*

Constraints (4.55) *are specific constraints of the extended unit stock formulation and do not impact on the total unimodularity of the matrix. Now, because the matrix corresponding to* (4.63)–(4.70) *is totally unimodular, the matrix corresponding to* (4.81)–(4.92) *is totally unimodular. The result directly follows.* □

Proposition 4.1. *The following inequalities are valid for* P_{CIRP}*:*

$$\begin{aligned} I_{ti} &\geq d_i(1 - z_{ti}) \quad \forall i \in \mathcal{I},\ \forall\, t \in \mathcal{T} \\ I_{ti} &\leq \left(T - 2 + z_{next(t)i}\right) d_i \quad \forall i \in \mathcal{I},\ \forall\, t \in \mathcal{T} \\ I_{ti} &\geq I_{prev(t)i} + (1 - T z_{ti})\, d_i \quad \forall i \in \mathcal{I},\ \forall\, t \in \mathcal{T} \\ I^i_{prev(t)0} &\geq (1 - z_{ti})\, d_i \quad \forall i \in \mathcal{I},\ \ \forall\, t \in \mathcal{T} \\ I^i_{t0} &\leq (T - 2 + z_{ti})\, d_i \quad \forall i \in \mathcal{I},\ \ \forall\, t \in \mathcal{T} \\ I^i_{t0} &\leq I^i_{prev(t)0} + T d_i z_{ti} - d_i \quad \forall i \in \mathcal{I},\ \ \forall\, t \in \mathcal{T} \\ I^i_{t0} &\leq \left(z_{prev(t)i} + 2 z_{ti}\right) d_i \quad \forall i \in \mathcal{I},\ \ \forall\, t \in \mathcal{T} = \{1, 2, 3\}. \end{aligned}$$

Proof. *To see that* (4.74) *is valid, if* $z_{ti} = 0$, *then* $I_{ti} = I_{prev(t)i} + d_i \geq d_i$ *by* (4.10); *otherwise, the inequality reduces to* (4.10). *To see that* (4.75) *is valid, if* $z_{next(t)i} = 0$, *then* $I_{next(t)i} \leq (T-1)d_i$ *by* (4.4) *and* $I_{next(t)i} = I_{ti} + d_i$ *by* (4.2) *and* (4.6), *resulting in* $I_{ti} = I_{next(t)i} - d_i \leq (T-1)d_i - d_i = (T-2)d_i$; *otherwise, the inequality holds due to* (4.4). *To see that* (4.76) *is valid, if* $z_{ti} = 1$, *then* $I_{ti} = 0$ *by* (4.4) *and we have* $I_{prev(t)i} \leq (T-1)d_i$ *which holds also due to* (4.4); *otherwise, the inequality holds due to*

(4.2) *and* (4.6)*. To see* (4.77) *is valid, if* $z_{ti} = 0$*, then* $I^i_{prev(t)0} \geq d_i$ *by* (4.3)*,* (4.6)*, and* (4.10)*; otherwise the inequality reduces to* (4.10)*.*

Inequalities (4.78)–(4.80) *rely upon* $I_{ti} + I^i_{t0} = I_{prev(t)i} + I^i_{prev(t)0}$ *which follows from* (4.2) *and* (4.3)*, as well as* $I_{ti} + I^i_{t0} \leq (T-1)d_i$ *which subsequently follows from* (4.2)*,* (4.3)*,* (4.4)*, and* (4.6)*. To see* (4.78) *is valid, if* $z_{ti} = 0$*, then* $I_{ti} \geq d_i$ *from* (4.2)*,* (4.6)*, and* (4.10) *and therefore* $I^i_{t0} \leq (T-1)d_i - I_{ti} \leq (T-2)d_i$*; otherwise, if* $z_{ti} = 1$*, then* $I_{ti} = 0$ *by* (4.4) *and* $I_{t0} \leq (T-1)d_i$*. To see* (4.79) *is valid, if* $z_{ti} = 1$*, then* $I_{ti} = 0$ *by* (4.4) *and therefore* $I^i_{t0} = I_{prev(t)i} + I^i_{prev(t)0} - I_{ti} \leq I^i_{prev(t)0} + (T-1)d_i$*; otherwise, the inequality holds from* (4.3)*.*

To see (4.80) *is valid for* $T = 3$*, first observe that when* $z_{prev(t)i} = 0$ *and* $z_{ti} = 0$*,* $I^i_{prev(t)0} = I^i_{prev(prev(t))0} - d_i$ *and* $I^i_{t0} = I^i_{prev(t)0} - d_i$ *by* (4.3)*. Thus,* $I^i_{t0} = I^i_{prev(prev(t))0} - 2d_i \leq (T-1)d_i - 2d_i = 0$*. If* $z_{ti} = 1$*, regardless the value of* $z_{prev(t)i}$*, we have* $I^i_{t0} \leq 2d_i = (T-1)d_i$*. If* $z_{prev(t)i} = 1$ *and* $z_{ti} = 0$*, then* $I^i_{t0} = I^i_{prev(t)0} - d_i \leq (T-1)d_i - d_i = 2d_i - d_i = d_i$ *by* (4.3)*.* □

4.C Additional Valid Inequalities

In addition to the single-item lot-sizing valid inequalities described in §4.4.3, we consider two other sets of valid inequalities.

We dynamically add valid inequalities based on the cocircuit inequalities (Barahona & Grötschel 1986) to the branch-and-cut in the third phase of our approach. The *parity inequalities* are:

$$\sum_{(i,j)\in\delta(S)\setminus H} x_{tij} \geq \sum_{(u,v)\in H} x_{tuv} - |H| + 1, \ \forall S \subset I \cup \{0\}, \ H \subseteq \delta(S), \ |H| \ odd, \forall t \in \mathcal{T}. \tag{4.104}$$

Proposition 4.2. *The parity inequalities are valid for* P_{CIRP}*.*

Proof. *Consider the parity inequalities. In the case which all the edges in* H *are traversed, then* $\sum_{(u,v)\in H} x_{tuv} = |H|$ *and the right-hand side of the inequality is 1. Because* $|H|$ *is odd, the cutset* $\delta(S) \setminus H$ *must be traversed at least once. Therefore,* $\sum_{(i,j)\in\delta(S)\setminus H} x_{tij} \geq 1$ *holds. In the case which not all the edges of* H *are travelled, then* $\sum_{(u,v)\in H} x_{tuv} \leq |H| - 1$ *and the right-hand side of the inequality is nonpositive. Therefore, the inequality is trivially satisfied.* □

In the branch-and-cut algorithm of the first and third phase, we add some known valid inequalities from Adulyasak et al. (2013) to the initial linear program at the root of the branch-and-bound tree. Specifically, we add to (**R**) *priority inequalities*:

$$z_{tki} \leq z_{tk0} \ \ \forall i \in \mathcal{I}, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \tag{4.105}$$

$$x_{tkij} \leq z_{tki} \ \ \forall (i,j) \in \mathcal{E} : i \neq 0, \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \tag{4.106}$$

symmetry breaking inequalities:

$$z_{tk0} \geq z_{0,next(k),t} \ \ \forall k \in \mathcal{K} - last(\mathcal{K}), \forall t \in \mathcal{T}, \tag{4.107}$$

$$\sum_{i=1}^{j} 2^{(j-i)} z_{tki} \geq \sum_{i=1}^{j} 2^{(j-i)} z_{i,next(k),t} \ \ \forall j \in \mathcal{I}, \forall k \in \mathcal{K} - last(\mathcal{K}), \forall t \in \mathcal{T}. \tag{4.108}$$

Table 4.D.1: Branch-and-Cut Performance on One-Vehicle, Three-Period Instances

Instance	Best LB	Total Cost	Time (sec)	Nodes	Capacitated Subtour Elimination Constraints			Parity Cuts	
					Import	Root	Tree	Root	Tree
CIRL3t3k1i35	6854.44	6855.12	3498.13	934	15964	467	2399	102	76
CIRL5t3k1i35	5645.12	5645.12	474.08	246	5456	469	682	83	39
CIRL2t3k1i40	6150.48	6151.00	5431.44	953	16284	792	2201	116	91
CIRL2t3k1i45	6495.91	6496.55	3572.86	625	9079	354	2973	71	91
CIRL3t3k1i45	6654.00	6654.31	462.64	63	13583	278	195	91	41
CIRL5t3k1i45	6150.75	6151.30	968.97	190	8076	771	1260	86	31
CIRL1t3k1i50	7061.57	7062.21	2351.91	241	16231	511	1282	113	38
CIRL2t3k1i50	7421.89	7422.57	11177.42	1189	16190	514	3910	111	133
CIRL3t3k1i50	6907.50	6907.94	1255.54	142	16182	303	1086	68	43
CIRL4t3k1i50	7312.26	7312.28	1353.11	120	17325	1218	1672	130	50
CIRL5t3k1i50	7284.15	7284.85	2748.12	630	7100	1054	3289	126	44
Average			3026.75	485	12861	612	1904	100	62

and *logical inequalities*:

$$\sum_{i \in \mathcal{I}} x_{tki0} \leq 1 \qquad \forall k \in \mathcal{K}, \forall t \in \mathcal{T}, \tag{4.109}$$

$$\sum_{j \in \mathcal{I}} x_{tk0j} \leq 1 \qquad \forall k \in \mathcal{K}, \forall t \in \mathcal{T}. \tag{4.110}$$

We add to (**P**) *priority inequalities:*

$$x_{tij} \leq z_{ti} \quad \forall (i,j) \in \mathcal{E} : i \neq 0, \forall t \in \mathcal{T}, \tag{4.111}$$

and *logical inequalities:*

$$\sum_{i \in \mathcal{I}} x_{ti0} = \sum_{j \in \mathcal{I}} x_{t0j} \qquad \forall t \in \mathcal{T}. \tag{4.112}$$

4.D Computational Analysis of Third Phase: Single-Vehicle Case

In this section, we examine the computational behavior of the third phase's branch-and-cut procedure on the single-vehicle instances. As listed in the first column of Table 4.D.1, we only consider the instances that are not solved to optimality by solving the extended lower bound formulation. The second column provides the value of the best lower bound. The third column and fourth column list the total cost of the solution obtained by the branch-and-cut algorithm and the computation time required to obtain it, respectively. The fifth column shows the number of nodes in the branch-and-cut. The sixth, seventh and eighth columns display the number of CSE constraints imported from the extended lower bound formulation, added to the linear program at the root node, and dynamically added to other nodes of the branch-and-cut tree, respectively. The ninth and tenth columns report the number of parity cuts added to the root node and to other nodes of the branch-and-cut tree.

From Table 4.D.1, we see that the branch-and-cut algorithm is able to find an optimal solution (within a 0.01% tolerance) in all one-vehicle, three-period instances. We import a large number of CSE constraints from the extended lower bound formulation that we add to the initial linear program at the root node and to other nodes of the branch-and-cut tree dynamically. We add fewer parity cuts.

Table 4.D.2: Branch-and-Cut Performance on One-Vehicle, Six-Period Instances

Instance	Best LB	Total Cost	Time (sec)	Nodes	Capacitated Subtour Elimination Constraints			Parity Cuts	
					Import	Root	Tree	Root	Tree
CIRL1t6k1i20	7730.89	7731.66	5879.62	15038	3632	45	638	23	142
CIRL4t6k1i20	8247.29	8271.78	TL	21276	3161	242	1661	81	249
CIRL2t6k1i25	9587.72	9675.64	TL	9112	3597	357	721	39	291
CIRL3t6k1i25	10310.93	10311.86	2477.67	2038	2806	152	133	54	199
CIRL4t6k1i25	8841.46	8842.34	1677.43	986	3233	122	228	107	145
CIRL5t6k1i25	10531.99	10533.04	2866.73	2120	3584	93	212	65	201
CIRL1t6k1i30	11022.25	11354.68	TL	4100	3721	390	828	129	295
CIRL1t6k1i35	10561.68	10798.68	TL	2502	5135	362	664	119	231
CIRL3t6k1i35	12950.98	12952.28	6876.80	939	4271	308	490	107	282
CIRL4t6k1i35	10295.01	10433.68	TL	2099	6805	195	204	84	275
CIRL5t6k1i35	10928.85	11046.44	TL	2614	6029	311	988	65	364
CIRL1t6k1i40	12171.11	12509.10	TL	900	7614	329	591	100	246
CIRL2t6k1i40	11496.81	11619.48	TL	1098	5307	431	405	144	219
CIRL3t6k1i40	12287.45	12387.84	TL	1626	6074	308	167	101	159
CIRL4t6k1i40	10660.19	10869.66	TL	990	4430	270	388	91	213
CIRL1t6k1i45	12447.95	13519.72	TL	502	5593	261	549	127	174
CIRL2t6k1i45	12228.34	14057.46	TL	357	5574	444	429	102	197
CIRL3t6k1i45	12482.83	12484.08	12996.85	1265	3069	325	605	155	118
CIRL4t6k1i45	13089.30	15576.92	TL	407	6404	451	244	71	257
CIRL1t6k1i50	13158.51	15140.84	TL	32	8315	329	8	142	62
CIRL2t6k1i50	13790.01	15078.00	TL	316	6702	679	461	150	182
CIRL3t6k1i50	13534.98	13944.14	TL	330	4086	262	195	150	66
CIRL4t6k1i50	13628.89	15216.52	TL	20	5997	404	13	155	98
CIRL5t6k1i50	13596.80	13870.78	TL	87	5109	390	85	119	108
Average			12165.63	2948	5010	311	454	103	199

Table 4.D.2 shows that the four-hour time limit is reached in 18 of the 24 one-vehicle, six-period instances. Comparing the one-vehicle instances, we add fewer CSE constraints and more parity cuts with a planning horizon of six periods versus a planning period of three periods.

Table 4.D.3 shows the impact of the valid inequalities on the lower bound at the root of the branch-and-cut in the instances with one vehicle. The second column displays the value of the lower bound at the root node when we apply all the valid inequalities (lot-sizing, classical, as well as the parity cuts and the CSE constraints imported from the extended lower bound formulation). The third column shows the value of the lower bound with only the lot-sizing and classical valid inequalities. The fourth column shows the value of the lower bound with only the lot-sizing valid inequalities. The fifth column shows the value of the lower bound with none of the valid inequalities. The sixth column computes the percent increase of the value of the lower bound at the root node due to the lot-sizing valid inequalities. The seventh column computes the percent increase of the value of the lower bound at the root node due to the lot-sizing and classical valid inequalities. The eighth column computes the percent increase of the value of the lower bound at the root node due to the entire set of valid inequalities.

Table 4.D.3 shows the effectiveness of the valid inequalities in improving the value of the lower bound at the root node. As a collection, all the valid inequalities provide a minimum lower bound increase of 18.56 percent in the one-vehicle instances. Moreover, the corresponding average increase of the lower bound resulting from applying all the valid inequalities is about 111 percent in the one-vehicle instances. By themselves, the lot-sizing valid inequalities contribute to an average increase of the lower bound of about 88 percent in the one-vehicle instances.

Table 4.D.3: Lower Bound at Root Node for One-Vehicle, Three-Period Instances

Instance	All	Lot Sizing & Class. V.I.	L.S. V.I.	None	L.S. V.I. % Increase	L.S. & Class. % Increase	All V.I. % Increase
CIRL1t3k1i05	1341.60	1341.60	1138.59	995.93	14.32	34.71	34.71
CIRL2t3k1i05	1111.78	1111.78	1111.78	844.50	31.65	31.65	31.65
CIRL3t3k1i05	2003.36	2003.36	1815.36	1300.67	39.57	54.03	54.03
CIRL4t3k1i05	1167.23	1167.23	1134.03	984.51	15.19	18.56	18.56
CIRL5t3k1i05	1357.65	1349.36	1349.36	966.24	39.65	39.65	40.51
CIRL1t3k1i10	2153.06	2153.06	2013.23	1202.43	67.43	79.06	79.06
CIRL2t3k1i10	2604.22	2558.22	2432.10	1796.99	35.34	42.36	44.92
CIRL3t3k1i10	2307.11	2306.93	2209.30	1517.35	45.60	52.04	52.05
CIRL4t3k1i10	2060.00	2060.00	1987.04	1226.00	62.08	68.03	68.03
CIRL5t3k1i10	2459.96	2459.96	2298.96	1226.00	87.52	100.65	100.65
CIRL1t3k1i15	2632.64	2629.64	2448.97	1544.60	58.55	70.25	70.44
CIRL2t3k1i15	2696.76	2696.75	2219.93	1238.07	79.31	117.82	117.82
CIRL3t3k1i15	3172.83	3135.27	2872.14	1831.58	56.81	71.18	73.23
CIRL4t3k1i15	2483.08	2361.00	2180.91	1425.67	52.97	65.61	74.17
CIRL5t3k1i15	2696.87	2696.87	2603.32	1377.83	88.94	95.73	95.73
CIRL1t3k1i20	3079.42	3035.42	2820.90	1571.84	79.46	93.11	95.91
CIRL2t3k1i20	3272.22	3082.22	2614.90	1181.04	121.41	160.98	177.06
CIRL3t3k1i20	3398.52	3255.16	3057.65	1648.00	85.54	97.52	106.22
CIRL4t3k1i20	3248.31	3116.55	2896.68	1691.08	71.29	84.29	92.08
CIRL5t3k1i20	3831.75	3801.25	3272.98	1614.81	102.69	135.40	137.29
CIRL1t3k1i25	3814.83	3421.41	3237.34	1832.95	76.62	86.66	108.13
CIRL2t3k1i25	3963.21	3791.21	3461.74	1821.02	90.10	108.19	117.64
CIRL3t3k1i25	4148.75	3979.25	3526.24	1871.16	88.45	112.66	121.72
CIRL4t3k1i25	3722.48	3658.19	3362.73	1843.94	82.37	98.39	101.88
CIRL5t3k1i25	4493.77	4417.63	4115.00	2163.75	90.18	104.17	107.68
CIRL1t3k1i30	4567.58	4326.55	4149.48	1971.00	110.53	119.51	131.74
CIRL2t3k1i30	4471.10	4421.18	3985.87	2188.06	82.16	102.06	104.34
CIRL3t3k1i30	4869.10	4731.56	4403.66	2119.31	107.79	123.26	129.75
CIRL4t3k1i30	4043.42	3959.94	3684.62	1972.00	86.85	100.81	105.04
CIRL5t3k1i30	4094.72	3931.78	3795.05	1729.01	119.49	127.40	136.82
CIRL1t3k1i35	4469.64	4225.21	4041.48	2021.33	99.94	109.03	121.12
CIRL2t3k1i35	4782.18	4588.85	4330.09	2220.00	95.05	106.70	115.41
CIRL3t3k1i35	5561.42	5497.41	5186.48	2480.33	109.10	121.64	124.22
CIRL4t3k1i35	4209.17	4198.33	3782.75	2009.97	88.20	108.88	109.41
CIRL5t3k1i35	4636.55	4574.82	3718.83	1841.98	101.89	148.36	151.72
CIRL1t3k1i40	5135.05	5016.13	4253.46	2048.97	107.59	144.81	150.62
CIRL2t3k1i40	5071.97	4749.94	4230.73	2251.97	87.87	110.92	125.22
CIRL3t3k1i40	5600.52	5115.98	4726.06	2465.98	91.65	107.46	127.11
CIRL4t3k1i40	4613.51	4505.76	4073.97	1843.97	120.93	144.35	150.19
CIRL5t3k1i40	5174.66	4855.65	4555.51	1943.03	134.45	149.9	166.32
CIRL1t3k1i45	5532.65	5213.19	4817.78	2305.00	109.01	126.17	140.03
CIRL2t3k1i45	5089.98	5020.22	4659.76	2061.98	125.98	143.47	146.85
CIRL3t3k1i45	5608.29	5402.46	5055.87	2359.00	114.32	129.01	137.74
CIRL4t3k1i45	5391.97	5289.44	4773.59	2345.03	103.56	125.56	129.93
CIRL5t3k1i45	5299.00	5093.63	4691.12	1919.65	144.37	165.34	176.04
CIRL1t3k1i50	5852.14	5653.25	5061.70	2338.95	116.41	141.70	150.20
CIRL2t3k1i50	6074.88	5754.37	5515.83	2845.95	93.81	102.2	113.46
CIRL3t3k1i50	6122.38	5941.91	5238.53	2616.00	100.25	127.14	134.04
CIRL4t3k1i50	6277.03	6160.59	5670.89	2641.02	114.72	133.27	137.67
CIRL5t3k1i50	6122.20	5935.91	5435.35	2101.00	158.70	182.53	191.39
Average					87.75	104.48	110.55

Table 4.E.1: Branch-and-Cut Performance on Two-Vehicle, Three-Period Instances

Instance	Best LB	Total Cost	Time (sec)	Nodes	Capacitated Subtour Elimination Constraints			Parity Cuts	
					Import	Root	Tree	Root	Tree
CIRL1t3k2i20	4801.73	5193.76	TL	13395	15	6	328	36	89
CIRL3t3k2i20	4834.01	4961.00	TL	14100	45	9	196	34	103
CIRL4t3k2i20	5130.10	5494.42	TL	11043	38	2	481	28	123
CIRL5t3k2i20	5736.48	5914.20	TL	15000	41	8	494	29	94
CIRL1t3k2i25	5062.32	5181.58	TL	10900	94	9	323	47	94
CIRL2t3k2i25	5573.18	5717.85	TL	7600	11	7	503	41	83
CIRL3t3k2i25	5996.97	6081.88	TL	7646	31	6	267	52	116
CIRL4t3k2i25	4892.93	5007.69	TL	9932	76	6	341	47	133
CIRL5t3k2i25	6120.23	6431.16	TL	9025	58	3	408	49	85
CIRL1t3k2i30	6267.15	6388.85	TL	5004	47	5	412	45	94
CIRL2t3k2i30	5898.94	6047.80	TL	4768	30	8	375	41	97
CIRL3t3k2i30	6092.44	6244.14	TL	5810	29	5	666	42	72
CIRL4t3k2i30	5444.46	5895.92	TL	4596	40	5	472	42	101
CIRL5t3k2i30	5156.41	5731.92	TL	4834	36	8	552	40	113
CIRL1t3k2i35	5839.22	6285.79	TL	3123	39	9	911	65	95
CIRL2t3k2i35	6224.25	6389.07	TL	3078	24	8	584	66	100
CIRL3t3k2i35	7356.81	8012.77	TL	3721	13	7	744	75	82
CIRL4t3k2i35	5811.34	6770.96	TL	2800	21	2	695	55	124
CIRL5t3k2i35	5937.54	6023.08	TL	3809	18	7	288	55	84
Average			14400	7378	37	6	476	47	100

4.E Computational Analysis of Third Phase: Multi-vehicle Case

In this section, we examine the computational behavior of the third phase's branch-and-cut procedure on the multi-vehicle instances. As listed in the first columns of Tables 4.E.1, 4.E.2, and 4.E.3, we only consider the instances that are not solved to optimality by solving the extended lower bound formulation.

Tables 4.E.1 and 4.E.2 show that the four-hour time limit is reached in all three-period instances with two and three vehicles. Compared to the one-vehicle instances in Table 4.D.1, we import fewer CSE constraints to be added to the root node or dynamically to other nodes in the branch-and-cut tree. However, we add more parity cuts in the two and three-vehicle instances than in the one-vehicle instances.

Table 4.E.3 shows that the four-hour time limit is reached in 20 of the 25 two-vehicle, six-period instances. The number of imported of CSE constraints to be added to the root node or dynamically to other nodes in the branch-and-cut tree increases significantly with respect to the single vehicle case. This is also due to the fact that the planning horizon is twice the case with three periods.

Table 4.E.4 shows the impact of the valid inequalities on the lower bound at the root of the branch-and-cut in the instances with two vehicle. All the valid inequalities provide a minimum lower bound increase of 21 percent. Moreover, the corresponding average increase of the lower bound resulting from applying all the valid inequalities is about 195.43 percent. By themselves, the lot-sizing valid inequalities contribute to an average increase of the lower bound of about 61.36 percent.

4.F Additional Computational Experiments on the Extended Lower Bound

Tables 4.F.1 and 4.F.2 examine the extended lower bound formulation's behavior on the one-vehicle and two-vehicle instances, respectively. The first column of these tables provides the instance name.

Table 4.E.2: Branch-and-Cut Performance on Three-Vehicle, Three-Period Instances

Instance	Best LB	Total Cost	Time (sec)	Nodes	Capacitated Subtour Elimination Constraints			Parity Cuts	
					Import	Root	Tree	Root	Tree
CIRL1t3k3i10	5053.81	5135.46	TL	141286	53	2	469	16	88
CIRL2t3k3i10	5602.58	5782.96	TL	152585	56	2	383	19	105
CIRL5t3k3i10	4171.13	4171.54	TL	39121	25	5	143	18	49
CIRL1t3k3i15	4331.14	4550.06	TL	30400	12	2	453	28	93
CIRL2t3k3i15	4697.66	4864.45	TL	39378	18	7	216	32	110
CIRL3t3k3i15	5532.42	5845.28	TL	26200	17	1	385	24	116
CIRL4t3k3i15	4547.92	4692.06	TL	40292	52	8	250	18	114
CIRL5t3k3i15	4978.30	5103.67	TL	36650	29	2	279	25	105
CIRL1t3k3i20	5563.72	6381.87	TL	11512	11	7	221	28	85
CIRL2t3k3i20	4750.79	4963.20	TL	10982	18	5	287	25	114
CIRL3t3k3i20	5393.85	5727.06	TL	12257	18	8	155	36	113
CIRL4t3k3i20	6025.82	6882.05	TL	10047	18	9	523	33	95
CIRL5t3k3i20	6802.26	8315.03	TL	13100	23	6	473	30	80
CIRL1t3k3i25	5965.52	7016.52	TL	9389	46	8	215	49	93
CIRL2t3k3i25	6268.91	7127.99	TL	6693	11	8	429	48	73
CIRL3t3k3i25	6835.50	7867.64	TL	7480	20	6	322	45	81
CIRL4t3k3i25	5324.98	5557.04	TL	7891	32	6	231	34	90
CIRL5t3k3i25	7020.89	8356.63	TL	8550	12	9	356	41	71
CIRL1t3k3i30	6862.49	8847.87	TL	4330	12	28	401	48	72
CIRL2t3k3i30	6307.17	7472.91	TL	4687	19	10	560	56	70
CIRL3t3k3i30	6412.47	7272.84	TL	4672	17	7	541	46	51
CIRL4t3k3i30	5925.84	7778.54	TL	3927	16	7	330	52	79
CIRL5t3k3i30	5606.42	6674.87	TL	3875	20	6	765	47	94
CIRL1t3k3i35	6334.59	8980.46	TL	2930	20	12	735	60	88
CIRL2t3k3i35	6649.49	7024.35	TL	2790	23	45	400	60	100
CIRL3t3k3i35	8103.08	10194.95	TL	3272	10	9	674	70	81
CIRL4t3k3i35	6301.67	8460.50	TL	2517	7	8	658	62	119
CIRL5t3k3i35	6435.98	7675.54	TL	3640	17	10	238	59	66
Average			14400	22837	23	9	396	40	89

Table 4.E.3: Branch-and-Cut Performance on Two-Vehicle, Six-Period Instances

Instance	Best LB	Total Cost	Time (sec)	Nodes	Capacitated Subtour Elimination Constraints			Parity Cuts	
					Import	Root	Tree	Root	Tree
CIRL1t6k2i10	7541.72	7825.70	TL	37716	26	4	208	45	109
CIRL2t6k2i10	8358.71	8778.44	TL	30576	15	2	101	37	120
CIRL3t6k2i10	6640.86	6887.18	TL	34576	43	5	178	43	120
CIRL4t6k2i10	6884.89	7353.06	TL	31970	16	1	97	17	100
CIRL5t6k2i10	6847.38	6913.00	TL	47741	52	6	147	54	110
CIRL1t6k2i15	6927.06	7867.48	TL	13500	39	6	368	32	146
CIRL2t6k2i15	7705.00	7901.80	TL	15239	52	5	180	63	155
CIRL3t6k2i15	9050.76	11331.28	TL	10868	19	4	319	48	171
CIRL4t6k2i15	7041.01	8847.10	TL	11240	78	31	274	53	125
CIRL5t6k2i15	8064.09	8761.10	TL	12943	43	9	380	69	133
CIRL1t6k2i20	8950.53	10209.22	TL	5765	44	7	459	68	207
CIRL2t6k2i20	8044.09	8347.74	TL	5956	71	5	477	66	206
CIRL3t6k2i20	8713.02	9163.76	TL	6642	51	6	345	82	155
CIRL4t6k2i20	9551.27	13039.90	TL	4653	34	6	498	81	171
CIRL5t6k2i20	11050.19	14566.66	TL	5352	47	59	530	56	146
CIRL1t6k2i25	9674.71	10046.92	TL	3165	150	36	486	118	148
CIRL2t6k2i25	10402.21	13459.94	TL	2644	31	9	878	96	178
CIRL3t6k2i25	11238.31	12261.94	TL	2700	68	59	573	78	186
CIRL4t6k2i25	9313.34	12017.60	TL	2905	107	18	555	112	167
CIRL5t6k2i25	11771.38	14396.32	TL	2930	32	11	541	93	191
Average			14400	14454	51	14	380	66	152

Table 4.E.4: Lower Bound at Root Node for Two-Vehicle, Three-Period Instances

Instance	All	Lot Sizing & Class. V.I.	L.S. V.I.	None	L.S. V.I. % Increase	L.S. & Class. % Increase	All V.I. % Increase
CIRL1t3k2i05	1758.67	1494.65	1352.65	1310.85	3.19	3.32	21.58
CIRL2t3k2i05	2582.08	2349.69	2149.32	2039.66	5.38	62.43	78.50
CIRL3t3k2i05	3444.35	3040.78	2683.78	2415.58	11.10	110.21	138.10
CIRL4t3k2i05	2294.91	2008.15	1857.99	1789.71	3.82	38.82	58.64
CIRL5t3k2i05	1916.85	1654.59	1634.59	1435.52	13.87	14.38	32.51
CIRL1t3k2i10	3114.51	2272.45	2223.50	1446.57	53.71	57.09	115.30
CIRL2t3k2i10	3927.71	3073.31	2906.70	2257.31	28.77	112.45	171.52
CIRL3t3k2i10	3128.39	2750.07	2558.40	2012.56	27.12	90.11	116.26
CIRL4t3k2i10	2954.05	2152.52	1879.35	1370.00	37.18	48.80	104.21
CIRL5t3k2i10	3294.28	2698.76	2447.76	1514.00	61.68	86.56	127.73
CIRL1t3k2i15	3070.02	2648.58	2478.41	1657.81	49.50	83.09	112.23
CIRL2t3k2i15	3619.97	2871.68	2397.52	1377.58	74.04	98.52	150.25
CIRL3t3k2i15	3975.63	3257.61	3006.34	2032.78	47.89	125.20	174.83
CIRL4t3k2i15	3165.43	2669.67	2461.62	1785.32	37.88	84.55	118.82
CIRL5t3k2i15	3586.50	3014.68	2899.26	1779.55	62.92	108.40	147.93
CIRL1t3k2i20	3713.94	3213.27	2941.49	1851.65	58.86	122.13	156.74
CIRL2t3k2i20	3900.31	3138.01	2670.73	1297.12	105.90	116.93	169.62
CIRL3t3k2i20	4336.45	3397.80	3121.24	1837.00	69.91	134.89	199.77
CIRL4t3k2i20	4560.90	3339.22	2982.76	1922.54	55.15	130.84	215.29
CIRL5t3k2i20	5469.69	4112.47	3798.46	2159.95	75.86	184.29	278.11
CIRL1t3k2i25	4815.88	3763.66	3523.81	2138.80	64.76	160.18	232.92
CIRL2t3k2i25	4837.71	3851.87	3524.14	1927.07	82.88	166.28	234.43
CIRL3t3k2i25	5561.11	4121.45	3621.14	1997.87	81.25	184.91	284.43
CIRL4t3k2i25	4609.21	3864.02	3562.92	2077.88	71.47	167.12	218.63
CIRL5t3k2i25	5681.89	4867.91	4629.35	2763.49	67.52	236.51	292.78
CIRL1t3k2i30	5974.73	4429.88	4245.54	2155.00	97.01	206.23	313.03
CIRL2t3k2i30	5541.10	4658.44	4241.53	2688.00	57.80	222.03	283.05
CIRL3t3k2i30	5691.32	4706.81	4254.84	2120.93	100.61	225.38	293.44
CIRL4t3k2i30	5050.86	4107.30	3785.07	2144.00	76.54	183.93	249.16
CIRL5t3k2i30	4786.07	3950.80	3743.46	1824.06	105.23	173.12	230.86
CIRL1t3k2i35	5378.06	4315.90	4101.08	2004.00	104.64	198.35	271.78
CIRL2t3k2i35	5745.17	4722.29	4494.71	2474.00	81.68	226.45	297.16
CIRL3t3k2i35	6991.93	5868.83	5569.26	2804.00	98.62	305.71	383.35
CIRL4t3k2i35	5325.99	4248.79	3875.21	2135.85	81.44	193.71	268.18
CIRL5t3k2i35	5770.44	4638.66	3893.78	2022.91	92.48	220.67	298.90
Average					61.63	139.53	195.43

The second column lists the total cost of the extended lower bound while the third column displays the computation time (in seconds) to achieve it. The fourth and the fifth columns provide the number of SECs added in the branching and at the root node, respectively. Finally, sixth column shows the number of nodes explored in the branch-and-cut algorithm.

As Table 4.F.1 and 4.F.2 show, the computation time substantially increases as the number of suppliers increases for both the one-vehicle and two-vehicle instances, but more dramatically for the two-vehicle instances. For one-vehicle instances, the computation time goes from less than 1 second for 5-supplier instances to the two-hour time limit for as few as 35 suppliers. For the two-vehicle instances, the two-hour time limit is reached consistently in the 20-supplier instances.

For the one-vehicle instances, the first phase dynamically adds a relatively large number of SECs, and the number dynamically added increases as the number of suppliers increases. Comparing the two-vehicle instances to the one-vehicle instances, the number of dynamically added SECs increases less sharply with the number of suppliers, but the number of SECs added to the root node increases more sharply with the number of suppliers.

Table 4.F.1: Extended Lower Bound Performance on One-Vehicle, Three-Period Instances

Instance	Total Cost	Time (sec)	Capacitated Subtour Elimination Constraints		Nodes
			Branching	Root Node	
CIRL1t3k1i05	1555.90	0.11	0	12	9
CIRL2t3k1i05	1493.92	0.54	30	10	26
CIRL3t3k1i05	2684.58	0.29	17	27	24
CIRL4t3k1i05	1567.86	0.32	11	27	27
CIRL5t3k1i05	1574.44	0.38	17	30	16
CIRL1t3k1i10	2933.56	1.12	95	41	29
CIRL2t3k1i10	3322.78	5.00	135	72	181
CIRL3t3k1i10	2888.04	3.02	108	73	108
CIRL4t3k1i10	2771.13	1.27	111	27	31
CIRL5t3k1i10	3161.71	4.75	179	37	165
CIRL1t3k1i15	3185.96	3.92	180	132	59
CIRL2t3k1i15	3413.42	4.41	275	139	63
CIRL3t3k1i15	4086.81	8.95	243	123	134
CIRL4t3k1i15	3142.98	26.67	206	145	362
CIRL5t3k1i15	3528.57	15.96	516	76	169
CIRL1t3k1i20	4051.09	75.57	892	94	433
CIRL2t3k1i20	3911.01	10.91	276	151	85
CIRL3t3k1i20	4282.28	34.05	596	71	186
CIRL4t3k1i20	4243.97	30.84	522	227	158
CIRL5t3k1i20	4795.15	11.83	283	235	74
CIRL1t3k1i25	4677.38	658.80	2512	226	1902
CIRL2t3k1i25	4943.56	86.68	1117	369	301
CIRL3t3k1i25	5301.89	186.82	545	481	605
CIRL4t3k1i25	4568.79	175.63	1261	128	577
CIRL5t3k1i25	5436.36	132.71	795	184	376
CIRL1t3k1i30	5793.88	134.62	667	660	293
CIRL2t3k1i30	5455.03	262.12	1643	123	463
CIRL3t3k1i30	5874.99	66.97	827	142	145
CIRL4t3k1i30	5030.12	351.15	1617	114	686
CIRL5t3k1i30	4843.31	69.92	929	190	150
CIRL1t3k1i35	5431.29	176.49	874	156	267
CIRL2t3k1i35	5874.49	694.38	4106	215	910
CIRL3t3k1i35	6855.12	7161.74	10417	589	4290
CIRL4t3k1i35	5265.81	261.44	1834	321	298
CIRL5t3k1i35	5639.50	346.43	2115	489	497
CIRL1t3k1i40	6485.55	715.32	2114	516	626
CIRL2t3k1i40	6177.87	7182.15	10771	338	2470
CIRL3t3k1i40	6427.60	115.70	750	106	136
CIRL4t3k1i40	5653.95	434.73	1747	271	352
CIRL5t3k1i40	6109.86	117.09	843	247	127
CIRL1t3k1i45	6657.52	3940.33	10601	406	2346
CIRL2t3k1i45	6480.05	1805.91	5642	277	769
CIRL3t3k1i45	6654.31	2784.64	8169	125	1514
CIRL4t3k1i45	6758.66	2823.22	7058	862	1264
CIRL5t3k1i45	6148.15	1607.54	3682	242	1128
CIRL1t3k1i50	7103.64	7092.04	9748	409	1654
CIRL2t3k1i50	7544.85	7106.26	8467	578	1776
CIRL3t3k1i50	6917.94	7069.56	12115	172	1646
CIRL4t3k1i50	7317.37	7009.15	10497	587	1691
CIRL5t3k1i50	7280.35	864.06	2403	738	388
Average		1233.47	2611	240	640

Table 4.F.2: Extended Lower Bound Performance on Two-Vehicle, Three-Period Instances

Instance	Total Cost	Time (sec)	Capacitated Subtour Elimination Constraints		Nodes
			Branching	Root Node	
CIRL1t3k2i05	1801.05	0.49	6	23	7
CIRL2t3k2i05	2665.44	0.42	18	29	31
CIRL3t3k2i05	4007.30	0.88	36	29	37
CIRL4t3k2i05	2438.84	0.93	16	13	27
CIRL5t3k2i05	2200.63	3.13	57	18	156
CIRL1t3k2i10	3972.29	111.24	404	166	1384
CIRL2t3k2i10	4442.25	1665.26	1577	75	12457
CIRL3t3k2i10	3610.82	276.88	735	88	2879
CIRL4t3k2i10	3781.87	369.24	1419	116	4839
CIRL5t3k2i10	3580.32	144.49	451	77	1620
CIRL1t3k2i15	3807.30	974.13	1773	221	4158
CIRL2t3k2i15	4077.82	1259.30	2674	199	4912
CIRL3t3k2i15	4823.47	786.04	1817	133	2755
CIRL4t3k2i15	3865.76	1861.93	2257	227	6947
CIRL5t3k2i15	4349.72	6758.02	4386	148	20788
CIRL1t3k2i20	5148.09	7163.36	4047	309	10451
CIRL2t3k2i20	4214.90	1744.49	1474	317	4217
CIRL3t3k2i20	5039.24	6574.19	2991	199	9676
CIRL4t3k2i20	5545.53	7084.16	6863	274	6902
CIRL5t3k2i20	5868.17	7024.86	5359	446	8027
CIRL1t3k2i25	5311.47	7148.45	3165	436	6674
CIRL2t3k2i25	5876.82	7169.49	5190	871	5745
CIRL3t3k2i25	6180.67	6943.03	4287	833	4924
CIRL4t3k2i25	5093.02	7135.04	3707	329	6510
CIRL5t3k2i25	6663.25	7168.34	9561	576	4405
CIRL1t3k2i30	6636.56	6980.26	10081	658	2073
CIRL2t3k2i30	6097.81	7193.62	4934	308	3774
CIRL3t3k2i30	6392.50	7155.34	5000	287	3306
CIRL4t3k2i30	5680.75	7102.63	4434	236	4323
CIRL5t3k2i30	5498.71	7139.37	6786	423	2846
CIRL1t3k2i35	6291.93	7070.59	7454	651	2211
CIRL2t3k2i35	6668.95	7134.92	5104	430	2963
CIRL3t3k2i35	8316.27	7039.60	7245	898	2154
CIRL4t3k2i35	6368.49	7138.09	6450	721	2218
CIRL5t3k2i35	6126.24	7069.42	5774	1009	2419
Average		4296.90	3644	336	4538

4.G Additional Computational Experiments on the Heuristic Algorithm

In this section, we examine the behavior of the heuristic procedure that derives a solution to (**P**) from the extended lower bound by solving the vehicle routing problem for each day in the planning horizon as a set of TSPs. We compare the performance of the heuristic when the time limit to solve the extended lower bound formulation is set to 900 seconds and to 7,200 seconds, respectively. Tables 4.G.1 and 4.G.2 show the performance of the heuristic algorithm in the one-vehicle and two-vehicle instances, respectively. The second column lists the total cost of the heuristic solution, the third column shows the corresponding optimality gap, and the fourth column contains the additional time to solve the set of daily TSPs derived from the extended lower bound obtained within 900 seconds. The fifth through seventh columns provide the analogous results starting from the extended lower bound obtained within 7200 seconds.

Tables 4.G.1 and 4.G.2 show that the heuristic is very effective when based on the extended lower bound obtained within 7200 seconds, achieving an average optimality gap of just 0.07 percent and 4

percent in the one-vehicle and two-vehicle instances, respectively. Moreover, Tables 4.G.1 and 4.G.2 demonstrate that the heuristic is still effective when operating from the extended lower bound obtained within just 900 seconds, achieving an average optimality gap of 0.55 percent and 7.02 percent for the one-vehicle and two-vehicle instances, respectively. We also observe that the computation time required to solve the set of daily TSPs is negligible as the one-vehicle instances require the solution of a total of three TSPs, and the two-vehicle instances require the solution of a total of six TSPs.

Table 4.G.1: Heuristic Performance on One-Vehicle, Three-Period Instances

Instance	Time Limit = 900 sec			Time Limit = 7200 sec		
	Total Cost	% Gap	Time (sec)	Total Cost	% Gap	Time (sec)
CIRL1t3k1i05	1555.90	0.00	0.00	1555.90	0.00	0.00
CIRL2t3k1i05	1493.92	0.00	0.00	1493.92	0.00	0.00
CIRL3t3k1i05	2684.58	0.00	0.00	2684.58	0.00	0.00
CIRL4t3k1i05	1567.86	0.00	0.00	1567.86	0.00	0.00
CIRL5t3k1i05	1574.44	0.00	0.00	1574.44	0.00	0.00
CIRL1t3k1i10	2933.56	0.00	0.00	2933.56	0.00	0.00
CIRL2t3k1i10	3322.78	0.00	0.00	3322.78	0.00	0.00
CIRL3t3k1i10	2888.04	0.00	0.00	2888.04	0.00	0.00
CIRL4t3k1i10	2771.13	0.00	0.00	2771.13	0.00	0.00
CIRL5t3k1i10	3161.71	0.00	0.00	3161.71	0.00	0.00
CIRL1t3k1i15	3185.96	0.00	0.00	3185.96	0.00	0.00
CIRL2t3k1i15	3413.42	0.00	0.00	3413.42	0.00	0.00
CIRL3t3k1i15	4086.81	0.00	0.00	4086.81	0.00	0.00
CIRL4t3k1i15	3142.98	0.00	0.00	3142.98	0.00	0.00
CIRL5t3k1i15	3528.57	0.00	0.00	3528.57	0.00	0.00
CIRL1t3k1i20	4051.09	0.00	0.00	4051.09	0.00	0.00
CIRL2t3k1i20	3911.01	0.00	0.00	3911.01	0.00	0.00
CIRL3t3k1i20	4282.28	0.00	0.00	4282.28	0.00	0.00
CIRL4t3k1i20	4243.97	0.00	0.00	4243.97	0.00	0.00
CIRL5t3k1i20	4795.15	0.00	0.00	4795.15	0.00	0.00
CIRL1t3k1i25	4677.38	0.00	0.00	4677.38	0.00	0.00
CIRL2t3k1i25	4943.56	0.00	0.00	4943.56	0.00	0.00
CIRL3t3k1i25	5301.89	0.00	0.00	5301.89	0.00	0.00
CIRL4t3k1i25	4568.79	0.00	0.00	4568.79	0.00	0.00
CIRL5t3k1i25	5436.36	0.00	0.00	5436.36	0.00	0.00
CIRL1t3k1i30	5793.88	0.00	0.00	5793.88	0.00	0.00
CIRL2t3k1i30	5455.03	0.00	0.00	5455.03	0.00	0.00
CIRL3t3k1i30	5874.99	0.00	0.00	5874.99	0.00	0.00
CIRL4t3k1i30	5030.12	0.00	0.00	5030.12	0.00	0.00
CIRL5t3k1i30	4843.31	0.00	0.00	4843.31	0.00	0.00
CIRL1t3k1i35	5431.29	0.00	0.00	5431.29	0.00	0.00
CIRL2t3k1i35	5874.49	0.00	0.00	5874.49	0.00	0.00
CIRL3t3k1i35	6914.46	0.87	0.05	6855.12	0.00	0.04
CIRL4t3k1i35	5265.81	0.00	0.00	5265.81	0.00	0.00
CIRL5t3k1i35	5661.50	0.29	0.02	5661.50	0.29	0.06
CIRL1t3k1i40	6485.55	0.00	0.00	6485.55	0.00	0.00
CIRL2t3k1i40	6251.86	1.64	0.03	6177.87	0.44	0.04
CIRL3t3k1i40	6427.60	0.00	0.00	6427.60	0.00	0.00
CIRL4t3k1i40	5653.95	0.00	0.00	5653.95	0.00	0.00
CIRL5t3k1i40	6109.86	0.00	0.00	6109.86	0.00	0.00
CIRL1t3k1i45	6663.36	0.09	0.02	6657.52	0.00	0.00
CIRL2t3k1i45	6689.82	2.97	0.01	6496.55	0.00	0.02
CIRL3t3k1i45	6766.37	1.68	0.00	6654.31	0.00	0.09
CIRL4t3k1i45	6826.38	1.00	0.06	6758.66	0.00	0.00
CIRL5t3k1i45	6343.99	3.13	0.06	6181.65	0.49	0.08
CIRL1t3k1i50	7320.87	3.66	0.08	7103.64	0.59	0.04
CIRL2t3k1i50	8158.13	9.91	0.04	7544.85	1.65	0.05
CIRL3t3k1i50	6986.28	1.13	0.02	6917.94	0.14	0.07
CIRL4t3k1i50	7375.87	0.87	0.03	7317.37	0.07	0.06
CIRL5t3k1i50	7284.85	0.00	0.06	7284.85	0.00	0.07
Average		0.54	0.01		0.07	0.01

Table 4.G.2: Heuristic Performance on Two-Vehicle Instances

Instance	Time Limit = 900 sec			Time Limit = 7200 sec		
	Total Cost	% Gap	Time (sec)	Total Cost	% Gap	Time (sec)
CIRL1t3k2n05	1801.05	0.00	0.00	1801.05	0.00	0.00
CIRL2t3k2n05	2665.44	0.00	0.00	2665.44	0.00	0.00
CIRL3t3k2n05	4007.30	0.00	0.00	4007.30	0.00	0.00
CIRL4t3k2n05	2438.84	0.00	0.00	2438.84	0.00	0.00
CIRL5t3k2n05	2200.63	0.00	0.00	2200.63	0.00	0.00
CIRL1t3k2n10	3972.29	0.00	0.00	3972.29	0.00	0.00
CIRL2t3k2n10	4521.00	1.77	0.00	4442.25	0.00	0.00
CIRL3t3k2n10	3610.82	0.00	0.00	3610.82	0.00	0.00
CIRL4t3k2n10	3781.87	0.00	0.00	3781.87	0.00	0.00
CIRL5t3k2n10	3580.32	0.00	0.00	3580.32	0.00	0.00
CIRL1t3k2n15	3807.30	0.00	0.02	3807.30	0.00	0.00
CIRL2t3k2n15	4077.82	0.00	0.01	4077.82	0.00	0.00
CIRL3t3k2n15	4823.47	0.00	0.00	4823.47	0.00	0.00
CIRL4t3k2n15	3928.06	1.61	0.01	3865.76	0.00	0.00
CIRL5t3k2n15	4519.59	3.91	0.02	4349.72	0.00	0.00
CIRL1t3k2n20	5340.38	8.05	0.01	5148.09	4.16	0.04
CIRL2t3k2n20	4301.68	2.06	0.01	4214.90	0.00	0.00
CIRL3t3k2n20	5132.00	5.57	0.01	5039.24	3.66	0.06
CIRL4t3k2n20	5823.19	14.26	0.01	5545.53	8.81	0.03
CIRL5t3k2n20	6015.72	5.55	0.03	5868.17	2.96	0.04
CIRL1t3k2n25	5419.21	5.93	0.02	5311.47	3.82	0.06
CIRL2t3k2n25	5991.49	10.91	0.03	5876.82	8.79	0.02
CIRL3t3k2n25	6508.96	9.25	0.01	6180.67	3.74	0.05
CIRL4t3k2n25	5315.20	8.02	0.01	5093.02	3.50	0.02
CIRL5t3k2n25	6750.32	11.48	0.02	6663.25	10.04	0.07
CIRL1t3k2n30	7231.42	20.40	0.02	6636.56	10.49	0.04
CIRL2t3k2n30	6569.09	10.85	0.03	6097.81	2.90	0.02
CIRL3t3k2n30	6502.61	6.66	0.04	6392.50	4.85	0.05
CIRL4t3k2n30	5909.22	10.87	0.01	5680.75	6.59	0.03
CIRL5t3k2n30	5838.03	15.67	0.03	5498.71	8.95	0.04
CIRL1t3k2n35	6942.11	20.68	0.02	6291.93	9.38	0.03
CIRL2t3k2n35	7205.49	17.01	0.03	6668.95	8.30	0.06
CIRL3t3k2n35	8316.27	16.88	0.04	8316.271	16.88	0.07
CIRL4t3k2n35	6912.35	25.55	0.01	6368.49	15.67	0.01
CIRL5t3k2n35	6492.51	12.95	0.02	6126.24	6.58	0.05
Average		7.03	0.01		4.00	0.02

Chapter 5

Sparse Routing for the Inventory Routing Problem*

Abstract

We introduce the notion of sparse routing to solve the finite-horizon Inventory Routing Problem with split deliveries. We define sparse routing as a broad class of matheuristics which solves a route-based formulation of the problem. To address the complexity of a formulation with all possible routes, we show how to design effective and efficient sparse routing, hence achieving the best tight worst-case performance bound known for this problem, i.e. $\lceil\sqrt{H}\rceil$ where H is the planning horizon. The sets of routes we obtained, together with sets of routes generated by other methods we propose, allow us to have solutions with very good average performance in a large set of instances. These solutions significantly dominate, both in terms of cost and computational time, the best solutions obtained by applying a branch-and-cut algorithm we design to a flow–based formulation of the problem.
***Keywords**: inventory routing, split delivery, route-based formulation, matheuristics, worst-case analysis, branch-and-cut.*

5.1 Introduction

Logistics is a trillion-dollar industry. In the United States alone, total business logistics costs in 2018 amounted to $1.635 trillion, up 11.4% from the previous year. The biggest portion comes from transportation costs of just over $1 trillion. These figures suggest the importance of solving routing optimization problems as even a mere 0.1% savings is equivalent to $1 billion (see Ward et al. (2019)).

One of the classic routing problems is the Vehicle Routing Problem (VRP), where a depot must deliver goods in given quantities to a given set of customers using a fleet of vehicles, each with a certain limited capacity. The objective is to determine the allocation of customers among routes and the sequence by which customers in each route are visited, in order to minimize the total transportation costs. The VRP, which adds a capacity dimension to the Traveling Salesman Problem (TSP), is well known to be computationally challenging (i.e. NP-hard). In order to generate savings, one can allow split deliveries in the VRP such that the quantities required by customers need not be delivered by the same vehicle or in the same route. This adds computational complexity to the VRP and is known as the VRP with Split Deliveries (SVRP) (see Dror et al. (1994)).

*This chapter is based on Bertazzi, Chua, Laganá & Paradiso (2019).

In integrated logistics and in supply chain management, a means to generate savings comes from the second largest driver of logistics costs, which is inventory carrying costs. In 2018, the total inventory carrying costs in the United States was almost half a trillion dollars, up almost 15% from the previous year (see Ward et al. (2019)). In this setting, solving a VRP (or a SVRP) problem just provides suboptimal solutions, as quantities required by customers can be delivered ahead of time, say one or even many periods earlier, potentially trading off high transportation costs for lower inventory holding costs. This problem is known as the Inventory Routing Problem (IRP), which has seen several industrial applications, ranging from chemical components to oil and gas, automobile components, groceries, cement, blood, livestock, and organic waste (see Coelho et al. (2013*a*)).

As the IRP adds a time dimension to the VRP, there is clearly greater computational complexity in the former. It is no wonder that the IRP has attracted and continues to attract many researchers, working on exact and heuristic solutions as well as variants of the IRP. To address this computational challenge, we introduce the notion of *sparse routing* for inventory routing problems. To properly define sparse routing, we must model the IRP as a route-based mixed-integer program where decision variables are declared for each route and each period. This route-based formulation (as opposed to traditional flow-based formulation) allows us to see that if all possible routes are considered, then we will have $O(2^n \cdot H)$ decision variables related to the routing part of the problem and the same number of TSP problems to solve to optimality, where n is the number of customers and H is the number of periods in the planning horizon. This makes the problem computational intractable.

We can now define sparse routing as a class of matheuristics for general IRP problems which only considers a *subset of all possible routes*, hence arriving at good solutions in reasonable computational times. A matheuristic is a heuristic method or algorithm that involves the solution of mathematical programming models, typically corresponding to subproblems of the original problem (see Archetti & Speranza (2014)). It should not be mistaken for a metaheuristic. In this paper, we demonstrate how sparse routing can be applied to the finite-horizon IRP with split deliveries (SDIRP). The main contributions in the literature for the SDIRP are based on different mathematical formulations and methods, which make it unclear which approach is most suitable, especially when split deliveries are allowed.

With sparse routing, we are able to solve the SDIRP both effectively and efficiently. However, because sparse routing considers only a subset of routes, there are numerous methods to carry out sparse routing. The objective of this paper is to find methods that generate a set of routes that is sparse and, at the same time, guarantees good solution quality. On one hand, the set of all routes has the best solution quality but is not sparse at all (thus cannot be solved to optimality in reasonable time). On the other hand, direct shipping (i.e. routes with only one customer) is very sparse but can be shown (in this paper) to have arbitrarily poor solution quality. In general, one can consider the set of all routes with at most r customers, where r is relatively low compared to n. We refer to this as *short routing*, which must be distinguished from the more general notion of sparse routing. The key difference lies in route length cardinality versus route set cardinality. It can be shown that the solution quality from short routing can be arbitrarily bad, such that one can construct instances whereby such sets can be as bad as direct shipping. The main reason for such poor performance is because we do not use instance information in generating the set of routes.

Using this insight, we propose a method that solves a (relatively) easier problem, such as the VRP or the SVRP, to generate the subset of routes for our matheuristics. Instance information is used to define the demand of each customer in the VRP or the SVRP we solve, e.g. demand equal to the quantity needed to

serve the customers daily, or to serve them just once over the planning horizon. Then, given this subset of routes, we solve the route-based model we formulate for this problem.

Theoretically, we provide a new lower bound for SDIRP based on the optimal SVRP cost. Moreover, we prove an upper bound on the performance (i.e. cost) of the SDIRP when the subset of routes is generated by solving the SVRP with demand of each customer equal to the quantity delivered in the SDIRP when an intershipment time of τ periods is used. These upper and lower bounds allow us to prove a worst-case bound on such subclass of matheuristics, which corresponds to a type of sparse routing. Then, we solve the SVRP with demand of each customer equal to the quantity delivered in the SDIRP when the intershipment time that minimizes the worst-case bound is used. By using this subset of routes in the SDIRP, we find that the best worst-case bound is $\lceil\sqrt{H}\rceil$. Moreover, we prove that this bound is tight. To our best knowledge, this is the best tight worst-case bound known for the SDIRP. We also provide worst-case bounds for instances with non-identical inventory holding costs, capacitated customers, and time-varying demand.

Numerically, the class of matheuristics we consider allows us to test various methods of sparse routing, preferably using instance information. We tinkered with SVRP, VRP, VRP with optimized capacity and combinations of the above. We report excellent average performance over a broad set of test instances. Specifically, our solutions significantly dominate, in terms of both cost and computational time, the best solutions obtained by applying a branch-and-cut algorithm we design for the traditional flow-based formulation of the SDIRP. The methods we proposed also demonstrate various trade-off points between computational time and solution quality.

Our main methodological contribution is to show that *worst-case analysis* can be useful to design efficient and effective matheuristics, with worst-case performance guarantee, whenever a route–based formulation for the IRP is used. Efficiency comes from the fact that the subsets of routes generated by the matheuristic is very sparse. Effectiveness is obtained by just adding, to the subset of routes used to prove the worst-case performance bound, a very small number of additional routes designed to be high performing on average. Worst-case performance guarantee comes from the fact that enlarging the subset of routes used to prove the worst-case performance bound can never worsen the worst-case performance bound. These matheuristics are also practical, as the subset of routes used to prove the worst-case performance bound can be added to the set of routes already used by the company, resulting in both better solutions on average and a guarantee in the worst case. We believe this approach is general, i.e. it can be applied to any problem for which a column–based model can be formulated. In this paper, we demonstrate that it performs well on a very challenging problem.

The remainder of the paper is organized as follows. Section 5.2 provides a brief discussion of the related literature. Section 5.3 presents the problem description and the route-based model formulation. We then discuss the analysis of the worst-case performance and the average performance of the matheuristics we propose in Section 5.4 and Section 5.5, respectively. Finally, Section 5.6 concludes and provides managerial insights.

5.2 Literature Review

The literature contains a wide range of work on the IRP formulations and algorithms (see tutorials by Bertazzi & Speranza (2012*a*) and Bertazzi & Speranza (2013), and survey by Coelho et al. (2013*a*)). A classical distinction is between uncertain (stochastic and robust) and deterministic IRPs. A few papers

have been published on uncertain IRPs (see for example Adelman (2004), Bertazzi et al. (2013), Bertazzi et al. (2016), Coelho et al. (2014), Kleywegt et al. (2002), Kleywegt et al. (2004), Solyalı et al. (2012)). Meanwhile, the primary stream in this literature has been on deterministic IRPs, where the models considered can be broadly classified according to the number of products and number of vehicles involved. In this section, we discuss the ones which are most relevant to our work. The single-product, single-vehicle IRP was introduced in Archetti et al. (2007). Exact algorithms were designed by Solyalı & Süral (2008), Coelho & Laporte (2014), Avella et al. (2015), and Avella et al. (2018), while effective matheuristics were also designed and successfully applied to a large variety of routing problems in more recent years (see for example, Archetti et al. (2012)). These algorithms were then extended to solve integrated logistics problems in which routing plays an important role. We refer the readers to to Archetti & Speranza (2014) for a survey on matheuristics for routing problems, and to Bertazzi & Speranza (2012*b*) for an introduction to matheuristics for solving IRPs. Various other works also examine the IRP where the final inventory levels must be equal to the initial inventory levels (see Aksen et al. (2012), Ekici et al. (2015), Raa & Aghezzaf (2008), Aghezzaf et al. (2012), Vansteenwegen & Mateo (2014), and Raa (2015)). More recently, several papers investigate the single-product, multi-vehicle IRP including Coelho et al. (2012) and Adulyasak et al. (2013) and effective matheuristics were designed by Archetti et al. (2017). The multi-product, multi-vehicle IRP was studied in Coelho & Laporte (2013*a*) and Cordeau et al. (2015). Exact algorithms include branch–and–cut, branch–and–price, and more recently, branch–and–price–and–cut (Desaulniers et al. (2016)). The above body of work considers the operational problem where initial inventory levels are fixed, while our work solves a tactical planning problem where the initial inventory levels are decision variables.

To address computational complexity, one stream of work looks into decomposition approaches for large scale instances of the IRP such as Campbell & Salvesbergh (2004) and in Cordeau et al. (2015). Another stream develops computationally efficient algorithms or heuristics that achieve nearly optimal solutions or provide performance guarantees. For instance, Anily & Federgruen (1993) and Anily & Federgruen (1990) develop heuristics for solving the IRP with and without inventories at the depot, respectively, that are asymptotically within a few percentage points of optimality when the number of customers grows to infinity. They also provide methods for computing lower bounds (i.e. performance guarantees) in an efficient manner. Meanwhile, Gallego & Simchi-Levi (1990) show that direct shipping can perform well for the IRP, provided the demand from every customer is sufficiently high. On the flip side, this also implies that the direct shipping can be arbitrarily bad for general customer demand profiles. Nevertheless, the above stream of work on performance analysis considers only infinite horizon problems with continuous review, and does not provide any constant or finite worse-case performance bound. Bertazzi (2008) studies the discrete case with direct shipping, providing constant worst-case performance bounds in several classes of instances, but not a constant or finite worst-case performance bound on all instances. In contrast, our work examines finite horizon problems with periodic review, and to our best knowledge, provides the first known finite worst-case bound for this problem.

The IRP is also related to other well known problems such as the VRP and the Joint Replenishment Problem (JRP). We discuss how these two problems are less challenging than the IRP, hence why some of the recent results for these problems are not comparable to ours. For instance, Bramel & Simchi-Levi (1995) and Bramel et al. (1992) provide polynomial-time heuristics for the VRP which are asymptotically optimal under a broad range of demand scenarios. While some of their formulations and heuristics can be

extended to the IRP, no performance guarantees can be obtained. Meanwhile, Nagarajan & Shi (2016) present a unified approach for the submodular JRP and the uncapacitated IRP without split deliveries. While the approach yields very efficient algorithms for both problems, it does not apply to the more general IRP with vehicle capacity and split deliveries (SDIRP). Similarly, Federgruen & Zheng (1992) derive efficient algorithms for the JRP assuming submodular costs. However, the vehicle capacity constraint in the SDIRP destroys both properties of submodularity and approximate subadditivity required for the results in the above papers. Hence, it is clear that the SDIRP is substantially more challenging than both VRP and JRP, and the performance bounds are not comparable (e.g. some JRP bounds can be less than 2).

5.3 Model

In this section, we first describe the finite-horizon IRP with split deliveries (SDIRP) as well as the notations necessary for its model formulation and analysis. We then model the problem as a route-based mixed-integer linear program, which constitutes *sparse routing* or the class of matheuristics we consider in this paper.

5.3.1 Problem Description

We study the SDIRP in which a product has to be shipped from a supplier 0 to a set I of n customers over a finite planning horizon of H periods. The logistic network is represented by the undirected graph $G(V, E)$, where $V = I \cup \{0\}$ is the set of nodes and E is the set of edges. The transportation cost matrix $\{c_{ij}\}$ is symmetric, i.e. $c_{ij} = c_{ji}$ for all $i, j \in V$, and satisfies the triangle inequality, i.e. $c_{ij} \leq c_{il} + c_{lj}$, for all $i, l, j \in V$ such that $i \neq l \neq j$. Shipments from the supplier to the customers can be performed on each day t in the set $T = \{0, 1, \ldots, H-1\}$ by vehicles having each transportation capacity Q. Let K be the set of all possible routes, c_k be the cost of the route $k \in K$ (i.e. the cost of the corresponding optimal TSP), R_k be the set of customers served by route $k \in K$ and K_i be the set of routes serving customer i. Split delivery is allowed, i.e. a customer can be served by more than one vehicle on each day, even if the quantity sent to the customer on this day is not greater than the transportation capacity. The product is made available at the supplier at a given constant production rate q and absorbed by each customer $i \in I$ at a given constant demand rate q_i, where $\sum_{i \in I} q_i = q$. A unit inventory cost h is charged at the supplier and at the customers for positive inventory levels at the end of each day $t \in T$. Our aim is to determine a shipping policy that minimizes the sum of the routing cost and of the inventory cost, both at the supplier and at the customers, in the time unit. This policy is defined by the following decision variables: the quantity s_{itk} to deliver to customer $i \in I$ on day $t \in T$ by using route $k \in K$; y_{tk} equal to 1 if route $k \in K$ is used on day $t \in T$ and 0 otherwise; the starting inventory level $\bar{I}_0$ at the supplier 0 and the starting inventory level $\bar{I}_i$ at each customer $i \in I$. Let us introduce the variables I_{0t} and I_{it} denoting the inventory level at the end of day $t \in T$ at the supplier 0 and at each customer $i \in I$, respectively. Note that, since our problem is not defined at an operational level but at a tactical planning level, the starting inventory levels at the supplier and at the customers are decision variables of the problem, and not given data. This allows us to compute an optimal starting inventory level to make available at the supplier and at the customers every H periods. Moreover, note that no maximum inventory level is defined at the customers. The combination of these two features makes our problem more difficult than the one studied in Archetti et al. (2007) and the subsequent papers.

5.3.2 Route-Based Formulation for Sparse Routing: A Matheuristic Class

The problem described in the previous section can be formulated as the following route–based mixed integer linear programming model.

Problem $\mathcal{R}$

$$\text{Min} \quad h\left(\bar{I}_0 + \sum_{i\in I} \bar{I}_i\right) + \frac{1}{H}\sum_{t\in T}\sum_{k\in K} c_k y_{tk} \tag{5.1}$$

$$\text{s.t.} \quad \sum_{t\in T}\sum_{k\in K_i} s_{itk} = q_i H \qquad i \in I \tag{5.2}$$

$$\sum_{i\in R_k} s_{itk} \le Q y_{tk} \qquad t \in T \tag{5.3}$$

$$I_{0t} = \bar{I}_0 + \sum_{i\in I} q_i t - \sum_{\rho=0}^{t}\sum_{k\in K}\sum_{i\in R_k} s_{i\rho k} \qquad t \in T \tag{5.4}$$

$$I_{it} = \bar{I}_i + \sum_{\rho=0}^{t}\sum_{k\in K_i} s_{i\rho k} - q_i(t+1) \qquad i \in I \quad t \in T \tag{5.5}$$

$$\bar{I}_0 \ge 0 \tag{5.6}$$

$$I_{0t} \ge 0 \qquad t \in T \tag{5.7}$$

$$\bar{I}_i \ge 0 \qquad i \in I \tag{5.8}$$

$$I_{it} \ge 0 \qquad i \in I \quad t \in T \tag{5.9}$$

$$s_{itk} \ge 0 \qquad t \in T \quad k \in K \quad i \in R_k \tag{5.10}$$

$$y_{tk} \ge 0 \quad \text{integer} \qquad t \in T \quad k \in K. \tag{5.11}$$

The objective function (5.1) minimizes the sum of the inventory cost at the supplier and at the customers, and of the routing cost, per day. Note that the total inventory cost per day can be defined in this way, as the total inventory level on each day t, i.e. $I_{0t} + \sum_{i\in I} I_{it}$, is constant over time thanks to (5.4) and (5.5). Constraints (5.2) guarantee that the total quantity sent to each customer i over the time horizon H is equal to the total demand $q_i H$. Constraints (5.3) guarantee that quantity delivered by route k on day t is not greater than the transportation capacity if route k is used on day t. Constraints (5.4) compute the inventory level at the supplier at the end of each day t, given by the starting inventory level, plus the cumulative quantity made available at the supplier up to day $t-1$, minus the cumulative quantity sent to the customers up to day t. Constraints (5.5) compute the inventory level at each customer i at the end of each day t, given by the starting inventory level, plus the cumulative quantity delivered to the customer up to day t, minus the cumulative demand at the customer up to day t. Finally, (5.6)–(5.11) define the decision variables. Note that (5.11) implies that each route k can be used several times on each day.

Observe that apart from the problem instance parameters, a critical input to this model is the subset K of routes, based upon which decision variables are defined for the model. The choice of K corresponds to the type of sparse routing selected to solve the SDIRP. Because there are be numerous ways to choose K, the above formulation represents the entire class of sparse routing heuristics (e.g. it includes short routing). Given the model inputs, the above problem can be solved using standard Branch-and-Cut methods. In the next sections, we discuss methods to generate the subset K of routes, and analyze their performance.

5.4 Analysis of Worst-Case Performance

In this section, we first characterize the worst-case performance of direct shipping and show that it can be arbitrarily bad. This brings us to a subclass of matheuristics using an SVRP-based method to generate route sets, which turns out to be efficient and effective sparse routing. We show that the optimal routing cost of SVRP with total planning horizon demand, divided by the planning horizon duration, provides a lower bound on the optimal routing cost in the SDIRP. We then use this result to prove the worst-case performance of the SVRP-based subclass. Finally, we extend the results to non-identical inventory holding costs, capacitated customers and time-varying demand.

5.4.1 Direct Shipping

Similar to Gallego & Simchi-Levi (1990) and Bertazzi (2008) but for finite-horizon IRP with periodic review, we show that direct shipping can be arbitrarily bad. To this end, we let D be the set of direct shipping routes, z^D be optimal cost of Problem $\mathcal{R}$ with $K = D$ and $\underline{q} = \min_{i \in I} q_i$. We present the following lemma that provides a lower bound on z^*.

Lemma 5.1. $z^* \geq \sum_{i \in I} hq_i + \sum_{i \in I} 2c_{0i} \frac{q_i}{Q}$

Proof. This lower bound is given by the sum of a lower bound on the inventory cost and a lower bound on the routing cost. Consider first the inventory cost. Since a quantity q_i is absorbed on day 0 by each customer $i \in I$ and the quantity $\sum_i q_i$ produced at the supplier on day 0 can be shipped to customers only on day 1, then $\bar{I}_0 + \sum_{i \in I} \bar{I}_i \geq \sum_{i \in I} q_i$. Since the total inventory level is constant over time, then the total inventory cost per day is not lower than $\sum_{i \in I} hq_i$. The lower bound on the routing cost is the lower bound by Gallego & Simchi-Levi (1990) over an infinite time horizon. □

With this lower bound, we can now present our main result on the worst-case performance of direct shipping.

Theorem 5.1. $\frac{z^D}{z^*} \leq \max \left\{ 2, \left\lceil \sqrt{\frac{Q}{\underline{q}}} \right\rceil, \frac{Q}{\underline{q}H} \right\}$ *and the bound is tight.*

Proof. We first compute an upper bound on z^D. Consider the feasible solution where direct shipments are performed to each customer i every τ days. An upper bound on the inventory cost is $\sum_{i \in I} hq_i\tau$, i.e. the maximum possible inventory cost to serve each customer $i \in I$ every τ days, obtained when all customers are served on the same days, for example on days $0, \tau, 2\tau, ...$ The routing cost is $\sum_{i \in I} \frac{2c_{0i}}{\tau} \left\lceil \frac{q_i\tau}{Q} \right\rceil$. Let us define $\hat{I} \subseteq I$ be the subset of customers such that $\frac{q_i\tau}{Q} \leq 1$. Then, the routing cost can be written as $\sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \left\lceil \frac{q_i\tau}{Q} \right\rceil$. Therefore, $z^D \leq \sum_{i \in I} hq_i\tau + \sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \left\lceil \frac{q_i\tau}{Q} \right\rceil$.

Since the lower bound on the optimal cost proved in Lemma 5.1 can be written as follows:

$z^* \geq \sum_{i \in I} hq_i + \sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} \frac{q_i\tau}{Q} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \frac{q_i\tau}{Q}$, then

$$\frac{z^D}{z^*} \leq \frac{\sum_{i \in I} hq_i\tau + \sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \left\lceil \frac{q_i\tau}{Q} \right\rceil}{\sum_{i \in I} hq_i + \sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} \frac{q_i\tau}{Q} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \frac{q_i\tau}{Q}} \leq \max \left\{ \tau, \frac{\sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \left\lceil \frac{q_i\tau}{Q} \right\rceil}{\sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} \frac{q_i\tau}{Q} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \frac{q_i\tau}{Q}} \right\}.$$

Since $\underline{q} \leq q_i$ for all customers $i \in I$ and $\left\lceil \frac{q_i\tau}{Q} \right\rceil \leq 2\frac{q_i\tau}{Q}$ for each customer $i \in I \setminus \hat{I}$,

$$\frac{\sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \left\lceil \frac{q_i\tau}{Q} \right\rceil}{\sum_{i \in \hat{I}} \frac{2c_{0i}}{\tau} \frac{q_i\tau}{Q} + \sum_{i \in I \setminus \hat{I}} \frac{2c_{0i}}{\tau} \frac{q_i\tau}{Q}} \leq \max \left\{ \frac{Q}{\underline{q}\tau}, 2 \right\}.$$

Therefore, $\frac{z^D}{z^*} \leq \max\left\{\tau, \max\left\{\frac{Q}{\underline{q}\tau}, 2\right\}\right\}$.

Let us now select an optimal value of τ. Note first that $\sqrt{\frac{Q}{\underline{q}}}$ is the non–negative solution of the equation $\tau = \frac{Q}{\underline{q}\tau}$. If $\sqrt{\frac{Q}{\underline{q}}} < 2$, $\max\left\{\tau, \max\left\{\frac{Q}{\underline{q}\tau}, 2\right\}\right\}$ is minimized for $\tau = 2$ and the corresponding value is 2. If $\sqrt{\frac{Q}{\underline{q}}} > H$, then the previous function is minimized for $\tau = H$ and the corresponding value is $\frac{Q}{\underline{q}H}$. Otherwise, the optimal value is the best between $\left\lfloor\sqrt{\frac{Q}{\underline{q}}}\right\rfloor$ and $\left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil$. Since $\max\{\tau, \frac{Q}{\underline{q}\tau}\} = \left\lceil\frac{Q}{\underline{q}\tau}\right\rceil$ when $\tau = \left\lceil\frac{Q}{\underline{q}\tau}\right\rceil$, then $\frac{z^D}{z^*} \leq \max\left\{2, \left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil, \frac{Q}{\underline{q}H}\right\}$.

Consider the following instance to prove that the bound $\max\left\{2, \left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil, \frac{Q}{\underline{q}H}\right\}$ is tight when it is equal to 2: period $H = 2$, $n \geq 2$ customers (even number), production/consumption rate $\underline{q} = q_i = \frac{Q}{2} + \frac{Q}{n}$ for all customers $i \in I$ ($0 < \epsilon \leq \frac{Q}{n}$), inventory cost $h = 0$, transportation cost $c_{0i} = 1\ \forall i \in I$ and $c_{ij} = \frac{1}{n}$ $\forall i, j$ such that $i \in I, j \in I, i \neq j$. Note that the triangle inequality is satisfied and that $\left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil = 2$ and $\frac{Q}{\underline{q}H} < 1$. An optimal solution of Problem $\mathcal{R}$ with $K = D$ is to serve each customer with direct shipping on each day. Therefore, $z^D = 2n$. Consider now a feasible solution of Problem $\mathcal{R}$ with all routes, where two customers at a time are served in the same route on each day, delivering $\frac{Q}{2}$ each. Moreover, an additional route serving $\frac{Q}{n}$ to all customers is used on each day. This solution provides the following upper bound on the optimal cost: $z^* \leq n + 2 + (n-1)\frac{1}{n} = n + 3 - \frac{1}{n}$. Therefore, in this instance: $\frac{z^D}{z^*} \geq \frac{2n}{n+3-\frac{1}{n}} \to 2$ for $n \to \infty$.

Consider now the following instance to prove that the bound $\max\left\{2, \left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil, \frac{Q}{\underline{q}H}\right\}$ is tight when it is equal to $\left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil$: period $H = \sqrt{\frac{Q}{\underline{q}}} \geq 2$ (integer number), $n = \frac{Q}{\underline{q}}$ customers, production/consumption rate $q_i = \underline{q}$ for all customers $i \in I$, inventory cost $h = 0$, transportation cost $c_{0i} = 1\ \forall i \in I$ and $c_{ij} = \epsilon$ $\forall i, j$ such that $i \in I, j \in I, i \neq j$, where $0 < \epsilon << 1$. Note that the triangle inequality is satisfied and that $\frac{Q}{\underline{q}H} = \left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil$. Since $h = 0$ and $\underline{q}H < Q$, an optimal solution of Problem $\mathcal{R}$ with $K = D$ is to serve each customer with direct shipping just once over H. Therefore, $z^D = \frac{2n}{H} = 2\sqrt{\frac{Q}{\underline{q}}}$.

Consider now a feasible solution of Problem $\mathcal{R}$ with all routes, where all customers $i \in I$ are served every day in the same route. This solution provides the following upper bound on the optimal cost: $z^* \leq 2 + (n-1)\epsilon = 2 + (\frac{Q}{\underline{q}} - 1)\epsilon$. Therefore, in this instance: $\frac{z^D}{z^*} \geq \frac{2\sqrt{\frac{Q}{\underline{q}}}}{2+(\frac{Q}{\underline{q}}-1)\epsilon} \to \left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil$ for $\epsilon \to 0$, as $\left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil = \sqrt{\frac{Q}{\underline{q}}}$.

Finally, consider the following instance to prove that the bound $\max\left\{2, \left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil, \frac{Q}{\underline{q}H}\right\}$ is tight when it is equal to $\frac{Q}{\underline{q}H}$: period $2 \leq H < \sqrt{\frac{Q}{\underline{q}}}$ (integer number) such that $\frac{Q}{\underline{q}H} > \left\lceil\sqrt{\frac{Q}{\underline{q}}}\right\rceil$, $n = \frac{Q}{\underline{q}}$ customers, production/consumption rate $q_i = \underline{q}$ for all customers $i \in I$, inventory cost $h = 0$, transportation cost $c_{0i} = 1\ \forall i \in I$ and $c_{ij} = \epsilon\ \forall i, j$ such that $i \in I, j \in I, i \neq j$, where $0 < \epsilon << 1$. Note that the triangle inequality is satisfied. Since $h = 0$ and $\underline{q}H < Q$, an optimal solution of Problem $\mathcal{R}$ with $K = D$ is to serve each customer with direct shipping just once over H. Therefore, $z^D = \frac{2n}{H} = 2\frac{Q}{\underline{q}H}$. Consider now a feasible solution of Problem $\mathcal{R}$ with all routes, where all customers $i \in I$ are served every day in the same route. This solution provides the following upper bound on the optimal cost: $z^* \leq 2 + (n-1)\epsilon = 2 + (\frac{Q}{\underline{q}} - 1)\epsilon$. Therefore, in this instance: $\frac{z^D}{z^*} \geq \frac{2\frac{Q}{\underline{q}H}}{2+(\frac{Q}{\underline{q}}-1)\epsilon} \to \frac{Q}{\underline{q}H}$ for $\epsilon \to 0$. $\square$

This theorem shows that when customers have arbitrarily low demand rates, direct shipping may result in an arbitrarily poor solution quality. In the next section, we present sparse routing that does not have this

limitation.

5.4.2 The SVRP Subclass

We consider sparse routing that takes into account instance information. That is, we use demand information (albeit, in various methods) to solve the SVRP (a substantially less challenging single-period problem) to determine which routes will be included in the route set K. We collectively call these methods *the SVRP subclass*. In this section, we provide a worst-case performance guarantee for this subclass. To this end, we first present the following new lower bound.

Let z^* be the optimal cost of Problem $\mathcal{R}$ with all possible routes, $SVRP(\phi_i, Q)$ be the set of routes obtained by optimally solving the SVRP that delivers the quantity ϕ_i to each customer $i \in I$ when the transportation capacity is equal to Q, and $\hat{z}^{SVRP(\phi_i,Q)}$ be the corresponding routing cost. For example, $SVRP(q_iH, Q)$ is the set of routes obtained by optimally solving the SVRP when the total demand q_iH required by each customer $i \in I$ over the period H is delivered in one day. The following lemma states a lower bound on z^*.

Lemma 5.2. $z^* \geq \sum_{i\in I} hq_i + \frac{\hat{z}^{SVRP(q_iH,Q)}}{H}$

Proof. This lower bound is given by the sum of a lower bound on the inventory cost and a lower bound on the routing cost. Consider first the inventory cost. Since a quantity q_i is absorbed on day 0 by each customer $i \in I$ and the quantity $\sum_i q_i$ produced at the supplier on day 0 can be shipped to customers only on day 1, then $\bar{I}_0 + \sum_{i\in I} \bar{I}_i \geq \sum_{i\in I} q_i$. Since the total inventory level is constant over time, then the total inventory cost per day is not lower than $\sum_{i\in I} hq_i$. Consider now the routing cost. The set of routes in any feasible solution of Problem $\mathcal{R}$ provides a feasible solution for the SVRP to deliver the quantity q_iH to each customer $i \in I$, when the transportation capacity is equal to Q. Note that if the routes in the feasible solution of Problem $\mathcal{R}$ are assigned to several days, a solution with the same cost is obtained by assigning all these routes to the same day. If we set $h = 0$ and remove constraints (5.4)–(5.9) in Problem $\mathcal{R}$, an optimal solution of this model provides an optimal set of routes $SVRP(q_iH, Q)$, having cost $\hat{z}^{SVRP(q_iH,Q)}$. Therefore, $\frac{\hat{z}^{SVRP(q_iH,Q)}}{H}$ is a lower bound on the routing cost per day. □

We now provide a worst-case analysis of an optimal shipping policy computed by solving Problem $\mathcal{R}$ with $K = S$, where S is a given subset of routes, with respect to an optimal shipping policy computed by solving Problem $\mathcal{R}$ with all possible routes. Let z^S be the cost of an optimal solution of Problem $\mathcal{R}$ when the set of routes is S.

To generate the sparse set of routes to consider in the solution of Problem $\mathcal{R}$, we focus on *Single Frequency policies*, i.e. delivery policies in which the intershipment time, i.e. the number of days between any two consecutive shipments, is constant over time. Let $\tau \in \{1, 2, \ldots, H\}$ be the intershipment time and $SVRP(q_i\tau, Q)$ be the set of routes obtained by optimally solving the SVRP to deliver a quantity $q_i\tau$ to each customer $i \in I$ every τ days, when the transportation capacity is Q.

Once the set of routes $SVRP(q_i\tau, Q)$ is computed for a given τ, Problem $\mathcal{R}$ with $K = SVRP(q_i\tau, Q)$ is optimally solved. Note that the obtained optimal solution uses only the routes in the set $SVRP(q_i\tau, Q)$, but the intershipment times assigned to each customer can be different than τ. Moreover, each customer can have different intershipment times, even not based on a single frequency or multiple frequencies. For example, a customer can be served by using intershipment times in sequence that are all different.

The following lemma provides a worst-case performance ratio of the corresponding optimal cost $z^{SVRP(q_i\tau,Q)}$ with respect to the optimal cost of Problem $\mathcal{R}$ with all possible routes.

Lemma 5.3. $\frac{z^{SVRP(q_i\tau,Q)}}{z^*} \leq \max\{\tau, \frac{H}{\tau}\}$.

Proof. We first compute an upper bound on $z^{SVRP(q_i\tau,Q)}$, by summing up an upper bound on the inventory cost and an upper bound on the routing cost. The upper bound on the inventory cost is $\sum_{i\in I} hq_i\tau$, i.e. the maximum possible inventory cost to serve each customer $i \in I$ every τ days, obtained when all customers are served on the same days, for example on days $0, \tau, 2\tau, ...$ The upper bound on the routing cost is $\frac{\hat{z}^{SVRP(q_iH,Q)}}{\tau}$, computed by taking into account that the set of routes $SVRP(q_iH, Q)$ is feasible to serve all customers every τ days, as the quantity $q_i\tau$ sent every τ days to each customer $i \in I$ is not greater than the quantity q_iH sent every H days, i.e. the quantity used to compute the routes in $SVRP(q_iH, Q)$. Therefore, $z^{SVRP(q_i\tau,Q)} \leq \sum_{i\in I} hq_i\tau + \frac{\hat{z}^{SVRP(q_iH,Q)}}{\tau}$.

Since a lower bound on the optimal cost is $z^* \geq \sum_{i\in I} hq_i + \frac{\hat{z}^{SVRP(q_iH,Q)}}{H}$ thanks to Lemma 5.2, then

$$\frac{z^{SVRP(q_i\tau,Q)}}{z^*} \leq \frac{\sum_{i\in I} hq_i\tau + \frac{\hat{z}^{SVRP(q_iH,Q)}}{\tau}}{\sum_{i\in I} hq_i + \frac{\hat{z}^{SVRP(q_iH,Q)}}{H}} = \frac{\tau\sum_{i\in I} hq_i + \frac{H}{\tau}\frac{\hat{z}^{SVRP(q_iH,Q)}}{H}}{\sum_{i\in I} hq_i + \frac{\hat{z}^{SVRP(q_iH,Q)}}{H}} \leq$$

$$\leq \frac{\max\{\tau, \frac{H}{\tau}\}\left(\sum_{i\in I} hq_i + \frac{\hat{z}^{SVRP(q_iH,Q)}}{H}\right)}{\sum_{i\in I} hq_i + \frac{\hat{z}^{SVRP(q_iH,Q)}}{H}} = \max\left\{\tau, \frac{H}{\tau}\right\}.$$

□

Let us now consider the following values of τ to compute the set of routes $SVRP(q_i\tau, Q)$: 1 (Every Day SVRP), H (Total Demand SVRP) and the value τ^* that minimizes the worst-case performance bound $\max\{\tau, \frac{H}{\tau}\}$ (Optimal Single Frequency SVRP). The following theorems hold.

Theorem 5.2. $\frac{z^{SVRP(q_i,Q)}}{z^*} \leq H$.

Theorem 5.3. $\frac{z^{SVRP(q_iH,Q)}}{z^*} \leq H$.

Theorem 5.4. $\frac{z^{SVRP(q_i\tau^*,Q)}}{z^*} \leq \lceil\sqrt{H}\rceil$ *and the bound is tight.*

Proof. Consider the worst-case performance bound $\max\{\tau, \frac{H}{\tau}\}$ provided in Lemma 5.3. Since the first term is an increasing function in τ, while the second is a decreasing function in τ, then solving the following equation $\tau = \frac{H}{\tau}$, we have that the optimal non–negative value of τ is $\sqrt{H}$. Since τ^* is an integer number and the function $\max\{\tau, \frac{H}{\tau}\}$ is convex, τ^* is either$\lceil\sqrt{H}\rceil$ or $\lfloor\sqrt{H}\rfloor$. Since $\max\{\tau, \frac{H}{\tau}\} = \lceil\sqrt{H}\rceil$ when $\tau = \lceil\sqrt{H}\rceil$, while it is equal to $\frac{H}{\lfloor\sqrt{H}\rfloor}$ when $\tau = \lfloor\sqrt{H}\rfloor$, then $\frac{z^{SVRP(q_i\tau^*,Q)}}{z^*} \leq \min\left\{\lceil\sqrt{H}\rceil, \frac{H}{\lfloor\sqrt{H}\rfloor}\right\} \leq \lceil\sqrt{H}\rceil$.

Consider the following instance to prove that this bound is tight: Period H such that $\sqrt{H}$ is an integer number (i.e., $\lfloor\sqrt{H}\rfloor = \lceil\sqrt{H}\rceil = \sqrt{H}$), number of customers $n = \sqrt{H}$, production/demand rate $q_i = \frac{1}{H}$, $\forall i \in I$, inventory cost $h = 0$, transportation capacity $Q = 1$, transportation cost $c_{0i} = 1\ \forall i \in I$ and $c_{ij} = 2 - \epsilon\ \forall i, j$ such that $i \in I, j \in I, i \neq j$, where $\epsilon << 1$. Note that the triangle inequality is satisfied. Let us first compute the set of routes $SVRP(q_i\sqrt{H}, Q)$. Since the triangle inequality holds and the total quantity $nq_i\sqrt{H}$ sent every $\sqrt{H}$ days is equal to the transportation capacity $Q = 1$, the optimal set of routes $SVRP(q_i\sqrt{H}, Q)$ is composed of just one route, that is the TSP route serving all customers. The corresponding cost is $2 + (2 - \epsilon)(n - 1)$. Consider now an optimal solution of Problem $\mathcal{R}$

with $K = SVRP(q_i\sqrt{H}, Q)$ and the corresponding optimal cost $z^{SVRP(q_i\sqrt{H},Q)}$. Since the TSP route serving all customers is the only available route, the vehicle is fully loaded if this route is used every $\sqrt{H}$ days and the unit inventory cost $h = 0$, an optimal solution is to serve all customers every $\sqrt{H}$ days. Therefore, $z^{SVRP(q_i\sqrt{H},Q)} = \frac{2+(2-\epsilon)(n-1)}{\sqrt{H}}$. Consider now a feasible solution of Problem $\mathcal{R}$ with all routes, where each customer $i \in I$ is directly served with a fully loaded vehicle every H days. This solution provides the following upper bound on the optimal cost: $z^* \leq \frac{2n}{H}$. Therefore, in this instance: $\frac{z^{SVRP(\sqrt{H},Q)}}{z^*} \geq \frac{\frac{2+(2-\epsilon)(n-1)}{\sqrt{H}}}{\frac{2n}{H}} \rightarrow \frac{H}{\sqrt{H}} = \lceil\sqrt{H}\rceil$ for $\epsilon \rightarrow 0$, as $\lceil\sqrt{H}\rceil = \sqrt{H}$. □

Note that $\lceil\sqrt{H}\rceil$ is feasible even if it is not a factor of H. For example, delivering every 2 days when $H = 3$ is feasible in our tactical planning approach, because even if only q_i units are delivered to customer i on day 0, the customer already has q_i units left from the previous time horizon. Therefore, if all customers are served every 2 days, the total inventory level in the time unit is still constant over time and the inventory cost equal to $\sum_i hq_i 2$.

We note that the value of $\lceil\sqrt{H}\rceil$ grows much more slowly compared to H, which bodes well for the performance of the SVRP method to generate the set of routes for Problem $\mathcal{R}$. For example, when $H = 3$ and 6, the worst-case bound are reasonably low at 2 and 3, respectively. To our best knowledge, this is the best tight worst-case bound known for this problem. Moreover, the SVRP subclass brings with it the additional benefit that routes generated already comes with their optimal TSP costs.

5.4.3 Extensions

In this section, we extend the above results to three features; namely, non-identical inventory holding costs, capacitated customers, and time-varying demand. We show how these features affect the worst-case performance guarantees.

Non-identical Holding Costs

To model non-identical inventory holding costs, we replace h in the above models with h_0 at the supplier and h_i at customer $i, \forall i \in I$. The objective function in Problem $\mathcal{R}$ shall be replaced with $\frac{1}{H}\left(\sum_{t\in T} h_0 I_{0t} + \sum_{t\in T}\sum_{i\in I} h_i I_{it} + \sum_{t\in T}\sum_{k\in K} c_k y_{tk}\right)$. Lemma 5.1 and Lemma 5.2 will be replaced with the following.

Lemma 5.4. $z^* \geq \sum_{i\in I}(h_0 + h_i)\frac{q_i}{2} + \sum_{i\in I} 2c_{0i}\frac{q_i}{Q}$

Lemma 5.5. $z^* \geq \sum_{i\in I}(h_0 + h_i)\frac{q_i}{2} + \frac{\hat{z}^{SVRP(q_i H,Q)}}{H}$

Proof. For both lemmas, the routing costs remain the same. For the inventory cost, since the minimum intershipment time is 1 day, a lower bound on the inventory cost at the supplier is $h_0\sum_{i\in I} q_i/2$, while it is $h_i q_i/2$ for each customer i. Therefore, a lower bound on the total inventory cost in the time unit is $\sum_{i\in I}(h_0 + h_i)\frac{q_i}{2}$. □

We can then show that Theorem 5.1, Lemma 5.3, Theorem 5.2 and Theorem 5.3 still hold, while Theorem 5.4 will depend on the value of H. This is because the optimal τ must be a factor of H in order for it to be feasible. For example, if $\lceil\sqrt{H}\rceil$ is a factor of H, then the result remains. On the other hand, if H is a prime number, then the worst-case bound increases to H. In any other case, the worst-case bound will be lower than H and determined by the factor of H that minimizes the bound.

Capacitated Customers

While our model assumes uncapacitated customers, our results also holds when all customers have sufficiently large capacities. Intuitively, low capacity at even just one customer restricts the feasible values of the intershipment time τ. To examine the effect of customer capacity on our performance guarantee, we suppose that r_i is the capacity at customer $i, \forall i \in I$. For direct shipping, Theorem 5.1 may not longer hold, which implies that direct shipping can be even worse.

Theorem 5.5. $\frac{z^D}{z^*} \leq \max\left\{\tau^*, \max\left\{\frac{Q}{q\tau^*}, 2\right\}\right\}$, *where* $\tau^* = \underset{\tau \in \mathbf{Z}, \tau \leq \frac{r_i}{q_i}, \forall i \in I}{\arg\min} \left(\max\left\{\frac{Q}{q\tau^*}, 2\right\}\right)$.

Proof. The proof is similar to Theorem 5.1 with the exception that the intershipment time τ must be low enough to ensure that every customer will have enough capacity to hold the shipments that arrive. □

For the SVRP subclass, we present the following result.

Theorem 5.6. $\frac{z^{SVRP(q_i\tau,Q)}}{z^*} \leq \max\{\tau^*, H/\tau^*\}$ *where* $\tau^* = \min\left\{\lceil\sqrt{H}\rceil, \lfloor\min_{i\in I} \frac{r_i}{q_i}\rfloor\right\}$

Proof. Recall that the best performance bound is obtained by choosing τ that minimizes the performance bound on the right-hand side of Lemma 5.3. However, this time around, τ must also honor the capacity constraint at each customer. That is, $\tau^* \leq \frac{r_i}{q_i}, \forall i \in I$. Since the performance bound $\max\{\tau, H/\tau\}$ is convex in τ and using similar argument from Theorem 5.4, we obtain the desired result. □

This theorem generalizes Theorem 5.4 in the sense that if all r_i values are sufficiently high, the former recovers the latter. On the other hand, in the worst case when capacity for a given customer is only good for one day, then the performance guarantee goes up to H. It is still a finite bound and therefore not arbitrarily bad.

Time-Varying Demand

Suppose that a time-varying demand q_{it} is given for each customer i and day t. Let q_i be the corresponding average demand, i.e., $q_i = \sum_t q_{it}/H$. Suppose that the production rate at the supplier is equal to $\sum_i q_i$. Constraints (5) in Problem $\mathcal{R}$ are now formulated as follows: $I_{it} = \bar{I}_i + \sum_{\rho=0}^{t} \sum_{k\in K_i} s_{i\rho k} - \sum_{\rho=0}^{t} q_{i\rho}$, $i \in I, t \in T$. Since the total inventory level in the system $I_0 + \sum_i I_{it}$ on each day t is $\bar{I}_0 + \sum_i \bar{I}_i + \sum_i q_i t - \sum_i \sum_{\rho=0}^{t} q_{i\rho}$, the inventory cost in the objective function is $h\left(\bar{I}_0 + \sum_i \bar{I}_i\right)$ as in the constant demand case, given that $\sum_t(\sum_i q_i t - \sum_i \sum_{\rho=0}^{t} q_{i\rho})$ is just a constant.

For this setting, we present a lower bound on the optimal cost that is dependent on the average customer demand, and subsequently the main result on the worst-case performance bound.

Lemma 5.6. $z^* \geq \sum_i hq_i + \frac{\hat{z}^{SVRP(q_iH,Q)}}{H}$.

Proof. The lower bound on the inventory cost is obtained by setting $\bar{I}_0$ equal to $\sum_i q_i$, i.e. its minimum value, obtained when the daily production $\sum_i q_i$ is sent to the customers on each day, and $\bar{I}_i = 0$ for all customers i. The lower bound on the routing cost is obtained as in the constant demand case. □

Theorem 5.7. $\frac{z^{SVRP(q_i\tau,Q)}}{z^*} \leq H$.

Proof. For any intershipment τ, the maximum inventory cost is obtained when, for all customers i, $q_{i0} = q_i H$ and $q_{it} = 0$ for $t > 0$. In this case, $\bar{I}_0 + \sum_i \bar{I}_i = \sum_i q_i H$ to guarantee feasibility. Note that any other case can have a lower value of $\bar{I}_0 + \sum_i \bar{I}_i$, as one can benefit from the production of $\sum_i q_i$ units on each day. Since an upper bound on the routing cost is given by $\frac{\hat{z}^{SVRP(q_i H,Q)}}{\tau}$ as in the constant demand case, $z^{SVRP(q_i\tau,Q)} \leq \sum_i hq_i H + \frac{\hat{z}^{SVRP(q_i H,Q)}}{\tau}$. Given the lower bound on z^* provided in Theorem 1, $\frac{z^{SVRP(q_i\tau,Q)}}{z^*} \leq \frac{\sum_i hq_i H + \frac{\hat{z}^{SVRP(q_i H,Q)}}{\tau}}{\sum_i hq_i + \frac{\hat{z}^{SVRP(q_i H,Q)}}{H}} \leq \max\left\{H, \frac{H}{\tau}\right\} = H.$ □

We observe that with time-varying demand, the performance guarantee worsens to H. Nevertheless, it remains a finite bound and therefore not arbitrarily bad.

5.5 Analysis of Average Performance

In this section, we study the average performance of various methods to generate the route set, equivalently various subclasses of our matheuristics (or various types of sparse routing). To this end, we first present the flow-based formulation as a benchmark and develop a three-phase exact solution algorithm to solve this formulation. Next, we allow the vehicle capacity Q to be optimized. We can then use this optimized capacity on the VRP to generate a route set for Problem $\mathcal{R}$. Finally, we perform a computational study on various methods to generate the route set, such as SVRP, VRP, VRP with optimized capacity and combinations of the above.

5.5.1 Flow-Based Formulation: The Benchmark

We provide a flow–based model formulation for the SDIRP and design an exact algorithm, based on a branch–and–cut method, to solve it. Within a pre-specified computational time limit, the said algorithm will return either the optimal solution to the SDIRP or, at worst, the best upper and the best lower bounds. The best upper bound will be used as benchmark value and the the best lower bound to compute the optimality gap, when the optimal solution is not available.

Let P be the set of available vehicles. The flow-based formulation is based on the following additional decision variables: σ_{tpi}: quantity delivered to customer $i \in I$ by vehicle $p \in P$ on day $t \in T$; x_{tpij}: 1 if the edge $(i, j) \in E$ is traveled by vehicle $p \in P$ on day $t \in T$, and 0 otherwise; z_{tpi}: 1 if node $i \in V$ is visited by vehicle $p \in P$ on day $t \in T$, and 0 otherwise.

The mixed integer linear programming model can be formulated as follows:

Problem $\mathcal{F}$

$$\text{Min} \quad h(\overline{I}_0 + \sum_{i\in I} \overline{I}_i) + \frac{1}{H} \sum_{t\in T} \sum_{p\in P} \sum_{(i,j)\in E} c_{ij} x_{tpij} \tag{5.12}$$

$$\text{s.t.} \quad \sum_{t\in T} \sum_{p\in P} \sigma_{tpi} = q_i H \qquad i \in I \tag{5.13}$$

$$\sum_{i\in I} \sigma_{tpi} \leq Q z_{tp0} \qquad t \in T \quad p \in P \tag{5.14}$$

$$I_{0t} = \overline{I}_0 + \sum_{i\in I} q_i t - \sum_{\tau=0}^{t} \sum_{p\in P} \sum_{i\in I} \sigma_{\tau pi} \qquad t \in T \tag{5.15}$$

$$I_{it} = \overline{I}_i + \sum_{\tau=0}^{t} \sum_{p\in P} \sigma_{\tau pi} - q_i(t+1) \qquad t \in T \quad i \in I \tag{5.16}$$

$$\sigma_{tpi} \leq \min\{Q, q_i H\} z_{tpi} \qquad i \in \mathcal{I} \quad p \in P \quad t \in T \tag{5.17}$$

$$\sum_{(j,j')\in\delta(i)} x_{tpjj'} = 2 z_{tpi} \qquad i \in V \quad p \in P \quad t \in T \tag{5.18}$$

$$\sum_{(i,j)\in\mathcal{E}(\mathcal{U})} x_{tpij} \leq \sum_{i\in\mathcal{U}} z_{tpi} - z_{tpu} \qquad \mathcal{U} \subseteq \mathcal{I}, |\mathcal{U}| \geq 2 \quad u \in \mathcal{U} \quad p \in P \quad t \in T \tag{5.19}$$

$$I_{0t} \geq 0 \qquad t \in T \tag{5.20}$$

$$\overline{I}_0 \geq 0 \qquad t \in T \tag{5.21}$$

$$I_{it} \geq 0 \qquad i \in I \quad t \in T \tag{5.22}$$

$$\overline{I}_i \geq 0 \qquad i \in I \quad t \in T \tag{5.23}$$

$$\sigma_{tpi} \geq 0 \qquad t \in T \quad p \in P \quad i \in I \tag{5.24}$$

$$x_{tpij} \in \{0,1\} \qquad (i,j) \in \mathcal{E} : i \notin \{0\} \quad p \in P \quad t \in T \tag{5.25}$$

$$x_{tp0j} \in \{0,1\} \qquad j \in I \quad p \in P \quad t \in T \tag{5.26}$$

$$x_{tpj0} \in \{0,1\} \qquad j \in I \quad p \in P \quad t \in T \tag{5.27}$$

$$z_{tpi} \in \{0,1\} \qquad i \in V \quad p \in P \quad t \in T. \tag{5.28}$$

We propose the following three-phase exact solution algorithm to solve this model.

First phase: Lower bound

The lower bound is computed by optimally solving, within a time limit of 7,200 seconds, the model (5.12)–(5.28), where (5.25) is replaced by

$$x_{tpij} \in [0,1] \quad (i,j) \in \mathcal{E} : i \notin \{0\} \quad p \in P \quad t \in T \tag{5.29}$$

and the subtours elimination constraints (5.19) are removed, as they are dynamically added. To strengthen the quality of the root node lower bound, the following valid inequalities are added to the initial LP relaxation:

$$\sum_{i\in I} x_{tpi0} \leq 1 \qquad p \in P \quad t \in T, \tag{5.30}$$

$$\sum_{j\in I} x_{tp0j} \leq 1 \qquad p \in P \quad t \in T. \tag{5.31}$$

Observe that the first inequalities cut off infeasible solutions in which more than one edge outgoing the depot is traversed by a given vehicle. For example, consider the following fractional solution of the LP, in which customers i, j and l are visited in this order in period t by vehicle k, such that the routing variables take the following values: $\overline{z}_{tpi} = \overline{z}_{tpj} = 1, \overline{z}_{tp0} = 1.25, \overline{z}_{tpl} = 0.75, \overline{x}_{tp0i} = \overline{x}_{tpij} = \overline{x}_{tpl0} = 1, \overline{x}_{tp0j} = \overline{x}_{tpjl} = 0.5$. This solution violates inequalities (5.30). The second inequalities work similarly to the first ones.

Second phase: Heuristic algorithm

The heuristic solution is computed by finding, for each day t and vehicle p, an optimal TSP tour on the complete undirected subgraph induced by the subset $Z_{tp} \cup \{0\}$, where $Z_{tp} = \{i \in I : z^{LB}_{tpi} = 1\}$ and z^{LB}_{tpi} is the optimal value of the variable z_{tpi} obtained in the First phase (see Algorithm 2 in Appendix 5.A).

Third phase: Branch-and-Cut algorithm

In the third phase, we optimally solve model (5.12)-(5.28), without the Subtours elimination constraints (5.19), through a branch-and-cut algorithm. The Subtours elimination constraints are dynamically added. They are separated along the lines of the heuristic and exact procedures designed by Ahr (2004). More precisely, at the root node the heuristic finds all the connected components in the auxiliary graph induced by all the edges such that $\overline{x}_{tpij} >= \epsilon + \tau$, where $\overline{x}_{tpij}$ is the value of variable x_{tpij} on edge (i, j) in the current LP, while $\epsilon \in \{0, 0.25, 0.50\}$ and τ is a tolerance. In all the other nodes of the branch–and–cut tree, the connected components are only those induced by the edges such that $\overline{x}_{tpij} >= \tau$, since $\epsilon = 0$. Whenever a subset of customers S_p not connected to depot 0 is found, the corresponding subtour elimination constraint is added for any vertex $u \in S_p$. If no violation is found by the heuristic procedure, then a max-flow problem based exact procedure is executed to find violations. The initial upper bound is set equal to the cost of the heuristic solution found in the Second phase. The branch-and-cut is implemented in CPLEX 12.6 and solved within a time limit of 14,400 seconds.

In order to improve the value of the lower bound, we add the following classical valid inequalities (see Adulyasak et al. (2013)) at the root of the branch-and-bound tree:

Priority inequalities:

$$z_{tpi} \leq z_{tp0} \quad i \in I \quad p \in P \quad t \in T, \tag{5.32}$$

$$x_{tpij} \leq z_{tpi} \quad (i,j) \in \mathcal{E} : i \neq 0 \quad p \in P \quad t \in T. \tag{5.33}$$

Symmetry breaking inequalities:

$$z_{tp0} \geq z_{t,next(p),0} \quad p \in P - last(P) \quad t \in T, \tag{5.34}$$

$$\sum_{i=1}^{j} 2^{(j-i)} z_{tpi} \geq \sum_{i=1}^{j} 2^{(j-i)} z_{t,next(p),i} \quad j \in I \quad p \in P - last(P) \quad t \in T. \tag{5.35}$$

Moreover, we add the following logical inequalities:

$$\sum_{i \in I} x_{tpi0} = \sum_{j \in I} x_{tp0j} \quad p \in P \quad t \in T, \tag{5.36}$$

$$\sum_{i \in I} x_{tpi0} = z_{tp0} \quad p \in P \quad t \in T. \tag{5.37}$$

Finally, we introduce the following valid *Disaggregate parity inequalities* based on the cocircuit inequalities (see Barahona & Grötschel (1986)), which we dynamically add to the branch-and-cut:

$$\sum_{(i,j)\in\delta(S)\setminus\mathcal{H}} x_{tpij} \geq \sum_{(u,v)\in\mathcal{H}} x_{thuv} - |\mathcal{H}| + 1, \forall p \in P,\ \forall S \subset V,\ \mathcal{H} \subseteq \delta(S),\ |\mathcal{H}|\ odd, \forall t \in T. \quad (5.38)$$

Inequalities (5.38) are separated by using heuristic and exact procedures similar to the ones provided by Aráoz et al. (2009). More precisely, the heuristic algorithm is called at the root node of the branch–and–cut tree and, if it fails, the exact method is executed. In all the other nodes of the branch–and–cut tree, only the heuristic procedure is applied.

5.5.2 Routes with Optimized Capacity

We now introduce another subclass; namely, the set of routes generated by solving the VRP with total planning horizon demand and optimized vehicle capacity. We call this method, and the corresponding set of routes, $VRP(q_iH, \alpha)$, where $0 < \alpha \leq Q$ is the optimized capacity. If $f(VRP(q_iH, \alpha))$ is an estimate of the cost of Problem $\mathcal{R}$ when this set of routes is used, this subclass can be modeled as follows:

$$\min\ f(VRP(q_iH, \alpha)) \quad (5.39a)$$

$$0 < \alpha \leq Q \quad (5.39b)$$

The rationale for this method is that, by reducing the capacity, the routes generated solving the VRP could be worse in terms of routing costs, but better in terms of inventory costs for the corresponding IRP. In fact, delivering a lower quantity on each routes implies more visits to the customers and therefore an increase in the routing costs, but a decrease in the inventory costs.

Algorithm 3 described in Appendix 5.A aims at heuristically solving problem (5.39a) - (5.39b). Because the quality of an a priori approximation of the objective function could be low, the idea is to estimate the objective function after explicitly knowing the set of routes. We refer to an estimating function as $f(S)$, where S is a generic set of routes. The algorithm explores solutions by considering different values of α and, for each of them, solves the corresponding $VRP(q_iH, \alpha)$ to obtain the route set and estimate the total cost. Note that, we let the algorithm also search for values of α greater than the capacity Q (therefore relaxing constraints (5.39b)). This is because, when routes are given as input to formulation (5.1)-(5.11), the delivered quantity is decided by the model and feasibility is guaranteed by constraints (5.3). However, the best accepted value of capacity, that is not greater than Q, is stored and the corresponding routes are returned at the end of the algorithm. Therefore, the algorithm is ensuring two, potentially different, sets of routes: one considering the global best accepted value of capacity, and another one considering the best accepted values of capacity that satisfies constraint (5.39b). In conclusion, the set of routes with optimized capacity $VRP(q_iH, \alpha)$ is the union of the sets of routes $VRP(q_iH, \tilde{\alpha})$ obtained by applying Algorithm 3 with the two definitions of $f(S)$ provided in Appendix 5.A.

5.5.3 Computational Results

In this section, our aim is to compare the optimal solution of various subclasses of our matheuristics (i.e. Problem $\mathcal{R}$ with various route sets) and the optimal solution (or best lower bound) for our benchmark (i.e. the flow-based formulation (5.12)–(4.12) obtained by applying the branch-and-cut algorithm we designed), for a given computational time limit.

Our data set is composed of a large set of instances derived from the benchmark instances by Archetti et al. (2007). We use the same locations of the supplier and the customers, the same demand rate of each customer and the same capacity, while we set the unit inventory cost of each customer equal to the inventory cost of the supplier. The time horizon H is 3 and 6. Note that the initial inventory levels at the supplier and at the customers are data in Archetti et al. (2007), while they are decision variables in our problem. Moreover, there is a maximum inventory level at the customers in Archetti et al. (2007), while we do not have a maximum inventory level. Therefore, the feasible region of our instances is larger than the one of the instances in Archetti et al. (2007).

The solution approaches were coded in C++ and compiled with g++ -O3. The computational experiments were carried out on a PC equipped with 2 Intel Xeon E5335 CPUs running at 2.00 GHz, with 6 GB of RAM. To limit the number of routes to be used in a day, we added the constraint $\sum_{k \in K} y_{tk} \leq \left\lceil \frac{\sum_{i \in I} q_i H}{Q} \right\rceil$ for each day t to the Route-based formulation (5.1)-(5.11) and, equivalently, we defined the cardinality of the set P of available vehicles as $|P| = \left\lceil \frac{\sum_{i \in I} q_i H}{Q} \right\rceil$ in the flow-based formulation (5.12)–(4.12).

The VRP and SVRP are solved using the local search metaheuristic (Record-to-Record Travel) implemented in the VRPH library (see Groër et al. (2010), https://projects.coin-or.org/VRPH), by setting its parameters as follows: size of the main loop in the diversification phase D: 30; deterioration of the objective function in the diversification phase δ: 0.1; number of local minima before perturbing the solution K: 5; size of the neighbor list when running the local search operators N: 25; number of times the solution is perturbed once the search is stuck in a local minimum P: 2; local search operators U: one point move, two point move, two opt, three opt.

The SVRP is heuristically solved using the same algorithm used to solve the VRP, but on a different network. In particular, the SVRP is solved solving a VRP on a complete auxiliary graph $G'(V', E')$ where V' is the set of vertices corresponding to the supplier 0 and η nodes for each customer $i \in I$. Each $i' \in V'$ generated from $i \in I$, has a) the same location of node i in graph $G(V, E)$ (therefore, all the η nodes generated by a node i have the same location), b) a demand equal to d_i/η. From a feasible solution of a VRP on graph $G'(V', E')$, it is possible to obtain a feasible solution for the SVRP problem on $G(V, E)$. The parameter η influences both solution quality and computational time of the procedure. In particular, higher values of η allow us to split more the demand and therefore to potentially have better solutions, but increase the number of nodes in the network $G'(V', E')$ and therefore the computational time. In the experiments, we set $\eta = 3$, as we realized that this value provides a good balance between solution quality and computational time. The Record-to-Record Travel algorithm, both for VRP and SVRP, is applied 10 times and the best solution is selected. In each of the 10 iterations, this algorithm starts from a different feasible solution computed by using the parametrized Clarke and Wright algorithm provided by the library, starting with a parameter equal to 0.25 and increasing it by 0.25 in each iteration. Then, each route is optimized separately by using Concorde TSP Solver (see http://www.math.uwaterloo.ca/tsp/concorde.html).

In the computational experiment, we consider the following methods to generate route sets:

- Method 1: $SVRP(q_i\tau^*, Q)$
- Method 2: $VRP(q_i\tau^*, Q)$
- Method 3: Method 1 $\cup$ Method 2

- Method 4: Method 3 $\cup\, VRP(q_i, Q) \cup VRP(q_i H, Q)$
- Method 5: Method 4 $\cup\, VRP(q_i H, \alpha)$
- Method 6: Method 4 $\cup\, SVRP(q_i, Q) \cup SVRP(q_i H, Q)$

The rationale for these methods is as follows. Method 1 corresponds to the SVRP subclass, for which we obtained our best worst-case performance bound $\lceil\sqrt{H}\rceil$. Method 2 is the VRP variant of Method 1 and it allows us to evaluate if using VRP instead of SVRP provides a good compromise between increase in total cost and reduction in computational time. Method 3 allows us to evaluate the benefit of using both VRP and SVRP subclasses together. Method 4 expands the route set in Method 3 to include routes generated from other VRP subclasses; namely, using single period demand and using total planning horizon demand. We choose VRP to add more routes as it is computationally faster than SVRP. Method 5 further adds to the route set in Method 4 routes generated from the VRP method using optimized capacity, introduced in Section 5.2. Finally, Method 6 provides an alternative to Method 5 by adding to the route set in Method 4 routes generated from other SVRP subclasses; namely, using single period demand and using total planning horizon demand. Examining both Method 5 and Method 6 allows us to evaluate whether the VRP method with optimized capacity or other SVRP subclasses, bring about more benefits in terms of solution quality and computational time.

Table 5.5.1: Average Performance for Methods 1 to 3 when $H = 3$

n	Flow-Based		Method 1			Method 2			Method 3		
	OG	CT	OG	CT	RG	OG	CT	RG	OG	CT	RG
5	0.00	56.24	37.21	0.24	2.00	40.36	0.13	2.00	36.11	0.31	2.40
10	0.00	40.58	24.76	0.49	2.00	24.75	0.17	2.00	24.75	0.56	2.20
15	0.00	127.39	25.08	1.25	2.00	25.08	0.23	2.00	25.08	1.24	2.00
20	0.00	1199.71	18.95	2.19	2.00	18.95	0.32	2.00	18.95	2.12	2.00
25	0.11	8581.79	16.80	3.59	2.00	16.74	0.48	2.00	16.73	3.32	2.40
30	1.26	13163.53	16.92	5.73	2.00	16.92	0.61	2.00	16.92	5.15	2.20
35	5.22	21600.00	14.07	7.99	2.00	14.15	0.77	2.00	14.01	7.10	2.40
40	8.85	21600.00	15.15	9.43	2.00	16.14	1.03	2.00	13.19	8.74	3.00
45	11.26	21600.00	17.52	13.88	2.00	17.60	1.21	2.00	17.52	12.21	2.60
50	22.13	21600.00	18.07	17.52	2.00	18.26	1.49	2.00	17.67	14.80	3.40
Average	4.88	10956.92	20.45	6.23	2.00	20.89	0.65	2.00	20.09	5.55	2.46

Table 5.5.2: Average Performance for Methods 4 to 6 when $H = 3$

n	Flow-Based		Method 4			Method 5			Method 6		
	OG	CT	OG	CT	RG	OG	CT	RG	OG	CT	RG
5	0.00	56.24	1.69	1.72	5.40	1.69	4.53	6.00	1.69	2.71	5.40
10	0.00	40.58	1.49	2.78	6.20	0.80	6.86	7.40	1.49	5.58	6.60
15	0.00	127.39	2.74	4.51	6.00	1.98	9.93	8.20	2.50	9.98	6.60
20	0.00	1199.71	3.56	7.05	5.80	1.25	15.09	8.80	3.56	16.04	7.00
25	0.11	8581.79	3.17	11.49	6.40	2.30	23.79	9.60	3.12	27.17	7.80
30	1.26	13163.53	6.27	17.09	6.20	2.90	32.84	9.40	6.13	36.77	6.80
35	5.22	21600.00	5.46	22.14	6.40	3.95	45.79	10.20	5.46	49.01	7.60
40	8.85	21600.00	6.86	27.09	7.00	3.11	63.56	12.20	5.75	65.15	8.80
45	11.26	21600.00	10.77	36.43	6.40	4.89	77.79	11.40	10.29	81.82	7.80
50	22.13	21600.00	12.17	45.34	7.40	9.74	110.31	11.60	10.65	105.78	9.40
Average	4.88	10956.92	5.42	17.56	6.32	3.26	39.05	9.48	5.06	40.00	7.38

We summarize our computational results in Tables 5.5.1 to 5.5.4. Table 5.5.1 shows the average performance for Methods 1 to 3, compared to the Flow-Based benchmark, when planning horizon $H = 3$. Table 5.5.2 shows the performance for Methods 4 to 6, also when $H = 3$. Meanwhile, Tables 5.5.3 and 5.5.4 do the same as Tables 5.5.1 and 5.5.2, respectively, for the longer planning horizon of $H = 6$. For each table, we report average values for the optimality gap (OG), the computational time (CT) and number of routes generated (RG) for each method. We also tabulate average OG and CT values for the flow-based benchmark. OG measures solution quality with a lower value corresponding to better

Table 5.5.3: Average Performance for Methods 1 to 3 when $H = 6$

n	Flow-Based		Method 1			Method 2			Method 3		
	OG	CT	OG	CT	RG	OG	CT	RG	OG	CT	RG
5	0.00	743.10	4.41	0.90	2.80	7.93	0.70	3.00	4.41	1.23	4.60
10	6.78	19839.16	10.57	1.05	3.00	10.57	0.92	3.00	10.57	1.40	3.00
15	11.05	21600.00	7.51	1.70	3.00	7.51	0.90	3.00	7.51	1.67	3.00
20	42.07	21600.00	10.06	2.84	3.00	9.84	1.04	3.00	9.84	3.68	3.60
25	52.39	21600.00	13.87	4.61	3.00	13.55	1.59	2.80	13.29	5.69	3.80
30	49.54	21600.00	16.23	5.77	3.00	14.28	1.70	3.00	14.28	11.60	4.60
35	-	21600.00	17.93	8.20	3.00	18.16	1.82	3.00	17.89	14.01	4.00
40	-	21600.00	17.85	10.22	2.80	17.77	2.26	3.00	17.41	22.15	5.00
45	-	21600.00	16.30	12.26	2.80	16.37	2.02	2.80	15.96	19.20	4.40
50	14.56	21600.00	17.01	16.67	3.00	16.67	2.36	3.00	16.51	29.91	4.40
Average	>24.11	19338.23	13.17	6.42	2.94	13.26	1.53	2.96	12.77	11.05	4.04

Table 5.5.4: Average Performance for Methods 4 to 6 when $H = 6$

n	Flow-Based		Method 4			Method 5			Method 6		
	OG	CT	OG	CT	RG	OG	CT	RG	OG	CT	RG
5	0.00	743.10	1.62	11.53	8.20	1.52	27.96	9.60	1.30	15.48	9.40
10	6.78	19839.16	8.99	8.53	8.20	8.99	51.82	11.80	8.61	27.71	10.80
15	11.05	21600.00	7.15	19.42	8.20	6.71	106.30	13.00	7.15	28.36	9.80
20	42.07	21600.00	9.77	53.93	9.00	9.35	417.61	15.40	9.35	166.72	11.60
25	52.39	21600.00	12.42	131.01	9.40	12.19	1163.06	16.60	12.42	155.00	10.20
30	49.54	21600.00	13.82	342.50	9.60	13.57	14835.21	19.20	13.56	1200.76	13.00
35	-	21600.00	16.72	297.26	9.20	16.26	14852.53	19.60	16.69	1586.10	11.20
40	-	21600.00	17.19	970.90	10.40	15.95	19050.15	19.00	16.91	8840.80	13.60
45	-	21600.00	15.67	769.81	9.60	16.03	27844.70	19.60	15.67	5936.47	12.00
50	14.56	21600.00	16.51	768.05	9.80	16.52	20836.34	19.80	16.51	5873.31	13.80
Average	>24.11	19338.23	11.99	337.29	9.16	11.71	9918.57	16.36	11.82	2383.07	11.54

quality. For the Flow-Based benchmark it is defined as (optimal objective value or best upper bound of benchmark)/(optimal objective or best lower bound of benchmark) – 1, while for the other methods it is defined as (optimal objective value of method)/(optimal objective or best lower bound of benchmark) – 1, and it is reported in percentage points. CT indicates the required computational resource whereas RG measures the sparsity of the route set generated by the method.

Of the 50 problem instances we considered, there are 5 instances for each problem size, where problem size is defined by number of customers n, which ranges from 5 to 10, 15, 20, ..., 50. The values of OG, CT, and RG reported in the tables are averages across all instances considered for each problem size.

For shorter planning horizon $H = 3$, we make the following observations based on Tables 5.5.1 and 5.5.2. First, Method 2 is substantially faster than Method 1 with only a slight deterioration in solution quality. Second, Methods 1 to 4 can all be completed under a minute with acceptable solution quality. In fact, Method 4's solution quality already comes very close to the benchmark (5.42% versus 4.88% on average). However, the latter requires an average of 3.04 hours of computational time and reaches the 6-hour time limit once problem size becomes 35 customers or more. Finally, it appears that Method 5 is a more efficient way to use computational resources compared to Method 6. While both methods increase the average CT to about 40 seconds, Method 5 substantially improves solution quality to 3.26% on average whereas Method 6 barely makes a smaller dent at 5.06%. Overall, our results seem to recommend Method 5 as the best trade-off between solution quality and computational time.

When it comes to longer planning horizons such as $H = 6$, we draw the following observations from Tables 5.5.3 and 5.5.4. First, we observe that the flow-based formulation produces very poor solutions. Moreover, for instances with 35 customers or more, only one feasible solution was obtained within the 6-hour time limit. Second, Method 2 is once again substantially faster than Method 1 with only a slight increase in cost. Third, as before, Methods 1 to 4 can all be completed relatively quickly with average computational time of 5.6 minutes (for Method 4) with good solution quality (11.99% on average for Method 4). Unlike the shorter planning horizon, these four methods start to outperform the benchmark as

soon as problem size grows to 20 customers. Finally, our observations on Methods 5 and 6 vary from those from $H = 3$. Between them, Method 5 generates a slightly better solution quality than Method 6, but it requires substantially more computational time. Furthermore, Methods 5 and 6 do not provide meaningful improvements in solution quality to justify the increase in computational time from an average of 5.6 minutes to 2.7 hours and 40 minutes, respectively. Overall, the results appear to recommend Method 4 as the best balance between solution quality and computational time, for problem instances with longer planning horizons. We think that this behavior on the instances with longer time horizon is correlated to the amplitude of the horizon. In fact, over a time horizon that is twice than the horizon with only three periods, the policy to ship the daily demand to some customers and the total planning horizon demand to other customers can be more effective than other policies in which the total demand is split in subdeliveries. In fact, the first policy aims at reducing the delivery frequency to the customers that have high consumption rate with respect to the ones with a smallest one. The latter leads to an increase in the delivery frequency. As a result, the modularity of this frequency has much impact on the overall routing cost over a longer time horizon than on a shorter planning.

Focusing on the contribution of the subsets of routes we used to prove our best worst-case bound $\lceil\sqrt{H}\rceil$ in finding good solutions of the SDIRP in short computational time, we can conclude that this subset is very good when $H = 6$. In fact, Method 1 has an average optimality gap of 20.45% when $H = 3$, while our best average optimality gap is 3.26% (Method 4). However, when $H = 6$, Method 1 has an optimality gap of 13.17%, while our best average optimality gap is 11.71% (Method 5). The average computational time is very low (less than 6.5 seconds). Moreover, by just adding to this subset of routes the corresponding subset obtained by solving the VRP, instead of the SVRP, and the subsets of routes we used to prove our second best worst-case bound H, we have a very good performance, both when $H = 3$ and $H = 6$. In fact, Method 4 has an average optimality gap of 5.42% when $H = 3$ (while our best average optimality gap is 3.26%) and 11.99% when $H = 6$ (while our best average optimality gap is 11.71%). Moreover, the average computational time is still practical: 17.56 seconds when $H = 3$ and 337.29 seconds when $H = 6$. So, we can conclude that the subsets of routes used to prove the worst-case bounds are effective in finding good solutions of the SDIRP in short computational time, in particular when $H = 6$. Therefore, these subsets of routes are good to handle scalability in the SDIRP with respect to planning horizon.

Finally, we tabulate all test instances considered and report the performance of the best performing methods; namely, Method 5 for $H = 3$ and Method 4 for $H = 6$ in Table 5.B.1 in Appendix 5.B. We also provide the performance of the flow-based benchmark for comparison.

5.6 Conclusion

In this paper, we introduce the notion of sparse routing for the finite-horizon IRP with split deliveries. Essentially a class of matheuristics, sparse routing solves a route-based mixed-integer program where decision variables are declared for each route and each period. Route-based formulations are very appealing for real applications, as the set of routes used in the model can be composed at the beginning by the routes already used by the company and then enlarged over time to find better and better solutions. To avoid the complexity of a formulation with all possible routes, we show how to design effective and efficient sparse routing, i.e. choose sparse yet effective sets of routes. We make a number of contributions to the IRP literature. First, we show that direct shipping can be arbitrarily bad for finite-horizon IRP

with split deliveries. Second, we show that the optimal routing cost of the SVRP with total planning horizon demand forms part of a lower bound on the optimal SDIRP cost. Third, we prove an upper bound on the performance of the set of routes generated by SVRP with demand from any number of periods. Fourth, we optimize the number of periods to obtain the best tight worst-case performance bound known for this problem, which is $\lceil \sqrt{H} \rceil$, where H is the planning horizon. Fifth, we extend our results to non-identical inventory holding costs, capacitated customers and time-varying demand. Finally, we perform a computational study on an extensive set of test instances to show the average performance of various methods to generate the set of routes for our class of matheuristics. As managerial insights, we show that the subsets of routes used to prove the worst-case bounds can be effective in finding good solutions of the SDIRP in short computational time. Moreover, we recommend ways to to select sets of routes able to strike a balance between solution quality and computational time and show how the choice differs depending on factors such as planning horizon. These subsets of routes can be added to the routes already used by the company to compute, in short computational time, an effective solution having at the same time a worst-case performance guarantee.

Appendix

5.A Algorithms

In this section we provide a formal description of the heuristic algorithm we use in the first step of our exact approach (Algorithm 2), and of the algorithm we apply to compute the routes with optimized capacity (Algorithm 3). Moreover, we show how we compute the estimate $f(S)$ of the objective function when the set S of routes is known.

Algorithm 2: Heuristic algorithm

for each day $t \in T$ **do**
  for each vehicle $p \in P$ **do**
    Compute the subset of visited customers $Z_{tp} = \{i \in I : z^{LB}_{tpi} = 1\}$.
    Find an optimal TSP tour on the complete undirected subgraph induced by $Z_{tp} \cup \{0\}$.
  end for
end for

Algorithm 3: Algorithm to compute $VRP(q_iH, \tilde{\alpha})$

Require: $0 < \lambda < 1, 0 < \epsilon < 1, f(S)$.
Ensure: The set of routes $VRP(q_iH, \tilde{\alpha})$.
$\tilde{\alpha}, \tilde{\alpha}_f := Q, \tilde{c} := f(VRP_{q_iH,Q})$, stop = $false$
while stop = $false$ **do**
  $\overline{\alpha} := \tilde{\alpha}(1 + \lambda)$
  $\underline{\alpha} := \tilde{\alpha}(1 - \lambda)$
  $\alpha_{\min} := arg\min\{f(VRP(q_iH, \underline{\alpha})), f(VRP(q_iH, \overline{\alpha}))\}$
  $c_{\min} := \min\{f(VRP(q_iH, \underline{\alpha})), f(VRP(q_iH, \overline{\alpha}))\}$
  if $c_{\min} < \tilde{c}$ **then**
    $\tilde{\alpha} := \alpha_{\min}$
    $c := c_{\min}$
    if $\alpha_{\min} \leq Q$ **then**
      $\tilde{\alpha}_f = \alpha_{\min}$
    end if
    $VRP(q_iH, \tilde{\alpha}) := VRP(q_iH, \alpha_{\min}) \cup VRP(q_iH, \tilde{\alpha}_f)$
    $\lambda := \lambda\epsilon$
  else
    stop = $true$
  end if
end while
return $VRP(q_iH, \tilde{\alpha})$.

Algorithm 3 starts with a given value of capacity (Q) and searches around this point for a better solution. At a generic iteration where the current value of capacity is $\tilde{\alpha}$, two different values of capacity are explored: $\overline{\alpha} := \tilde{\alpha}(1 + \lambda)$ and $\underline{\alpha} := \tilde{\alpha}(1 - \lambda)$. For these two values, $VRP(q_iH, \overline{\alpha})$ and $VRP(q_iH, \underline{\alpha})$ are computed to estimate the costs of each sets. If the least cost set has a cost lower than the current

solution, the values of $\tilde{\alpha}$ is updated. This schema is repeated until no improvements are found.

We consider two different definitions of $f(S)$. In both definitions, the routing cost is the sum of the cost of the routes in the set S divided by the planning horizon duration. In the first definition of $f(S)$, the inventory cost per day is set equal to $h\tilde{q}$, where $\tilde{q} = \max\{q_p, q_{max}\}$ is the sum of the quantity delivered in the least $\left\lceil \frac{|S|}{H} \right\rceil$ loaded vehicles in the set S, and q_{max} is equal to the quantity delivered by the most loaded vehicle. The rationale behind this approximation is that, if each route is used exactly once in Problem $\mathcal{R}$ (like in the routing cost approximation), there will be one day in which at least $\tilde{q}$ units will delivered and, therefore, on hand at the supplier. In the second definition, the inventory cost is computed by optimally solving the following mixed integer linear programming model, in which each route in the set S can be used only once over the period H.

Problem $\mathcal{I}$

$$\text{Min} \quad h\left(\overline{I}_0 + \sum_{i\in I} \overline{I}_i\right) \tag{5.40}$$

$$\text{s.t.} \quad I_{0t} = \overline{I}_0 + \sum_{i\in I} q_i t - \sum_{\tau=0}^{t} \sum_{k\in S} \sum_{i\in\mathcal{R}_k} q_i H y_{\tau k} \qquad t \in T \tag{5.41}$$

$$I_{it} = \overline{I}_i + \sum_{\tau=0}^{t} \sum_{k\in\mathcal{K}_i} q_i H y_{\tau k} - q_i(t+1) \qquad i \in I \quad t \in T \tag{5.42}$$

$$\sum_{t=0}^{H-1} y_{tk} = 1 \qquad k \in S \tag{5.43}$$

$$\overline{I}_0 \geq 0 \tag{5.44}$$

$$I_{0t} \geq 0 \qquad t \in T \tag{5.45}$$

$$\overline{I}_i \geq 0 \qquad i \in I \tag{5.46}$$

$$I_{it} \geq 0 \qquad i \in I \quad t \in T \tag{5.47}$$

$$y_{tk} \in \{0,1\} \qquad i \in I \quad t \in T \quad k \in S. \tag{5.48}$$

5.B Detailed computational results

In this section we tabulate all test instances and report the performance of the flow–based benchmark and of the best performing methods, namely, Method 5 for $H = 3$ and Method 4 for $H = 6$.

Table 5.B.1: Flow-Based Benchmark vs. Best Route-Based Method

Instance	$H = 3$					$H = 6$				
	Flow-Based Benchmark		Route-Based Method 5			Flow-based Benchmark		Route-Based Method 4		
	OG	CT	OG	CT	RG	OG	CT	OG	CT	RG
abs1n05	0.00	273.54	0.00	5.76	5	0.00	255.64	1.86	7.96	7
abs2n05	0.00	0.75	0.00	4.80	6	0.00	325.17	0.00	9.44	8
abs3n05	0.00	2.05	2.14	4.04	7	0.00	294.65	1.06	9.23	8
abs4n05	0.00	3.98	0.00	4.75	7	0.00	600.17	2.80	16.37	9
abs5n05	0.00	0.89	6.33	3.30	5	0.00	2239.89	2.37	14.63	9
abs1n10	0.00	26.16	1.56	6.32	8	12.06	21600.00	11.12	9.9	8
abs2n10	0.00	135.92	0.00	6.27	7	8.58	21600.00	13.65	9.39	8
abs3n10	0.00	13.09	0.00	7.28	6	0.00	12795.82	6.04	7.04	8
abs4n10	0.00	14.59	0.00	7.05	9	10.09	21600.00	10.20	7.16	8
abs5n10	0.00	13.16	2.46	7.36	7	3.16	21600.00	3.97	9.15	9
abs1n15	0.00	26.70	3.05	10.83	7	13.11	21600.00	6.15	23.4	8
abs2n15	0.00	27.94	0.00	10.74	9	9.36	21600.00	9.29	19.24	8
abs3n15	0.00	275.08	3.63	8.71	8	17.36	21600.00	5.02	22.18	9
abs4n15	0.00	40.83	0.37	9.88	8	2.04	21600.00	7.34	15.94	8
abs5n15	0.00	266.39	2.87	9.51	9	13.37	21600.00	7.96	16.33	8
abs1n20	0.00	261.93	1.72	15.30	7	25.98	21600.00	6.75	32.89	8
abs2n20	0.00	788.82	0.06	15.54	10	20.89	21600.00	9.59	42.18	9
abs3n20	0.00	294.79	0.00	15.17	9	97.80	21600.00	6.35	30.82	8
abs4n20	0.00	2134.24	2.33	14.34	8	39.12	21600.00	9.34	45.89	10
abs5n20	0.00	2518.79	2.15	15.10	10	26.57	21600.00	16.79	117.88	10
abs1n25	0.00	586.32	0.21	21.09	6	38.81	21600.00	6.38	306.68	11
abs2n25	0.00	4827.96	2.75	23.59	11	56.63	21600.00	15.90	56.87	9
abs3n25	0.56	21600.00	3.21	24.50	8	64.86	21600.00	14.35	189.34	10
abs4n25	0.00	840.26	3.90	25.17	12	46.86	21600.00	3.79	39.48	9
abs5n25	0.01	15054.39	1.39	24.61	11	54.77	21600.00	21.68	62.67	8
abs1n30	0.00	3219.95	5.28	32.53	8	-	21600.00	15.99	80.66	8
abs2n30	0.75	21600.00	2.98	28.07	9	50.78	21600.00	15.23	65.04	8
abs3n30	0.00	1752.33	1.06	31.94	9	-	21600.00	11.03	584.33	10
abs4n30	0.00	17645.38	1.95	39.81	13	-	21600.00	10.95	495.53	12
abs5n30	5.55	21600.00	3.23	31.87	8	48.29	21600.00	15.92	486.96	10
abs1n35	2.86	21600.00	2.72	61.29	13	-	21600.00	14.06	696.78	11
abs2n35	0.66	21600.00	2.59	31.35	8	-	21600.00	12.74	93.20	8
abs3n35	9.55	21600.00	5.65	42.35	10	-	21600.00	17.01	83.61	8
abs4n35	5.89	21600.00	4.40	47.73	8	-	21600.00	17.62	489.84	11
abs5n35	7.14	21600.00	4.40	46.24	12	-	21600.00	22.19	122.85	8
abs1n40	12.08	21600.00	4.74	54.95	9	-	21600.00	16.15	655.90	10
abs2n40	7.96	21600.00	1.26	65.07	13	-	21600.00	20.70	120.44	8
abs3n40	9.93	21600.00	4.31	64.05	12	-	21600.00	14.59	1945.82	12
abs4n40	4.66	21600.00	2.03	64.18	14	-	21600.00	19.48	971.77	
abs5n40	9.66	21600.00	3.20	69.55	13	-	21600.00	15.01	1160.59	11
abs1n45	10.10	21600.00	2.76	80.31	12	-	21600.00	16.05	771.01	10
abs2n45	13.02	21600.00	7.46	89.46	13	-	21600.00	19.92	894.76	10
abs3n45	0.25	21600.00	3.34	75.60	9	-	21600.00	12.63	180.05	8
abs4n45	29.58	21600.00	8.49	77.15	12	-	21600.00	18.70	1024.94	10
abs5n45	3.35	21600.00	2.41	66.44	11	-	21600.00	11.05	978.30	10
abs1n50	19.16	21600.00	12.67	113.59	11	-	21600.00	19.31	1427.41	12
abs2n50	23.37	21600.00	9.84	95.83	9	-	21600.00	17.12	160.22	9
abs3n50	6.24	21600.00	10.24	129.75	12	-	21600.00	15.82	1131.21	10
abs4n50	45.51	21600.00	9.13	115.73	13	14.56	21600.00	16.16	150.48	8
abs5n50	16.38	21600.00	6.85	96.66	13	-	21600.00	14.16	970.91	10
Average	4.88	10956.92	3.26	39.05	9.48	>24.11	19338.23	11.99	337.29	9.16

Bibliography

Adelman, D. (2004), 'A price-directed approach to stochastic inventory/routing', *Operations Research* **52**(4), 499–514.

Adulyasak, Y., Cordeau, J.-F. & Jans, R. (2013), 'Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems', *INFORMS Journal on Computing* **26**(1), 103–120.

Aghezzaf, E.-H., Raa, B. & Van Landeghem, H. (2006), 'Modeling inventory routing problems in supply chains of high consumption products', *European Journal of Operational Research* **169**(3), 1048–1063.

Aghezzaf, E.-H., Zhong, Y., Raa, B. & Mateo, M. (2012), 'Analysis of the single-vehicle cyclic inventory routing problem', *International Journal of Systems Science* **43**(11), 2040–2049.

Ahr, D. (2004), Contributions to multiple postmen problems, PhD thesis.

Ahuja, R., Orlin, J., Pallottino, S., Scaparra, M. & Scutellà, M. (2004), 'A multi-exchange heuristic for the single-source capacitated facility location problem', *Management Science* **50**(6), 749–760.

Ait-Alla, A., Quandt, M. & Lutjen, M. (2013), Aggregate installation planning of offshore wind farms, *in* 'Proceedings of the 7th International conference on communications and information technology'.

Aksen, D., Kaya, O., Salman, F. & Akça, Y. (2012), 'Selective and periodic inventory routing problem for waste vegetable oil collection', *Optimization Letters* **6**(6), 1063–1080.

Anaya-Arenas, A. M., Chabot, T., Renaud, J. & Ruiz, A. (2016), 'Biomedical sample transportation in the province of quebec: a case study', *International Journal of Production Research* **54**(2), 602–615.

Andersson, H., Hoff, A., Christiansen, M., Hasle, G. & Løkketangen, A. (2010), 'Industrial aspects and literature survey: Combined inventory management and routing', *Computers & Operations Research* **37**(9), 1515 – 1536.
URL: *http://www.sciencedirect.com/science/article/pii/S0305054809002962*

Anily, S. & Federgruen, A. (1990), 'One warehouse multiple retailer systems with vehicle routing costs', *Management Science* **36**(1), 92–114.

Anily, S. & Federgruen, A. (1993), 'Two-echelon distribution systems with vehicle routing costs and central inventories', *Operations Research* **41**(1), 37–47.

Aráoz, J., Fernández, E. & Meza, O. (2009), 'Solving the prize-collecting rural postman problem', *European Journal of Operational Research* **196**(3), 886–896.

Archetti, C., Bertazzi, L., Hertz, A. & Speranza, M. G. (2012), 'A hybrid heuristic for an inventory routing problem', *INFORMS Journal on Computing* **24**(1), 101–116.

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

Archetti, C., Boland, N. & Speranza, M. G. (2017), 'A matheuristic for the multivehicle inventory routing problem', *INFORMS Journal on Computing* **29**(3), 377–387.

Archetti, C. & Speranza, M.-G. (2014), 'A survey on matheuristics for routing problems', *EURO Journal on Computational Optimization* **2**(4), 223–246.

Archetti, C., Speranza, M. G. & Vigo, D. (2014), Chapter 10: Vehicle routing problems with profits, *in* 'Vehicle Routing: Problems, Methods, and Applications, Second Edition', SIAM, pp. 273–297.

Avella, P., Boccia, M. & Wolsey, L. (2015), 'Single-item reformulations for a vendor managed inventory routing problem: Computational experience with benchmark instances', *Networks* **65**(2), 129–138.

Avella, P., Boccia, M. & Wolsey, L. (2018), 'Single-period cutting planes for inventory routing problems', *Transportation Science* **52**(3), 497–508.

Azi, N., Gendreau, M. & Potvin, J.-Y. (2007), 'An exact algorithm for a single-vehicle routing problem with time windows and multiple routes', *European journal of operational research* **178**(3), 755–766.

Azi, N., Gendreau, M. & Potvin, J.-Y. (2010), 'An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles', *European Journal of Operational Research* **202**(3), 756–763.

Azi, N., Gendreau, M. & Potvin, J.-Y. (2014), 'An adaptive large neighborhood search for a vehicle routing problem with multiple routes', *Computers & Operations Research* **41**(1), 167–173.

Baldacci, R., Mingozzi, A. & Roberti, R. (2011), 'New route relaxation and pricing strategies for the vehicle routing problem', *Operations Research* **59**(5), 1269–1283.

Baldacci, R., Mingozzi, A. & Roberti, R. (2012), 'New state-space relaxations for solving the traveling salesman problem with time windows', *INFORMS Journal on Computing* **24**(3), 356–371.

Barahona, F. & Grötschel, M. (1986), 'On the cycle polytope of a binary matroid', *Journal of Combinatorial Theory, Series B* **40**(1), 40–62.

Bard, J. & Nananukul, N. (2010), 'A branch-and-price algorithm for an integrated production and inventory routing problem', *Computers & Operations Research* **37**(12), 2202–2217.

Battarra, M., Monaci, M. & Vigo, D. (2009), 'An adaptive guidance approach for the heuristic solution of a minimum multiple trip vehicle routing problem', *Computers & Operations Research* **36**(11), 3041–3050.

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

Bertazzi, L. (2008), 'Analysis of direct shipping policies in an inventory-routing problem with discrete shipping times', *Management Science* **54**(4), 748–762.

Bertazzi, L., Bosco, A., Guerriero, F. & Laganà, D. (2013), 'A stochastic inventory routing problem with stock-out', *Transportation Research Part C: Emerging Technologies* **27**, 89–107.

Bertazzi, L., Bosco, A. & Laganà, D. (2016), 'Min–max exact and heuristic policies for a two-echelon supply chain with inventory and transportation procurement decisions', *Transportation Research Part E: Logistics and Transportation Review* **93**, 57–70.

Bertazzi, L., Chua, G. A., Laganá, D. & Paradiso, R. (2019), 'Sparse routing for the inventory routing problem', *Submitted to Operations Research* .

Bertazzi, L., Laganà, D., Ohlmann, J. W. & Paradiso, R. (2019), 'An exact approach for cyclic inbound inventory routing in a level production system', *European Journal of Operational Research* .

Bertazzi, L. & Speranza, M. G. (2012*a*), 'Inventory routing problems: an introduction', *EURO Journal on Transportation and Logistics* **1**(4), 307–326.

Bertazzi, L. & Speranza, M. G. (2012*b*), Matheuristics for inventory routing problems, *in* 'Hybrid algorithms for service, computing and manufacturing systems: Routing and scheduling solutions', IGI Global, pp. 1–14.

Bertazzi, L. & Speranza, M. G. (2013), 'Inventory routing problems with multiple customers', *EURO Journal on Transportation and Logistics* **2**(3), 255–275.

Bramel, J., Coffman, E. G., Shor, P. W. & Simchi-Levi, D. (1992), 'Probabilistic analysis of the capacitated vehicle routing problem with unsplit demands', *Operations Research* **40**(6), 1095–1106.

Bramel, J. & Simchi-Levi, D. (1995), 'A location based heuristic for general routing problems', *Operations Research* **43**(4), 649–660.

Campbell, A. & Salvesbergh, M. (2004), 'A decomposition approach for the inventory routing problem', *Transportation Science* **38**(4), 488–502.

Cattaruzza, D., Absi, N. & Feillet, D. (2016*a*), 'The multi-trip vehicle routing problem with time windows and release dates', *Transportation Science* **50**(2), 676–693.

Cattaruzza, D., Absi, N. & Feillet, D. (2016*b*), 'Vehicle routing problems with multiple trips', *4OR* **14**(3), 223–259.

Cattaruzza, D., Absi, N., Feillet, D. & González-Feliu, J. (2017), 'Vehicle routing problems for city logistics', *EURO Journal on Transportation and Logistics* **6**(1), 51–79.

Cattaruzza, D., Absi, N., Feillet, D. & Vigo, D. (2014), 'An iterated local search for the multi-commodity multi-trip vehicle routing problem with time windows', *Computers & Operations Research* **51**, 257 – 267.

Cheng, C., Adulyasak, Y. & Rousseau, L. M. (2018), Formulations and exact algorithms for drone routing problem, Technical Report CIRRELT-2018-31, CIRRELT.

Chitsaz, M., Cordeau, J.-F. & Jans, R. (2019), 'A unified decomposition matheuristic for assembly, production, and inventory routing', *INFORMS Journal on Computing* **31**(1), 134–152.

Chitsaz, M., Divsalar, A. & Vansteenwegen, P. (2016), 'A two-phase algorithm for the cyclic inventory routing problem', *European Journal of Operational Research* **254**(2), 410–426.

Chopra, S. & Meindl, P. (2007), Supply chain management. strategy, planning & operation, *in* 'Das summa summarum des management', Springer, pp. 265–275.

Christopher, M. (2016), *Logistics & supply chain management*, Pearson UK.

Chuah, K. H. & Yingling, J. (2005), 'Routing for a just-in-time supply pickup and delivery system', *Transportation Science* **39**(3), 328–339.

Coelho, L., Cordeau, J.-F. & Laporte, G. (2012), 'Consistency in multi-vehicle inventory-routing', *Transportation Research Part C: Emerging Technologies* **24**, 270–287.

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

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

Coelho, L., Cordeau, J.-F. & Laporte, G. (2014), 'Heuristics for dynamic and stochastic inventory-routing', *Computers & Operations Research* **52**, 55–67.

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

Coelho, L. & Laporte, G. (2013*b*), 'The exact solution of several classes of inventory-routing problems', *Computers & Operations Research* **40**(2), 558–565.

Coelho, L. & Laporte, G. (2014), 'Improved solutions for inventory-routing problems through valid inequalities and input ordering', *International Journal of Production Economics* **155**, 391–397.

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

Desaulniers, G., Rakke, J. G. & Coelho, L. (2016), 'A branch-price-and-cut algorithm for the inventory-routing problem', *Transportation Science* **50**(3), 1060–1076.

Dong, Z. & Turnquist, M. (2015), 'Combining service frequency and vehicle routing for managing supplier shipments', *Transportation Research Part E: Logistics and Transportation Review* **79**, 231–243.

Dorling, K., Heinrichs, J., Messier, G. G. & Magierowski, S. (2017), 'Vehicle Routing Problems for Drone Delivery', *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **47**(1), 70–85.

Dror, M., Laporte, G. & Trudeau, P. (1994), 'Vehicle routing with split deliveries', *Discrete Applied Mathematics* **50**(3), 239–254.

Ekici, A., Özener, O. & Kuyzu, G. (2015), 'Cyclic delivery schedules for an inventory routing problem', *Transportation Science* **49**(4), 817–829.

Federgruen, A. & Zheng, Y.-S. (1992), 'The joint replenishment problem with general joint cost structures', *Operations Research* **40**(2), 384–403.

Fischetti, M., Gangini, L., Paradiso, R. & Roberti, R. (2019), 'Optimization models for the installation planning of offshore wind farms', *Submitted to Omega* .

Fischetti, M. & Pisinger, D. (2019), 'Mathematical optimization and algorithms for offshore wind farm design: An overview', *Business & Information Systems Engineering* **61**(4), 469–485.

Fleischmann, B. (1990), 'The vehicle routing problem with multiple use of vehicles', *Fachbereich Wirtschaftswissenschaften, Universität Hamburg* .

Francis, P. & Smilowitz, K. (2006), 'The period vehicle routing problem with service choice', *Transportation Science* **40**(4), 439–454.

Gallego, G. & Simchi-Levi, D. (1990), 'On the effectiveness of direct shipping strategy for the one-warehouse multi-retailer r-systems', *Management Science* **36**(2), 240–243.

Gehring, H. & Homberger, J. (1999), A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows, *in* 'Proceedings of EUROGEN99', Vol. 2, pp. 57–64.

Ghiani, G., Laporte, G. & Musmanno, R. (2013), *Introduction to logistics systems management*, John Wiley & Sons.

Golden, B. (1984), 'Analysis of a large scale vehicle routing problem with an inventory component', *Large scale systems* **7**, 181–190.

Groër, C., Golden, B. & Wasil, E. (2010), 'A library of local search heuristics for the vehicle routing problem', *Mathematical Programming Computation* **2**, 79–101.

Hein, F. & Almeder, C. (2016), 'Quantitative insights into the integrated supply vehicle routing and production planning problem', *International Journal of Production Economics* **177**, 66–76.

Hernandez, F., Feillet, D., Giroudeau, R. & Naud, O. (2014), 'A new exact algorithm to solve the multi-trip vehicle routing problem with time windows and limited duration', *4OR* **12**(3), 235–259.

Hernandez, F., Feillet, D., Giroudeau, R. & Naud, O. (2016), 'Branch-and-price algorithms for the solution of the multi-trip vehicle routing problem with time windows', *Eur. J. Oper. Res.* **249**(2), 551–559.

Irawan, C., Jones, D. & Ouelhadj, D. (2017), 'Bi-objective optimisation model for installation scheduling in offshore wind farms', *Computers & Operations Research* **78**, 393–407.

Irawan, C., Ouelhadj, D., Jones, D., Stålhane, M. & Sperstad, I. (2017), 'Optimisation of maintenance routing and scheduling for offshore wind farms', *European Journal of Operational Research* **256**(1), 76–89.

Jepsen, M., Petersen, B., Spoorendonk, S. & Pisinger, D. (2008), 'Subset-row inequalities applied to the vehicle-routing problem with time windows', *Operations Research* **56**(2), 497–511.

Kleywegt, A., Nori, V. & Savelsbergh, M. (2002), 'The stochastic inventory routing problem with direct deliveries', *Transportation Science* **36**(1), 94–118.

Kleywegt, A., Nori, V. & Savelsbergh, M. (2004), 'Dynamic programming approximations for a stochastic inventory routing problem', *Transportation Science* **38**(1), 42–70.

Liker, J. K. (2005), *The toyota way*, Esensi.

Lim, A., Zhang, Z. & Qin, H. (2017), 'Pickup and delivery service with manpower planning in hong kong public hospitals', *Transportation Science* **51**(2), 688–705.

Macedo, R., Alves, C., de Carvalho, J. V., Clautiaux, F. & Hanafi, S. (2011), 'Solving the vehicle routing problem with time windows and multiple routes exactly using a pseudo-polynomial model', *European Journal of Operational Research* **214**(3), 536–545.

Martínez, L. & Amaya, C.-A. (2013), 'A vehicle routing problem with multi-trips and time windows for circular items', *Journal of the Operational Research Society* **64**(11), 1630–1643.

Nagarajan, V. & Shi, C. (2016), 'Approximation algorithms for inventory problems with submodular or routing costs', *Mathematical Programming* **160**(1-2), 225–244.

Natarajarathinam, M., Stacey, J. & Sox, C. (2012), 'Near-optimal heuristics and managerial insights for the storage constrained, inbound inventory routing problem', *International Journal of Physical Distribution & Logistics Management* **42**(2), 152–173.

Nguyen, P., Crainic, T. & Toulouse, M. (2013), 'A tabu search for time-dependent multi-zone multi-trip vehicle routing problem with time windows', *European Journal of Operational Research* **231**(1), 43–56.

Nguyen, P., Crainic, T. & Toulouse, M. (2017), 'Multi-trip pickup and delivery problem with time windows and synchronization', *Annals of Operations Research* **253**(2), 899–934.

Ohlmann, J., Fry, M. & Thomas, B. (2008), 'Route design for lean production systems', *Transportation Science* **42**(3), 352–370.

Paradiso, R., Roberti, R., Laganá, D. & Dullaert, W. (2019), 'An exact solution framework for multi-trip vehicle routing problems with time windows', *Operations Research* **forthcoming**.

Picard, J.-C. & Ratliff, H. D. (1973), 'A graph-theoretic equivalence for integer programs', *Operations Research* **21**(1), 261–269.

Qin, H., Ming, W., Zhang, Z., Xie, Y. & Lim, A. (2015), 'A tabu search algorithm for the multi-period inspector scheduling problem', *Computers & Operations Research* **59**, 78–93.

Raa, B. (2015), 'Fleet optimization for cyclic inventory routing problems', *International Journal of Production Economics* **160**, 172–181.

Raa, B. & Aghezzaf, E.-H. (2008), 'Designing distribution patterns for long-term inventory routing with constant demand rates', *International Journal of Production Economics* **112**(1), 255–263.

Raa, B. & Aghezzaf, E.-H. (2009), 'A practical solution approach for the cyclic inventory routing problem', *European Journal of Operational Research* **192**(2), 429–441.

Rusdiansyah, A. & Tsao, D. (2005), 'An integrated model of the periodic delivery problems for vending-machine supply chains', *Journal of Food Engineering* **70**(3), 421–434.

Saidur, R., Rahim, N., Islam, M. & Solangi, K. (2011), 'Environmental impact of wind energy', *Renewable and Sustainable Energy Reviews* **15**(5), 2423–2430.

Satoglu, S. & Sahin, I. (2013), 'Design of a just-in-time periodic material supply system for the assembly lines and an application in electronics industry', *The International Journal of Advanced Manufacturing Technology* **65**(1-4), 319–332.

Schmid, V., Doerner, K. F. & Laporte, G. (2013), 'Rich routing problems arising in supply chain management', *European Journal of Operational Research* **224**(3), 435–448.

Scholz-Reiter, B., Heger, J., Lütjen, M. & Schweizer, A. (2011), 'A MILP for installation scheduling of offshore wind farms', *International Journal of Mathematical Models and Methods in Applied Sciences* **5**(2), 371–378.

Scholz-Reiter, B., Karimi, H., Lütjen, M., Heger, J. & Schweizer, A. (2011), Towards a heuristic for scheduling offshore installation processes, *in* 'Proceedings of the 24th International Congress on Condition Monitoring (COMADEM)', pp. 999–1008.

Simchi-Levi, D., Kaminsky, P., Simchi-Levi, E. & Shankar, R. (2008), *Designing and managing the supply chain: concepts, strategies and case studies*, Tata McGraw-Hill Education.

Snyder, B. & Kaiser, M. (2009), 'Ecological and economic cost-benefit analysis of offshore wind energy', *Renewable Energy* **34**(6), 1567–1578.

Solomon, M. M. (1987), 'Algorithms for the vehicle routing and scheduling problems with time window constraints', *Operations research* **35**(2), 254–265.

Solyalı, O., Cordeau, J.-F. & Laporte, G. (2012), 'Robust inventory routing under demand uncertainty', *Transportation Science* **46**(3), 327–340.

Solyalı, O. & Süral, H. (2008), 'A single supplier–single retailer system with an order-up-to level inventory policy', *Operations Research Letters* **36**(5), 543–546.

Stacey, J., Natarajarathinam, M. & Sox, C. (2007), 'The storage constrained, inbound inventory routing problem', *International Journal of Physical Distribution & Logistics Management* **37**(6), 484–500.

Stadtler, H. (2015), Supply chain management: An overview, *in* 'Supply chain management and advanced planning', Springer, pp. 3–28.

Tilk, C. & Irnich, S. (2016), 'Dynamic programming for the minimum tour duration problem', *Transportation Science* **51**(2), 549–565.

Vaidyanathan, B., Matson, J., Miller, D. & Matson, J. (1999), 'A capacitated vehicle routing problem for just-in-time delivery', *IIE Transactions* **31**(11), 1083–1092.

Vansteenwegen, P. & Mateo, M. (2014), 'An iterated local search algorithm for the single-vehicle cyclic inventory routing problem', *European Journal of Operational Research* **237**(3), 802–813.

Vansteenwegen, P., Souffriau, W., Berghe, G. V. & Van Oudheusden, D. (2009), 'Iterated local search for the team orienteering problem with time windows', *Computers & Operations Research* **36**(12), 3281–3290.

Wang, Z., Liang, W. & Hu, X. (2014), 'A metaheuristic based on a pool of routes for the vehicle routing problem with multiple trips and time windows', *Journal of the Operational Research Society* **65**(1), 37–48.

Ward, J., Oca, A., Zimmerman, M., Acar, K., Balika, S. & Sun, Y. (2019), '2019 State of logistics report, AT Kearney'.

Zenker, M., Emde, S. & Boysen, N. (2016), 'Cyclic inventory routing in a line-shaped network', *European Journal of Operational Research* **250**(1), 164–178.

Zhao, Q.-H., Wang, S.-Y. & Lai, K. K. (2007), 'A partition approach to the inventory/routing problem', *European Journal of Operational Research* **177**(2), 786–802.

Zhong, Y. & Aghezzaf, E.-H. (2011), 'Combining dc-programming and steepest-descent to solve the single-vehicle inventory routing problem', *Computers & Industrial Engineering* **61**(2), 313–321.

Zhong, Y. & Aghezzaf, E.-H. (2012), 'Effective local search approaches for the single-vehicle cyclic inventory routing problem', *International Journal of Services Operations and Informatics* **7**(4), 260–279.