UNIVERSITY OF BERGAMO

School of Doctoral Studies

Doctoral Degree in Analytics for Economics and Business

XXIX Cycle

SSD: MAT/09

TITLE
Consolidation and coordination of routes in urban distribution

Advisor

Chiar.ma Prof.ssa M. Grazia Speranza

Doctoral Thesis

Andrea MOR

Student ID 1039536

Academic year 2017/18

Abstract

In this dissertation, two relevant aspects of routing in the urban environment are studied. In the first part of the thesis, the implications of considering release dates when planning for delivery from distribution centers are discussed. The routing problems with release dates are contextualized in the class of routing problems, and in particular, in the class of routing problems in which timing decisions must be considered. The Traveling Salesman Problem with release dates and completion time minimization is studied. Properties are introduced for the problem and a formulation is proposed. Two variants of a heuristic algorithm are tested against the optimal solution and shown to provide high quality results. The benefits of considering release dates are assessed by comparing the results with those obtained disregarding the release dates and delivering the parcels as soon as they arrive to the distribution center. The study on routing problems with release dates is then expanded to consider the stochastic and dynamic nature of the release dates. A reoptimization technique is proposed to tackle the dynamic aspect of the problem. Three reoptimization policies are proposed, with increasing reoptimization frequency, together with two models for the solution of the problem. The first is a stochastic model, considering the entire probabilistic information available for the release dates, and the second is a deterministic model, where a point estimation is used. The stochastic model is shown to perform better than the deterministic model, at the expense of the computational time required to evaluate any of the solutions explored.

The second part of the thesis is focused on the management of the loading and unloading areas in the city center. Urban distribution requires vehicles to temporarily stop to perform the last leg of the delivery by foot. If a spot is not available, vehicles resort to double parking which is a known cause of road congestion. Two booking management systems and the arising routing problems are presented. The solutions provided by the two systems are compared with the current state of the distribution.

Acknowledgements

I owe my gratitude, and perhaps more, to many people, but I am not very good at being thankful on paper, so I will be brief.

I wish to thank my supervisor Prof. M. Grazia Speranza, who patiently guided me in these years, and Prof. Claudia Archetti, for her patience and support.

I also wish to thank Prof. Dominique Feillet, who co-authored two fo the works presented in this thesis and hosted me in the Ecole de Mines the Saint-Etienne, and Prof. Harilaos N. Psaraftis, who hosted me at the Technical University of Denmark. Thanks to Prof. José M. Viegas for sharing his experience on real world problems.

Finally, I wish to thank all the members of the OR group at the University of Brescia for welcoming me so warmly and for making even the most relaxed moment an occasion for learning and sharing.

Contents

Abstract **i**

Acknowledgements **iii**

1 Introduction **3**

1.1 Context 3

1.2 Contributions 5

2 Vehicle routing problems over time: a survey **9**

2.1 Abstract 9

2.2 Introduction 10

2.3 Classes of vehicle routing problems 12

2.4 Routing over time 16

2.4.1 Periodic routing problems 17

2.4.2 Inventory routing problems 21

2.4.3 Vehicle routing problems with release dates 28

2.4.4 Multi-trip vehicle routing problems 31

2.5 Conclusions and future research directions 34

3 An Iterated Local Search for the Traveling Salesman Problem with release dates 37

3.1 Abstract 37

3.2 Introduction 38

3.3 Literature review 41

3.4 The TSP-rd(time) 43

3.4.1 Properties 44

3.4.2 Approximation algorithm 47

3.4.3 Mathematical formulation 48

3.5 An iterated local search algorithm for the TSP-rd(time) 51

3.5.1 Initial solution 52

3.5.2 Destroy-and-repair 53

3.5.3 Local Search 55

3.6 Computational experiments 57

3.6.1 Instances 57

3.6.2 Results 60

3.7 Conclusions 79

4 Dynamic traveling salesman problem with stochastic release dates 83

4.1 Abstract 83

4.2 Introduction . . . 84

4.3 The DTSP-srd . . . 88

4.4 Solution approach to the dynamic problem . . . 90

4.5 Optimization models . . . 92

4.5.1 Stochastic model . . . 93

4.5.2 Deterministic model . . . 97

4.5.3 Illustrative example . . . 97

4.5.4 A heuristic algorithm . . . 101

4.6 Instance generation . . . 103

4.6.1 Input instances . . . 103

4.6.2 Initial customer partitioning . . . 104

4.6.3 Release dates generation . . . 105

4.6.4 Release dates updating . . . 107

4.6.5 Parameters for the instance generation . . . 109

4.7 Computational experiments . . . 110

4.7.1 Preliminary experiments . . . 112

4.7.2 Final results . . . 115

4.8 Conclusions . . . 122

4.9 Appendix: Results . . . 125

5 Booking of loading/unloading areas **129**

5.1 Abstract . . . 129

5.2 Introduction 130

5.3 Literature review 132

5.4 Routing with loading/unloading areas 136

5.5 Booking of L/U areas 137

5.6 Sequential booking 139

5.6.1 The Traveling Salesman Problem with multiple Time Windows . 140

5.7 Centralized booking 142

5.7.1 The multiple Traveling Salesman Problem with node scheduling 143

5.8 Computational experiments 146

5.8.1 Instance generation 146

5.8.2 Solution method 148

5.8.3 Results 150

5.9 Conclusions 151

5.10 Acknowledgements 154

6 Conclusions **155**

List of Figures

2.1 Decisions of the classical VRPs. 13

2.2 Periodic Routing Problem: an example. 18

2.3 Inventory Routing Problem: an example. 22

2.4 Vehicle Routing Problem with release dates: an example. 29

2.5 Multi-trip Vehicle Routing Problem: an example. 32

3.1 Property 3: an example. 45

3.2 Property 4: an example. 46

3.3 Theorem 3.4.1: an example. 48

3.4 Plots of the percentage gap of the current best solution over the final solution found in the two hours run of HeuTSPrd on the instances with $n = 100$ and $\beta = 3$. 75

4.1 The RDW policy. 92

4.2 An example. 97

4.3 The heuristic scheme from Archetti et al. (2018). 102

4.4 The effect of different values of β on the release dates. $\beta = 0.5, 1, 1.5$ from top to bottom. 111

4.5 The evolution of the release dates when $\delta = 1$. 112

5.1 The layout of the instances. 147

5.2 The step function of the occupancy leval of a L/U area with the full booking intervals highlighted. 149

5.3 The function of the occupancy level of a L/U area with the overbooked interval highlighted. 150

5.4 A comparison of the routing costs for the instance with $|U| = 2$ and $|K| = 6$. 152

List of Tables

3.1 The symmetric instances selected from the TSPLIB, by instance size. . . 59

3.2 Number of instances solved to optimality. 61

3.3 Results on the $n = 20$ instances for different values of the MILP time limit. 62

3.4 Results on the $n = 50$ instances for different values of the MILP time limit. 63

3.5 Results on the $n = 100$ instances for different values of the MILP time limit. 64

3.6 Results on the $n = 50$ instances for different values of ρ. MILP time limit is set to 5 seconds for all runs. 65

3.7 Results on the $n = 100$ instances for different values of ρ. MILP time limit is set to 20 seconds for all runs. 66

3.8 Comparison with optimal solutions for instances with $n = 10$. 69

3.9 Comparison with optimal solutions for instances with $n = 15$. 70

3.10 Comparison with optimal solutions for instances with $n = 20$. 71

3.11 Growth of the average optimal value with respect to the value of β. . . 72

3.12 Comparison of MathTSPrd and HeuTSPrd on instances with $n = 50$. . 73

3.13 Comparison of MathTSPrd and HeuTSPrd on instances with $n = 100$. 74

3.14 Comparison of MathTSPrd and HeuTSPrd on the instances derived from TSPLIB. 77

3.15 Comparison of MathTSPrd and HeuTSPrd on the instances derived from TSPLIB. 78

3.16 Summary of results on the instances generated from TSPLIB. 81

3.17 Comparison MathTSPrd and HeuTSPrd on the asymmetric instances derived from TSPLIB. 82

4.1 Expected values of the solutions. 100

4.2 Results in the four scenarios¡. 100

4.3 The results of the tests for each $T_{\max}$. 113

4.4 Results on the average number of ILS iterations for each model in the instance. 114

4.5 Time in seconds to perform one ILS iteration. 115

4.6 Average percentage gap from the best solution. 117

4.7 Detailed results for the average percentage gap from the best solution. . 118

4.8 Average time required for each simulation (in minutes). 120

4.9 Average number of reoptimizations. 121

4.10 Average number of routes. 122

4.11 Average time (in minutes) over the number of routes. 123

4.12 Average percentage waiting time. 124

4.13 Detailed results for the total time required for each simulation (in minutes). 125

4.14 Detailed results for the total time over the number of routes (in minutes). 125

4.15 Detailed results for number of reoptimizations. 126

4.16 Detailed results for the number of routes. 126

4.17 Detailed results for the percentage waiting time. 127

5.1 The results of the solution of the mTSPns different size of the instance. 151

5.2 A comparison of the routing costs for the instance with $|U| = 2$ and $|K| = 6$. 152

Chapter 1

Introduction

1.1 Context

Although the numbers vary with the source, it is of wide consensus that the percentage of the population living in urban areas is increasing. This trend has the effect of centralizing the expression of transportation needs, both of people and of goods, at rate unmatched by the transportation infrastructure. This phenomenon has been further worsened with the advent of e-commerce and the wide adoption of home delivery, which have caused a change in the paradigm of goods distribution from customers going to goods to goods going to customers. It is therefore becoming increasingly important to efficiently satisfy the transportation requests in the urban environment, to minimize the negative effects on traffic, pollution, and, broadly, to improve the quality of the service provided to the citizens, the livability of the urban environment and the quality of life in general. The broad scientific field that studies this subject is called *city logistics*.

Various approaches to the problem have been proposed and studied on the literature of operations research. With respect to the reduction of the emissions and the improvement of the distribution in terms of vehicle routing and loading, one of the most valued approaches consists in the design of multi-echelon distribution networks and the implementation of distribution centers, allowing for a better consolidation of the parcels before the delivery is performed in the city center.

Improvements in the road network use can also be envisioned with the aid of technological advancement, especially in two prominent directions. A better use of the capacity of the current network, resulting in more vehicles being able to travel on the same road, can be predicted with the advent of automated and connected vehicles. The adoption of such vehicles would allow, for instance, a reduction in safety distances and potentially even the dismissal of traffic lights, resulting in a higher vehicle density on the road, a higher congestion threshold, and reduced probability of disruptions caused by accidents. A reduction in the need of road freight transportation, at least for small and light packages, could be achieved with the introduction of unmanned aerial vehicles to perform the last leg of the delivery.

The ultimate cause of the pollution resulting from freight transportation is the use of fossil fuel powered vehicles. A sensible reduction in the air pollution has been made available with the introduction of alternative fuel vehicles and even more, also in terms of noise pollution, with electric powered vehicles. The former, however, suffer from a general lack of refueling stations and the latter from slow recharging times and short traveling range. As the adoption of these vehicles gets wider, however, these downsides could be mitigated by an increased economic viability of this solution and technological advancements.

Another important aspect in the strive to improve the efficiency of network usage, with perhaps more immediate effects and less investments required than the other approaches discussed, is represented by the trends in road use over time, characterized by daily peaks at rush hours. One way to discourage freight transportation in such hours is represented by dynamic road pricing based on traffic level. Another way to smoothen transportation demand in the city center through the day, both spatially and temporally, is the adoption of limited entrance zones.

1.2 Contributions

In this dissertation, two relevant aspects of the distribution in city centers are studied.

As above introduced, an important role in the reduction of the negative effects and the overall improvement of the efficiency of the distribution in the city centers is parcels consolidation. One way this could be achieved is through the introduction of distribution centers where the parcels for customers are delivered and consolidated before delivery. The advantages of the introduction of distribution centers is twofold. The first benefit is the reduced circulation of large long haul trucks in the city center, in favor of small short range vehicles, often powered by electricity. The second benefit is to allow for an overall reduction of the number of vehicles required to perform the distribution, as parcels can be batched together and delivered in a more efficient way. An important role in the viability of distribution centers is represented by the efficiency in the processing and delivery of the parcels. In this sense, the implementation of distribution centers must not represent a bottleneck in the distribution in the city or the cause of a degradation in the quality of service perceived by the customers.

A key aspect to better plan the deliveries in the city from a distribution center is to take into account the time at which the parcels reach the distribution centers and are ready to be delivered. Such time is called the release date of the parcel. The study of the implication of considering release dates for the parcels of customers when planning the routing for the delivery is the subject of Chapters 2, 3, and 4.

In Chapter 2 the subject of vehicle routing problems (VRP) with release dates is contextualized in the literature of routing problems. A classification of the VRPs is proposed based on the decisions to be taken when planning for the distribution. After presenting various classes of routing problems based on this criterion, the class of problems where timing decisions must be considered in addition to the more traditional characteristics of VRPs is discussed. This class of routing problems includes the periodic routing problem, the inventory routing problem, the routing problems with release dates, and the multi-trip routing problems. A review of the literature on these problems is presented.

To better understand the effects of considering release dates in routing problems when planning for the distribution, Chapter 3 focuses on the study of the Traveling Salesman Problem (TSP) with release dates, where a single uncapacitated vehicle delivers goods arriving at the distribution center over time, seeking the minimization of the total time required for the completion of the distribution, given by the sum of the travel times for the deliveries to customers and waiting times at the depot for the parcels to be delivered to the distribution center. Properties are introduced for the problem and a mixed integer formulation is proposed. A heuristic is presented based on an iterated local search where the perturbation is performed by means of a destroy-and-repair method. Two alternative repair operators, one simple and fast and the other based on a mathematical programming model, are proposed, which give rise to two variants of the heuristic. The mathematical formulation is used to find the optimal solution on instances with up to 20 customers, built from benchmark instances for the classical TSP. The two variants are shown to provide high quality solutions compared to the optimal solutions. Further comparison is reported on larger instances and the benefits of considering release dates is shown by comparing the results of the heuristic with the solution obtained by disregarding known arrival times and delivering the parcels as soon as they reach the distribution center.

In Chapter 4, the study of the TSP with release dates is expanded to consider the stochastic and dynamic nature of the release dates as the representation of the arrival times of vehicles delivering parcels to the distribution center, subject to delays and uncertainty in travel times but also able, with the aid of connected devices, to provide updates on the estimated time of arrival. A reoptimization technique is proposed to tackle the dynamic aspect of the problem. Three reoptimization policies are introduced, with increasing reoptimization frequency. The benefits and drawbacks of considering stochastic release dates are highlighted by comparing the results of the stochastic model proposed for the problem, considering the entire stochastic information for the release dates, and a deterministic model, where a point estimation for the release dates is used.

In Chapter 5 a different problem arising in the context of city logistics is considered. As above discussed, one of the issues in the distribution in urban areas is the inefficient use of the transportation network. Most literature discussing city logistics, however, focuses solely on the use that vehicles make of the network when traveling, disregarding the moments in which the vehicles must stop to load, unload and for the delivery to take place and therefore assuming that a parking spot is always available or that the vehicle resorts to double parking if none is found. In practice, however, space is a scarce resource for the allocation of loading and unloading areas as it is for the road infrastructure. Not considering the effects of double parking on traffic when discussing the distribution problems of urban areas can cause more issues that the ones that were aimed to be solved.

Conclusions and future research are presented in Chapter 6.

Chapter 2

Vehicle routing problems over time: a survey

This chapter is based on the homonymous article currently submitted for publication. The work therein has been performed by the author in collaboration with Prof. M. Grazia Speranza.

2.1 Abstract

In this chapter the literature on vehicle routing problems (VRPs) that include a timing decision in addition to the more classical characteristics of this class of problems, is reviewed. After discussing the different kinds of decisions taken in different classes of vehicle routing problems, the class where decisions have to be taken about when the routes start from the depot is considered. This class of problems, that we call VRPs over time, includes the periodic vehicle routing problems, the inventory routing problems, the routing problems with release dates, the multi-trip routing problems.

2.2 Introduction

Vehicle routing problems (VRPs) are among the most studied problems in the field of combinatorial optimization. Despite the long tradition, the literature on routing problems is expanding at a faster-than-ever pace (Eksioglu et al. (2009) reports an exponential growth with an annual growth rate of 6%), with more variants being introduced and faster and better solution methods being devised. The knowledge gained on these problems and the technological advancements have allowed researchers to tackle more complex variants, also narrowing in this way the gap between the scientific literature and real life applications. In this context, it is of great importance to have surveys and classifications of the scientific contributions that organize the literature and, on one hand help, understanding what has been achieved and, on the hand, what are the open and promising directions for future research.

Taxonomies and surveys on VRPs have been published over the years aimed at classifying the VRPs and overview recent evolutions and trends. A recent taxonomy of VRPs is provided in Braekers et al. (2016), updating and expanding the work of Eksioglu et al. (2009). The authors classify the papers based on five macro-categories: type of study, scenario characteristics, problem physical characteristics, information characteristics and data characteristics. A classification of routing problems that takes into account the management components of the problem is briefly introduced in Lahyani et al. (2015) for the rich VRPs. The authors consider the cases where routing decisions are taken together with inventory, location, driver scheduling and production and distribution planning.

One may think about the Traveling Salesman Problem (TSP) as the most basic problem of the class of the VRPs. In fact, in the VRPs a vertex is a depot and the other vertices are customers. Each customer has a demand and vehicles, with limited capacity, are available to serve the customers. The most basic problem of the class of the VRPs is considered to be the capacitated VRP, sometimes simply called the VRP, where a set of customers, each with a given demand, have to be served with a fleet of identical vehicles. Each vehicle performs exactly one route. The number of vehicles/routes is given. All routes start and end at a depot. The VRP consists in assigning customers to routes and in sequencing the customers in each route in such a way that the total routing cost is minimized. Many extensions and variants of the capacitated VRP have been studied with the largest number of VRPs focused on the assignment and sequencing decisions.

In this chapter, the VRPs are classified according to the additional, with respect to the assignment of customers to routes and the sequencing of customers in each route, decisions to be taken. Such additional decisions usually imply a substantial change in the mathematical programming formulation and the design of substantially different solution approaches. This classification is aimed at identifying the class of the *VRPs over time* for which we will overview the literature. In the VRPs over time, the decision about when a vehicle starts from the depot has to be taken. This implies that not all routes start at the same time from the depot. The VRPs over time include the periodic vehicle routing problems, the inventory routing problems, the routing problems with release dates, the multi-trip routing problems.

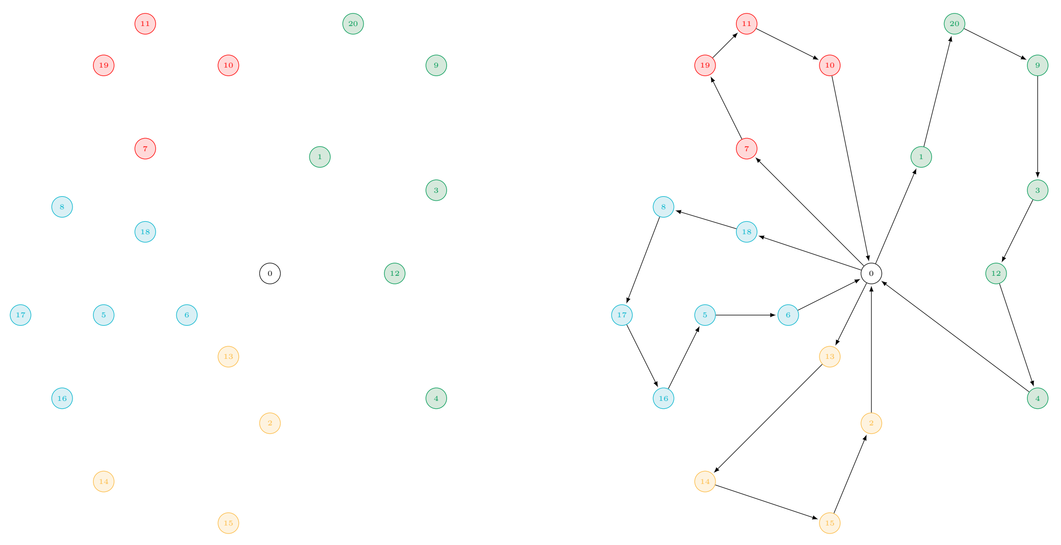
The chapter is organized as follows. In Section 2.3 we describe some of the classes of VRPs that arise when different decisions are considered. In Section 2.4 the class of VRPs over time is surveyed.

2.3 Classes of vehicle routing problems

In this section we structure the VRPs according to the kinds of decisions that have to be taken to solve them. We start from the most classical VRPs, where customers are assigned to vehicles and ordered. Then, we consider other classes of VRPs, such as the VRPs where the decision about which customers to serve (not necessarily all) has to be also taken, or the VRPs where the decision about the quantity (not necessarily total demand) to be served has to be taken. The main goal is to identify as a specific class, that of the VRPs over time, to which Section 2.4 is devoted. In the following, if not specified, for the sake of simplicity, we will refer to distribution problems where customers are delivery customers while in most cases we could also refer to collection problems where customers are pickup customers.

With which vehicle? In which order? The most classical VRPs include the Capacitated VRP (CVRP). In this problem, each vehicle starts and returns to the same depot. The number of vehicles available is given and coincides with the number of routes. It is understood that each vehicle performs exactly one route. We may say that a vehicle starts from the depot or, equivalently, that a route starts from the depot. All routes can start at the same time. All customers must be served. The demand of each customer is entirely served by one route. One single commodity is considered. Formally, the *basic CVRP*, as described in Toth and Vigo (2014), is defined on a directed graph $G = (V, A)$ where the set of vertices V is composed of the depot, vertex 0, and the set $N = \{1, \ldots, |N|\}$ of customers. The demand of customer $i \in N$ is indicated as q_i, expressing the weight or the volume of goods to be delivered. The set of vehicles available for the distribution is denoted by $K = \{1, \ldots, |K|\}$. The vehicles are homogeneous and have a capacity Q. The cost of traveling from i to j is denoted by c_{ij}. Routes must be created, that is, customers must be assigned to vehicles and the customers assigned to each vehicle must be ordered. The minimization of the routing cost is sought while satisfying the following constraints: all customers are served, each vehicle departing from the depot serves a subset of customers and returns to the depot, each vehicle performs at most one route, the total demand of the customers served in the same route does not exceed the vehicle capacity, and no customer is visited more than once.

The VRPs in this class, that we call the *classical VRPs*, include all the problems where the decisions to be taken concern the assignment of customers to routes and the sequencing of the customers assigned to each route. These problems extend the CVRP with additional constraints or features on the vehicles or on the customers, and include, for example, the CVRP with time windows and several variants of the CVRP with pick-ups and deliveries.



With which vehicle? In which order?

Figure 2.1: Decisions of the classical VRPs.

Which customers to serve? In the classical VRPs, all customers must be served. This implies that a decision about which customers to serve has already been taken at an earlier stage. In general, the customers to be served in a day may be decided on the basis of convenience or urgency. Also, customers, offered by other companies or web sites, may be accepted or not. When the decision about which customers to serve is considered, we have the class of the *VRPs with profits*. Examples of VRPs with profits are the orienteering problem and the team orienteering problem (see Archetti et al. (2014b) for a review).

How much to a customer? In the classical VRPs all customers are visited once. This implies that the demand of a customer does not exceed the capacity of a vehicle. No decision about the quantity to deliver to a customer has to be taken as the entire demand is served by one vehicle. While this is often a realistic assumption, it may be beneficial to visit customers more than once, even when the demand does not exceed the capacity of a vehicle.

The Split Delivery Vehicle Routing Problem (SDVRP) is the CVRP where the quantity to be delivered to a customer by a vehicle has to be decided, which implies that a customer may be visited by multiple vehicles. A survey on the classes of *VRPs with split deliveries* can be found in Archetti and Speranza (2012).

What commodity? Only one commodity is considered in the classical VRPs. This does not mean that in the real-life applications of the models only one commodity is delivered but that one modeled commodity is sufficient to model the whole set of real commodities. This is possible under the assumption that each vehicle can deliver anything and the depot contains everything that has to be distributed to customers. The total demand of the customers, possibly of multiple real commodities, and the capacity of the vehicles are measured with the same unit of measure, typically weight or volume.

There are interesting relevant vehicle routing applications that can be modeled only by explicitly considering multiple commodities. This happens, for example, when the vehicles have multiple compartments, each with limited capacity, to keep multiple commodities separated (see Yahyaoui et al. (2018) for a recent review of the literature). Also, when different commodities are available for the distribution in different locations, multiple commodities have to be explicitly modeled. In these cases, an additional decision to be taken concerns the commodity to be loaded on a vehicle. To the best of our knowledge, no survey is available on the class of *VRPs with multiple commodities*.

When does a route start? In the classical VRPs, all routes (or vehicles, given that each vehicle performs one route) start from the depot at the same time. Thus, these models consider a period of time where all vehicles start their route at the same time and assume that the decision on which customers to serve in that period has already been taken.

There are applications where it is worth considering multiple routes of the same vehicle, that can be for example limited by a total time duration, or that there is some flexibility on the period of time where to serve a customer. In such cases, an additional kind of decision has to be taken, that is when a route should start. We call *VRPs over time* the class of problems which consider the decision of when to start a route. Problems of this class are attracting increasing interest in the last decades. The next section is devoted to the VRPs over time.

2.4 Routing over time

In the classical VRPs, the decisions are how to assign customers to vehicles and in which order each vehicle serves the customers assigned to it. As already mentioned, these problems may take into consideration, for example, capacity constraints or constraints on the order of visits in a route, e.g., backhaul, pick-up and delivery. In the VRPs over time, an additional decision can and has to be taken, that is, when to start a route. The reason to consider the time aspect lies in the specific setting of the problem. It might be a consequence of the characteristics of customers requests, requiring the distributor to consider when or how each customer has been served in the past. It may also be the result of the resources available to perform the distribution.

A consequence of taking into account the dimension of time in routing problems is that, while in the classical VRPs routes can start all together, in the VRPs over time routes can start separately from one another, as a result of the peculiarities of the problem that make evaluating the decision on routing over time necessary. While in the classical VRPs, the decision about when to serve customers has been taken beforehand, in the VRPs over time the decision is taken jointly with the decisions about assignment of customers to routes and order of visit in each route.

In this section we overview the contributions to VRPs over time by organizing the discussion according to the characteristics of the studied problems. We will focus on the characteristics that make these problems different from the classical VRPs and different from each other. The literature on Periodic Routing Problems (PRPs) will be reviewed first (see Section 2.4.1), followed by the literature on Inventory Routing Problems (IRPs) (see Section 2.4.2). These two types of problems cover the majority of the contributions. In both cases, a homogeneous fleet of capacitated vehicles is available. In the PRPs possible sequences of days of visit (schedules) are pre-defined for each customer. Moreover, given a sequence of days for a customer, the quantity to be delivered to that customer is given. In the IRPs different solutions may have the same times of visit but different quantities delivered. While in the PRPs the decision is when to visit, and this decision implies the quantity to deliver, in the IRPs both decisions about when and how much to deliver have to be taken. Other two types of VRPs over time will then be discussed, namely the VRPs with release dates (see Section 2.4.3) and the multi-trip VRPs (see Section 2.4.4). In the former case there is a lower bound, that is the release date, on the time a vehicle can start its route to visit a customer, in the latter the same vehicle can perform multiple routes.

2.4.1 Periodic routing problems

The PRPs are characterized by the fact that each customer is required to be visited in multiple periods of a planning horizon. Such periods can be specified in different ways but typically are either the result of a given frequency for the visits (e.g. every two days), of a given number of visits within the planning horizon (e.g. twice a week), or of a fixed set of periods specified by the customer (e.g. Monday and Thursday or Tuesday and Friday). We call each possible set of periods (e.g., Monday and Thursday) a visiting option. The most common objective is the minimization of the routing cost. The decision about the visiting option of each customer, among the possible alternatives, is taken jointly with the decisions about the assignment of customers to routes and the sequencing of customers in each route. The quantity to be delivered to customers is not a decision to be taken, is implicit and fixed.

The *basic PRP*, as defined in Campbell and Wilson (2014), is the problem of assigning to each customer one of its feasible visiting options over a planning horizon with the following requirements: the route of each vehicle starts and ends at the same depot, the amount of goods to be delivered to each customer is known and is

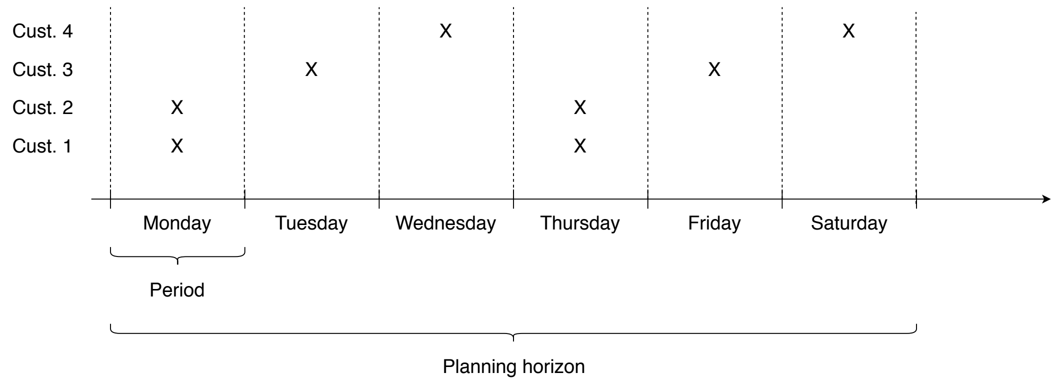


Figure 2.2: Periodic Routing Problem: an example.

entirely served by one vehicle, the size of the available fleet is given, and the total travel time of each vehicle is limited. Formally, let $G = (V, A)$ be a directed graph, where the set of vertices V is composed of the vertex 0, denoting the depot, and the set of vertices $N = \{1, \ldots, |N|\}$, denoting the customers. Over a discretized planning horizon $T = \{1, \ldots, |T|\}$, each customer $i \in N$ is characterized by a set of feasible visiting options, where each visiting option is a subset of T. The set of vehicles available for the distribution is denoted by $K = \{1, \ldots, |K|\}$. The vehicles are homogeneous and have a capacity Q. The cost of traveling from i to j is denoted by c_{ij}. One visiting option for each customer must be chosen and routes for each period must be created. The objective is to minimize the total routing cost over the planning horizon. The solution of a PRP is exemplified in Figure 2.2.

The introduction of the first routing problem which is periodic in nature is credited to Beltrami and Bodin (1974), where the problem was motivated by an application in the context of waste collection. The paper also introduces heuristics for the problem. The definition of the problem is extended in Russell and Igo (1979). The first paper to identify the problem as a PRP is Christofides and Beasley (1984). The instances proposed therein have become part of the standard benchmark instance set that newly devised algorithms are tested on.

The most recent survey on PRPs is provided in Campbell and Wilson (2014). The authors discuss the advancements since the paper of Beltrami and Bodin (1974).

Several contributions have been proposed after the survey Campbell and Wilson (2014), mostly on variants of the basic PRP. The PRP with time windows (PRPTW) is a variant of the PRP in which customers are allowed to be served within a specific time interval of each period. A generational genetic algorithm is proposed for the PRPTW in Nguyen et al. (2014). A study of the performance of a particle swarm optimization approach is presented in Norouzi et al. (2015) for the PRPTW in a competitive environment, in which a fraction of the customer demand is served by the first competitor to arrive at the customer location.

Another variant that has received considerable interest is the multi-depot PRP in which the customers are allowed to be served from multiple depots, where the vehicle returns at the end of a route. A heuristic for the problem is investigated in Mirabi (2014) combining elements derived from the mechanics of electromagnetism and simulated annealing. The fleet sizing problem faced when dealing with a multi-depot PRPTW is investigated in Rahimi-Vahed et al. (2015).

Typically, in the basic PRP the problem faced by the distributor is either to deliver goods or to pickup goods at customer locations. A variant of the basic PRP is introduced in Jayakumar N. et al. (2016), where the additional issue of dealing with both the pickup and the delivery phases is considered.

In Archetti et al. (2015b) the multi-period VRP with due dates (MPVRPD) is studied. The problem considers the case in which a set of customers is characterized by a release and a due date. These express the first and last period in which each customer must be served, respectively. The problem can be seen as a PRP as each period between the release and the due date is a visiting option and the amount to be delivered to the customers is given. The multi-depot periodic VRP with due dates and time windows is considered in Cantu-Funes et al. (2018). The work is inspired by a real case of a brewing company. Heterogeneous vehicles are used to serve distribution centers from multiple depots. Due to the fact that empty bottles are collected at the distribution center, only one distribution center can be served in each route. However, each vehicle is allowed to perform multiple routes in each period. Deliveries must be performed within a time window, specified for each customer, which is the same in each period. The request of each customer must be fulfilled before a due date, expressed as the last period in which the customer can be served. Again, each period before the due date can be seen as a visiting option.

While the basic PRP and the above mentioned contributions refer to problems defined on graphs where customers are represented by vertices, recently, problems where customers are represented by arcs have received some interest. The Periodic Capacitated Arc Routing Problem (PCARP) is introduced in Lacomme et al. (2005) as extension to a periodic setting of the Capacitated Arc Routing problem, as the PRP is for the VRP. The authors describe several versions of the PCARP, a classification scheme for the problem, and a memetic algorithm based on a crossover operator considering both planning and scheduling decisions. In recent years the PCARP has been investigated in Zhang et al. (2017), proposing a memetic algorithm together with a route decomposition operator, and in Riquelme-Rodríguez et al. (2014). The authors pose an adaptive large neighborhood search is presented for the PCARP with inventory constraints. The problem arises in open-pit mines, where unpaved roads have to be watered to prevent the dust from damaging the equipment. A maximum capacity is considered for the water depot and water demand is modeled as a function of time and humidity level.

2.4.2 Inventory routing problems

The IRPs are problems characterized by the fact that deliveries take place over time without pre-defined schedules and the quantity to be delivered to any customer at any time is a decision variable. The quantity to be delivered is such that the inventory capacity of the customer is not exceeded. In the IRPs several decisions are simultaneously taken: when to visit customers, how much to deliver to customers, how to assign customers to routes and how to order customers visit in a route.

In the terminology of the IRP often a supplier has the role of the depot and retailers the role of customers. We will use both terms in this section. The *basic IRP* is defined on a directed graph where a vertex represents a supplier and the other vertices the retailers. A discretized planning horizon is given. The quantity available at the supplier and demanded by each retailer in each period of the horizon is known. Each retailer has an inventory holding capacity that cannot be exceeded. Supplier and retailers may incur inventory holding costs. Stock-out situations are not allowed, that is, the inventory at each retailer must be sufficient to satisfy the demand in each period. The objective is to minimize the total distribution cost which includes the routing cost and, if relevant, the inventory cost. The basic IRP with a single vehicle was introduced in Archetti et al. (2007). The problem is formally defined as follows. Let $G = (V, A)$ be a directed graph, where the vertex 0 represents the common supplier and the set of vertices $N = \{1, \ldots, |N|\}$ the retailers. The set of vehicles available for the distribution is denoted by $K = \{1, \ldots, |K|\}$. The vehicles are homogeneous and have a capacity Q. The cost of traveling from i to j is denoted by c_{ij}. At each period t over the planning horizon $T = \{1, \ldots, |T|\}$ the quantity r_{0t} is made available at the supplier and the quantity r_{it} is consumed at retailer $i \in N$. The initial inventory of the supplier is known. Each retailer i has a maximum inventory level and an initial inventory level. A unit inventory holding cost is defined for both the supplier and

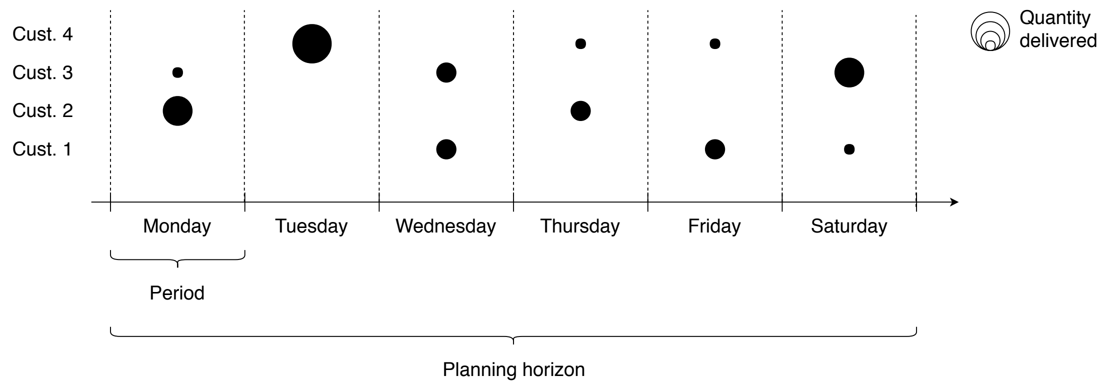


Figure 2.3: Inventory Routing Problem: an example.

the retailers. The customers to be visited in each period must be chosen, together with the quantity to be delivered in each visit and routes must be created for each period. The minimization of the inventory and routing costs is sought while avoiding stock-out at the retailers. An example of the solution of an IRP is shown in Figure 2.3.

The first paper where the expression *inventory routing problem* was used is Bell et al. (1983), where the authors discuss the integration of the inventory management at retailer locations in the industrial gases supply chain. Whereas the study of the PRPs was motivated by applications in waste collection, IRPs were motivated by applications aimed at integrating inventory management in distribution problems.

Various reviews of the scientific contributions have been presented in recent years. In Andersson et al. (2010) the industrial aspects of integrating inventory management and routing are analyzed. In Bertazzi and Speranza (2012) and Bertazzi and Speranza (2013) tutorials on the IRPs are provided, with a classification of characteristics that make the IRPs different from each other, starting from the case where there is one retailer only. A survey of the contributions to the IRPs is given in Coelho et al. (2013), where problems are categorized with respect to their structural variants and the availability of information on retailer demand.

We review here only the papers published after these surveys. In Archetti et al. (2014a) a comparison of different formulations for the basic IRP with multiple vehicles is presented. It is shown that a vehicle-indexed formulation appears to perform better that more compact formulations. Exact approaches to the solution of the basic IRP with multiple vehicles have been studied in Coelho and Laporte (2014a) and Desaulniers et al. (2015). The former introduces new valid inequalities. The authors also assess how input ordering affects solution of the model. The latter presents a branch-and-price-and-cut algorithm to solve a newly devised formulation, providing the optimal solution for additional 54 instances over the standard benchmark instance set of 640 instances introduced in Archetti et al. (2007). An analysis of the benefits of integrating inventory management and routing in the supply chain management is provided in Archetti and Speranza (2016) by comparing the solution obtained by considering inventory management and routing separately and by solving the basic IRP.

Various new variants of the basic IRP have been recently introduced. The IRP where lost sales are allowed for customers is defined in Park et al. (2016). A genetic algorithm is proposed for the solution of the problem. In Niakan and Rahimi (2015) a multi-objective IRP arising in the medicinal drug distribution to healthcare facilities is presented. Together with inventory and routing costs and greenhouse gases emissions, the objective function considers product shortage and expired drugs minimization. A hybridized possibilistic method in synergy with an interactive fuzzy approach is proposed for the solution of the problem. The pickup and delivery IRP is considered in Iassinovskaia et al. (2017), discussing the IRP arising in supply chains where reusable packets (Returnable Transport Items, RTI) are used. Products are delivered in RTIs and, simultaneously, empty RTIs are collected. The IRP with perishable goods is considered in Coelho and Laporte (2014b) where the decisions of when, how and how much to replenish customers with such goods is analyzed. A single perishable product is considered in Azadeh et al. (2017) discussing the IRP with transshipment. The authors propose a genetic algorithm for the problem where the parameters of the algorithm are tuned using the Taguchi method.

The so called Flexible Periodic Vehicle Routing Problem (FPVRP) is presented in Archetti et al. (2017). In this problem each customer has a total demand that must be served within a planning horizon and a limit on the maximum quantity that can be delivered at each visit is defined. No predefined schedules are given and the quantity to be delivered is a decision variable. While the problem is called periodic by the authors, it shares the basic features of an IRP.

The multi-product variant of the basic IRP has been studied in Cordeau et al. (2015) where a tree-phases heuristic solution approach is presented based on the decomposition of the decision process. In the first phase a Lagrangian based method is used to plan customer replenishment. In the second phase the routing among customers is obtained, allowing for split delivery. Finally, the solution is improved by means of a feedback model. In the multi-product variant studied in Mjirda et al. (2014) vehicles visit suppliers to collect products to be delivered to assembly plants. Each supplier provides one product and can be visited multiple times during a period. A two-phase variable neighborhood search is proposed. In the first phase an initial solution is build by solving a CVRP for each period and in the second phase this solution is iteratively improved minimizing both the routing and inventory costs. In Shaabani and Kamalabadi (2016) a population-based simulated annealing heuristic is proposed for the multi-product multi-retailer IRP of perishable goods and assess its performances comparing the algorithm with a simulated annealing and with a genetic algorithm. A multi-product IRP is also considered in Laganà et al. (2015) in the context of the supermarket distribution industry. Two mixed integer problem formulations are presented and a decomposition approach is proposed.

The multi-depot IRP is studied in Bertazzi et al. (2017) in a city logistic environment. A formulation for the problem is presented and a branch-and-cut algorithm and a three-phases matheuristic is presented, comprised of a clustering, a routing construction and an optimization phase. Benchmark instances for the multi-depot IRP have been proposed in Noor and Shuib (2015) by applying clustering techniques to single-depot IRP benchmark instances.

In the context of maritime inventory routing, Papageorgiou et al. (2014a) presents an approximate dynamic programming approach for the deterministic maritime IRP with a long planning horizon. Two decomposition algorithms for a single product maritime IRP are presented in Papageorgiou et al. (2014b). The IRP of liquefied natural gas is introduced in Andersson et al. (2016). The peculiarity of the problem lies in the fact that a constant rate of the cargo evaporates in the tanks each day and is used as fuel during transportation. The authors present a path flow formulation that is solved with a decomposition algorithm.

The literature presented so far models the IRP as a problem on a planning horizon of a finite set of periods. A different setting is studied in the literature considering the periodic IRP (PIRP). In this problem a replenishment schedule must be identified over a finite period with the additional constraint that the initial inventory levels must be equal to those at the end of the planning horizon, allowing the schedule to be repeated. A heuristic is proposed in Qin et al. (2014) for the problem where the inventory and routing components are solved by means of a local search and a tabu search, respectively, with the two components being iteratively executed. Liu et al. (2016) introduces a hybrid heuristic consisting of an iteration of a particle swarm algorithm, a local search improving each particle found in the previous step and a large neighborhood search to avoid being trapped in local optima solutions. The algorithm is shown to outperform the one proposed in Qin et al. (2014) by more than 10% on average over a set of 10 instances. Finally, the selective and periodic inventory routing problem (SPIRP) is studied in Aksen et al. (2014), where an adaptive large neighborhood search is proposed. Montagné et al. (2018) extends the work of Aksen et al. (2014) for the real case of reusable waste oil collection in Canada, introducing the use of a relaxation of the model and a constructive heuristic to solve instances with up to 3000 customers in a 30-day time horizon.

As reported in Andersson et al. (2010), the other options considered in the literature for the horizon over which the IRP is studied, are instant horizon, when the planning horizon is so short that at most one visit per customer is needed, and infinite horizon, when the decision focuses on the distribution strategies rather than schedules. An instant planning horizon is considered in Li et al. (2014) studying the IRP arising the petrochemical industry where specific constraints are in place, like hours-of-service of the vehicles. A measure of workload balance is considered in the objective function, namely the solution minimizing the maximum route travel time. A single period horizon is considered in Juan et al. (2014), proposing a simheuristic, a solution algorithm combining simulation and heuristics to solve, for the solution of the IRP with stochastic demands and stock-out. An infinite planning horizon is considered in Van Anholt et al. (2016) discussing the IRP with pickups and deliveries arising in the contest of ATM replenishment in the Netherlands. The authors propose to decompose the model into treatable subproblems that are then solved by a branch-and-cut algorithm.

Variants of the problems considering stochastic information have also been studied recently. In Coelho et al. (2014) some heuristics are proposed for the dynamic stochastic IRP and it is shown how considering the stochastic information is beneficial for the quality of the solution at the expenses of computational time. Other findings of the paper include that a longer rolling horizon step does improve the solution and that allowing for consistent solutions is more beneficial in a static setting rather than in a dynamic one. The IRP with transportation procurement and stochastic demand is studied in Bertazzi et al. (2015). The authors show the benefit of considering the entire stochastic information instead of the average demand. A stochastic dynamic programming formulation is proposed and a matheuristic is devised for the problem. The IRP with stochastic stationary demand rates is proposed in Abdul Rahim et al. (2014), where a Lagrangian relaxation approach is proposed to solve the problem. Mes et al. (2014) describes the IRP arising in waste collection from sensor equipped underground containers. The waste to be collected is assumed to be stochastic and dynamically evolving. Collection costs are minimized together with a function of customer satisfaction. Stochastic demand is also considered in Crama et al. (2018) when studying the IRP with perishable products. Four solution methods are presented and compared. Perishable products are also considered in the IRP variant studied in Soysal et al. (2018) where collaboration is considered among multiple suppliers. The benefits of collaboration are investigated while accounting for demand uncertainty. Expected inventory costs, waste costs, fuel and driver costs are minimized. Within the context of IRPs with stochastic information, literature on the IRP considering multiple objectives has also been recently presented. In Nolz et al. (2014a), the authors extend their previous work (see Nolz et al. (2014b)) on the stochastic IRP for medical waste collection to consider a bi-objective IRP in the context of waste disposal. In particular, together with the routing and inventory costs, the authors consider the quality of service as part of the objective function. A multi-objective IRP is studied in Rahimi et al. (2017) where, in addition to the inventory and routing costs, the authors consider service level and the environmental footprint as part of the objective function. In Yadollahi et al. (2017), a chance-constrained formulation is proposed for the IRP with stochastic demand. A safety stock-based deterministic optimization model is used to determine near-optimal solutions to the chance-constrained optimization problem. Different safety stock models are investigated and insights on the setting of the safety stock level are obtained.

2.4.3 Vehicle routing problems with release dates

The definition of the classical VRPs imply that the goods to be delivered to the customers are ready for delivery at the depot at the beginning of the planning period. This is, however, not the case in many practical applications. For example, in consolidation and distribution centers, goods to be distributed in the city center arrive during the distribution period, while the delivery of other goods that have previously arrived at the distribution center is taking place. In this case, one has to decide whether it is better to deliver the goods to the customers or to wait for more goods to reach the distribution center. Similar problems arise in the context of cross docking and same day delivery problems. The former is a logistic solution aimed at reducing the stocking costs by loading the parcel of inbound trucks directly on outbound vehicles. In the latter, customers place orders dynamically during the same day the order has to be fulfilled. Same day delivery problems are faced by many distributors with online purchases. Distributors must quickly react to incoming orders to deliver goods on time.

We propose the *basic VRP with release dates* (VRP-rd) as follows. Let $G = (V, A)$ be a directed graph. The set V is composed of the vertex 0, identifying the depot, and the set $N = \{1, \cdots, |N|\}$, representing the customers. The demand of customer i is defined by a quantity $q_i \geq 0$ and a release date $r_i \geq 0$ is associated with customer i. A set $K = \{1, \ldots, |K|\}$ of homogeneous vehicles with capacity Q

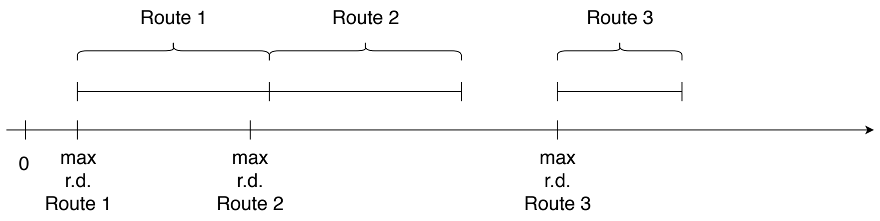


Figure 2.4: Vehicle Routing Problem with release dates: an example.

is available to perform the distribution. Each arc of the graph is characterized by a traveling time and a distance, which are assumed to be identical. The time and distance from i to j is denoted as c_{ij}. Routes must be created and assigned to each vehicle in such a way that the starting time of a vehicle cannot be lower than the maximum release date of the customers assigned to it. The objective function is the minimization of completion time.

An example of solution of a basic VRP-rd is provided in Figure 2.4 where 'r.d.' means 'release date'. The first route can start when the last of the parcels to be delivered in that route reaches the depot. As the vehicle is traveling when all the parcels to be delivered in the second route have reached the depot, the second route starts as the vehicle becomes available again, i.e., when it returns to the depot. Finally, the vehicle waits for all the parcels to be delivered in the last route to reach the depot and performs its final route.

The possibility that customers have a release date has only recently been introduced in VRPs. To the best of our knowledge, the first paper to consider release dates for parcel delivery is Cattaruzza et al. (2016a). Archetti et al. (2015a) investigates the complexity of both the TSP and the uncapacitated VRP with release dates and unlimited fleet in the case of completion time minimization and total distance minimization with a deadline on special topologies, namely the line and the star. In

Reyes et al. (2018) the study of the complexity of the problem is extended to consider a service guarantee for the deliveries to the customers. In Archetti et al. (2018) the authors provide a formulation for the TSP with release dates and completion time minimization. Two variants of an iterated local search are presented for the problem and compared. Cattaruzza et al. (2016a) introduces the multi-trip VRP with time windows and release dates. A hybrid genetic algorithm is proposed for the solution of the problem, making use of a route decomposition technique for chromosome decoding and a local search to improve the solution. In Shelbourne et al. (2017) the VRP with release and due dates is considered. A due date is the time by which the order should be delivered to the customer, is also associated with each customer. A convex combination of operational costs and customer service level is considered. The former is expressed as the total traveled distance and the latter as the total weighted delivery tardiness. The authors present a path relinking algorithm for the problem.

The literature discussed above studies the case in which known customers have a deterministic release date. However, customer requests may arrive during the day, with probabilistic information on potential requests. This is the case of the same day delivery problem. Various papers investigate the routing problem arising in the logistic of same day deliveries. In these problems the information on customers request is assumed to be stochastic either in time or both spatially and temporally. Klapp et al. (2016) studies the dynamic dispatch waves problem (DDWP) of a single vehicle on a line, where the distributor has to decide whether to dispatch a vehicle to serve known customers or to wait for potential requests that may arrive later, with the objective to minimize expected vehicle operating costs and penalties for unserved requests. Klapp et al. (2018) the authors extend their previous work to a general network. A deterministic model is used to find an optimal a priori solution to the stochastic variant and two dynamic policies are developed. The trade-off between minimizing operational costs and maximizing the total order coverage is studied. In Voccia et al. (2017) investigates the same day delivery (SDD) problem, where a fleet of vehicles is used to serve requests characterized by time windows or a delivery deadline. Probabilistic information about the arrival rate of future requests at known locations is available. The authors identify the circumstances that make waiting at the depot beneficial to maximize the number of requests that are served on time. Ulmer et al. (2016) considers the SDD problem in with vehicles are allowed to return to the depot before having completed its distribution to load the parcels of new customers. Unknown customers are characterized by a spatial and temporal probability distribution. Ulmer et al. (2017) introduces the restaurant meal delivery problem of picking up meals at a restaurant chosen by the customer and delivering the meal at his location. The probability distributions on the time and location of meal requests are known. Before the delivery, the selected vehicle has to pick up the meal at the restaurant. The meal preparation time is random and therefore the vehicle could be waiting at the restaurant to pick up the meal.

2.4.4 Multi-trip vehicle routing problems

The general assumption in the classical VRPs is that each vehicle only performs one route. Consideration on the multiple use of a vehicle may be made *ex post*, once a solution is obtained, with the underlying assumption that non overlapping routes can be performed by the same vehicle. In the multi-trip VRPs (MTVRPs) each vehicle is explicitly allowed to perform multiple trips during its service time.

The *basic MTVRP*, as defined in Cattaruzza et al. (2016b), is a CVRP in which each vehicle performs a set of routes in such a way that the total demand of customers served in each route does not exceed its capacity and that at the end of the last route the vehicle returns to the depot before a given deadline. The total routing time is

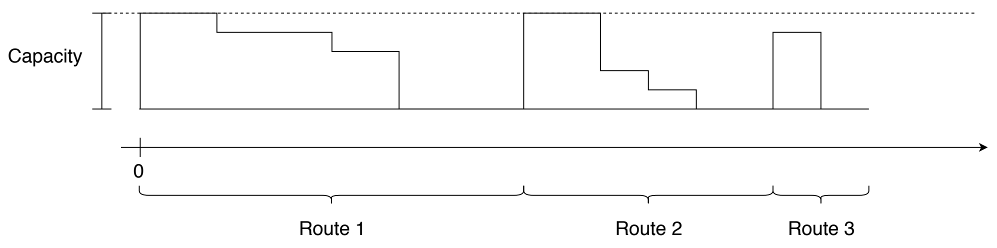


Figure 2.5: Multi-trip Vehicle Routing Problem: an example.

minimized. Formally, the problem is defined on a directed graph $G = (V, A)$, where the set V is composed of the vertex 0, representing the depot, and the set N of customers is defined as $N = \{1, \ldots, |N|\}$. The set of arcs is defined as $A = \{(i, j)|i, j \in V\}$ with the cost of traveling on the arc $(i, j) \in A$ being denoted as t_{ij}. The set of vehicles available for the distribution is denoted by $K = \{1, \ldots, |K|\}$. The vehicles are homogeneous and have a capacity Q. Each vehicle is available at time 0 and has to complete all deliveries within time T_H. The cost of traveling from i to j is denoted by c_{ij}. Customer i demand is represented as $q_i, q_i \geq 0, i \in N$. Routes must be created and assigned to a vehicle such that the total routing time is minimized, subject to the following constraints: each route starts and ends at the depot, each customer is visited exactly once, the sum of the demands of the customers in each route does not exceed the capacity of the vehicle, the sum of the duration of the routes assigned to the same vehicle does not exceed T_H.

An example solution of a MTVRP with one vehicle is presented in Figure 2.5. At the beginning of the distribution the vehicle is fully loaded and departs for its first route, serving three customers. Three are the steps in the vehicle load. After serving the three customers, the vehicle returns empty to the depot. A second route serving three customers is performed, with the vehicle leaving the depot fully loaded. The final route serves only one customer with the vehicle loaded only to serve its demand.

A recent review of the literature on the MTVRP is provided in Cattaruzza et al. (2016b). The formulations introduced so far for the MTVRP are presented, together with the instances created for the problem and the exact and heuristic algorithms proposed. The same topics are discussed for the known variants of the problem.

Various contributions have been introduced after the survey from Cattaruzza et al. (2016b). In François et al. (2016) a large neighborhood search is proposed to solve a relaxed version of the problem where the deadline on the service time of the vehicles is not considered as a constraint but overtime is penalized in the objective function. In Liu et al. (2018) the multi-trip repairmen problem with time windows is introduced. The problem arises from a real application of a repair company facing the additional costs of the allowances that are paid to the repairmen when stationed at customers locations. The trade-off of returning to the depot (increasing the routing costs) and waiting at the customer location (increasing the allowance costs) is investigated. A branch-and-price method is proposed for the problem. The MTVRP with backhauls is introduced in Wassan et al. (2017) where trips must be constructed on the condition that, in each route, backhaul customers, if any, must be visited after all linehaul customers have been served. A formulation is presented for the problem and a two level Variable Neighborhood Search heuristic is proposed. The multi-trip pickup and delivery problem with time windows and synchronization (MT-PDTWS) is introduced in Nguyen et al. (2017). A tabu search heuristic is devised for the problem and its performance is assessed. In Tirkolaee et al. (2018) the multi-trip capacitated arc routing problem arising in waste collection is studied where depots and disposal facilities are in different locations. A formulation is devised for the problem and an ant colony algorithm is proposed for the problem.

Stochastic information is considered in two papers. In Chu et al. (2017) the MTVRP is studied where split-deliveries are allowed and soft time windows are considered to achieve customer inventory replenishment. Stochastic travel times are assumed. A two stage heuristic is proposed. In the first stage a solution for the problem is built and in the second stage the solution is improved by balancing the load of the vehicles to reduce delays and penalties and idling. In Tirkolaee et al. (2017) the robust MTVRP of perishable products is introduced with customers demand uncertainty and time windows and intermediate depots. A formulation is presented for the problem and results assessing its robustness are illustrated.

2.5 Conclusions and future research directions

In this chapter a classification of vehicle routing problems based on the decisions that have to be taken has been proposed. Then, the literature on vehicle routing problems over time has been surveyed. In case of previous surveys, only the papers published after the surveys have been discussed here. In this class of problems, in addition to the classical decisions about the assignment of customers to vehicles and the sequencing of customers in each route performed by vehicles, the decision about when a route starts from the depot has to be taken. Besides the most studied problems in this class, periodic routing problems and inventory routing problems, we survey the literature on recently studied problems, namely vehicle routing problems with release dates, and multi-trip vehicle routing problems.

The decision about when each route starts increases the computational complexity of the problems, and implies the need of additional variables in the mathematical programming models, with respect to the more classical problems, and specific solution methods. At the same time, the vehicle routing problems over time model more integrated problems whose solution allows savings with respect to sequential approaches.

Future research directions include the study of the most appropriate formulations and solution methods of already studied problems but also the study of deterministic and stochastic variants. Dynamic vehicle routing problems over time would also deserve attention, especially considering the technological evolution towards the use of digital devices that allow continuous generation and transmission of data.

Chapter 3

An Iterated Local Search for the Traveling Salesman Problem with release dates

This chapter is based on the article:

C. Archetti, D. Feillet, A. Mor, and M.G. Speranza. An iterated local search for the traveling salesman problem with release dates and completion time minimization. *Computers & Operations Research*, 98:24-37, 2018.

The work there has been performed by the author in collaboration with Prof. Claudia Archetti, Prof. Dominique Feillet, and Prof. M. Grazia Speranza.

3.1 Abstract

In the Traveling Salesman Problem (TSP) with release dates and completion time minimization an uncapacitated vehicle delivers to customers goods which arrive at the depot over time. A customer cannot be served before the demanded goods arrive at the depot. A release date is associated with each customer which represents the time at which the goods requested by the customer arrive at the depot. The vehicle may perform multiple routes, all starting and ending at the depot. The release dates of the customers served in each route must be not larger than the time at which the route starts. The objective of the problem is to minimize the total time needed to serve all customers, given by the sum of the traveling time and the waiting time at the depot. The waiting time is due to the fact that the vehicle has to wait at the depot until the latest release date of the customers it is going to serve in the next route. We introduce some properties, propose a mathematical programming formulation and present a heuristic approach based on an iterated local search where the perturbation is performed by means of a destroy-and-repair method. Two alternative repair operators, one simple and fast and the other based on a mathematical programming model, are proposed, which give rise to two variants of the heuristic. The mathematical formulation is used to find the optimal solution on instances with up to 20 customers, built from benchmark instances for the classical TSP. Comparison with optimal solutions shows that both algorithms provide high-quality solutions. Tests are also made on larger instances to compare the performance of the two variants of the heuristic.

3.2 Introduction

A common trait of the classical Vehicle Routing Problems (VRP) is the assumption that the goods to be distributed are available at the depot when the distribution starts. This implies that all vehicle routes may start immediately to distribute goods to customers. However, there are different settings in which this assumption is not satisfied, i.e., goods are not all available at the depot when the distribution starts and arrive at the depot over time. In this case, vehicles need to wait at the depot for the goods to arrive before distributing them. One example is related to city logistics and city distribution centers. Such an application has been studied in Cattaruzza et al. (2016a) where the authors introduce the Multi-Trip Vehicle Routing Problem with Time Windows and Release Dates (MTVRPTW-R) in which the goods that have to be distributed arrive at the depot over time, i.e., they become available when the distribution has already started. This poses the additional question of whether it is better to wait for additional goods to arrive and have a better loaded vehicle, or to start a route with the currently available goods. The arrival time at the depot of the goods to be delivered to a customer is called its release date. Another example arises in same day delivery problems related to e-commerce logistics. In this case, customer orders arrive online when the distribution (of previously received orders) has already started. Thus, the newly received orders have to be integrated in the distribution plan by designing vehicle routes that perform multiple trips, i.e., vehicles return to the depot multiple times in order to pickup the newly arrived orders that need to be distributed.

The focus of this chapter is to study a routing problem arising in the applications mentioned above. In particular, we consider the TSP with release date and completion time minimization (TSP-rd(time)), that is the problem where each customer is associated with a release date and a single uncapacitated vehicle is allowed to perform multiple trips during the time horizon (say, the day), one after the other. No restriction is imposed on the maximum time taken by the vehicle to serve all customers and the objective is to minimize the completion time of the service, that is the time at which the vehicle is back to the depot and has served all customers. The completion time is given by the sum of the traveling time and waiting time. We consider the static problem where all information about the customers and the associated release dates are known in advance. As the TSP-rd(time) is relatively new in the literature, we believe that a deeper understanding of the static and deterministic version of the problem can be useful when solving the problem where release dates are characterized by dynamicity and uncertainty.

The contributions of this chapter are summarized as follows. We first devise some properties of the problem and describe an approximation algorithm derived from Christofides approximation algorithm for the TSP. Then, we propose a mathematical programming formulation and present a heuristic approach based on an iterated local search where the perturbation is performed by means of a destroy-and-repair operator. Two alternative repair operators are proposed, one simple and fast and the other based on a mathematical programming model, which gives rise to two variants of the proposed heuristic. Ad hoc neighborhoods for the local search are introduced which are based on the characteristics of the problem. The mathematical programming formulation is used to find the optimal solution on instances with up to 20 customers built from benchmark instances for the classical TSP. On all but one instances the heuristic with the simple repair operator finds the optimal solution.

The chapter is organized as follows. In Section 3.3 we review the literature related to similar problems. In Section 4.3 the TSP-rd(time) is defined and the properties and the mathematical programming formulation are presented. The heuristic is described in Section 3.5, whereas the computational experiments are presented in Section 3.6. Finally, conclusions are drawn in Section 3.7.

3.3 Literature review

To the best of our knowledge, Cattaruzza et al. (2016a) is the first work where the concept of release date is introduced in a routing setting. The problem studied is a multi-vehicle routing problem with time windows. The authors propose a hybrid genetic algorithm to solve the problem and test it on instances generated from Solomon's instances for the VRP with time windows (Solomon (1987)). The vehicle routing problem with release dates with a single uncapacitated vehicle is introduced in Archetti et al. (2015a). The authors call this problem the Travelling Salesman Problem with release dates (TSP-rd). Two variants of the TSP-rd are proposed: one takes into consideration a deadline for completing the distribution and minimizes the total traveling time, the other one has no deadline and seeks the minimization of the time needed to complete the distribution. The former is referred to as TSP-rd(distance) and the latter as TSP-rd(time). The complexity of the two variants is analyzed for special topologies of the graph representing the distribution network, i.e., a line, modelling a distribution along a road, and a star, modelling the situation where the depot is the center of the distribution area. Further complexity analysis is carried out in Reyes et al. (2018), where, in addition to a distribution deadline, a service guarantee is considered, enforcing a maximum delay between the release date and the delivery to the customer. Shelbourne et al. (2017) consider the VRP with release and due dates, where the due date is the time by which the order should be delivered to the customer. The authors minimize a convex combination of operational costs and customer service level, measured by the total traveled distance and the total weighted delivery tardiness and no waiting time is considered, contrary to what happens in the TSP-rd(time). The authors present path relinking algorithm for the problem. The algorithm relies, among other things, on a parameter penalizing the solution that are infeasible with respect to the capacity constraint which makes it not suited for the

TSP-rd(time), where such constraint is not considered.

As mentioned in Section 3.2, the vehicle routing problem with release dates finds applications in the context of the same-day delivery service. Recent contributions that study the same day delivery problem are Klapp et al. (2016), Klapp et al. (2018), and Voccia et al. (2017). In Klapp et al. (2016) the authors study the dynamic dispatch waves problem (DDWP) of a single vehicle on a line, where the problem is to decide whether to dispatch a vehicle to serve known customers or to wait for potential requests that may arrive later, with the objective to minimize expected vehicle operating costs and penalties for unserved requests. The study is extended to the case of a general network in Klapp et al. (2018). In Voccia et al. (2017) the same day delivery problem is investigated. A fleet of vehicles is used to serve requests characterized by time windows or a delivery deadline. Future requests are unknown but probabilistic information is available. The authors identify the circumstances that make waiting at the depot beneficial to maximize the number of requests that are served on time.

Finally, a problem which is strictly related to the one analyzed in this chapter is the Multi-Trip Vehicle Routing Problem (MTVRP), where each vehicle may perform multiple trips and, thus, visit the depot multiple times. See Cattaruzza et al. (2016b) for a recent survey. The need of visiting the depot multiple times may come from the fact that routes need to have a short duration (see Azi et al. (2007) for potential applications) or because of capacity constraints. The problem with a single vehicle performing multiple routes within one workday is investigated in Azi et al. (2007), where customer requests with time windows are considered. The authors propose an exact algorithm for the problem. The work is extended in Azi et al. (2010) and in Azi et al. (2014) to consider multiple vehicles. An exact algorithm is designed in Azi et al. (2010) while an adaptive large neighborhood search is proposed in Azi et al. (2014).

In this chapter we focus on the TSP-rd(time) which is introduced in Archetti et al. (2015a). We propose the first mathematical formulation and solution approach for the problem. The TSP-rd(time) differs from the problem analyzed in Azi et al. (2007) as it considers release dates and does not consider duration constraints and time windows for the customers. It also differs from Klapp et al. (2016), Klapp et al. (2018), and Voccia et al. (2017) where a dynamic problem is studied and decision epochs are defined a priori, contrary to what happens in TSP-rd(time).

3.4 The Traveling Salesman Problem with release dates and completion time minimization

The TSP-rd(time) is defined as follows. Let $G = (V, A)$ be a complete graph. A traveling time and a traveling distance are associated with each arc $(i, j) \in A$. These two values are assumed identical and are denoted by t_{ij}. It is also assumed that the triangle inequality is satisfied. The set of vertices V is composed by vertex 0, which identifies the depot, and the set N of customers, with $|N| = n$. The release date for customer $i \in N$ is denoted by r_i, $r_i \geq 0$. This means that the goods for customer i can either arrive at time r_i, then $r_i > 0$, or be at the depot at the beginning of the distribution, e.g., because they arrived overnight, then $r_i = 0$. A single vehicle is allowed to perform a sequence of trips. Capacity constraints are not considered. The objective is to minimize the completion time, that is, the total traveling time plus the waiting time at the depot, and serve all customers. The waiting time is due to the fact that the vehicle has to wait at the depot until the latest release date of the customers it is going to serve in the next route. Without loss of generality, we assume that customers are ordered in non-decreasing order of their release dates, i.e., $r_i \leq r_j$ for $i < j, i, j = 1, \ldots, n$. Given a solution to the TSP-rd(time), we call *route* a trip starting and ending at the depot and not visiting the depot in between.

3.4.1 Properties

In this section some properties of the problem are presented which are used to enhance the mathematical formulation (Properties 1 and 2), to define the structure of a solution in the heuristic algorithm (Property 1) and to build an initial solution for the heuristic algorithm (Property 3). Property 4 shows a link between the optimal solution of the TSP-rd(time) and the optimal solution of the TSP. In addition, an approximation algorithm is presented in Section 3.4.2.

We adopt the following notation. We let $t(S)$ denote the completion time and $d(S)$ the traveled distance of solution S. S^* denotes the optimal solution of the TSP-rd(time) and d_{TSP} the value of the optimal solution of the Traveling Salesman Problem (TSP) defined on graph G with t_{ij} as the cost of traversing an arc.

The TSP-rd(time) is NP-hard as it has the TSP as special case when all release dates are zero. Furthermore, the problem always admits feasible solutions. In particular, the solution that waits until time r_n and executes an optimal TSP tour is feasible and gives an upper bound $r_n + d_{TSP}$ on the value of the solution.

Properties 1 and 2 give some simple characteristics satisfied by at least one optimal solution. Property 3 presents a lower bound on the value $t(S^*)$ of the optimal solution. Property 4 shows that release dates may have a huge impact on the total distance traveled.

Property 1. *There exists an optimal solution with no waiting time after the departure of the first route.*

This property is derived from property 4.2 in Klapp et al. (2016) and thus we omit the proof.

Property 2. *There exists an optimal solution with exactly one route starting not earlier than time r_n.*

Proof: First, at least one route starts not earlier than r_n in every solution because customer n has to be served. Assume that several routes start not earlier than r_n in an optimal solution. Then, all the products are available when the first of these routes starts. Thus, they can be replaced by a single route restricted to the set of customers served in these routes without incrementing the traveling cost. □

Properties 1 and 2 will be used to reinforce the formulation in Section 3.4.3 and to define a solution in the heuristic algorithm.

Property 3. *Inequality $r_n + d_{TSP} \leq 2 \times t(S^*)$ holds and the ratio is tight.*

Proof: $t(S^*) \geq r_n$ and $t(S^*) \geq d_{TSP}$, which proves the inequality. We now demonstrate that the ratio of 2 is tight. Consider the example shown in Figure 3.1, where $n = 2$.

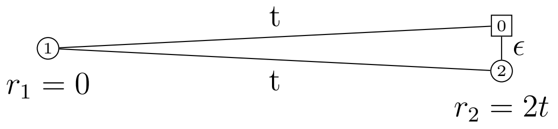


Figure 3.1: Property 3: an example.

On this example, $r_n = 2t$ and $d_{TSP} = 2t + \epsilon$. Solution S^* is the solution starting at time 0 with a route visiting customer 1 and continuing with a route visiting customer 2. Its completion time is $t(S^*) = 2t + 2\epsilon$. Setting ϵ to a small value, the ratio $\frac{r_n + d_{TSP}}{t(S^*)}$ can be as close as we want to 2, thus showing that the ratio is tight for $n = 2$. The example can be easily generalized to n customers. Set $t_{1i} = t_{i1} = t$ for

$i \in V \setminus \{1\}$, $t_{ij} = t_{ji} = \epsilon$ for $i, j \in V \setminus \{1\}$, $r_1 = 0$, $r_i = 2t$ for $i \in V \setminus \{1\}$. Again $r_n = 2t$ and $d_{TSP} = 2t + (n-1)\epsilon$. Solution S^* serves customer 1 in a first route and all remaining customers in a second route: $t(S^*) = 2t + n\epsilon$. When ϵ tends to 0, $\frac{r_n + d_{TSP}}{t(S^*)}$ tends to 2. $\square$

Property 3 provides a lower bound for the value $t(S^*)$ but also a performance guarantee for the initial solution of the proposed heuristic, introduced in Section 3.5.1.

Property 4. *Inequality $d(S^*) \leq n \times d_{TSP}$ holds and the ratio is tight.*

Proof: Solution S^* has at most n routes and, because of the triangle inequality, the distance traversed by any route is not larger than d_{TSP}. Consequently, $d(S^*) \leq n \times d_{TSP}$, which permits to conclude that the inequality holds. Let us now show that the ratio is tight, by considering the example shown in Figure 3.2 where $n = 2$.

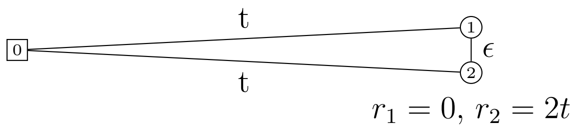


Figure 3.2: Property 4: an example.

On this example, $d_{TSP} = 2t + \epsilon$. Solution S^* is formed by a first route starting at time 0 and visiting customer 1 and a second route starting at time $2t$ and visiting customer 2, with value $t(S^*) = 4t$. The traveled distance in solution S^* is $d(S^*) = 4t$. Setting ϵ to a small value, the ratio $\frac{d(S^*)}{d_{TSP}}$ can be as close as we want to 2, thus showing that the ratio is tight for $n = 2$. The example can be easily generalized to n customers. Set $t_{0i} = t_{i0} = t$ for $i \in V \setminus \{0\}$, $t_{ij} = t_{ji} = \epsilon$ for $i, j \in V \setminus \{0\}$ and $r_i = 2(i-1)t$ for $i \in V \setminus \{0\}$. Now, $d_{TSP} = 2t + (n-1)\epsilon$. Solution S^* is given by n successive single-customer routes starting at time 0, with completion time $t(S^*) = 2nt$ and traveled distance $d(S^*) = 2nt$. When ϵ tends to 0, $\frac{d(S^*)}{d_{TSP}}$ tends to n. $\square$

The proof of Property 4 is helpful in understanding the impact of the release dates on the total traveled distance $d(S^*)$. In fact, release dates may have a big impact on the value of the optimal solution of the problem, as it is shown in the analysis of the computational results (see Section 3.6.2).

3.4.2 Approximation algorithm

The following approximation algorithm for the TSP-rd(time) relies on the well-known Christofides $\frac{3}{2}$-approximation algorithm for the TSP.

Algorithm 1 Approximation algorithm for TSP-rd(time)

```
1: Relax release dates and apply Christofides $\frac{3}{2}$-approximation algorithm
2: Start the TSP tour obtained at time $r_n$
```

Theorem 3.4.1. *Algorithm 1 has a performance guarantee of 2.5 for the TSP-rd(time) and the ratio is tight. Its running time is $O(n^3)$.*

Proof: $t(S^*) \geq r_n$ and $t(S^*) \geq d_{TSP}$. Thus, $r_n + \frac{3}{2}d_{TSP} \leq 2.5t(S^*)$. Furthermore, we know that the completion time of the solution given by Algorithm 1 is not larger than $r_n + \frac{3}{2}d_{TSP}$, which proves the theorem. We now show that the ratio is tight. Consider a graph with $2k+2$ vertices. The first $2k+1$ are arranged as shown in Figure 3.3, with k vertices at the upper level and $k+1$ vertices at the bottom level (where one vertex, vertex 0, is the depot). The final vertex, vertex n, overlaid to vertex 0, i.e., $t_{0n} = t_{n0} = 0$. The solid edges have distance one and the dashed ones have distance $1+\epsilon$. Let the release dates be $r_i = 0$, $i = 1, \ldots, 2k+1$ and $r_n = d_{TSP}$. In addition, let there be an edge connecting vertex 0 and vertex $k+1$ at cost $k(1+\epsilon)$.

The optimal tour for the TSP on this graph is composed by the dashed edges and the rightmost and leftmost solid edges. It has cost $d_{TSP} = (2k-1)(1+\epsilon)+2$. The

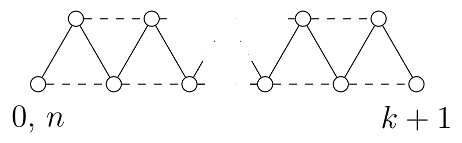


Figure 3.3: Theorem 3.4.1: an example.

solution of the Christofides algorithm consists of the solid edges, i.e., the minimum spanning tree of the graph, and the edge connecting vertices 0 and $k+1$. It has cost $d_{CH} = 3k + \epsilon$. The optimal solution of the TSP-rd(time) is composed of one route starting at time 0 and serving all customers except customer n followed by a second route serving customer n. This solution has $t(S^*) = (2k-1)(1+\epsilon)+2$. The solution of Algorithm 1 has $t(S_{Alg.1}) = (2k-1)(1+\epsilon)+2+3k+\epsilon$. Setting ϵ to a small value, as k increases the ratio $\frac{t(S^*)}{t(S_{Alg.1})} \approx \frac{5}{2}$. The running time of Algorithm 1 corresponds to the one of Christofides algorithm which is $O(n^3)$. $\square$

3.4.3 Mathematical formulation

We propose a 3-index formulation for the TSP-rd(time), with flow variables indexed by the route in which the edge is traversed. We thus introduce the set of routes $K = \{1, \ldots, |K|\}$, where $|K|$ is an upper bound on the optimal number of routes. A simple upper bound is n, but it leads to a weak formulation with a large number of variables. In the following section, we present a way to determine a heuristic value for $|K|$. Also, empty routes are allowed, meaning that there may be a subset of the routes in K visiting the depot only. Thus, the number of routes effectively used to serve customers is given by $|K|$ minus the number of empty routes.

The formulation relies on the following decision variables:

- $x_{ij}^k = \begin{cases} 1 \text{ if route } k \in K \text{ travels through edge } (i,j) \in A, \\ 0 \text{ otherwise,} \end{cases}$
- $y_i^k = \begin{cases} 1 \text{ if route } k \in K \text{ visits vertex } i \in N, \\ 0 \text{ otherwise,} \end{cases}$
- $\tau_{start}^k =$ the starting time of route $k \in K$,
- $\tau_{end}^k =$ the ending time of route $k \in K$.

In addition, flow variables u_{ij}^k are added to enforce subtour elimination. Note that $x_{00}^k = 1$ means that the route k is an empty route as it visits no customers.

The resulting model is:

$$\min \tau_{end}^K \tag{3.1}$$

s.t.

$$\sum_{k \in K} y_i^k = 1 \qquad i \in N, \tag{3.2}$$

$$\sum_{j \in V} x_{ij}^k = \sum_{j \in V} x_{ji}^k = y_i^k \qquad i \in V, k \in K, \tag{3.3}$$

$$\sum_{j \in V} u_{ji}^k - \sum_{j \in V} u_{ij}^k = y_i^k \qquad i \in N, k \in K, \tag{3.4}$$

$$u_{ij}^k \leq (n-1) x_{ij}^k \qquad (i,j) \in A, k \in K, \tag{3.5}$$

$$\tau_{end}^k = \tau_{start}^k + \sum_{(i,j) \in A} t_{ij} x_{ij}^k \qquad k \in K, \tag{3.6}$$

$$\tau_{end}^k \leq \tau_{start}^{k+1} \qquad k \in K \setminus \{|K|\}, \tag{3.7}$$

$$\tau_{start}^k \geq r_i y_i^k \qquad k \in K, i \in N, \tag{3.8}$$

$$\tau_{end}^k = \tau_{start}^{k+1} \qquad k \in K \setminus \{|K|\}, \tag{3.9}$$

$$\tau_{start}^k \leq r_n \qquad k \in K \setminus \{|K|\}, \tag{3.10}$$

$$x_{ij}^k \leq 1 - x_{00}^k \qquad (i,j) \in A, k \in K \setminus \{|K|\}, \tag{3.11}$$

$$x_{00}^k \geq x_{00}^{k+1} \qquad k \in K \setminus \{|K|\}, \tag{3.12}$$

$$x_{ij}^k \in \{0,1\} \qquad (i,j) \in A, k \in K, \tag{3.13}$$

$$y_i^k \in \{0,1\} \qquad i \in V, k \in K, \tag{3.14}$$

$$\tau_{start}^k, \tau_{end}^k \geq 0 \qquad k \in K, \tag{3.15}$$

$$u_{ij}^k \geq 0 \qquad (i,j) \in A, k \in K. \tag{3.16}$$

The objective function (3.1) minimizes the ending time of the last route. This is equivalent to the minimization of the total completion time computed as the total traveling time plus the waiting time at the depot. Constraints (4.2) ensure the visit of all customers. Constraints (4.3)-(4.5) impose that each route is a circuit connected to the depot. In particular, constraints (4.5) generate a flow that decreases while the vehicle visits customers, which prevents subtours. This set of constraints has been first proposed in Gavish and Graves (1978) and its performance has been assessed in Öncan et al. (2009). Constraints (4.11)-(4.12) set the relationship between time variables τ_{start}^k and τ_{end}^k. Constraints (4.13) establish that a route cannot start before the last release date of the customers served in it. Constraints (3.9) and (3.10) are added to reinforce the formulation. Constraints (3.9) follow from Property 1. Constraints (3.10) are deduced from Property 2. Constraints (4.6) and (4.7) are symmetry breaking constraints. In particular, constraints (4.6) set $x_{00}^k = 1$ if route k is empty, $x_{00}^k = 0$ otherwise, and constraints (4.7) impose that all the empty routes, if any, must precede all the non-empty routes, i.e., if route k is empty then all routes k' must be empty, with $k' = 1, \ldots, k-1$.

3.5 An iterated local search algorithm for the TSP-rd(time)

In this section we describe the heuristic algorithm we devised for the solution of the TSP-rd(time). It is an Iterated Local Search (ILS) algorithm which combines a local search phase with a perturbation phase (see Lourenço et al. (2010) for a detailed description of the ILS scheme). In particular, the perturbation phase is performed through a destroy-and-repair procedure.

The general scheme of the approach is presented in Algorithm 2. The idea is to build an initial solution through a construction phase and to iteratively find new solutions through destroy-and-repair (DR) followed by local search (LS). DR takes as input the current solution S as well as a parameter α which determines the size of the perturbation. The solution S' found by DR is given as input to LS. Finally, S_{best} represents the best solution found by the algorithm.

Function UpdateBest(S_{best}, S) updates the best solution found in case $t(S) < t(S_{best})$ while the other functions are described in Sections 3.5.1–3.5.3.

The ILS makes use of an estimation of the unknown cardinality of the set of routes K. We call C_K this value:

$$C_K = 1 + \left\lfloor \frac{r_n \times (n-1)}{\sum_{i \in N}(t_{0i} + t_{i0})} \right\rfloor . \tag{3.17}$$

The rationale behind this formula is that the number of routes is 1 when $r_n = 0$ while the number of routes could be n when the maximal release date is equal to $\sum_{i \in N}(t_{0i} + t_{i0})$. Indeed, in the latter case, the optimal solution would have n routes if the values of release dates are $r_j = \sum_{i=1}^{j}(t_{0i} + t_{i0})$. Note that C_K is not a valid upper bound on the number of routes, as shown by the following example. Let $\sum_{i \in N}(t_{0i} + t_{i0}) > r_n \times (n-1)$, $r_1 = 0$, $t_{01} = t_{10} = 1$, and $r_2 = 2$; therefore $r_i \geq 2$ for $i > 2$.

Algorithm 2 The ILS algorithm for the TSP-rd(time)

```
1: $S \leftarrow$ InitialSolution
2: $S_{best} \leftarrow S$
3: $\alpha \leftarrow$ InitializeDRParameter
4: **repeat**
5:     $S' \leftarrow$ DR($S$, $\alpha$)
6:     **if** a solution is found **then**
7:         $S \leftarrow$ LS($S'$)
8:         $S_{best} \leftarrow$ UpdateBest($S_{best}$, $S$)
9:         $\alpha \leftarrow$ UpdateDRParameter($\alpha$, $S_{best}$, $S$)
10:    **end if**
11: **until** the ending conditions are satisfied
12: return $S_{best}$
```

It follows that $C_K = 1$ while the upper bound on the number of routes is at least 2. Despite this and considering the heuristic nature of the algorithms, the C_K value has been kept instead of the upper bound equal to n mentioned in the previous section which proved to be extremely weak.

A solution S is composed by a vector of C_K routes. Because of Property 1, there is no waiting time between populated routes and all empty routes occupy the first positions of the vector, while populated routes are all at the end.

3.5.1 Initial solution

The initial solution is built by solving the TSP on the complete graph G with the Lin-Kernighan (LK) heuristic, described in Lin and Kernighan (1973), in the implementation provided by Helsgaun (2000). The starting time of this single route is then set to r_n. This solution is feasible and has a completion time greater than or equal to

$r_n + d_{TSP}$.

3.5.2 Destroy-and-repair

The Destroy-and-Repair (DR) procedure works as follows. Given a solution S and an integer α, the DR first applies a destroy operator that removes a set $\tilde{N}$ of α customers from S. A repair operator is used to reassign customers from $\tilde{N}$ to a route, either empty or populated, in S. A final step optimizes each route in S.

The parameter α is initialized to a lower bound $\alpha_{\min}$ and has upper bound equal to $\alpha_{\max}$. The initial value of α is set to $\alpha_{\min}$ and is updated after each successful destroy-and-repair call. If an improving solution is found by the local search, the value is reset to $\alpha_{\min}$, otherwise α is increased by one. If the new value exceeds the upper bound $\alpha_{\max}$, then it is set to $\alpha_{\min}$. The updating of α is described in Algorithm 3.

Algorithm 3 UpdateDRParameter(α, S_{best}, S)

```
1: if $t(S) = t(S_{best})$ then
2:     $\alpha = \alpha_{\min}$
3: else
4:     $\alpha = \alpha + 1$
5:     if $\alpha > \alpha_{\max}$ then
6:         $\alpha = \alpha_{\min}$
7:     end if
8: end if
```

In order to choose the customers that are removed, we first rank all customers on the basis of one of the following two criteria:

- the starting time saving obtained by removing a customer from the corresponding route, i.e., how much the starting time of the route decreases if the vertex is removed from it;

- the classical detour saving, without taking into account changes in the starting time of the route.

Each time the DR is called, one of these two criteria is chosen at random. Then, α customers are removed as follows: customers are considered sequentially in the order of the ranking and each customer is removed with a probability equal to 50% until α customers are removed.

The assignment of the vertices from $\tilde{N}$ to the routes of S is accomplished by a repair operator. Two different operators are used which give rise to two variants of the heuristic: the first one is based on the mathematical formulation proposed in Section 3.4.3, called MIP hereafter, while the second one is a random repair approach.

MIP repair

Given $\tilde{N}$, a mixed integer linear program is obtained from MIP with the following two changes. First, we force the clustering of customers as in S, except for the removed customers. A cluster is given by the customers served in the same route. In order to do that, the following constraints are added to MIP:

$$\sum_{i \in N_k \setminus \tilde{N}} y_i^k = |N_k \setminus \tilde{N}| y_t^k \qquad k \in K, t \in N_k \setminus \tilde{N}, \tag{3.18}$$

where N_k is the set of customers visited in route k in S. Constraints (3.18) guarantee to preserve the clustering of customers of the current solution (except for the removed customers) without fixing the index k of the routes. In this way we can anticipate or postpone clusters with respect to S. Second, we relax the integrality condition on the edge variables. We call this problem *modified MIP*.

A maximum time limit is given to the solution of the modified MIP. Thus, the MIP repair operator is successful if a feasible solution is found within the time limit. The solution of the modified MIP provides a clustering for the customers. The routes are computed by applying the LK heuristic on each cluster and a solution is then obtained by computing the earliest possible starting time for each route and by scheduling the routes in the increasing order of these values.

Random repair

This repair operator randomly assigns each vertex in $\tilde{N}$ to a cluster N_k, $k \in K$, different from the cluster of the vertex in S. As in the MIP repair, a solution is then built by applying the LK heuristic on each cluster, sorting the obtained routes by the largest release date and computing the earliest possible starting time for each route. The aim of the random repair operator is to quickly find a feasible solution. Note that the solution found by the random repair operator is always feasible, contrary to what may happen with the MIP repair where the modified MIP may run out of time without finding any feasible solution.

We call MathTSPrd the algorithm originating from the scheme described in Algorithm 2 when the MIP repair operator is used, HeuTSPrd the one where the random repair operator is used.

3.5.3 Local Search

After each DR move or, in the case of the MIP repair, each time a feasible solution is found, the Local Search (LS) is applied. The proposed LS is defined by seeking the first improvement in a neighborhood of four possible moves:

- *Vertex relocation*: a vertex is removed from its current route and relocated in a different route, either empty or populated. The new vertex position in the route is found through cheapest insertion.

- *Depot insertion*: a depot visit is added to a route, splitting it in two different routes. This move is not applied in case the number of populated routes is equal to C_K.
- *Depot shift*: a visit to the depot between two routes is either moved forward or backward in the solution. Let r and r' be two consecutive routes and let us consider the shift of the depot visit between r and r'. When the depot visit is moved forward, the set of vertices visited at the beginning of r' and before the new position of the depot will be shifted to r. On the contrary, when the depot visit is moved backward, the set of vertices visited at the end of r and before the new position of the depot will be shifted to r'. Backward shifting is evaluated up to the case where route r visits one customer only. Similarly, forward shifting is evaluated up to the case where r' visits one customer only.
- *Depot removal*: a depot visit between two routes is removed, thus merging the two routes.

All moves are evaluated by the corresponding value of the objective function, i.e., the ending time of the last route, therefore potentially accounting for savings in both the travelled distance and the starting time of each route. A scheme of the LS is presented in Algorithm 4:

Note that, while the vertex relocation move is quite standard in heuristics for routing problems, moves involving the depot are instead specific for the TSP-rd(time) as they have an impact on the starting time of each route.

Algorithm 4 LS algorithm

```
1: repeat
2:   Randomly sort all vertices (including the depot)
3:   for $i = 0$ to $n$ do
4:     if the i-th vertex is the depot then
5:       Evaluate the *depot insertion*, *shift*, and *removal* moves
6:     else
7:       Evaluate *vertex relocation* move
8:     end if
9:     if an improving move is found then
10:      Apply the improving move
11:    end if
12:  end for
13: until no improving solution is found
14: Apply LK to each populated route
```

3.6 Computational experiments

In this section computational results are presented for instances derived from benchmark TSP instances. In Section 3.6.1 we describe how the instances have been generated while computational results are presented in Section 3.6.2.

3.6.1 Instances

The instances have been derived from Solomon instances (Solomon (1987)) and from the "TSPLIB" library (Reinelt (1991)), and are characterized by three values:

- n: the cardinality of the vertex set (not including the depot);
- d_{TSP}: the optimal TSP value on the resulting graph;
- β: the width of the interval in which release dates are defined.

Solomon's instances are composed by 6 sets of instances: $C1$, $C2$, $R1$, $R2$, $RC1$ and $RC2$. All the instances in the same set have the same coordinates of the vertices and differ in time windows only. As we do not consider time windows, we kept one instance from each set. In addition, we discarded $R2$ and $RC2$ as they have the same coordinates as $R1$ and $RC1$, respectively. The four remaining problem sets allow us to assess the performance of the heuristic when vertices are spread according to different schemes.

Given an instance $\mathcal{I}$ from Solomon (1987), the instance $\mathcal{I}\_n\_\beta$ for the TSP-rd(time) is generated as follows:

- data from the Solomon's instance are truncated after $n + 1$ vertices;
- the first vertex of the instance is set to be the depot;
- let i be the index of the vertices in the instance (the depot receives index $i = 0$, the first vertex index $i = 1$ and so on). Its release date is defined as $r_i = \lfloor \beta \times d_{TSP} \times Y \rceil$, where $Y \sim U(0, 1)$, therefore assigning to each customer a uniformly random integral release date in $[0, \beta \times d_{TSP}]$.

Instances have been generated with $\beta \in \{0.5, 1, \ldots, 3\}$ resulting in 24 instances for each value of n.

From the TSPLIB we have chosen symmetric instances with less than 500 vertices, explicit vertex coordinates, edge weights computed in the two dimensional Euclidean space, and with no constraints on the TSP solution. The resulting selection comprises a total of 42 instances, reported in Table 3.1 according to the instance size.

We also tested the ILS algorithm on asymmetric instances in order to check whether the behavior of the two variants of the algorithm is affected by the asymmetry of the distance matrix. The following five asymmetric instances have been chosen:

`ftv33`, `ft53`, `ftv70`, `kro124p`, and `rbg403`.

50 – 100 nodes	101 – 150 nodes	151 – 250 nodes	251 – 500 nodes
eil51	eil101	pr152	gil262
berlin52	lin105	u159	pr264
st70	pr107	rat195	a280
eil76	pr124	d198	pr299
pr76	bier127	kroA200	lin318
rat99	ch130	kroB200	rd400
kroA100	pr136	ts225	fl417
kroB100	pr144	tsp225	pr439
kroC100	ch150	pr226	pcb442
kroD100	kroA150		d493
kroE100	kroB150		
rd100			

Table 3.1: The symmetric instances selected from the TSPLIB, by instance size.

As for Solomon instances, the first vertex is set for the depot. Then, the same procedure is used to generate the release dates, with $\beta \in \{0.5, 1, \ldots, 3\}$. Thus, we obtain 252 symmetric and 30 asymmetric instances in total.

All the instances can be downloaded at `http://or-brescia.unibs.it/instances`.

3.6.2 Results

This section is organized as follows. In the first part we identify the largest instances that can be solved to optimality. Secondly, we focus on the tuning of the parameters of the MathTSPrd, investigating how the time limit allowed to solve each modified MIP and the range of α affect the quality of the solution. Finally, the performance of the ILS is analyzed: firstly on small instances, in comparison with the optimal solution found by solving the formulation in Section 3.4.3 and secondly by comparing the two variants of the ILS on larger instances, generated both from Solomon's and the TSPLIB instances.

Note that, throughout this section, the ending conditions for Algorithm 2 are the maximum time allowed for each run and the maximum number of iterations without improvement. Unless otherwise specified, the maximum time is set to ten minutes and the maximum number of iterations is set to one thousand. For all the tested instances, the time required to build the initial solution for the ILS is below one second. Thus, it does not have an impact on the performance of the ILS.

The ILS was implemented in C++ and run using a 3.5 GHz Intel Xeon E5-1650v2 processor with 64 GB of RAM, using CPLEX 12.6 as MILP solver, when necessary.

Largest instances solved to optimality

To determine the largest instance size for which CPLEX finds an optimal solution to formulation (3.1)-(4.10) within one hour, instances have been generated with $n \in \{10, 15, 20, 25, 30\}$ starting from Solomon instances. A total of 24 instances are generated for each size, given by the combination between the values of β, and the 4 Solomon instances (C101, C201, R101, RC101). As Table 3.2 shows, the maximum size for which the optimal solution is obtained within one hour is $n = 20$ customers.

For $n = 25$ only 13 instances out of 24 are solved to optimality. For $n = 30$ the number of optimal solutions further decreases to 7.

n	# optimal solutions
10	24
15	24
20	24
25	13
30	7

Table 3.2: Number of instances solved to optimality.

Tuning of the parameters

The parameters to be tuned in the MathTSPrd are the time limit for each MIP repair and the range for the number α of customers to be removed by the DR. Note that the parameter α has also an impact on the HeuTSPrd. Thus, it was fine tuned on the MathTSPrd and the best value was kept for HeuTSPrd as well. The tuning has been carried out on the instances derived from Solomon with $n \in \{20, 50, 100\}$.

The impact of changes in the time limit parameter for each MIP repair solution in the MathTSPrd algorithm are reported in Table 3.3 for $n = 20$ instances, Table 3.4 for $n = 50$ instances and Table 3.5 for $n = 100$ instances. The results have been obtained with the bounds for α set to $\alpha_{\min} = \min\{5, \lfloor n/10 \rfloor\}$ and $\alpha_{\max} = \max\{\min\{10, n-1\}, \lfloor n/2 \rfloor\}$ and are reported for instances grouped by the value of β. The column "Avg. % Best" reports the average percentage gap with respect to the best solution found for each instance group. "Avg. # MILP" indicates the average number of modified MIPs successfully solved, i.e., the average number of modified MIPs for which a feasible solution has been found within the time limit. The best results have been found for the time limit equal to $n/10$ seconds when $n = 20$ and $n = 50$, corresponding to 2 and 5 seconds, respectively, and to $n/5 = 20$ seconds when $n = 100$. Note also that, while on instances with $n = 20$ the difference in the performance of MathTSPrd for different values of the MIP repair time limit is negligible, the value shows an increase when $n = 50$ and becomes remarkable when $n = 100$. Thus, the main message we may keep from these results is that the MIP repair needs more time to be effective on large instances.

	MILP: n/50 = 0.4s		MILP: n/10 = 2s		MILP: n/5 = 4s		MILP: n/2 = 10s	
β	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP
0.5	0	1017.25	0	974	0	974.75	0	974.25
1	0	1053.00	0	560.00	0	517.25	0	505.75
1.5	0.10	927.75	0	378.25	0.10	275.50	0.10	228.50
2	0.09	879.75	0	322.25	0.18	215.75	0.18	148.75
2.5	0	843.00	0	295.50	0	193.25	0.04	124.00
3	0	796.25	0.03	285.75	0	178.00	0	111.25
Avg.	0.03		0.01		0.04		0.05	

Table 3.3: Results on the $n = 20$ instances for different values of the MILP time limit.

We also investigated how the range of α, the number of vertices removed and inserted by the DR, affects the performance of the MathTSPrd algorithm. The value of $\alpha_{\min}$ has been fixed to $\alpha_{\min} = \min\{5, \lfloor n/10 \rfloor\}$ while $\alpha_{\max}$ has been set to $\alpha_{\max} = \max\{\min\{10, n-1\}, \lfloor \rho \rfloor\}$ and different values have been tested for ρ, i.e., $\rho \in \{n/2, n/4, n/8\}$. Table 3.6 presents the results for the $n = 50$ instances and Table 3.7 for the $n = 100$ instances. In both tables the MILP time limit chosen is the one

	MILP: n/50 = 1s		MILP: n/10 = 5s		MILP: n/5 = 10s		MILP: n/2 = 25s	
β	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP
0.5	0.08	365.00	0.13	126.00	0.24	69.25	0.20	54.00
1	0.03	254.25	0.92	89.75	0.73	58.50	0.87	30.00
1.5	0.17	266.00	0.53	90.50	0.50	51.00	1.13	25.50
2	0.47	245.00	0.34	95.25	1.26	50.50	1.55	22.50
2.5	1.97	153.25	0.18	94.50	0.20	53.00	1.02	22.25
3	3.35	12.75	0.26	94.50	0.08	52.50	0.30	23.25
Avg.	1.01		0.39		0.50		0.85	

Table 3.4: Results on the $n = 50$ instances for different values of the MILP time limit.

that has shown the best results in the previous experiments, i.e., 5 seconds and 20 seconds, respectively. The results show an improving trend as ρ decreases hence the default value for the different runs reported hereafter has been fixed to $\rho = n/8$.

Comparison with optimal solutions

Tables 3.8, 3.9, and 3.10 report the results of the comparison between the ILS and the optimal solution. In particular, we compare the optimal solution with the value of the initial solution (computed as a TSP tour starting at the latest release date), the value of the solution provided by the MathTSPrd and the value of the solution provided by HeuTSPrd. The comparison is made on instances with up to 20 customers. The column "% Exact" reports the percentage gap of the corresponding "Value" column to the exact solution value. The time limit for the modified MIP in the MIP repair operator is set to $n/10 = 2$ seconds. The tables shows that on these instances the

β	MILP: n/50 = 2s		MILP: n/10 = 10s		MILP: n/5 = 20s		MILP: n/2 = 50s	
	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP
0.5	1.33	19.50	0.79	30.00	0.48	26.00	1.59	15.50
1	3.23	1.75	1.03	7.00	1.28	10.00	0.14	9.00
1.5	4.72	1.00	1.37	13.25	0.02	12.00	0.23	8.25
2	2.35	1.00	0.35	11.75	1.22	12.50	1.18	8.25
2.5	4.47	1.00	1.19	8.50	0	18.00	0.93	10.75
3	7.00	0.75	1.21	4.00	0	15.75	0.94	10.25
Avg.	3.85		0.99		0.50		0.84	

Table 3.5: Results on the $n = 100$ instances for different values of the MILP time limit.

two algorithms have a good performance with all the instances of size $n = 10$ solved to optimality and with no percentage gap above 1% in the larger instances. The HeuTSPrd shows a slight advantage, finding the optimal solution in all but one of the 24 instances in both the instances of size $n = 15$ and $n = 20$. The MathTSPrd finds the optimal solution in 22 instances of size $n = 15$ and in 22 instances of size $n = 20$. Table 3.10 also reports the running time, in seconds, required to find the incumbent solution by MathTSPrd and HeuTSPrd, respectively. The computing time, in seconds, is reported in the "TI" column. It can be observed that, while the MathTSPrd is shown to be the fastest on average, the HeuTSPrd has the least maximum time.

By looking at the results presented in Tables 3.8-3.10, it is also interesting to note the impact of release dates on the value of the optimal solution of the TSP-rd(time).

β	$\rho = n/2$ Avg. % Best	$\rho = n/2$ Avg. # MILP	$\rho = n/4$ Avg. % Best	$\rho = n/4$ Avg. # MILP	$\rho = n/8$ Avg. % Best	$\rho = n/8$ Avg. # MILP
0.5	0	127.75	0	158.00	0.25	175.00
1	0.87	93.25	0.19	102.25	0.40	109.00
1.5	0.45	89.00	0.76	96.50	0.21	97.25
2	0.16	95.25	0.17	96.50	0.47	99.25
2.5	0.50	94.50	0.50	96.75	0.11	97.25
3	0.55	96.00	0.30	96.00	0.02	100.75
Avg.	0.42		0.32		0.25	

Table 3.6: Results on the $n = 50$ instances for different values of ρ. MILP time limit is set to 5 seconds for all runs.

To better highlight this impact, in Table 3.11 we report the average percentage increase in the value of the optimal solution of the TSP-rd(time) for different values of β with respect to the case with $\beta = 0.5$. Each cell reports the average increase over the four instances with the value of β of the corresponding row and the value of n of the corresponding column. We can notice that the more the release dates are spread out, the more the value of the optimal solution increases with a maximum increase of nearly 130% when $\beta = 3$. This is due to the fact that more routes are needed when release dates are more dispersed. In addition, the increase in the objective function seems to be not affected (or only slightly affected) by the size of the instance.

	$\rho = n/2$		$\rho = n/4$		$\rho = n/8$	
β	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP	Avg. % Best	Avg. # MILP
0.5	1.06	23.50	1.20	23.50	0.13	26.50
1	0	9.50	0	9.75	0	10.50
1.5	0	12.00	0	12.00	0	12.75
2	0	12.75	0	12.50	0	12.00
2.5	0	18.00	0.33	17.75	0	17.00
3	0.28	15.575	0.20	15.75	0.58	14.50
Avg.	0.22		0.29		0.12	

Table 3.7: Results on the $n = 100$ instances for different values of ρ. MILP time limit is set to 20 seconds for all runs.

Comparison of the two variants of the heuristic

A comparison of the two variants has also been carried out on larger instances. Tables 3.12 and 3.13 presents the results for the Solomon instances with $n = 50$ and $n = 100$, respectively. The column “TI” identifies the time in seconds after which the incumbent solution has been found and the column “TI MathTSPrd” reports the time at which HeuTSPrd has found the first solution at least as good as the one found by MathTSPrd. In both the instances with $n = 50$ and $n = 100$ it can be observed that the MathTSPrd is outperformed by the HeuTSPrd in all but one instance. In the instances of size $n = 50$ the average percentage gap of the MathTSPrd from the HeuTSPrd is 0.86%. The best solutions of the MathTSPrd are found in an average time of 265 seconds, while the HeuTSPrd takes an average time of 43 seconds to find the first solution at least as good, if any. For the instances of size $n = 100$ the average percentage gap of the MathTSPrd from the HeuTSPrd is 3.33%. The MathTSPrd finds the best solution in an average time of 246 seconds, while the HeuTSPrd takes an average time of 47 seconds to match the result of the MathTSPrd, when this is achieved.

The number of routes of each solution is reported in the column "# Routes". The value appears to follow a positive trend with respect to the value of the paramenter β. The trend is shared by both the solutions found by the MathTSPrd and the HeuTSPrd version of the heuristic. This leads to the conclusion that the more the release dates get spread out in time compared to the cost of visiting the vertices, measured as the cost of the TSP solution on the instance, the more it is convenient to visit customers as soon as they are available. To further confirm these results, the two variants of the ILS have been compared with a "myopic" heuristic visiting customers as they become available, on instances of $n = 50$ vertices. This heuristic works as follows: every time the vehicle reaches the depot, it either departs to serve the customers whose goods arrived while the vehicle was traveling or waits for the next customer that can be served. The results show the benefit of the ILS compared to the intuitive rule of serving customers as soon as they become available. The "myopic" algorithm performs on average 16% worse than the best solution, with the worst case taking 33.25% more time to complete the distribution than the solution found by the HeuTSPrd version. The percentage gap from the best solution has a negative trend with respect to the parameter β, confirming the conclusion drawn above.

An indicator of the operational implications of the solution is given in the column "% Wait" where the percentage of time spent waiting at the depot over the total time required to complete the distribution is reported. We observe that in most of the tested instances the waiting time is less than 10% of the total service time and only one solution is found with more than 15% of waiting time for the instances with size $n = 50$ and $n = 100$. The average percentage waiting time appears to decrease as the instance size increases. In particular, focusing on the solution found by the HeuTSPrd, the value goes from 6.51% to 4.29% when the instance size goes from $n = 50$ to $n = 100$. We also observe that the average waiting time of the HeuTSPrd is greater that the one of the MathTSPrd for both the instances of size $n = 50$ and $n = 100$, indicating that, on average, the best solution known for each instance has a higher waiting time to total service time ratio. A further comparison with the "myopic" heuristic can be carried out on instances with $n = 50$. The "myopic" heuristic has a very little waiting time component but the solutions found with this method are poor compared to those found by the HeuTSPrd, as discussed above. This validates the conclusion that the choice of serving customers versus waiting for more goods to arrive at the depot is non-trivial and deserves a thorough analysis.

The results obtained indicate that the time required by the operator based on the MIP, the one adopted in the MathTSPrd, limits the number of iterations performed in the given computational time to a value that is much lower than the number of iterations performed when the random operator is adopted in the HeuTSPrd. This implies that exploring a broader portion of the solution space is more effective than spending more time to find better solutions to feed the LS.

		Optimal	Initial solution		MathTSPrd		HeuTSPrd	
β	Instance	Value	Value	% Exact	Value	% Exact	Value	% Exact
0.5	C101	79	79	0	79	0	79	0
	C201	186	189	1.61	186	0	186	0
	R101	217	244	12.44	217	0	217	0
	RC101	195	195	0	195	0	195	0
1	C101	101	102	0.99	101	0	101	0
	C201	221	245	10.86	221	0	221	0
	R101	261	316	21.07	261	0	261	0
	RC101	241	252	4.56	241	0	241	0
1.5	C101	113	125	10.62	113	0	113	0
	C201	250	300	20.00	250	0	250	0
	R101	323	388	20.12	323	0	323	0
	RC101	289	309	6.82	289	0	289	0
2	C101	134	149	11.19	134	0	134	0
	C201	297	356	19.87	297	0	297	0
	R101	373	460	23.32	373	0	373	0
	RC101	347	367	5.76	347	0	347	0
2.5	C101	157	172	9.55	157	0	157	0
	C201	338	412	21.89	338	0	338	0
	R101	432	532	23.15	432	0	432	0
	RC101	395	424	7.34	395	0	395	0
3	C101	180	195	8.33	180	0	180	0
	C201	386	467	20.89	386	0	386	0
	R101	495	604	22.02	495	0	495	0
	RC101	444	481	8.33	444	0	444	0
Avg.				12.12		0		0

Table 3.8: Comparison with optimal solutions for instances with $n = 10$.

		Optimal	Initial solution		MathTSPrd		HeuTSPrd	
β	Instance	Value	Value	% Exact	Value	% Exact	Value	% Exact
0.5	C101	145	145	0	145	0	145	0
	C201	228	241	5.70	228	0	228	0
	R101	288	327	13.54	288	0	288	0
	RC101	220	220	0	220	0	220	0
1	C101	176	189	7.39	176	0	176	0
	C201	271	314	15.39	271	0	271	0
	R101	361	426	18.01	364	0.83	361	0
	RC101	266	286	7.52	266	0	266	0
1.5	C101	217	7.37	9.23	217	0	217	0
	C201	327	387	18.35	327	0	327	0
	R101	427	524	22.72	427	0	427	0
	RC101	330	353	6.97	330	0	330	0
2	C101	261	277	6.13	261	0	261	0
	C201	375	460	22.67	375	0	375	0
	R101	509	623	22.40	510	0.20	510	0.20
	RC101	387	419	8.27	387	0	387	0
2.5	C101	304	321	5.59	304	0	304	0
	C201	446	533	19.51	446	0	446	0
	R101	572	722	26.22	572	0	572	0
	RC101	439	485	10.48	439	0	439	0
3	C101	347	365	5.19	347	0	347	0
	C201	517	605	17.02	517	0	517	0
	R101	663	820	23.68	663	0	663	0
	RC101	494	552	11.74	494	0	494	0
Avg.				12.60		0.04		0.01

Table 3.9: Comparison with optimal solutions for instances with $n = 15$.

		Optimal		Initial solution		MathTSPrd			HeuTSPrd		
β	Instance	Value	TI	Value	% Exact	Value	% Exact	TI	Value	% Exact	TI
0.5	C101	177	0	177	0	177	0	0	177	0	0
	C201	247	5	258	4.45	247	0	13	247	0	115
	R101	326	3	376	15.34	326	0	7	326	0	118
	RC101	305	3	313	2.62	305	0	1	305	0	129
1	C101	212	6	230	8.49	212	0	1	212	0	160
	C201	285	11	336	17.89	285	0	1	285	0	153
	R101	409	42	484	18.34	409	0	110	409	0	139
	RC101	374	15	407	8.82	374	0	62	374	0	160
1.5	C101	260	39	284	9.23	260	0	2	260	0	172
	C201	349	74	414	18.62	349	0	6	349	0	169
	R101	475	129	596	25.47	479	0.84	536	475	0	216
	RC101	422	25	502	18.96	422	0	12	422	0	177
2	C101	306	67	337	10.13	306	0	83	306	0	191
	C201	402	180	492	22.39	402	0	9	402	0	174
	R101	568	871	708	24.65	571	0.53	214	571	0.53	174
	RC101	508	77	596	17.32	508	0	17	508	0	192
2.5	C101	356	271	391	9.83	356	0	21	356	0	201
	C201	478	403	570	19.25	478	0	57	478	0	193
	R101	648	333	820	26.54	648	0	141	648	0	180
	RC101	586	97	690	17.75	586	0	46	586	0	199
3	C101	406	1426	444	9.36	406	0	5	406	0	199
	C201	551	1801	648	17.60	551	0	5	551	0	195
	R101	750	185	932	24.27	750	0	67	750	0	166
	RC101	654	110	785	20.03	654	0	2	654	0	184
Avg.			257.21		15.31		0.06	59.08		0.02	164.83

Table 3.10: Comparison with optimal solutions for instances with $n = 20$.

β	$n = 10$	$n = 15$	$n = 20$
0.5	100.00	100.00	100.00
1	121.71	121.91	121.95
1.5	144.02	147.67	143.61
2	170.01	173.89	170.23
2.5	195.27	199.89	197.33
3	222.30	229.40	225.29

Table 3.11: Growth of the average optimal value with respect to the value of β.

		MathTSPrd					HeuTSPrd						Myopic		
β	Instance	Value	% Best	TI	# Routes	% Wait	Value	% Best	TI	# Routes	% Wait	TI MathTSPrd	Value	% Best	% Wait
0.5	C101	333	0	284	3	11.71	333	0	190	3	11.71	190	412	23.72	0
	C201	415	0	7	3	6.47	415	0	3	3	6.27	3	553	33.25	0
	R101	594	1.02	408	5	2.02	588	0	59	5	3.91	3	691	17.52	0
	RC101	530	0	88	3	2.64	530	0	22	3	2.64	22	730	37.74	0
1	C101	422	0	126	5	3.55	423	0.24	34	4	4.73	-	547	29.62	0
	C201	534	1.14	74	5	2.43	528	0	60	5	1.14	15	675	27.84	0
	R101	745	2.34	270	6	4.83	728	0	407	7	1.10	1	851	16.90	0
	RC101	661	0	144	4	7.41	661	0	14	4	7.41	14	737	11.50	0
1.5	C101	523	0.58	29	7	1.53	520	0	262	6	4.62	101	623	19.81	0
	C201	668	1.21	349	6	2.25	660	0	2	7	3.48	2	787	19.24	0
	R101	893	1.48	245	9	4.70	880	0	188	9	1.02	105	973	10.57	0
	RC101	824	0.61	579	6	3.16	819	0	119	6	5.37	6	916	11.84	0
2	C101	625	1.46	141	7	9.28	616	0	200	8	4.55	3	713	15.75	0
	C201	793	1.67	410	8	5.04	780	0	418	8	10.77	133	875	12.18	0
	R101	1087	2.94	461	10	6.99	1056	0	519	10	6.82	3	1195	13.16	0.17
	RC101	978	1.14	21	7	7.98	967	0	40	8	6.93	13	1148	18.72	0.09
2.5	C101	719	0.56	279	10	1.25	715	0	80	8	15.10	16	797	11.47	0
	C201	921	0.22	503	11	3.37	919	0	148	10	2.07	148	997	8.49	0.10
	R101	1265	0.64	452	14	6.64	1257	0	27	13	9.39	26	1349	7.32	0.74
	RC101	1132	1.34	482	10	6.89	1117	0	175	8	6.27	6	1286	15.13	0.08
3	C101	823	1.11	387	13	6.32	814	0	178	12	11.67	5	853	4.79	0.11
	C201	1073	0.28	318	11	2.98	1070	0	182	11	12.71	33	1158	8.22	0.09
	R101	1462	0.62	172	18	5.75	1453	0	418	21	7.85	18	1513	4.13	1.19
	RC101	1284	0.23	128	10	5.84	1281	0	490	11	8.59	122	1384	8.04	0.07
Avg.			0.86	264.88		5.04		0.01			6.51	42.96		16.12	0.11

Table 3.12: Comparison of MathTSPrd and HeuTSPrd on instances with $n = 50$.

		MathTSPrd					HeuTSPrd					
β	Instance	Value	% Best	TI	# Routes	% Wait	Value	% Best	TI	# Routes	% Wait	TI MathTSPrd
0.5	C101	677	0	414	5	2.81	677	0	338	5	2.07	338
	C201	749	1.90	272	4	3.20	735	0	537	5	4.90	2
	R101	840	2.07	292	5	2.26	823	0	68	4	1.58	47
	RC101	860	2.50	600	5	0	839	0	112	5	5.60	10
1	C101	906	2.95	124	8	3.09	880	0	51	7	7.50	10
	C201	975	3.17	365	6	4.62	945	0	577	5	6.77	1
	R101	1078	7.37	208	10	3.53	1004	0	524	9	0.10	6
	RC101	1090	5.31	284	8	3.67	1035	0	78	7	0.10	1
1.5	C101	1111	3.25	209	12	7.20	1076	0	61	9	7.53	9
	C201	1192	4.29	484	10	4.70	1143	0	589	8	4.37	12
	R101	1286	4.64	9	12	3.89	1229	0	409	11	2.60	5
	RC101	1295	2.70	202	9	2.08	1261	0	406	10	0.87	4
2	C101	1351	4.97	10	16	2.37	1287	0	148	13	1.17	7
	C201	1412	3.44	229	10	4.18	1365	0	537	12	6.45	1
	R101	1478	0	8	16	3.38	1483	0.34	106	15	0.34	-
	RC101	1629	8.10	170	13	1.10	1507	0	155	11	1.66	2
2.5	C101	1519	0.73	56	17	3.75	1508	0	120	15	6.23	120
	C201	1619	2.86	235	16	2.53	1574	0	360	15	7.37	71
	R101	1746	1.45	174	20	6.30	1721	0	396	22	3.49	348
	RC101	1844	3.71	291	15	6.72	1778	0	12	13	4.05	12
3	C101	1757	3.35	286	22	4.67	1700	0	354	16	8.18	17
	C201	1858	2.77	494	20	5.33	1808	0	64	20	4.15	33
	R101	2056	3.26	10	19	10.02	1991	0	143	22	13.51	8
	RC101	2165	5.15	458	18	7.21	2059	0	529	18	2.28	2
Avg.			3.33	245.17		4.11		0.01			4.29	46.35

Table 3.13: Comparison of MathTSPrd and HeuTSPrd on instances with $n = 100$.

Given the better results of HeuTSPrd, we investigated the impact of the ending conditions on its performance, by extending the time limit to two hours and removing the limit on the number of iterations without improvement. The results of such runs are plotted in Figure 3.4 for instances with $\beta = 3$. The x-axis reports the time in seconds in logarithmic scale and the y-axis the percentage gap over the best solution found after two hours. The figure shows that it is not beneficial to give a longer maximum time limit to HeuTSPrd. In fact, in all instances the solution obtained after 10 minutes is less that 2.5% worse than the one obtained in two hours and the time required to improve the solution obtained after such time limit is often very large (we recall that time is on a logarithmic scale).

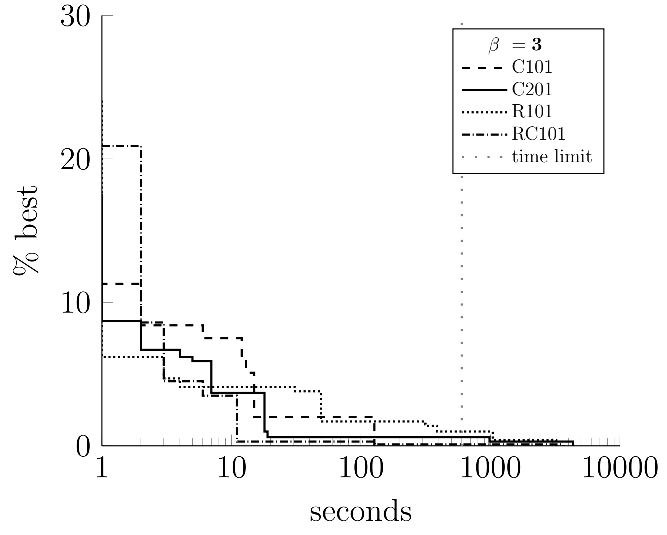


Figure 3.4: Plots of the percentage gap of the current best solution over the final solution found in the two hours run of HeuTSPrd on the instances with $n = 100$ and $\beta = 3$.

Further comparisons of the performance of the two variants have been carried out on the instances originated from the TSPLIB. The selected subset of instances spans from a minimum size of 50 vertices (i.e. `eil51`) to a maximum of 492 vertices (i.e. `d493`). In the MathTSPrd the time limit for the MILP is set to 20 second as this is the best value for large instances and a shorter time provides almost the same results on small instances, as shown in Section 3.6.2. Tables 3.14 and 3.15 report the results of the two algorithms on 252 instances for the different values of β. The column "HeuTSPrd Value" gives the value of the solution found by the HeuTSPrd algorithm and the column "% MathTSPrd" the percentage gap of the value of the solution found by the MathTSPrd over the corresponding value in the "HeuTSPrd Value" column to the left. In this set of instances the MathTSPrd is systematically outperformed by the HeuTSPrd. The average percentage gaps of the MathTSPrd from the HeuTSPrd does not show a specific trend.

Instance	$\beta = 0.5$		$\beta = 1$		$\beta = 1.5$		$\beta = 2$		$\beta = 2.5$		$\beta = 3$	
	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd
eil51	545	0.18	664	4.97	850	0.94	1027	1.46	1212	1.82	1386	1.08
berlin52	9984	1.75	12000	4.19	14930	1.15	18169	1.00	21529	1.04	24892	0.48
st70	911	0.44	1112	5.76	1364	4.40	1626	3.63	1917	2.66	2217	0.45
eil76	703	2.42	868	4.26	1027	4.87	1236	4.85	1456	2.20	1687	3.26
pr76	141425	1.99	172651	8.88	211749	3.56	255415	4.49	301773	0.69	345979	1.86
rat99	1664	1.44	2063	1.21	2523	1.55	3010	2.76	3527	1.90	4080	1.35
kroA100	28071	2.18	36167	2.21	44772	3.03	53563	4.40	62322	5.36	71122	3.62
kroB100	29868	1.13	37906	3.27	46076	1.60	54261	3.28	63010	4.35	72596	4.84
kroC100	27999	1.23	34929	4.14	43153	2.40	51722	1.80	60880	2.34	70322	2.61
kroD100	28715	1.02	36501	3.08	44956	3.34	53455	3.01	61877	7.33	71573	5.37
kroE100	29387	1.73	37219	4.15	45835	3.38	55079	2.86	63941	3.19	73748	2.61
rd100	10546	1.38	13458	2.21	16649	3.06	19928	2.66	23169	1.35	26738	1.52
eil101	823	0.97	1028	4.77	1233	1.62	1470	2.72	1728	4.22	2008	4.28
lin105	20153	0.29	25640	2.19	31358	2.29	37347	1.41	43711	2.75	50013	2.44
pr107	62841	0.28	81590	1.05	99526	2.04	117771	1.82	137778	1.64	157633	1.28
pr124	81592	0.24	105827	1.29	131301	2.67	154628	5.16	182232	2.21	209157	2.59
bier127	153870	1.32	190369	5.01	234997	8.39	283277	9.13	332902	8.58	389597	6.70
ch130	8238	3.45	10305	8.22	12806	4.24	15324	3.39	17826	4.02	20445	19.40
pr136	131354	2.39	164984	1.65	202170	2.77	241218	5.72	284208	4.13	328775	2.72
pr144	84662	1.68	111381	2.35	136360	4.55	162547	3.49	187253	5.45	213833	9.12

Table 3.14: Comparison of MathTSPrd and HeuTSPrd on the instances derived from TSPLIB.

	$\beta = 0.5$		$\beta = 1$		$\beta = 1.5$		$\beta = 2$		$\beta = 2.5$		$\beta = 3$	
Instance	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd	Heu-TSPrd Value	% Math-TSPrd
ch150	8814	3.14	11041	6.63	13670	7.24	16296	2.41	19045	3.52	22029	4.02
kroA150	35462	1.58	45060	4.40	55676	2.59	66775	3.50	79010	3.34	90724	2.01
kroB150	35269	0.73	44446	9.06	54459	5.70	65446	7.18	77219	2.46	88263	4.50
pr152	105710	0.32	137972	0.92	170156	2.42	203653	0.93	236474	2.94	267376	4.70
u159	57578	1.68	74511	2.19	90709	4.40	108865	3.59	126932	4.84	145806	5.43
rat195	3179	2.49	4042	5.24	4810	5.18	5861	6.69	6863	3.82	7867	1.91
d198	23045	2.66	29621	0.45	36595	2.09	43710	2.39	51107	2.65	58513	1.96
kroA200	40123	0.75	50710	6.21	61891	5.21	74533	2.93	86712	2.23	99878	3.78
kroB200	40037	1.71	51429	1.35	62562	2.69	74816	4.47	87065	6.03	99910	6.72
ts225	178901	2.57	229069	1.53	278429	5.72	334784	2.97	390320	5.22	446692	3.55
tsp225	5431	3.94	6839	4.87	8396	5.32	9819	10.41	11374	7.97	13348	5.60
pr226	111997	7.58	148556	2.85	184889	2.29	221846	1.24	259471	2.34	297197	1.36
gil262	3337	0.87	4200	5.33	5169	1.70	6087	5.06	7170	1,81	8226	1.92
pr264	69397	0.95	89983	2.12	110907	3.89	133099	2.47	154463	4.01	178378	5.14
a280	3626	5.38	4602	2.28	5503	5.91	6659	4.23	7740	5.63	8889	15.91
pr299	67752	0.55	86133	1.23	105509	2.54	125318	2.86	146635	1.90	167359	1.17
lin318	59008	1.52	75732	3.78	93531	1.52	111255	3.70	129062	13.84	147677	13.70
rd400	20649	10.94	26602	14.79	32719	16.64	38882	17.77	45556	17.26	52452	16.39
fl417	17519	1.49	22036	7.55	27321	8.42	33125	7.30	38324	8.19	43698	8.43
pr439	150834	6.57	194527	10.14	237929	12.54	282937	13.56	330485	13.41	375922	13.94
pcb442	68173	11.66	88116	15.16	107594	17.87	130733	16.40	153334	15.77	175161	15.81
d493	50548	3.81	64404	8.61	79553	9.89	94619	10.86	107956	13.35	125521	11.40
Avg.		7.00		4.56		4.61		4.81		4.99		5.40

Table 3.15: Comparison of MathTSPrd and HeuTSPrd on the instances derived from TSPLIB.

To better highlight the comparison of the two variants of the ILS with respect to the instance size, Table 3.16 summarizes the results obtained on symmetric instances derived from TSPLIB, grouped by the following size ranges: 50 to 100, 101 to 150, 151 to 250 and 251 to 500 vertices. It can be observed that the average percentage gap of MathTSPrd has a positive trend with respect to the instance size, while the HeuTSPrd shows consistent results on the same metric, with a slight increase in the $251-500$ instance group. For both algorithms, the spread with the initial solution decreases as the size increases, from 14.26% in the $50-100$ range to 4.06% in the $251-500$ range for MathTSPrd and from 17.40% in the $50-100$ range to 12.16% in the $251-500$ for HeuTSPrd, indicating the increasing difficulty to find a solution that improves the initial one.

Furthermore, a set of experiments on asymmetric instances has been carried out to check whether the behavior of the two variants is affected by the asymmetry of the distance matrix. From the results, reported in Table 3.17, we observe that while the HeuTSPrd variant is able to greatly improve over the initial solution, the asymmetry of the distance matrix considerably worsens the performance of the MathTSPrd. In particular, in all but 4 instances, all having $\beta = 0.5$, the algorithm is not able to find an improvement of the initial solution.

3.7 Conclusions

In this chapter the Traveling Salesman Problem (TSP) with release dates and completion time minimization, known as the TSP-rd(time) problem, is studied. The derived properties contribute to a deeper understanding of a problem belonging to a relatively unexplored class of routing problems. The mathematical formulation has allowed us to solve to optimality instances with up to 20 customers and to assess the quality of the two proposed variants of a heuristic based on an iterated local search, where the perturbation is performed by means of a destroy-and-repair procedure. The two variants differ in the repair operator, aimed at inserting customers in a partial solution. The variant named MathTSPrd uses a MILP model as repair operator while the one named HeuTSPrd inserts customers at random. The computational results show that the HeuTSPrd outperforms the MathTSPrd. Heuristics that make use of MILP models, the often so called matheuristics, have been shown to be effective in several cases when compared to other more traditional heuristics. However, in our study, using MILP models as a repair operators in an iterative framework has not turned out to be beneficial for the solution of the TSP-rd(time).

Several research directions remain open. An interesting direction is related to designing new heuristics for the problem and investigating the situations where the use of a MILP model is effective in a heuristic framework, and possibly identifying characteristics of the problems or the matheuristic for which this happens. It would be interesting also to study the extension of the TSP-rd(time) problem to the case where a fleet of vehicles is available. Finally, it would be interesting to study stochastic and dynamic versions of the problem, which are closer to the practical applications described in the introduction.

		MathTSPrd		HeuTSPrd	
Size	β	Avg % of initial	Avg % Best	Avg % of initial	Avg % Best
$50-100$	0.5	9.69	1.41	11.23	0
	1	13.62	4.03	18.20	0
	1.5	16.32	2.77	19.54	0
	2	15.87	3.02	19.36	0
	2.5	15.37	2.85	18.61	0
	3	14.69	2.42	17.44	0
		14.26	2.75	17.40	0
$101-150$	0.5	8.15	1.46	9.72	0
	1	10.48	4.24	15.18	0
	1.5	12.61	4.01	17.11	0
	2	12.97	4.18	17.68	0
	2.5	12.71	3.85	17.05	0
	3	10.40	5.37	16.14	0
		11.22	3.85	15.48	0
$151-250$	0.5	5.45	2.63	8.19	0
	1	8.46	2.85	11.56	0
	1.5	9.78	3.92	14.11	0
	2	9.93	3.96	14.29	0
	2.5	9.76	4.23	14.42	0
	3	9.60	3.89	13.87	0
		8.83	3.58	12.74	0
$250-500$	0.5	2.35	4.37	6.75	0
	1	4.11	7.10	11.32	0
	1.5	5.18	8.09	13.43	0
	2	5.19	8.42	13.81	0
	2.5	4.38	9.52	14.03	0
	3	3.17	10.38	13.62	0
		4.06	7.98	12.16	0

Table 3.16: Summary of results on the instances generated from TSPLIB.

		Initial solution		MathTSPrd		HeuTSPrd	
β	Instance	Value	% Best	Value	% Best	Value	% Best
0.5	ftv33	2424	31.38	1845	0	1854	0.49
	ft53	14690	25.99	12677	8.72	11660	0
	ftv70	4137	21.82	3842	13.13	3396	0
	kro124	59583	17.00	53247	4.56	50925	0
	rbg403	5606	14.99	5606	14.99	4875	0
1	ftv33	3037	36.49	3037	36.49	2225	0
	ft53	17980	28.66	17980	28.66	13975	0
	ftv70	5067	27.86	5067	27.86	3963	0
	kro124	77033	22.34	77033	22.34	62964	0
	rbg403	6837	22.55	6837	22.55	5579	0
1.5	ftv33	3650	42.58	3650	42.58	2560	0
	ft53	21271	32.27	21271	32.27	16081	0
	ftv70	5996	24.30	5996	24.30	4824	0
	kro124	94484	27.54	94484	27.54	74083	0
	rbg403	8067	29.80	8067	29.80	6215	0
2	ftv33	4262	37.75	4262	37.75	3094	0
	ft53	24562	30.87	24562	30.87	18768	0
	ftv70	6925	26.65	6925	26.65	5468	0
	kro124	111934	25.90	111934	25.90	88904	0
	rbg403	9298	24.35	9298	24.35	7477	0
2.5	ftv33	4875	35.12	4875	35.12	3608	0
	ft53	27852	33.42	27852	33.42	20875	0
	ftv70	7854	24.00	7854	24.00	6334	0
	kro124	129385	26.90	129385	26.90	101961	0
	rbg403	10528	22.09	10528	22.09	8623	0
3	ftv33	5488	34.41	5488	34.41	4083	0
	ft53	31143	31.00	31143	31.00	23774	0
	ftv70	8784	26.77	8784	26.77	6929	0
	kro124	146835	24.85	146835	24.85	117612	0
	rbg403	11759	20.14	11759	20.14	9788	0
Avg.			27.66		25.33		0.02

Table 3.17: Comparison MathTSPrd and HeuTSPrd on the asymmetric instances derived from TSPLIB.

Chapter 4

Dynamic traveling salesman problem with stochastic release dates

This chapter is based on the homonymous article currently submitted for publication. The work therein has been performed by the author in collaboration with Prof. Claudia Archetti, Prof. Dominique Feillet, and Prof. M. Grazia Speranza.

4.1 Abstract

The dynamic traveling salesman problem with stochastic release dates (DTSP-srd) is the problem in which a distributor has to deliver parcels to its customers and part of these parcels are delivered to its depot while the distribution is taking place. The arrival time of a parcel to the depot is called its release date. In the DTSP-srd, release dates are stochastic and dynamically updated as the distribution takes

place. The objective of the problem is to minimize the total time needed to serve all customers, given by the sum of the traveling time and the waiting time at the depot. A reoptimization technique is proposed to tackle the dynamic aspect of the problem. To define the reoptimization epochs, three policies are introduced, with increasing reoptimization frequency. Two models are proposed for the problem to be solved at each decision epoch. The first one is a stochastic model exploiting the entire probabilistic information available for the release dates. The second one is a deterministic model where a point estimation of the release dates is used. An instance generation procedure is proposed to simulate the evolution of the information about the release dates and computational tests are performed. The results show that a more frequent reoptimization provides better results across all tested instances and that the stochastic model performs better than the deterministic model. The main drawback of the stochastic model lies in the computational time required to evaluate any of the solution explored, which makes an iteration of the heuristic substantially longer.

4.2 Introduction

In a last mile distribution system, the distributor is often called to face the uncertainties of the delivery of goods to its depot. Information therefore plays a crucial role in an efficient distribution planning. In this setting, we study the dynamic traveling salesman problem with stochastic release dates (DTSP-srd) that is a problem in which a distributor receives from its suppliers goods to be distributed to customers. The goods to be delivered are available for the distribution only after they have arrived to the depot of the distributor. The arrival time of a parcel to the depot is called its release date. In the DTSP-srd, release dates are considered to be stochastic and dynamically updated as the distribution takes place. The DTSP-srd finds application in many real world problems. The steadily growing interest for an environment-aware supply chain and distribution organization has led to a significant research for an eco-friendly delivery to city centers. In this context, the distribution is implemented by means of distribution centers, where goods are unloaded from the trucks, consolidated, and delivered to the customers by means of hybrid or electric vehicles. The distributor has to face the uncertainty of the arrival time of the trucks to plan for an efficient distribution. The goal is to define an efficient distribution plan serving all customers. The DTSP-srd also finds application in cross docking operations, where the routing of outbound vehicles is planned according to the arrival time of inbound vehicles.

While problems with release dates, defined as the earliest availabilities of jobs for processing, have been widely studied in the context of machine sequencing (see Pinedo (2016)), only in recent years the concept of release dates has been introduced for vehicle routing problems. Cattaruzza et al. (2016a) introduces the multi-trip vehicle routing problem with time windows and release dates (MTVRPTW-R) in which all information is assumed to be static and deterministic. The authors propose a genetic algorithm for the problem with an underlying giant tour decomposition procedure. The complexity of the traveling salesman problem with release dates (TSP-rd) is studied in Archetti et al. (2015a). The authors introduce two variants of the problem, one considering a deadline for the completion of the distribution and seeking the minimization of the total travel time, referred to as TSP-rd(distance), and the other minimizing the total time required to complete the distribution, referred to as TSP-rd(time). The complexity of the problem is studied on special topologies of the graph. Reyes et al. (2018) extend this work to consider service guarantee, a fixed maximum delay between the release date and the delivery time. The TSP-rd(time) is studied in Archetti et al. (2018) where a mathematical programming formulation is described and an iterated local search is developed. Two versions of the heuristic are tested on instances derived from TSP instances by Solomon (1987) and from the TSPLIB (see Reinelt (1991)).

Stochasticity has been widely studied in the literature on distribution problems. We refer to Gendreau et al. (2016) for a review of the recent advances and future directions of the stochastic vehicle routing literature and to Ritzinger et al. (2016) for a survey on the dynamic and stochastic vehicle routing problems. Despite this, little work has been done in the past regarding the uncertainty in the release dates. Uncertain arrival times are considered in Klapp et al. (2016), where the authors study the dispatch wave problem. At any decision epoch (wave), requests are either known or potential. The stochasticity of potential requests considers both the probability of non-arrival and the probability of the arrival to a specific wave. The minimization of the expected vehicle operating costs and the penalties for unserved requests is sought on a special topology of the graph, i.e., the line. In Klapp et al. (2018) the authors extend the work to a general network. A deterministic model is used to find an optimal a priori solution to the stochastic variant and two dynamic policies are developed. Finally, the trade-off between minimizing operational costs and maximizing the total order coverage is studied.

In this chapter we introduce the DTSP-srd as a generalization of the TSP-rd(time) where the information on the release dates is assumed to be stochastic and dynamically updated as the distribution takes place. A reoptimization technique is proposed to tackle the dynamic aspect of the problem and three dynamic policies with increasing reoptimization frequency are introduced. Two models for the solution of the problem at each decision epoch are discussed. The first one is a stochastic model exploiting the entire probabilistic information available for the release dates. The second one is a deterministic model where a point estimation of the release dates is used. Both models are solved with an iterated local search, adapted from the heuristic presented for the TSP-rd(time) in Archetti et al. (2018). The two models are also compared to a myopic policy serving customers as their parcel arrives to the depot, without considering the information available about future release dates. An instance generation procedure is proposed to simulate the evolution of the information about the release dates and computational tests are performed. The results, obtained on instances with 50 customers, show that a more frequent reoptimization provides better results across all tested instances. The myopic policy is shown to perform more the 12% worse than the best solution found by the two models. The deterministic and stochastic model have an average percentage gap from the best solution found across the three dynamic policies of 2.92% and 1.16%, respectively. While the stochastic model performs better, its main drawback lies in the computational time required to evaluate any of the solution explored, which makes an iteration of the heuristic substantially longer.

The chapter is organized as follows. In Section 4.3 the dynamic traveling salesman problem with stochastic release dates is defined. The solution approach to the dynamic problem is described in Section 4.4. In Section 4.5 the stochastic and deterministic optimization models are introduced, an illustrative example of the two models is presented and the proposed heuristic algorithm is described. The instance generation procedure is described in Section 4.6. Computational experiments are reported in Section 4.7. Finally, conclusions are drawn in Section 4.8.

4.3 The dynamic traveling salesman problem with stochastic release dates

The DTSP-srd is defined as follows. Let $G = (V, A)$ be a complete graph. A traveling time and a cost are associated with each arc $(i, j) \in A$. These two values are assumed identical and denoted by d_{ij}. It is also assumed that the triangle inequality is satisfied. The set of vertices V is composed by vertex 0, which identifies the depot, and the set N of customers. Each customer is characterized by the arrival time of its parcel to the depot. We call this value its release date. A single vehicle is allowed to perform a sequence of tours to deliver the goods to the customers. Capacity constraints are not considered. We call *route* a tour starting and ending at the depot and not visiting the depot in between. We call $K = \{1, \dots, |K|\}$ the set of routes. The objective is to minimize the expected completion time to serve all customers, that is, the total travel time plus the expected waiting time at the depot. The waiting time is due to the fact that the vehicle has to wait at the depot until the latest release date of the customers that it is going to serve in the next route. Time is assumed to be discrete, i.e., $t \in \mathbb{N}$. While this is a modeling simplification, a sufficiently fine granularity of time should not hinder the quality of the conclusions. We assume that $t = 0$ is the starting of the distribution plan.

As the name implies, the DTSP-srd is characterized by a dynamic and a stochastic aspect. The dynamic aspect of the problem lies in the fact that the information available for each release date evolves over time, as the parcels are delivered to the depot and the distribution to the customers takes place. The stochasticity is found in the fact that the release dates of the parcels that have not yet been delivered to the depot are assumed to be stochastic. This represents the situation where the vehicle carrying the parcel to the depot is still en route and might encounter unforeseen situations, such as lighter or heavier traffic than expected at the beginning of the distribution.

At any time t, as the distribution is being carried out, some customers might have already been served, meaning that their parcels have already been delivered to the depot and loaded on the vehicle that has left from the depot. We call $N_t^{served} \subseteq N$ the set of such customers. Note that $N_0^{served} = \varnothing$.

Unserved customers, i.e., customers in $N \setminus N_t^{served}$, are classified based on the information available for their release date. We call $N_t^{unserved}$ the set of these customers. If the parcel for customer has already been delivered to the depot, its release date is past and certain and the customer can be served. We call $N_t^{known} \subseteq N_t^{unserved}$ the set of these customers. If the parcel is still to be delivered to the depot, the release date is future and stochastic. Two types of customers with stochastic release date are defined, according to how such stochasticity is updated over time. The first type of these customers are those for which the stochastic information about the release date is not updated until the vehicle reaches the depot. This can be seen as the case of non-GPS-equipped vehicles: as no new information is available about the state of the vehicle, the information about the release date is not updated until the parcel is delivered to the depot. We call $N_t^{static} \subseteq N_t^{unserved}$ the set of these customers. The second type of customers are those whose information about the stochastic release date is refined over time. This is the case of parcels delivered to the depot by means of GPS-equipped vehicles, transmitting updated information as they travel. We call $N_t^{dynamic} \subseteq N_t^{unserved}$ this set of customers. Note that, at any time, $\{N_t^{known}, N_t^{static}, N_t^{dynamic}\}$ is a partition of $N_t^{unserved}$ and $\{N_t^{served}, N_t^{unserved}\}$ is a partition of N.

The vehicles delivering the parcels to the depot are assumed to travel independently from one another, i.e., the release dates are assumed to be independent.

The release date of customer $i \in N_t^{unserved}$ at time t is denoted by $\tilde{r}_i^t$. Based on the type of customer, $\tilde{r}_i^t$ is defined according to the following:

- if $i \in N_t^{known}$, $\tilde{r}_i^t$ is the known release date, $\tilde{r}_i^t = r_i$, where r_i is the observed arrival time of the parcel for customer i, i.e. $r_i \leq t$. At time $t = 0$, the set N_t^{known} is composed by those customers whose parcel arrived before the starting time of the distribution plan, e.g., overnight, and $r_i = 0$;
- if $i \in N_t^{static}$, $\tilde{r}_i^t$ is a random variable describing the release date with the information available at the beginning of the distribution, $\tilde{r}_i^t = \tilde{r}_i^0$, $\forall t > 0$, i.e., an estimation of the release date is done at time $t = 0$ and never updated as no newer information is available before the parcel reaches the depot;
- if $i \in N_t^{dynamic}$, $\tilde{r}_i^t$ is a random variable describing the release date at time t. In this case $\tilde{r}_i^t$ is dynamically updated with the information available at time t.

The random variables are assumed to be discrete and bounded, with the lower and upper bound of $\tilde{r}_i^t$ denoted as l_i^t and u_i^t, respectively. As the random variable represents a future release date, the lower bound is assumed to be greater than the time t, i.e, $l_i^t \geq t$.

4.4 Solution approach to the dynamic problem

As the information on the release dates is dynamically updated, a reoptimization technique is proposed for the problem. A policy is therefore required to identify the reoptimization epochs. At each reoptimization epoch, a problem is solved with updated information about the release dates for the customers in $N_t^{unserved}$. Three different policies are proposed: the first and simplest one solves the problem at the beginning of the distribution and each time the vehicle returns to the depot at the end of a route. This policy is identified as policy R. The second policy reoptimizes at every return to the depot and every time the vehicle should depart from the depot, i.e., when the earliest time at which the next route can start is reached. If the first route of the newly obtained solution can start, this route is implemented. Otherwise, the process is repeated. This policy is identified as policy RD. The last policy reoptimizes at every time the vehicle returns to the depot, every time the vehicle should depart from the depot, and at intervals while the vehicle is waiting at the depot. During waiting times, the next reoptimization epoch is defined as the middle point between the current time and the expected starting time of the next route. If the interval between the current time and the next starting time is less than a minimum time span between reoptimization epochs, the next reoptimization is performed at the expected starting time of the next route. This policy is identified as RDW policy. The flowchart of the RDW policy is presented in Figure 4.1, where τ_{start}^{next} is the actual starting time of the next route, meaning the time at which all the parcels of the customers to be served in the next route have been delivered at the depot and thus the vehicle can start the route and serve the customers.

The highlighted box in Figure 4.1 involves the solution of the DTSP-srd with the information available at time t, i.e., the set of customers in $N_t^{unserved}$ and the information on the corresponding release dates. When a route is performed, the customers that have been served are removed from the set N_t^{known} and added to the set N_t^{served}.

We devise two optimization models for the solution of the problem. The distribution problem can be solved in a stochastic way, by handling the release dates as random variables, or in a deterministic way, deriving a point estimation of the release date from the random variable $\tilde{r}_i^t$ of customers $i \in N_t^{static} \cup N_t^{dynamic}$.

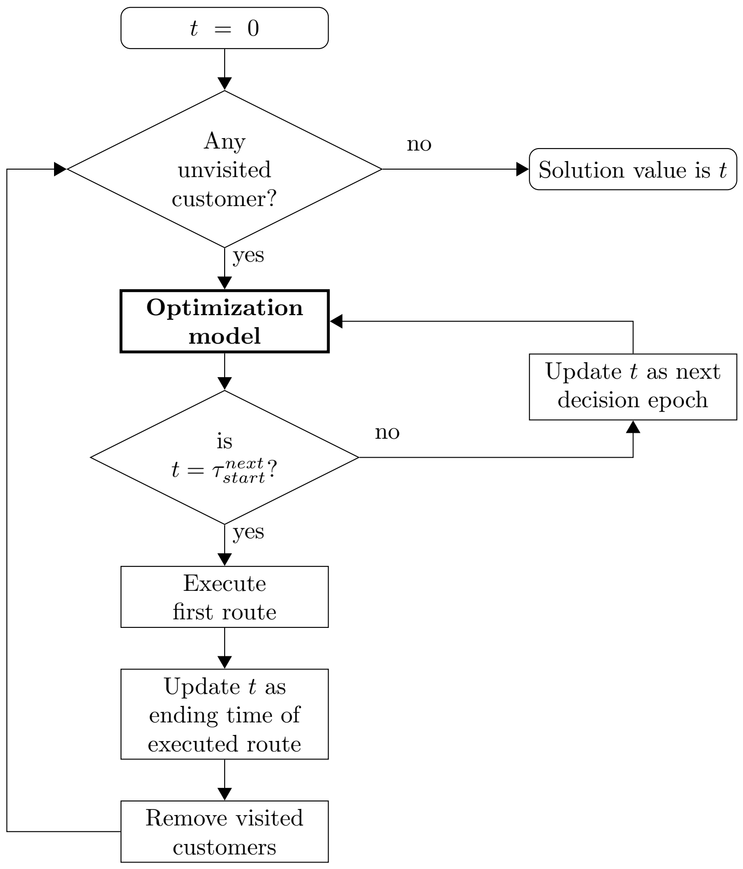


Figure 4.1: The RDW policy.

4.5 Optimization models

In this section the two models proposed for the solution of the problem at each reoptimization epoch are presented. The stochastic model is introduced in Section 4.5.1. Then, in Section 4.5.2, the deterministic model is described. To illustrate the differences of the two models an example is presented in Section 4.5.3. Finally, the solution method proposed for the two models is described in Section 4.5.4.

4.5.1 Stochastic model

If the full stochastic information on the release dates is considered, the proposed stochastic mixed-integer linear program for the problem at time t is defined as follows. The formulation relies on the following decision variables: x_{ij}^k are arc variables, with $x_{ij}^k = 1$ if route k traverses arc (i,j) and $x_{ij}^k = 0$ otherwise, y_i^k are route variables, with $y_i^k = 1$ if customer i is visited in route k and $y_i^k = 0$ otherwise. Flow variables u_{ij}^k are added to enforce subtour elimination.

$$\min \sum_{k \in K} \sum_{(i,j) \in A} d_{ij} x_{ij}^k + \mathbb{E}\left[Q(x, \tilde{r}^t)\right] \tag{4.1}$$

s.t.

$$\sum_{k \in K} y_i^k = 1 \qquad i \in N_t^{unserved}, \tag{4.2}$$

$$\sum_{j \in V} x_{ij}^k = \sum_{j \in V} x_{ji}^k = y_i^k \qquad i \in N_t^{unserved} \cup \{0\}, k \in K, \tag{4.3}$$

$$\sum_{j \in V} u_{ji}^k - \sum_{j \in V} u_{ij}^k = y_i^k \qquad i \in N_t^{unserved}, k \in K, \tag{4.4}$$

$$u_{ij}^k \leq (n-1) x_{ij}^k \qquad (i,j) \in A, k \in K, \tag{4.5}$$

$$x_{ij}^k \leq 1 - x_{00}^k \qquad (i,j) \in A, k \in K \setminus \{|K|\}, \tag{4.6}$$

$$x_{00}^k \geq x_{00}^{k+1} \qquad k \in K \setminus \{|K|\}, \tag{4.7}$$

$$x_{ij}^k \in \{0,1\} \qquad (i,j) \in A, k \in K, \tag{4.8}$$

$$y_i^k \in \{0,1\} \qquad i \in N_t^{unserved} \cup \{0\}, k \in K, \tag{4.9}$$

$$u_{ij}^k \geq 0 \qquad (i,j) \in A, k \in K. \tag{4.10}$$

The objective function (4.1) minimizes the total serving time as sum of the travel times and the expected waiting times at the depot, where $x = (x_{ij}^k)_{(i,j) \in A, k \in K \setminus \{|K|\}}$ and $\tilde{r}^t = (\tilde{r}_i^t)_{i \in N_t^{unserved}}$ is the random vector of the release dates. Constraints (4.2) ensure the visit of all customers. Constraints (4.3)-(4.5) impose that each route is a circuit connected to the depot. In particular, constraints (4.5) generate a flow that decreases while the vehicle visits customers, which prevents subtours. Constraints (4.6) and (4.7) are symmetry breaking constraints. In particular, constraints (4.6) set $x_{00}^k = 1$ if route k is empty, $x_{00}^k = 0$ otherwise, and constraints (4.7) impose that all the empty routes, if any, must precede all the non-empty routes, i.e., if route k is empty then all routes k' must be empty, with $k' = 1, \dots, k1$.

Given a solution and a realization for the random vector $\tilde{r}^t$, $Q(x, \tilde{r}^t)$ denotes the waiting time at the depot. Waiting occurs if the starting time of a route is greater than the ending time of the previous one, or, in the case of the first route, if it is not starting at time $t = 0$. The ending time of a route is computed as its starting time plus its travel time. The starting time of a route must be greater than or equal to the release date of the customers in the route and the ending time of the previous route. Let τ_{start}^k and τ_{end}^k be the starting and ending time of route $k \in K$. Then:

$$\tau_{end}^k = \tau_{start}^k + \sum_{(i,j)\in A} d_{ij} x_{ij}^k \qquad k \in K, \tag{4.11}$$

$$\tau_{end}^k \leq \tau_{start}^{k+1} \qquad k \in K \setminus \{|K|\}, \tag{4.12}$$

$$\tau_{start}^k \geq \tilde{r}_i^t y_i^k \qquad i \in N, k \in K. \tag{4.13}$$

It follows that

$$\mathbb{E}\left[Q(x, \tilde{r}^t)\right] = \mathbb{E}\left[\sum_{k\in K\setminus\{|K|\}} \left(\tau_{start}^{k+1} - \tau_{end}^k\right)\right]. \tag{4.14}$$

Therefore, the objective function (4.1) can also be computed as the expected ending time of the last route:

$$\sum_{k\in K} \sum_{(i,j)\in A} d_{ij} x_{ij}^k + \mathbb{E}\left[Q(x, \tilde{r}^t)\right] = \mathbb{E}\left[\tau_{end}^{|K|}\right]. \tag{4.15}$$

The model (4.1)-(4.10) is solved with the solution method presented in Section 4.5.4 where solutions are evaluated by the expected ending time of the last route. In the remainder of this section the computation of this expected value is described. Let $\mathcal{S}$ be a solution of (4.1)-(4.10), obtained at time t. The vehicle is assumed to be located at the depot and $\mathcal{S}$ provides the delivery routes for customers in $N_t^{unserved}$. The waiting time depends on the release dates $\tilde{r}_i^t$. In the remainder of this section, $\tilde{r}_i^t$ is assumed to be a random variable. Note that the known release date of customers in N_t^{known} can be seen as a random variable with bounds $l_i^t = u_i^t = r_i$ and $P(\tilde{r}_i^t = r_i) = 1$. In what follows the index t is removed to improve readability.

The expected completion time of delivery in $\mathcal{S}$ is computed as $\mathbb{E}\left[\tau_{end}^{|K_\mathcal{S}|}\right]$, where $K_\mathcal{S} = \{1, \ldots, |K_\mathcal{S}|\}$ is the set of routes of $\mathcal{S}$. From equations (4.11) and (4.12) it follows that the starting time of a route can be computed as:

$$\tau_{start}^k = \max\left\{\tau_{end}^{k-1}, \max_{i \in N_k}\{\tilde{r}_i\}\right\}, \qquad k \in K_\mathcal{S} \setminus \{1\}, \tag{4.16}$$

where $N_k := \{i \in N_t^{unserved} | y_i^k = 1\}$ is the set of customers served in route k. The computation makes use of the cumulative distribution function of the maximum of the release dates of the customers in route k, $\max_{i \in N_k}\{\tilde{r}_i\}$, computed as:

$$P\left(\max_{i \in N_k}\{\tilde{r}_i\} \leq t'\right) = \prod_{i \in N_k} P\left(\tilde{r}_i \leq t'\right), \qquad k \in K_\mathcal{S}. \tag{4.17}$$

Equations (4.16) and (4.17) allow us to define the cumulative distribution function of τ_{start}^k as:

$$P\left(\tau_{start}^k \leq t'\right) = \begin{cases} P\left(\max_{i \in N_k}\{\tilde{r}_i\} \leq t'\right) & \text{for } k = 1 \\ P\left(\tau_{end}^{k-1} \leq t'\right) \cdot P\left(\max_{i \in N_k}\{\tilde{r}_i\} \leq t'\right) & \text{for } 2 \leq k \leq |K_\mathcal{S}|. \end{cases} \tag{4.18}$$

The support of the random variable τ^k_{start} is defined as follows. Let l^k_{start} and u^k_{start} be the lower and upper bound of τ^k_{start}, respectively. The two bounds are computed as:

$$l^k_{start} = \begin{cases} \max_{i \in N_k}\{l_i\} & \text{for } k = 1 \\ \max\left\{l^{k-1}_{end}, \max_{i \in N_k}\{l_i\}\right\} & \text{for } 2 \leq k \leq |K_{\mathcal{S}}| \end{cases} \tag{4.19}$$

$$u^k_{start} = \begin{cases} \max_{i \in N_k}\{u_i\} & \text{for } k = 1 \\ \max\left\{u^{k-1}_{end}, \max_{i \in N_k}\{u_i\}\right\} & \text{for } 2 \leq k \leq |K_{\mathcal{S}}|. \end{cases} \tag{4.20}$$

Because of equation (4.11), the bounds on τ^k_{end} are computed as $l^k_{end} = l^k_{start} + d_k$ and $u^k_{end} = u^k_{start} + d_k$, where d_k is the traveling time of route k.

The expected value of solution $\mathcal{S}$ is therefore computed as:

$$\mathbb{E}\left[\tau^{|K_{\mathcal{S}}|}_{end}\right] = \sum_{t=l^{|K_{\mathcal{S}}|}_{end}}^{u^{|K_{\mathcal{S}}|}_{end}} t \cdot \left[P\left(\tau^{|K_{\mathcal{S}}|}_{end} \leq t\right) - P\left(\tau^{|K_{\mathcal{S}}|}_{end} \leq t-1\right)\right]. \tag{4.21}$$

where $P\left(\tau^{|K_{\mathcal{S}}|}_{end} \leq t\right)$ follows from expressions (4.11) and (4.18). Thanks to (4.15) this allows us to compute the objective function for any solution.

The complexity of computing the expected value of the objective function is $\mathcal{O}(|S_{\tau^{|K_{\mathcal{S}}|}_{end}}| \cdot |N|)$, where $S_{\tau^{|K_{\mathcal{S}}|}_{end}}$ is the support of the random variable $\tau^{|K_{\mathcal{S}}|}_{end}$. The term $|S_{\tau^{|K_{\mathcal{S}}|}_{end}}|$ is due to the fact that the computation of the expected value involves a sum over all the elements of the support, as described in expression (4.21). The term $|N|$ is due to the fact that the computation of the cdf of $\max_{i \in N_k}\{\tilde{r}_i\}$, having cost $\mathcal{O}(|N_k|)$ as per expression (4.17), must be performed for all routes in K, as defined in expression (4.18), and therefore $\sum_{k=1}^{|K|} \mathcal{O}(|N_k|) = \mathcal{O}(|N|)$.

4.5.2 Deterministic model

The second of the two proposed models for the DTSP-srd makes use of a point estimation for the stochastic release dates. The information to be used in the model is the observed release date of customers in N_t^{known}, and the expected value of the stochastic release dates for customers in $i \in N_t^{static} \cup N_t^{dynamic}$. The problem to be solved at any time t when the vehicle is at the depot is a TSP-rd(time) as proposed in Archetti et al. (2018).

4.5.3 Illustrative example

An example is now shown to illustrate the differences between the two models. As the aim of the example is a comparison of the two models without taking into consideration the effect of the different reoptimization policies and the effect of the dynamic information on the solution, the release dates are considered as static and only the solution obtained at time $t = 0$ is shown. Without affecting the validity of the example, in what follows the assumption that $t \in \mathbb{N}$ is waived to allow for a leaner text. The example can be made valid with $t \in \mathbb{N}$ by multiplying the arc costs and the values of the support of the release dates by 1000.

Let there be two customers arranged as in Figure 4.2. The number close to each arc is d_{ij}. The random variables describing the release dates of the two customers,

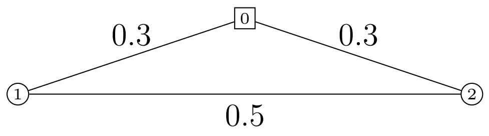


Figure 4.2: An example.

denoted as $\tilde{r}_1$ and $\tilde{r}_2$, are defined by the following probability density functions (pdf):

- Customer 1:

$$f_{\tilde{r}_1}(t) = \begin{cases} 0.5 & \text{if } t = 1 \\ 0.5 & \text{if } t = 2 \\ 0 & \text{otherwise.} \end{cases}$$

- Customer 2:

$$f_{\tilde{r}_2}(t) = \begin{cases} 0.5 & \text{if } t = 1 \\ 0.5 & \text{if } t = 2.1 \\ 0 & \text{otherwise.} \end{cases}$$

Let us consider two solutions for the DTSP-srd problem. The first solution, $\mathcal{S}_1$, consists of two routes, $\mathcal{S}_{1,1}$ and $\mathcal{S}_{1,2}$, visiting customer 1 and customer 2, respectively. The traveling time of the routes is $c_{\mathcal{S}_{1,1}} = c_{\mathcal{S}_{1,2}} = 0.6$. The second solution, $\mathcal{S}_2$, consists of a single route $\mathcal{S}_{2,1}$, visiting both customers 1 and 2, with travel time $c_{\mathcal{S}_{2,1}} = 1.1$.

The deterministic model evaluates the two solutions in the following way:

- solution $\mathcal{S}_1$ has an estimated value of 2.7, as route $\mathcal{S}_{1,1}$ has an estimated starting time $\tau_{start}^{\mathcal{S}_{1,1}} = \mathbb{E}[\tilde{r}_1] = 1.5$ and route $\mathcal{S}_{1,2}$ has an estimated starting time $\tau_{start}^{\mathcal{S}_{1,2}} = \max\{\tau_{end}^{\mathcal{S}_{1,1}}, \mathbb{E}[\tilde{r}_2]\} = 2.1$;

- solution $\mathcal{S}_2$ has an estimated value of 2.65 as route $\mathcal{S}_{2,1}$ has an estimated starting time $\tau_{start}^{\mathcal{S}_{2,1}} = \max\{\mathbb{E}[\tilde{r}_1], \mathbb{E}[\tilde{r}_2]\} = 1.55$.

The stochastic model evaluates the two solutions in the following way:

- solution $\mathcal{S}_1$ is evaluated according to the procedure described in Section 4.5.1:
  - the starting time of $\mathcal{S}_{1,1}$ is defined as $\tau_{start}^{\mathcal{S}_{1,1}} = \tilde{r}_1$, meaning that $f_{\tau_{start}^{\mathcal{S}_{1,1}}} = f_{\tilde{r}_1}$;
  - because of equation (4.11), $\tau_{end}^{\mathcal{S}_{1,1}}$ is defined as $\tau_{end}^{\mathcal{S}_{1,1}} = \tau_{start}^{\mathcal{S}_{1,1}} + c_{\mathcal{S}_{1,1}}$. The pdf

of $\tau_{end}^{\mathcal{S}_{1,1}}$ is defined as:

$$f_{\tau_{end}^{\mathcal{S}_{1,2}}}(t) = \begin{cases} 0.5 & \text{if } t = 1.6 \\ 0.5 & \text{if } t = 2.6 \\ 0 & \text{otherwise;} \end{cases}$$

– the starting time of the second route is defined as $\tau_{start}^{\mathcal{S}_{1,2}} = \max\{\tau_{end}^{\mathcal{S}_{1,1}}, \tilde{r}_2\}$. The cumulative distribution function is obtained as described in (4.18): $F_{\tau_{start}^{\mathcal{S}_{1,2}}}(t) = F_{\tau_{end}^{\mathcal{S}_{1,1}}}(t) \cdot F_{r_2}(t)$. The resulting pdf of $\tau_{start}^{\mathcal{S}_{1,2}}$ is the following:

$$f_{\tau_{start}^{\mathcal{S}_{1,2}}}(t) = \begin{cases} 0.25 & \text{if } t = 1.6 \\ 0.25 & \text{if } t = 2.1 \\ 0.5 & \text{if } t = 2.6 \\ 0 & \text{otherwise;} \end{cases}$$

– combining equation (4.11) with the pdf of $\tau_{start}^{\mathcal{S}_{1,2}}$ allows us to compute the expected ending time of $\mathcal{S}_{1,2}$ as $\mathbb{E}[\tau_{end}^{\mathcal{S}_{1,2}}] = 2.825$;

- solution $\mathcal{S}_2$ has expected value 2.9. The random variable of the starting time is defined as $\tau_{start}^{\mathcal{S}_{2,1}} = \max\{\tilde{r}_1, \tilde{r}_2\}$ and therefore has $l_{\tau_{start}^{\mathcal{S}_{2,1}}} = \max_{i\in\{1,2\}}\{l_{\tilde{r}_i}\} = 1$ and $u_{\tau_{start}^{\mathcal{S}_{2,1}}} = \max_{i\in\{1,2\}}\{u_{\tilde{r}_i}\} = 2.1$. The pdf of $\tau_{start}^{\mathcal{S}_{1,2}}$ is:

$$f_{\tau_{start}^{\mathcal{S}_{1,2}}}(t) = \begin{cases} 0.25 & \text{if } t = 1 \\ 0.25 & \text{if } t = 2 \\ 0.5 & \text{if } t = 2.1 \\ 0 & \text{otherwise,} \end{cases}$$

thus, $\mathbb{E}[\tau_{start}^{\mathcal{S}_{2,1}}] = 1.8$ and $\mathbb{E}[\tau_{end}^{\mathcal{S}_{2,1}}] = 2.9$.

Table 4.1 reports the expected value of the two solutions according to the stochastic and the deterministic models.

Model	$\mathcal{S}_1$	$\mathcal{S}_2$
Deterministic	2.7	**2.65**
Stochastic	**2.825**	2.9

Table 4.1: Expected values of the solutions.

This result shows how, in the presented example, the deterministic model chooses the solution that waits for the two customers while the stochastic model chooses to split the service of the two customers, anticipating the visit of customer 1. The two solutions have been evaluated in the four scenarios, originating from the combination of the realization of the random variables representing the release dates of the two customers. The value of each solution in every scenario is reported in Table 4.2.

Scenario	$\tilde{r}_2 = 1$	$\tilde{r}_2 = 2.1$
$\tilde{r}_1 = 1$	$\mathcal{S}_1 = 2.2$	$\mathcal{S}_1 = \mathbf{2.7}$
	$\mathcal{S}_2 = \mathbf{2.1}$	$\mathcal{S}_2 = 3.2$
$\tilde{r}_1 = 2$	$\mathcal{S}_1 = 3.2$	$\mathcal{S}_1 = \mathbf{3.2}$
	$\mathcal{S}_2 = \mathbf{3.1}$	$\mathcal{S}_2 = \mathbf{3.2}$

Table 4.2: Results in the four scenarios¡.

When looking at the best solution for each scenario, it can be observed that $\mathcal{S}_1$, the solution chosen by the stochastic model, finds the best solution in 2 over the 4 scenarios and $\mathcal{S}_2$, the solution chosen by the deterministic model, is the best solution in all but one scenarios. However, when evaluating the difference in terms of average percentage gap from the best solution in each scenario, $\mathcal{S}_1$ has an average gap of 2% while $\mathcal{S}_2$ has an average gap of 4.63%. This result is explained by the fact that, as highlighted in Table 4.2, $\mathcal{S}_2$ is greatly penalized in the scenario in which customer 1 is early and customer 2 is late.

4.5.4 A heuristic algorithm

The solution of the two models is obtained through a heuristic method derived from the one presented in Archetti et al. (2018), which we briefly recall. The authors present an iterated local search (ILS) that is composed by two phases: a perturbation, consisting in a destroy-and-repair (DR) procedure, followed by a local search (LS). The two phases are repeated iteratively as shown in Figure 4.3. The DR is carried out by removing a number of customers from the solution and adding them back in a randomly chosen route, different from the one they were originally included in. The LS considers a set of four neighborhoods: the relocation of a customer to a different route, the merging of two consecutive routes, the split of a route in two separate ones, and the anticipation or postponement of a depot visit. The vertices are randomly sorted and considered sequentially: if the vertex is not the depot the customer relocation move is considered, otherwise the remaining three moves are evaluated. The first improving move is implemented and the process is repeated until no improving move is found. The ending conditions for the heuristic are the maximum time allowed and the maximum number of iterations without improvement. Each LS iteration has complexity $\mathcal{O}(|N|^2)$, as the node insertion move is evaluated for all nodes in all the possible positions within a solution.

The algorithm from Archetti et al. (2018) was shown to have good performance, finding the optimal solution in all but one of the 24 instances tested with size $|N| = 20$, the largest that could be solved to optimality within one hour with CPLEX. It has been modified to reflect the dynamic setting. In particular, the local search algorithm has been revised to become a best improvement local search where solutions will null improvement over the current best solution are implemented with the condition that

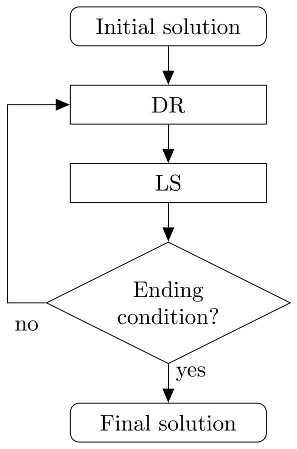


Figure 4.3: The heuristic scheme from Archetti et al. (2018).

the structure of the solution is improved. When the deterministic model is solved, between two equivalent solutions, the one minimizing the sum over all routes of the difference between the expected starting time and the maximum release date is chosen. This criterion is expressed as $\sum_{k\in K}\left(\tau^k_{start} - \max_{i\in N_k}\{\hat{r}^t_i\}\right)$. This allows us to select the solution anticipating the visit to customers as much as possible. This property is intuitively useful in a dynamic setting, where the information about arrivals in near future is expected to be less uncertain than that of customers whose goods will arrive further away in time. Given a solution and a move, the evaluation of the neighboring solution of the deterministic model has complexity $\mathcal{O}(1)$. Each LS iteration for the deterministic model has therefore complexity $\mathcal{O}(|N|^2)$.

For the stochastic model, the solution is obtained by adopting the same heuristic scheme presented for the deterministic model modified to consider the stochastic component of the waiting times. In the local search, solutions are kept if they improve the sum of the total travel time and expected waiting times, i.e., the expected value of the solution, computed as described in Section 4.5.1. Similarly to the deterministic case, solutions with null improvement are kept only if the structure of the solution is improved. As the release dates are stochastic, this criterion is measured as the expected value of the sum, for all routes, of the difference between the starting time and the maximum release date, which are both random variables, i.e., $\mathbb{E}\left[\sum_{k\in K}\left(\tau^k_{start} - \max_{i\in N_k}\{\tilde{r}^t_i\}\right)\right] = \sum_{k\in K}\left(\mathbb{E}\left[\tau^k_{start}\right] - \mathbb{E}\left[\max_{i\in N_k}\{\tilde{r}^t_i\}\right]\right)$. The solution for which such value is minimum is kept.

As reported in Section 4.5.1, the evaluation of a solution of the stochastic model has complexity $\mathcal{O}(|S_{\tau_{end}^{|K_{\mathcal{S}}|}}|\cdot|N|)$. Each LS iteration has therefore complexity $\mathcal{O}(|S_{\tau_{end}^{|K_{\mathcal{S}}|}}|\cdot|N|^3)$.

4.6 Instance generation

The following sections describe every step of the instance generation, firstly describing the instances from which the DTSP-srd instances are derived, and secondly defining how customers are partitioned and how the release dates of each customer are generated and updated over time. In a brief overview, the instance generation process works as follows. The release dates of customers are generated by simulating the traveling of the vehicles delivering the parcels to the depot. The dynamic release dates are updated as the vehicles travel to the depot. When the vehicle reaches the depot the release date becomes known.

4.6.1 Input instances

Instances have been generated starting from the instances for the TSP-rd(time) described in Archetti et al. (2018), and in particular from the subset of instances derived from Solomon (1987). Each instance is denoted as $\mathcal{I}\_\beta$ where $\mathcal{I}$ is the name of the instance from which the vertices coordinates are taken, i.e., C101, C201, R101, and RC101, and β represents how spread out in time the release dates can be, compared to the TSP travel time of the instance. Denoting d_{TSP} the TSP travel time of the instance, each release date is uniformly sampled in the interval $[0, \beta \times d_{TSP}]$. We denote $\bar{r}_i$ the release date of customer i from the instance for the TSP-rd(time).

4.6.2 Initial customer partitioning

A time $t = 0$, each customer of the instance is assigned to the set N_0^{known}, N_0^{static}, or $N_0^{dynamic}$ according to the following rule. All customers for which $\bar{r}_i = 0$, meaning all customers whose parcel is at the depot at time $t = 0$ in the input instance, are included in the set N_0^{known}, hence $N_0^{known} = \{i \in N | \bar{r}_i = 0\}$. Customers in $N \setminus N_0^{known}$ are assigned to N_0^{static} or $N_0^{dynamic}$ according to a parameter $\delta \in [0, 1]$ indicating the rate of customers with a dynamically updated release date, i.e., the customers in set $N_0^{dynamic}$ over the total number of customers in $N \setminus N_0^{known}$. In particular, $\delta \lceil |N \setminus N_0^{known}| \rceil$ randomly chosen customers are assigned to set $N_0^{dynamic}$ and the remaining customers are assigned to N_0^{static}, i.e., $N_0^{static} = N \setminus (N_0^{known} \cup N_0^{dynamic})$. The set of served customers is initialized as empty, i.e., $N_0^{served} = \varnothing$. The customers partitioning procedure is described in Algorithm 5.

Algorithm 5 InitialCustomerPartitioning($\mathcal{I}\_\beta$, δ)

```
1: $N_0^{known} = \varnothing, N_0^{static} = \varnothing, N_0^{dynamic} = \varnothing, N_0^{served} = \varnothing$
2: $N_0^{known} = \{i \in N | \bar{r}_i = 0\}$
3: while $|N_0^{dynamic}| < \delta \lceil |N \setminus N_0^{known}| \rceil$ do
4:     Randomly select $i \in N \setminus (N_0^{known} \cup N_0^{dynamic})$
5:     $N_0^{dynamic} = N_0^{dynamic} \cup \{i\}$
6: end while
7: $N_0^{static} = N \setminus (N_0^{known} \cup N_0^{dynamic})$
```

Instances with $\mathcal{I} \in \{C101, C201, R101, RC101\}$, $\beta \in \{0.5, 1, 1.5\}$, and $\delta \in \{0, 0.5, 1\}$ have been tested.

4.6.3 Release dates generation

In this section we present the procedure to generate the random variables describing the initial release dates for the customers in $N_t^{static} \cup N_t^{dynamic}$, i.e., $\tilde{r}_i^0$. As defined in Section 4.3, the release dates are assumed to be bounded and discrete. The discrete and bounded distribution for each release date are generated as follows. A continuous probability distribution is chosen for all the release dates of customers in $N_t^{static} \cup N_t^{dynamic}$. Then, for each customer, the parameters of the continuous probability density function ${}_cf_{\tilde{r}_i^0}$ are computed, the truncating points are defined, the pdf ${}_cf'_{\tilde{r}_i^0}$ of the resulting truncated distribution is obtained, and the discrete pdf $f_{\tilde{r}_i^0}$ is derived.

Concerning the choice of the continuous probability distribution, various distributions have been considered in the vehicle routing literature modelling stochasticity in travel times, including uniform (see Malandraki and Daskin (1992)), Gaussian (see Li et al. (2010)), log-normal (see Kaparias et al. (2008)), gamma (see Fan et al. (2005) and Taş et al. (2013)) and shifted gamma (see Russell and Urban (2008)). The computational results reported in the following sections have been obtained using a Gaussian distribution. Note, however, that the proposed models and the related solution methods are not distribution dependent. The parameters of the continuous Gaussian pdf ${}_cf_{\tilde{r}_i^0}$ are the expectation and the variance of the distribution, denoted as $\mu_{\tilde{r}_i^0}$ and $\sigma^2_{\tilde{r}_i^0}$, respectively. The truncating points for the distribution are defined as follows. The lower bound l_i^0 is taken as the maximum between 0 and the floor rounding of the 1st percentile and the upper bound u_i^0 as the ceil rounding of the 99th percentile of $\mathcal{N}\left(\mu_{\tilde{r}_i^0}, \sigma^2_{\tilde{r}_i^0}\right)$.

The truncated pdf is defined as

$$ {}_cf'_{\tilde{r}_i^0}(r) = \begin{cases} \frac{{}_cf_{\tilde{r}_i^0}(r)}{{}_cF_{\tilde{r}_i^0}(u_i^0) - {}_cF_{\tilde{r}_i^0}(l_i^0)} & l_i^0 < r \leq u_i^0 \\ 0 & \text{otherwise.} \end{cases} $$

The discrete truncated distribution $f_{\tilde{r}_i^0}$ is computed as

$$ f_{\tilde{r}_i^0}(r) = P(r-1 < \tilde{r}_i^0 \leq r) = {}_cF'_{\tilde{r}_i^0}(r) - {}_cF'_{\tilde{r}_i^0}(r-1). $$

The parameters $\mu_{\tilde{r}_i^0}$ and $\sigma^2_{\tilde{r}_i^0}$ of the Gaussian distribution of each release date are obtained by simulating the arrival of the vehicles delivering the parcels to the depot. Each vehicle is assumed to travel at a speed in the interval $[10, 100]$. The release dates generation procedure is described in Algorithm 6. In line 2 the distance at time $t = 0$ of each vehicle is computed as $d_0^i = v \cdot \bar{r}_i$ where v is the central point of the speed interval, i.e. $v = 55$. In line 3, the notation $\mathcal{N}'(\mu, \sigma, a, b)$ represents the truncated normal distribution with expected value μ, standard deviation σ, lower bound a, and upper bound b, while $x \leftarrow \mathcal{N}'$ indicates that x is a sample of distribution $\mathcal{N}'$. In particular, v_i^0, the speed of each vehicle at time $t = 0$, is sampled from a Gaussian distribution having expected value $\mu = v$ and standard deviation $\sigma = 19.3465$. This value for σ is chosen to let the bounds for the speed, i.e., $[10, 100]$, be the 1st and 99th percentile of the non-truncated Gaussian distribution. In lines 4, 5, and 6 the initial parameters $\mu_{\tilde{r}_i^0}$ and $\sigma^2_{\tilde{r}_i^0}$ of the random variable describing the release date of each customer are computed as the estimated time of arrival and the estimated traveling time left, respectively. These values are computed according to the current distance and speed of each vehicle and they are set to the same value.

In the case of customers in N_t^{static}, the information is not updated for $t > 0$. For these customers the actual release date r_i is assumed to fall within the interval $[l_i^0, u_i^0]$. The value is sampled from the random variable $\tilde{r}_i^0$, as shown at line 8 of Algorithm 6 where $r_i \leftarrow \tilde{r}_i^0$ indicates that r_i is sampled from $\tilde{r}_i^0$. Instead, for customers in D the information on the release date is updated as described in Section 4.6.4.

Algorithm 6 ReleaseDateGeneration

```
1: for all $i \in N_t^{static} \cup N_t^{dynamic}$ do
2:     $d_0^i = v \cdot \bar{r}_i$
3:     $v_i^0 \leftarrow \mathcal{N}'(55, 19.3465, 10, 100)$
4:     $\mu_{\tilde{r}_i^0} = \frac{d_i^0}{v_i^0}$
5:     $\sigma^2_{\tilde{r}_i^0} = \frac{d_i^0}{v_i^0}$
6:     $l_i^0 = \max\left\{\left\lfloor {}_cF^{-1}_{\tilde{r}_i^0}(0.01)\right\rfloor, 0\right\}$, $u_i^0 = \left\lceil {}_cF^{-1}_{\tilde{r}_i^0}(0.99)\right\rceil$
7:     if $i \in N_t^{static}$ then
8:         $r_i \leftarrow \tilde{r}_i^0$
9:     end if
10: end for
```

4.6.4 Release dates updating

At each time $t > 0$ the release date of each customer is updated as follows. The information available for customers in N_t^{static} is only updated if the parcel has arrived to the depot, meaning that if $t = r_i$, the value sampled in the generation phase, the customer is removed from set N_t^{static} and included in the set N_t^{known} of customers that can be served. For customers in $N_t^{dynamic}$, the release date is updated by simulating the traveling of the vehicles delivering the parcels to the depot. The distance of each vehicle is updated by decreasing its value by the amount travelled in the previous time unit, i.e., $d_i^t = d_i^{t-1} - v_i^{t-1}$. If the updated distance from the depot is null (or negative) the vehicle has reached the depot and the customer is included in the set N_t^{known} with release date $r_i = t$. Otherwise, the speed of the vehicle is updated and the new parameters of the release date are computed. $\mu_{\tilde{r}_i^t}$ is computed as the expected arrival time of the parcel, i.e., $t + \frac{d_t^i}{v_t^i}$, and $\sigma^2_{\tilde{r}_i^t}$ as the estimated residual traveling time, i.e., $\frac{d_t^i}{v_t^i}$. The lower bound l_i^t is computed as the floor rounding of the 1st percentile and the upper bound u_i^t as the ceil rounding of the 99th percentile of $\mathcal{N}\left(\mu_{\tilde{r}_i^t}, \sigma^2_{\tilde{r}_i^t}\right)$. As the lower bound of the release date cannot be past, the lower bound is computed as the maximum between the rounding of the 1st percentile and t.

Algorithm 7 describes the updating procedure for the release dates.

Algorithm 7 ReleaseDateUpdate

```
 1: for all $i \in N_t^{static} \cup N_t^{dynamic}$ do
 2:     if $i \in N_t^{static}$ then
 3:         if $t > l_i^t$ then
 4:             $l_i^t = t$
 5:         end if
 6:         if $t = r_i$ then
 7:             $N_t^{static} = N_t^{static} \setminus \{i\}$, $N_t^{known} = N_t^{known} \cup \{i\}$
 8:         end if
 9:     end if
10:     if $i \in N_t^{dynamic}$ then
11:         $d_i^t = d_i^{t-1} - v_i^{t-1}$
12:         if $d_i^t \leq 0$ then
13:             $N_t^{dynamic} = N_t^{dynamic} \setminus \{i\}$, $N_t^{known} = N_t^{known} \cup \{i\}$
14:             $r_i^t = t$
15:         else
16:             UpdateSpeed($i$)
17:             $\mu_{\tilde{r}_i^t} = t + \frac{d_t^i}{v_t^i}$
18:             $\sigma^2_{\tilde{r}_i^t} = \frac{d_t^i}{v_t^i}$
19:             $l_i^t = \max\left\{\left\lfloor {}_cF^{-1}_{\tilde{r}_i^t}(0.01)\right\rfloor, t\right\}$, $u_i^t = \left\lceil {}_cF^{-1}_{\tilde{r}_i^t}(0.99)\right\rceil$
20:         end if
21:     end if
22: end for
```

The speed is updated as a truncated random walk process with Gaussian steps. In particular, v_i^t, the speed at time t, is computed as the sum of the speed at time $t-1$, v_i^{t-1}, and a stochastic component, ϵ_i^t, sampled from the Gaussian distribution with $\mu = 0$ and $\sigma = 1$. The truncation is achieved by checking if the bounds on the speed, i.e., $[10, 100]$, have been violated and by setting v_i^t accordingly. The speed updating procedure is described in Algorithm 8.

Algorithm 8 UpdateSpeed(i), with $i \in N_t^{dynamic}$

```
1: $\epsilon_i^t \leftarrow N(0,1)$
2: $v_i^t = v_i^{t-1} + \epsilon_i^t$
3: **if** $v_i^t < 10$ **then**
4:     $v_i^t = 10$
5: **end if**
6: **if** $v_i^t > 100$ **then**
7:     $v_i^t = 100$
8: **end if**
```

4.6.5 Parameters for the instance generation

As described in Sections 4.6.1-4.6.4, the instance generation procedure takes three input: $\mathcal{I}$, the instance name, β, the parameter describing how spread out in time the release dates are compared to the TSP travel time of the instance, and δ, describing the ratio of customers in $N_t^{dynamic}$ with respect to those in $N_t^{static} \cup N_t^{dynamic}$, at the beginning of the distribution. In this section the impact of the parameters β and δ is analyzed, to allow for a better understanding of the role they play in the instance generation process.

Figure 4.4 presents the effect of β on the distribution of the supports of the release dates in time. The plots refer to three instances with value of $\beta = 0.5, 1, 1.5$, from top to bottom, and with $\delta = 0$. The x-axis reports the time and the y-axis the customers of the instance. The horizontal lines of each plot represent the support of the distribution function of the release dates of customers. Each line corresponds to one customer. Customers are sorted by increasing value of the expected release date, from bottom to top. As β increases, the release dates are sampled in a broader interval with respect to d_{TSP}. This allows us to investigate both the cases in which it is better to wait for more customers to serve in longer routes, i.e., when β is small, and the one where it is more advantageous to serve customers as their parcel reaches the depot, i.e., when β increases.

Figure 4.5 shows the evolution of the expected value and bounds of the release dates when $\delta = 1$. The figure reports the evolution of the release date of six customers, sorted by the value of the initial expected release dates, allowing us to understand the effect of describing the speed of the vehicles as a truncated random walk on the evolution of the release dates in time. At the beginning of the distribution, the release dates have a given variance, based on the distance from the depot. As time goes on, the expected arrival time each vehicle might be advanced or delayed, based on various conditions, e.g., traffic and deliveries to other distribution centers. As the vehicle travels the potential causes of uncertainty decrease and the estimation of the arrival time gets more precise, until the moment in which the vehicle reaches the depot.

4.7 Computational experiments

In this section we describe the computational experiments that have been carried out to assess the performance of the two models and the three dynamic policies. The aim is to understand whether, in the presented reoptimization and stochastic setting, it is better to consider a point estimation or the entire stochastic information for

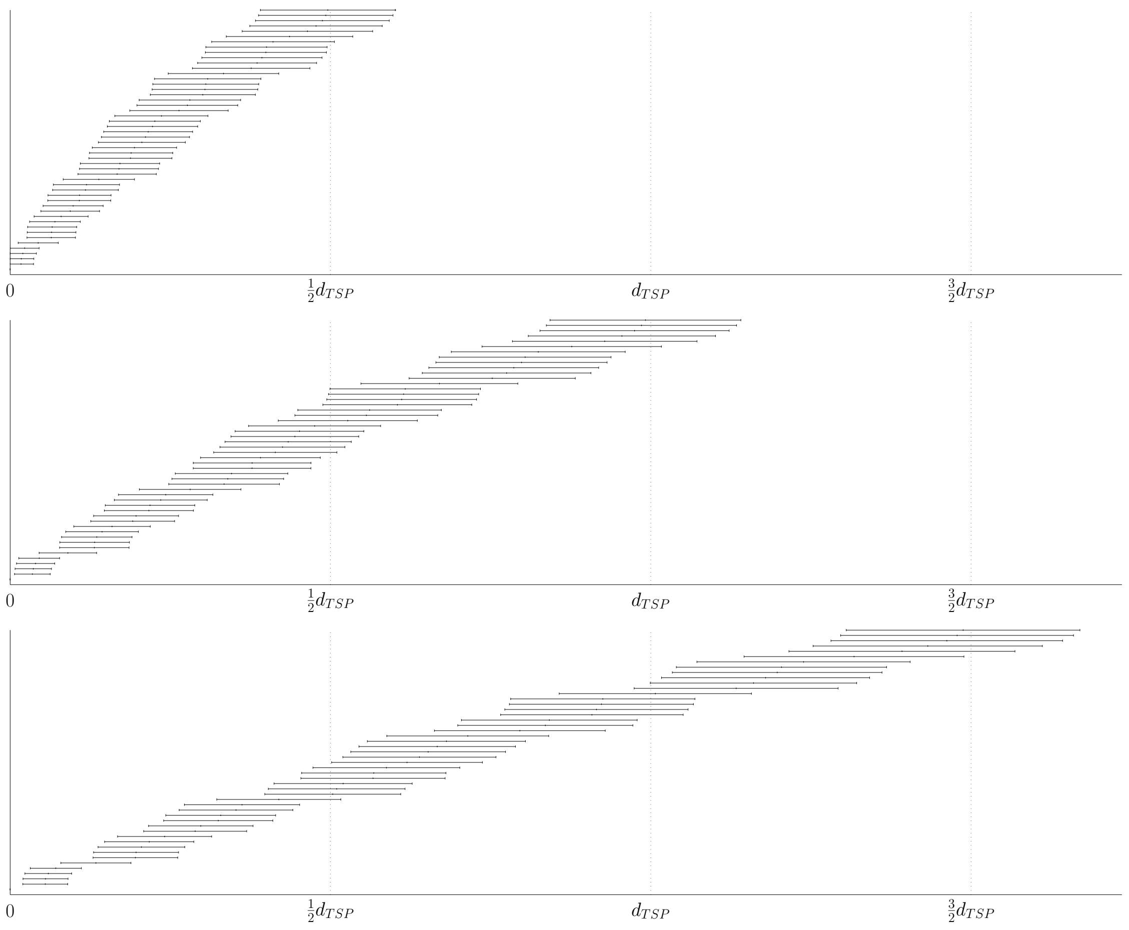


Figure 4.4: The effect of different values of β on the release dates. $\beta = 0.5, 1, 1.5$ from top to bottom.

the release dates and if increasing the number of reoptimization epochs improves the quality of the solution.

In Section 4.7.1, preliminary experiments are preformed. Then, in Section 4.7.2, the final results are presented.

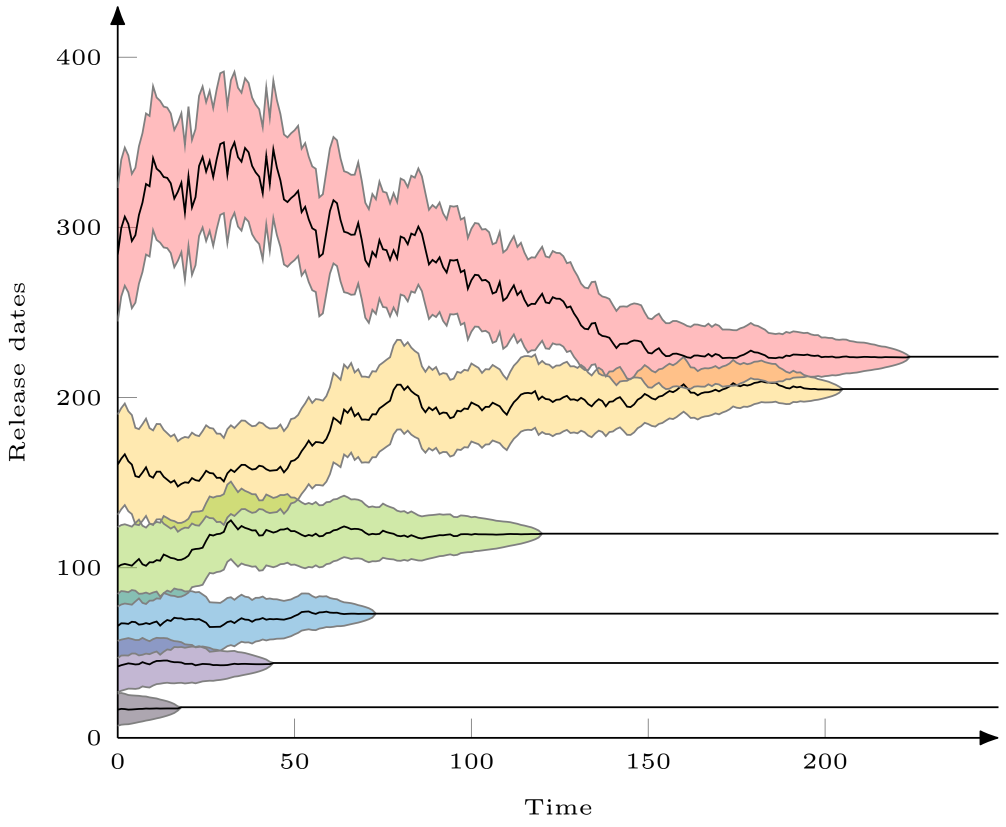


Figure 4.5: The evolution of the release dates when $\delta = 1$.

4.7.1 Preliminary experiments

In this section preliminary experiments are reported to identify the time limit for each reoptimization and investigate the relationship between the instance size and the number of ILS iterations performed.

The time limit $T_{\max}$ for each reoptimization, expressed in minutes, has been obtained through a set of preliminary experiments. Tests have been carried out for $T_{\max} \in \{1, 5, 10\}$ on the instances with $\mathcal{I} = C101$, $\beta \in \{0.5, 1, 1.5\}$, and $\delta = \{0, 0.5, 1\}$. It must be noted that, as the time required for each reoptimization is not taken into account when computing the value of the objective function, the implicit assumption is that the considered time unit is larger than the value of $T_{\max}$. The maximum number of iterations without improvement has been set to $100 \cdot T_{\max}$. The results are reported in Table 4.3. Column "Avg" shows, for each tested value of $T_{\max}$, the average performance gap of each model from the best solution found by that model across all the tested values of $T_{\max}$. As the deterministic model has shown some outlying results for the instance with $\beta = 1.5$ and $\delta = 0.5$, the average percentage gap excluding such instance, for both the deterministic and stochastic models, is reported in the column "Avg*". This value shows an increase in the quality of the solution found by the deterministic model as the time allowed for each reoptimization increases, while the stochastic model shows no benefit when $T_{\max}$ is increased from 5 to 10 minutes. The column "Max Sim. Time" shows the maximum time required to run each simulation, i.e., all the reoptimizations required to solve an instance, for each value of $T_{\max}$, across all model-policy combinations. The value $T_{\max} = 5$ has been chosen as it allowed to run each simulation in under two hours without significantly undermining the performance of the two models.

	Deterministic		Stochastic		
$T_{\max}$ (minutes)	Avg	Avg*	Avg	Avg*	Max Sim. Time (minutes)
1	1.63	1.35	1.37	1.37	21
5	1.43	1.27	0.66	0.61	102
10	1.68	1.21	0.66	0.62	201

Table 4.3: The results of the tests for each $T_{\max}$.

The performance of the ILS depends on the number of iterations and therefore on the time that each iteration requires. Table 4.4 reports for instance $\mathcal{I} = C101$, $\beta = 1$, $\delta = 0.5$, the average number of iterations performed by each model by size of the subgraph of customers in $N_t^{unserved}$, on the three policies. The "-" symbol indicates that the model has not been solved for subgraphs of that size. It is shown how the number of iterations required when solving the problem considering the full probabilistic information for the release dates is substantially smaller than in the deterministic case, especially in the early stages of the simulation, when the size of the subgraph considered in the reoptimization is larger.

	Number of iterations	
Size	Deterministic	Stochastic
50-46	718	18
45-41	715	63
40-36	-	201
35-31	613	226
30-26	2248	500
25-21	1765	500
5-1	500	500

Table 4.4: Results on the average number of ILS iterations for each model in the instance.

As the number of ILS performed within $T_{\max}$ appears to be related to the size of the instance to be solved, tests have been conducted to measure the time to perform one ILS iteration for different instance sizes. The results are reported in Table 4.5 for instance $\mathcal{I} = C101$, $\beta = 1$, $\delta = 0.5$. Column "t_{LS}" reports the total time required for the LS, "n_{LS}" the number of LS iterations performed and "$\bar{t}_{LS}$" the average time required for one LS iteration. It is shown how the average time required to perform a LS iteration for the deterministic model appears to be constant with respect to the instance size while it increases when the stochastic model is solved. It must also be noted that when the instance size is greater or equal to 70 nodes the cost of one LS iteration for the stochastic model becomes greater than $T_{\max}$.

	Deterministic			Stochastic		
Size	t_{LS}	n_{LS}	$\bar{t}_{LS}$	t_{LS}	n_{LS}	$\bar{t}_{LS}$
50	0.11	8	0.014	49.14	16	3.07
60	0.14	8	0.017	46.34	14	3.31
70	0.28	22	0.013	335.93	36	9.33
80	0.41	58	0.007	406.88	15	27.13
90	0.25	42	0.006	557.44	15	37.16
100	0.34	31	0.011	310.07	16	19.38

Table 4.5: Time in seconds to perform one ILS iteration.

4.7.2 Final results

Considering the results in Section 4.7.1, all computational results are performed on instances with 50 customers. The six model-policy combinations have been compared on the entire instance set described in Section 4.6. The results are shown by comparing the performance of the two models (deterministic vs. stochastic) and the three reoptimization policies (R, RD and RDW) and of the myopic policy. Then, an analysis of the solutions is carried out. As the instance generation process is dependent on the sampling of random variables, five different seeds are generated for each combination of $\mathcal{I}$, β, and δ. The results in this section are computed as averages of these five different scenarios.

Performance of models and policies

Table 4.6 shows the average of the percentage gap of each solution from the best solution found by all model-policy combinations. In the top left corner the performance of the two models is shown with respect to the parameter δ. On average, the stochastic model is shown to find better solutions across all values of δ. A higher rate of dynamic customers appears to have a negative impact on the performance of the two models: the percentage gap of the deterministic model shows a positive trend with respect to δ, while the performance of the stochastic model worsens as δ goes from 0 to 0.5 and shows only a slight reduction when δ is increased to 1. The bottom left corner reports the performance of the models with different values of β. Again, the stochastic model finds on average the best solution for all tested values. Interestingly, $\beta = 1$ appears to be the value for which the stochastic model has the largest advantage over the deterministic one. This could be explained by the fact that $\beta = 1$ is the case in which it is harder to find the best trade-off between serving customers and waiting at the depot, as no component of the objective function, i.e., the waiting time at the depot caused by the release dates and the cost of traveling, outweighs the other. It appears that, by exploiting the full probabilistic information, the stochastic model better copes with the increased challenge.

The top central part of Table 4.6 allows us to analyze the different reoptimization policies for different values of δ. It is shown that the quality of the solution improves as the number of reoptimizations increases and the benefit increases for an increasing value of δ. The bottom central part shows a comparison of the three policies for each value of β. The RDW policy is again the one finding the best solution across each value of β. Furthermore, with the sole exception of the R policy, the case with $\beta = 1$ appears to be the most challenging one, reflecting the result found when comparing the two models.

To show the benefits of the proposed reoptimization technique, the instances have also been solved with a myopic policy. Such heuristic works as follows: every time the vehicle is at the depot, it immediately starts a route to serve the customers in N_t^{known}. If $N_t^{known} = \emptyset$, it waits for the first parcel arriving to the depot and leaves as soon as it arrives. The results are reported in the rightmost column of Table 4.6. On average, the solutions found by such heuristic are shown to be more than 12% worse than the best solution found. The performance of the myopic policy improves as β increases. This is explained by the fact that as β increases, it gets more beneficial to serve customers in shorter routes, i.e., serving less customers per route with a higher number of routes and reducing waiting times.

	Avg. Deterministic	Avg. Stochastic	Avg. R	Avg. RD	Avg. RDW	Avg. Myopic
$\delta = 0$	2.69	0.72	1.77	1.68	1.67	11.96
$\delta = 0.5$	2.96	1.40	2.35	2.17	2.02	14.74
$\delta = 1$	3.10	1.34	3.03	2.39	2.25	10.85
$\beta = 0.5$	1.91	1.28	1.67	1.61	1.51	24.22
$\beta = 1$	3.79	0.90	2.37	2.56	2.11	11.10
$\beta = 1.5$	3.05	1.29	2.40	2.09	2.02	2.23
Avg.	2.92	1.16	2.14	2.09	1.88	12.52

Table 4.6: Average percentage gap from the best solution.

More detailed results of the average percentage gap are reported for each model-policy pair and for the myopic policy in Table 4.7. It is worth noting that, although the results of the myopic policy are on average worse than the ones found by the reoptimization approach, the case when $\delta = 0$ and $\beta = 1.5$ is best solved by serving customers as soon as their parcel is delivered to the depot. This is due to the fact that when release dates are sufficiently spread out in time to make serving each customer individually become the optimal solution, the myopic policy finds the optimal solution. Table 4.7 also highlights how, on average, the best results are found by the stochastic model in combination with the RDW policy.

		Deterministic			Stochastic			
δ	β	R	RD	RDW	R	RD	RDW	Myopic
0	0.5	0.61	0.41	0.46	0.76	0.82	0.73	25.19
	1	4.79	4.55	4.24	0.48	0.22	0.46	10.69
	1.5	2.92	3.07	3.19	1.05	1.05	0.92	0
	Avg.	2.77	2.67	2.63	0.76	0.70	0.70	11.96
0.5	0.5	2.29	2.33	2.31	1.76	1.03	0.99	25.89
	1	4.10	3.69	2.97	0.85	2.20	1.49	15.09
	1.5	3.15	2.57	3.19	1.92	1.21	1.18	3.26
	Avg.	3.18	2.87	2.82	1.51	1.48	1.22	14.74
1	0.5	2.47	3.44	2.85	2.11	1.61	1.71	21.58
	1	3.12	3.57	3.06	0.86	1.12	0.42	7.53
	1.5	3.71	2.71	2.94	1.65	1.94	0.68	3.43
	Avg.	3.10	3.24	2.95	1.54	1.56	0.94	10.85
Avg.		3.02	2.93	2.80	1.27	1.24	0.95	12.52

Table 4.7: Detailed results for the average percentage gap from the best solution.

Analysis of the solutions

To allow for a better comparison of the proposed models and to better understand the differences in the solutions found by the different models and policies, Tables 4.8, 4.9, 4.10, 4.11, and 4.12 are presented, replicating the structure of Table 4.6. Table 4.8 reports the average total time required to run each simulation, Table 4.9 reports the number of reoptimizations carried out, Table 4.10 reports the average number of routes across all tested instances, Table 4.11 shows the average reoptimization time per route, and Table 4.12 the average percentage of waiting time over the total cost of the solution for each model and reoptimization policy and for all values of δ and β. The results of the myopic policy are omitted when not relevant. More detailed results are reported, for each of the metrics, in the Appendix (Section 4.9).

Table 4.8 reports the average total time required to run each simulation. Notably, when comparing the two models for different values of δ, it is shown that the stochastic model consistently requires about 5 minutes more than the deterministic one. When comparing the two models with respect to the parameter β, the time required on average for each simulation increases for both models. It is shown, however, that the difference in the required time for the stochastic model to the deterministic one decreases as β increases. The total time is larger when the reoptimization is carried out more frequently. In particular, the RDW policy takes more than twice the time required by the R policy.

Table 4.9 shows a comparison of the models and policies with respect to the number of reoptimizations. The two models show a very similar number of reoptimizations across all values of δ and β. As expected, the number of reoptimizations increases from the R policy to the RD and RDW policies. Such increase is shown to be correlated with the parameter β. This is due to the fact that the more release dates are spread out in time, the more it is beneficial to serve customers in a higher num-

	Avg. Deterministic	Avg. Stochastic	Avg. R	Avg. RD	Avg. RDW
$\delta = 0$	40.36	46.29	23.20	36.11	70.65
$\delta = 0.5$	32.12	37.04	22.28	33.55	47.90
$\delta = 1$	37.79	43.57	25.64	40.04	56.35
$\beta = 0.5$	16.62	26.39	14.21	21.39	28.92
$\beta = 1$	37.69	43.37	25.10	38.95	57.54
$\beta = 1.5$	55.96	57.13	31.81	49.36	88.45
Avg.	36.75	42.30	23.71	36.57	58.30

Table 4.8: Average time required for each simulation (in minutes).

ber of shorter routes. As the reoptimization epochs are dependent on the number of routes and on the starting time of such routes, when β increases more reoptimizations are performed.

Table 4.10 reports a comparison of the number of routes for each model and policy. It is observed that, on average, the stochastic model finds solutions where the distribution is completed with less routes than the ones found by the deterministic model. This result holds across all values of δ. When the models are compared with respect to the parameter β it only holds for $\beta = 1.5$. Table 4.10 confirms the positive correlation between the parameter β and the average number of routes in each solution. This result also validates the increasing competitiveness of the myopic policy, highlighted in Table 4.6. While this result might seem contradictory with the fact that the solutions found by the myopic policy have on average less routes than the ones found with any of the proposed reoptimization policies, it must be noted that the case in which the myopic policy performs better, i.e., $\beta = 1.5$, is also the case in which the difference in the number of routes with any of the reoptimization policies

	Avg. Deterministic	Avg. Stochastic	Avg. R	Avg. RD	Avg. RDW
$\delta = 0$	10.71	10.63	5.98	9.12	16.90
$\delta = 0.5$	9.07	9.11	5.89	8.79	12.58
$\delta = 1$	10.41	10.69	6.66	10.10	14.88
$\beta = 0.5$	6.37	6.33	4.33	6.27	8.46
$\beta = 1$	10.49	10.78	6.61	10.12	15.18
$\beta = 1.5$	13.32	13.31	7.59	11.63	20.73
Avg.	10.06	10.14	6.18	9.34	14.79

Table 4.9: Average number of reoptimizations.

is minimum. Finally, the number of routes appears to be consistent across the three different reoptimization policies.

Table 4.11 allows us to investigate the behavior of the ratio between the total time required to run each simulation over the number of routes. This value can be interpreted as the average time spent solving the model in order to obtain the next route to be implemented. When comparing the two models, the stochastic one is shown to require more time per route across all values of β and δ. Looking at the results for different values of δ, the case of $\delta = 0$ is the one requiring the longest time both when considering the two models and when comparing the three policies. Both the two models and the three policies show a positive trend with respect to the parameter β.

Further analysis of the structure of the solutions found has been carried out by comparing the percentage of time spent waiting at the depot over the total time required for the distribution. The results are reported in Table 4.12. It can be

	Avg. Deterministic	Avg. Stochastic	Avg. R	Avg. RD	Avg. RDW	Avg. Myopic
$\delta = 0$	6.11	5.79	5.98	5.86	6.01	5.25
$\delta = 0.5$	6.13	5.93	5.92	5.96	6.21	4.97
$\delta = 1$	7.08	6.33	6.66	6.60	6.85	5.38
$\beta = 0.5$	4.15	4.53	4.33	4.33	4.36	3.70
$\beta = 1$	6.58	6.67	6.61	6.58	6.69	5.08
$\beta = 1.5$	8.59	6.84	7.62	7.52	8.02	6.82
Avg.	6.44	6.01	6.19	6.14	6.36	5.20

Table 4.10: Average number of routes.

observed that the stochastic model has on average a higher rate of waiting time over the deterministic model. This result confirms those found when looking at the number of routes of the solutions: as the stochastic model tends to serve customers with less routes compared to the deterministic model, more time is spent waiting at the depot for the parcels of the customers. The results found for the myopic policy reflect its strategy of serving customers as soon as possible: as this approach only waits at the depot if there are no customers that can be served, the percentage waiting time is shown to be lower compared to the solutions found when solving the models considering the information available about future release dates.

4.8 Conclusions

In this chapter than dynamic traveling salesman with stochastic release dates (DTSP-srd) is introduced. A solution approach is proposed based on a reoptimization technique. Three policies are introduced to define the reoptimization epochs, with in-

	Avg. Deterministic	Avg. Stochastic	Avg. R	Avg. RD	Avg. RDW
$\delta = 0$	6.28	7.90	3.75	6.00	11.51
$\delta = 0.5$	5.12	6.28	3.61	5.63	7.87
$\delta = 1$	5.19	6.82	3.71	6.00	8.31
$\beta = 0.5$	4.06	6.01	3.19	4.97	6.94
$\beta = 1$	5.75	6.58	3.74	5.94	8.81
$\beta = 1.5$	6.79	8.41	4.14	6.72	11.93
Avg.	5.53	7.00	3.69	5.88	9.23

Table 4.11: Average time (in minutes) over the number of routes.

creasing frequency of reoptimization, and two models are proposed for the solution of the problem at each epoch. The first one considers a point estimation of the release dates and the second one makes use of the entire probabilistic information available. The models have been solved with an iterated local search heuristic. Extensive computational tests have been carried out on a large instance set. The results show that a more frequent reoptimization provides better results across all tested instances. Furthermore, it has been observed that, on average, on instances with 50 customers, better solutions are found when the problem is solved with the stochastic model. In particular, the myopic policy is shown to perform more the 12% worse than the best solution found by the two models. The deterministic and stochastic model have an average percentage gap from the best solution found across the three dynamic policies of 2.92% and 1.16%, respectively. When considering this result together with the number of iterations performed by the heuristic for each of the two models, the deterministic model appears to be a good compromise, especially when the number of customers increases.

	Avg. Deterministic	Avg. Stochastic	Avg. R	Avg. RD	Avg. RDW	Avg. Myopic
$\delta = 0$	9.13	11.00	9.79	10.23	10.18	1.67
$\delta = 0.5$	5.88	6.53	6.08	6.13	6.40	1.43
$\delta = 1$	6.64	8.37	7.49	8.02	7.00	2.28
$\beta = 0.5$	2.81	2.13	2.32	2.51	2.57	0
$\beta = 1$	7.91	6.93	7.13	7.87	7.25	0.77
$\beta = 1.5$	10.93	16.85	13.91	14.00	13.76	4.60
Avg.	7.22	8.63	7.79	8.13	7.86	1.79

Table 4.12: Average percentage waiting time.

Several research directions remain open. As in the proposed approach no action is considered when the vehicle is not at the depot, such option could be considered by reoptimizing while the vehicle is at customers location and allowing for an anticipated return to the depot, as a further effort towards a better distribution. Furthermore, different probability distributions could be considered for the release dates and the case where the distribution assumed is different from the actual distribution of the release dates could be investigated. The DTSP-srd considers one uncapacitated vehicle to perform the distribution. A natural extension of the problem would allow multiple capacitated vehicles to be deployed.

4.9 Appendix: Results

		Deterministic			Stochastic		
δ	β	R	RD	RDW	R	RD	RDW
0	0.5	9.82	14.71	22.42	17.24	24.35	33.24
	1	21.97	32.30	63.16	28.78	42.96	67.37
	1.5	35.02	51.35	112.46	26.35	51.02	125.27
0.5	0.5	8.82	16.23	21.91	19.18	25.98	33.82
	1	21.28	34.11	43.10	25.09	36.28	52.98
	1.5	34.40	46.13	63.07	24.91	42.57	72.54
1	0.5	11.53	19.05	25.06	18.65	28.03	37.05
	1	27.84	42.91	52.53	25.63	45.12	66.11
	1.5	38.99	54.12	68.08	31.22	51.00	89.29

Table 4.13: Detailed results for the total time required for each simulation (in minutes).

		Deterministic			Stochastic		
δ	β	R	RD	RDW	R	RD	RDW
0	0.5	2.41	3.62	5.62	3.99	5.80	8.32
	1	3.27	5.10	9.77	4.10	6.00	9.61
	1.5	4.33	6.87	15.00	4.44	8.59	19.61
0.5	0.5	2.27	4.19	5.19	4.06	5.70	7.79
	1	3.53	5.70	6.98	3.90	5.86	8.11
	1.5	4.18	5.60	7.92	3.90	6.44	9.92
1	0.5	2.65	4.34	6.04	3.90	5.98	8.06
	1	3.76	5.94	7.26	3.94	6.90	10.02
	1.5	4.05	5.90	6.57	4.04	6.81	11.18

Table 4.14: Detailed results for the total time over the number of routes (in minutes).

		Deterministic			Stochastic		
δ	β	R	RD	RDW	R	RD	RDW
0	0.5	4.10	5.85	8.85	4.30	5.70	7.40
	1	6.60	9.60	17.35	7.00	10.45	17.00
	1.5	8.00	11.60	24.40	5.90	11.50	26.40
0.5	0.5	3.85	6.15	8.20	4.70	6.20	7.85
	1	6.00	9.60	12.50	6.40	8.95	13.05
	1.5	8.05	11.45	15.85	6.35	10.40	18.05
1	0.5	4.30	6.90	9.15	4.75	6.80	9.30
	1	7.20	11.10	14.45	6.45	11.00	16.70
	1.5	9.55	12.90	18.10	7.70	11.90	21.60

Table 4.15: Detailed results for number of reoptimizations.

		Deterministic			Stochastic		
δ	β	R	RD	RDW	R	RD	RDW
0	0.5	4.10	4.05	4.05	4.30	4.20	4.20
	1	6.60	6.25	6.40	7.00	7.15	7.00
	1.5	8.00	7.60	7.95	5.90	5.90	6.45
0.5	0.5	3.85	3.95	4.35	4.70	4.60	4.55
	1	6.00	6.05	6.30	6.40	6.25	6.65
	1.5	8.20	8.30	8.15	6.35	6.60	7.25
1	0.5	4.30	4.40	4.30	4.75	4.75	4.70
	1	7.20	7.20	7.20	6.45	6.55	6.60
	1.5	9.55	9.20	10.35	7.70	7.50	7.95

Table 4.16: Detailed results for the number of routes.

		Deterministic			Stochastic			
δ	β	R	RD	RDW	R	RD	RDW	Myopic
0	0.5	1.35	1.36	2.61	0.45	0.61	0.48	0.00
	1	9.65	10.09	9.84	6.72	6.28	6.87	0.00
	1.5	14.37	16.67	16.23	26.20	26.40	25.04	5.00
0.5	0.5	2.48	2.52	2.27	1.77	1.61	1.75	0.00
	1	6.08	6.89	5.01	4.97	5.52	6.16	0.24
	1.5	8.62	9.05	10.05	12.57	11.22	13.18	4.06
1	0.5	3.92	4.93	3.85	3.94	4.06	4.47	0.00
	1	7.22	8.79	7.58	8.13	9.65	8.05	2.08
	1.5	8.72	7.47	7.23	12.99	13.22	10.82	4.75

Table 4.17: Detailed results for the percentage waiting time.

Chapter 5

Booking of loading/unloading areas

This chapter is based on the result of ongoing research project on the subject of booking managements systems for loading and unloading areas. The work has been performed by the author in collaboration with Prof. M. Grazia Speranza and Prof. José M. Viegas.

5.1 Abstract

City distribution usually requires vehicles to temporarily stop at roadside to allow for the driver to perform the last leg of the delivery by foot. The stops take place in designated areas, called loading/unloading (L/U) areas. In this chapter the introduction of a booking system for the management of the L/U areas in a city center is studies as a way to eliminate double parking. Two booking management system and the arising routing problems are presented. The first booking management proposed in the one where distributors book in sequence, accounting for the reservations that have already been placed. The second considers a centralized system that collects all the required

stops at L/U areas and finds a reservation for each distributor. The optimization problems arising in each of the two approaches are presented. The solutions provided by the two booking systems are discussed and compared with a representation of the current use of the L/U areas, where the distributors do not consider the availability of a parking spot and resort to double if none is available.

5.2 Introduction

The rapid increase of demand of goods and services in the city center and the comparatively static nature of the road infrastructure, among other aspects, make transportation in the urban areas a very relevant topic. This increase in the demand in urban areas is the result of various factors, such as the increase rate of people moving to cities, with the rate of urbanized population expected to reach 80% worldwide by 2050 (see Bettencourt and West (2010)), social and economic growth. These phenomena, together with technological advancement and the wide adoption of e-commerce, have also caused the need of a fast and readily available distribution system. The management of the road network is therefore assuming a more and more relevant role in the quality of the distribution system of a urban area and in the quality of life of its citizens. These and many other aspects of the challenges and opportunities of city logistics are discussed in Savelsbergh and Van Woensel (2016).

In this context, we propose to consider the management of loading and unloading zones reserved by the municipality to commercial vehicles. City distribution usually requires vehicles to temporarily stop at roadside to allow for the driver to perform the last leg of the delivery by foot. The stops take place in designated areas, called loading/unloading areas (henceforth shortened L/U areas), each with a specific number of available parking spots. The high demand for L/U areas and the scarcity of space available in city centers makes the management of such areas a crucial factor in the city distribution for all the stakeholders involved: the municipality wants to reduce double parking and its effects on traffic, the distributors want to easily access L/U areas to minimize the discomfort during the delivery, including fines for double parking, and do not want the infrastructure to have a negative impact on the service they provide. Such designated areas are identified by the municipality but typically little to no management is performed after the locations have been established. In some cases, areas are forced to be shared in time between the vehicles by imposing a time limit on each stop (e.g. by means of a parking disc). Such limit is enforced by the traffic police.

One of the ways to deal with double parking arising from L/U areas misuse would be to regulate the use of these locations through a reservation system, controlled by the municipality, with compulsory reservation made by the distributors. In particular, distributors would have to specify the expected time of arrival and the duration of the occupation. The booking would be made possible by a simple front-end interface, e.g., a web interface. Several systems could be conceived to control the parking on the L/U areas, such as GPS or RFID-based electronic toll collection systems with the latter also being capable of directly imputing the (possible) price to be paid for the stop.

The management of L/U areas requires the involvement of all the stakeholders (customers, distributors, municipality, traffic police) and their acceptance that it is in everybody interest for the system to perform well as it allows for less discomfort both in terms of delivery delays and traffic flow. In the early stages of adoption the traffic police would have to be on top of the new system, educating any distributors that comes without reservation so that they go away, ask their office to make the reservation for some time later (informing the customer once it is accepted) and then come back at the time when a delivery position is available. Meetings among all the stakeholders could be scheduled to share some statistics of adoption of the system, occupation of the delivery positions across the hours of the day and days of the week, and check on the problems each category is having on the use of the system. This is a system of devolved responsibility that only imposes a minimum of discipline in the use of scarce resources, and generates the information necessary for a revision of the number of places available and times of preferential use of those spaces for deliveries. In this paper, different booking management systems for the L/U areas are proposed. We study how the routing of each distributor would be affected if such system were to be put in place, with respect to the current situation, where distributors resort to double parking. The comparison is presented on instances derived from real data of the city of Lisbon.

The remainder of this paper is organized as follows. In Section 5.3 a review of the related literature on L/U areas, the issue of double parking and how this problem is discussed in the field of optimization is presented. In Section 5.4 the routing problems with loading and unloading areas is discussed. Two booking management systems are introduced in Section 5.5. The first, where reservations for the L/U areas are placed sequentially, is introduced in Section 5.6, while the second, where the reservations are handled centrally by the municipality, is discussed in Section 5.7. Computational results are presented in Section 5.8. Conclusions are drawn in Section 5.9.

5.3 Literature review

The impact of parking availability on commercial vehicle costs and operations is studied in Figliozzi and Tipagornwong (2017). Among other aspects, the authors investigate how the insufficient availability of on-street parking and L/U areas during certain periods of the day in dense and congested urban areas, affects the behavior of the distributors. The authors present a modeling framework for parking availability that combines queuing and logistical models and conclude that double parking is unlikely to disappear from urban areas unless more dedicated L/U areas are made available at peak times and that increasing parking fines and enforcement of the regulations can discourage double parking but will not eradicate the problem for a sufficiently high demand/supply ratio. This result is also confirmed by the fact that double parking fines are being considered as part of the business costs by some large deliver companies, such as FedEx or UPS, and that therefore a more productive long-term policy would be to require enough parking spaces for freight and service vehicles.

The shortage of available parking spots for commercial vehicles is caused not only by an undersized infrastructure but frequently also by the misuse of the reserved spots by private vehicles, as reported by Aiura and Taniguchi (2005) and Alho et al. (2018). Aiura and Taniguchi (2005) studies the facility location problem of the planning of on-street L/U spaces by minimizing delay penalties, fixed costs, taking in consideration the behavior of pickup-delivery vehicles as well as passengers cars. The authors conclude that, together with infrastructural improvements, the management of such infrastructure needs to be reviewed both at the planning and evaluation level. Alho et al. (2018) proposes to study the reduction of the double parking of freight vehicles by changing the spatial configuration and the non-freight vehicles parking rule compliance level. The authors adopt a microsimulation approach with a dedicated representation of double parking for freight vehicles. Among the findings of the paper, the authors report the disproportionate effects of the externalities caused by double parking, which can be the cause of delays ranging from 10% up to 63% and decrease the average speed by 4% up to 17% on the network.

A parking choice modeling simulation is presented in Nourinejad et al. (2014)

for the study of truck parking policies, such as time restrictions, pricing policies and space management and enforcement. The simulation captures various dimensions of the parking activity such as walking distance, congestion impact and parking search times. Two scenarios based on the Toronto area are presented to validate the model.

Considering the management of L/U areas in a routing problem introduces aspects that are similar to those typical of scheduling problems. While integrating routing and scheduling is not new in the literature, the scheduling aspect is often focused on the resources on the distributor side, such as the scheduling of vehicles or personnel, whereas we propose to consider the scheduling of L/U areas, which are infrastructural resources. The scheduling aspect of routing with L/U areas shares some similarities with the parking allocation problem. Within this field the paper that most closely describes the dynamics of the L/U areas presented in this paper is Roca-Riu et al. (2015). The authors study a system in which carriers are required to park in designated areas only and have to communicate to the municipality a time window for the beginning and the duration of their operation in advance. The authors define the Parking slot Assignment Problem (PAP) on one L/U area with a given capacity as the problem of finding a feasible assignment of the requests to the parking spaces of the area within the operation horizon, such that the time window for the beginning of each request is satisfied and that the capacity of the area is not violated. Since a feasible solution for the problem might not exist, the time window specified by each carrier is assumed to be flexible, in the sense that the one they are assigned to might not be the one they requested. It is also assumed that carriers will accept and respect the assigned interval. The authors evaluate different measures to induce fairness in the allocation of the resources and to penalize and measure the non-accomplishment of the requests. It is worth noting that, with regard to the acceptance and respect of the assigned interval, the study presented in this paper could yield to higher acceptance rate of the interval assigned to each carrier, as the cost of routing is also considered.

Roca-Riu et al. (2017) studies the dynamic allocation of driving lanes to L/U operations to maximize delivery opportunities while reducing traffic disruption. A simulation study is used to evaluate the model and estimate its benefits compared to real situations where commercial vehicles resort to double parking. The authors devise the conditions under which temporarily allocating shoulder lanes as L/U areas reduces the vehicles delay compared to the case where commercial vehicles resort to double parking.

The reservation of capacitated areas and its effects on the vehicle routing is a new topic in the vehicle routing literature. To the best of our knowledge, no publication exists that addresses this problem. Few known variants of the Vehicle Routing Problem (VRP), however, are related to its setting and, in particular, to the way the proposed problem discusses interdependencies in the routing of the vehicles, constraints on the capacity of the visited locations, and routing in urban areas in general. The first case is represented by variants such as the VRP with synchronization, in which dependencies between vehicle visits are considered, such as the visit at the same node of two vehicles happening simultaneously or in a before-after fashion. An extensive survey on VRPs with synchronization is provided in Drexl (2012). We refer, in particular, to the section considering resource synchronization, defined as the case where different vehicles compete for a common scarce resource, and the literature cited therein. One work that stands out is Hempsch and Irnich (2008) where, among other restrictions, the authors consider a limit in the number of docking stations at the depots. Considering the booking of parking spots could also be beneficial to the routing of electric vehicles, as it allows for a reduction of the uncertainty in the availability of a free spot at capacitated charging stations. Capacitated charging stations in a routing problem with electric vehicles are considered in Froger et al. (2017). The authors study an electric VRP with non-linear charging function and capacitated charging stations. Different formulations are presented and a two stage solution method is proposed for the problem.

The VRP with L/U areas can be seen as part of a wider trend in the optimization literature, and of routing problems in particular, aiming to improve the efficiency in city logistics and reduce its effect on traffic, environment and the quality of life in general. A survey on the routing problems in city logistics is provided in Cattaruzza et al. (2017).

5.4 Routing with loading/unloading areas

Several optimization problems arise when dealing with the logistics of routing with loading and unloading areas when different levels of integration of the decision process between each distributor and the booking system administrator, e.g., the municipality, are considered.

The first optimization problem considered in this paper is the one that more closely resembles the current state of the management of L/U areas. Each vehicle has to fulfill customer requests in a city center and to do so it must stop at a number of L/U areas for the driver to perform the last leg of the distribution by walking to the customers while the vehicle remains parked. Each distributor acts independently and finds the best routing among the L/U areas required to perform its deliveries, regardless of the occupancy level of the L/U area at the time the vehicle arrives. The length of the stay at each L/U area is known and the routing problem to be solved is a Traveling Salesman Problem (TSP) on the chosen L/U areas. The solution for all the vehicles having to deliver in the city center can be seen as a representation of the current state of the distribution of the city center, where vehicles resort to double parking when a fully occupied L/U area is found. We will refer to this as the “Independent TSPs” solution.

Various optimization problems arise according to the different ways the municipality could deal with the management of the L/U areas to increase the efficiency of the L/U areas usage. We propose to study the effects of a booking system that could be introduced, where all vehicles having to deliver parcels to customers in the city center can place reservations for the L/U areas. We discuss different optimization problems arising based on how the booking system is managed. These booking systems and the related problems are introduced in Section 5.5.

5.5 Booking of L/U areas

Distributors often face bottlenecks at L/U areas due to various factors, such as peaks in the delivery requests either in time, in space, or both, e.g., all shops in the fashion district requiring the clothes to be delivered in the morning, or systemic problems in the allocation of L/U areas, e.g., scarcity or poor placement of the spots. Vehicles response to the unavailability of L/U spots is often double parking. One way to deal with this problem is the introduction of a booking system.

What would happen when such system is put in place is that the use of L/U areas would shift from being similar to the one of a car park, with vehicle occupying a spot as soon as it becomes available and double park or cruising for parking if all the spots in the desired area are taken, to one where each distributor has to place a reservation to the L/U areas booking system to access a spot. A reservation for an L/U area would consist of the starting time and the length of the stop. Various strategies could be devised for the booking system to handle the reservations and consequently different optimization problems would have to be solved. In this paper we investigate two different systems. In the first one, reservations are placed by each distributor over time, in sequence. Each time a distributor wants to place a reservation, it queries the booking system which replies with the free time slots for each L/U area, meaning the time intervals in which at least one parking spot of the L/U area is not booked. The distributor can then place a reservation only if is compatible with the time slots currently available in the requested areas. As the booking system is only in charge of providing the current state of the booking of the area and checking if the new reservation violates the capacity constraints of the areas, reservations can be placed at any time and an immediate confirmation is given to the distributor.

The second strategy we propose is a centralized management of the bookings. All the distributors that need to stop at an L/U area provide the booking system with the required starting time and length of the stops. As the reservations provided by the distributors do not consider the capacity constraints of the L/U areas (and in fact might very well be the result of the Independent TSPs described in Section 5.4), no confirmation can be given at the time the reservations are placed and the starting time of the booking for each L/U area can be taken as preferred starting time, meaning that the system is allowed to make changes to obtain a feasible solution. The reservations are collected up to a deadline (e.g., a given hour of the previous day) after which an optimization method is run to seek a feasible reservation for each distributor, taking into account the routing cost between the areas the distributor has placed a reservation for. As in this approach the booking system is also in charge of solving an optimization problem, this approach would likely be implemented as a one-day-look-ahead system, where distributors can place reservation requests up to a certain hour of the day before the one the requests are for. The first booking system is described in Section 5.6 and the second is described in Section 5.7.

5.6 Sequential booking

In the case of sequential booking, distributors can place a reservation at any time, with the constraint that the booking of an area cannot overlap with times in which the area has already been fully booked. On the distributor side, this system works as follows. When its set of deliveries to be performed is known, meaning the set of L/U areas to be visited and the length of the stay in each selected area, the distributor can obtain from the booking system the set of available time slots for the requested L/U areas (e.g., by logging-in in the booking website). Once all these information are known, a routing problem is solved, where each visit to an L/U area must happen within one of the time windows in which the area has not been fully booked by previous reservations, and whose span is enough to accommodate the length of the stay required by the distributor. We call the problem faced by each distributor the Traveling Salesman Problem with multiple Time Windows (TSPmTW). The formal definition of the problem and the proposed formulation are presented in Section 5.6.1.

Once the distributor has planned its distribution it books the stops it needs. The reservations happen in sequence, meaning that the reservation of the k^{th} distributor has to take into account all the reservations placed by the $k-1$ distributors that have already booked.

A booking system based on the sequential solution of a TSPmTW by each distributor has the benefit of being a relatively simple method to introduce booking as a way to manage L/U areas and eliminate double parking. It would require little effort from the distributors and therefore has a high chance of being well-received. It also allows for real time checking of the availability of an L/U area time slot and immediate confirmation of the booking. This approach has, however, various downsides. Firstly, the booking of each area is the result of the convenience of the distributors, who reserve the areas with the unique objective of minimizing routing costs, which could result in a fragmented and therefore inefficient use of the L/U areas. Secondly, booking sequentially can be highly penalizing for the late booking distributor to the point of encouraging the distributors to disregard booking times.

It is worth noting that, as the solution of each TSPmTW takes into account the availability limitation caused by the reservations already placed by other distributors, the solution obtained by solving a TSPmTW for each distributor is free of double parking.

5.6.1 The Traveling Salesman Problem with multiple Time Windows

The Traveling Salesman Problem with multiple Time Windows is defined as follows. Let $G = (V, A)$ be a directed graph, where the vertex 0 represents the depot and the vertices $U = \{1, \dots, |U|\}$ represent the L/U areas in which the vehicle has to stop. The required length of the stop at area u is denoted as $s_u > 0$. Each area $u \in U$ is characterized by a set of time windows TW_u in which it has not been fully booked by previous reservations. Time windows in TW_u are denoted as $TW_{u,1}, TW_{u,2}$, etc, with the lower and upper bound of the h^{th} time window of L/U area u denoted as $TW^a_{u,h}$ and $TW^b_{u,h}$, respectively, meaning that the area u is not fully booked from time $TW^a_{u,h}$ to $TW^b_{u,h}$, included. Time windows $TW_{u,h} \in TW_u$ are such that the span is broad enough to accommodate the length of the stop of the vehicle. We denote as H_u the index set of TW_u, i.e., $H_u := \{h \in \mathbb{N} | TW_{u,h} \in TW_u\}$. The TSPmTW is the problem of finding the shortest route starting and ending at the depot such that the vehicle stops in one of the allowed time windows for each of the L/U areas.

For modeling purposes a sink depot is added as returning point of the route. We call this the return depot, denoted as $|U|+1$. Furthermore, we let the stop lengths be defined for all $i \in V \cup \{|U|+1\}$ with stop length at the depot set to zero, i.e., $s_0 = 0$. The formulation relies on the following variables:

- $x_{ij} = \begin{cases} 1 \text{ if the route travels through edge } (i,j) \in A, \\ 0 \text{ otherwise,} \end{cases}$

- $T_i \geq 0$, the arrival time of the vehicle to vertex i,

- $y_{u,h} = \begin{cases} 1 \text{ if the vehicle stops at area } u \text{ in the time window } TW_{u,h}, \\ 0 \text{ otherwise,} \end{cases}$

Time is assumed to be discrete, meaning that $t_{ij}, s_i, T_i, \in \mathbb{N}$, for all $i, j \in V \cup |U+1|$, and $TW^a_{u,h}, TW^b_{u,h} \in \mathbb{N}$, for all $u \in U, h \in H_u$. The resulting model is the following:

$$\min T_{|U|+1} \tag{5.1}$$

s.t.

$$\sum_{u \in U} x_{0u} = 1 \tag{5.2}$$

$$\sum_{u \in U} x_{u(|U|+1)} = 1 \tag{5.3}$$

$$\sum_{i \in \{0\} \cup U} x_{iu} = \sum_{i \in U \cup \{|U|+1\}} x_{ui} = 1 \qquad u \in U, \tag{5.4}$$

$$(T_i + t_{ij} + s_i - T_j) \leq M(1 - x_{ij}) \qquad i \in \{0\} \cup U, j \in U \cup \{|U|+1\}, \tag{5.5}$$

$$T_u \geq TW^a_{u,m} y_{u,m} \qquad u \in U, m \in M_u, \tag{5.6}$$

$$T_u + s_u \leq TW^b_{u,m} + M(1 - y_{u,m}) \qquad u \in U, m \in M_u, \tag{5.7}$$

$$\sum_{m \in M_u} y_{u,m} = 1 \qquad u \in U, \tag{5.8}$$

$$x_{ij} \in \{0,1\} \qquad i \in \{0\} \cup U, j \in U \cup \{|U|+1\}, \tag{5.9}$$

$$T_i \geq 0 \qquad i \in \{v_k, v_k+1\} \cup U, \tag{5.10}$$

$$y_{u,m} \in \{0,1\} \qquad u \in U, m \in M_u. \tag{5.11}$$

The objective function (5.1) minimizes the arrival time of the vehicle to the return depot. Constraints (5.2) and (5.3) impose that the route starts at the depot and ends at the return depot. Constraints (5.4) are flow conservation constraints. Constraints (5.5) regulate the arrival time of the vehicle at each vertex. Constraints (5.6) and (5.7) impose that the vehicle must arrive and depart within the chosen time window for each L/U area and constraints (5.8) impose that only one time window per L/U area is used. Constraints (5.9)-(5.11) are domain constraints.

5.7 Centralized booking

The L/U areas booking system could be built in such a way that the booking system itself is in charge of finding a routing plan and a schedule for the stops of each distributor. This would let the municipality the possibility to manage the L/U areas and reservations and to find a solution that takes into account the stops required by each distributor and creates less fragmented time slots for any last minute/same day reservations, providing the distributors with a solution that does not penalize late bookers and has a lower average additional routing cost over the TSP cost, compared to the sequential booking. This approach, however, its downsides. As the system would be in charge of finding a feasible reservation for all distributors minimizing routing costs, the reservations would have to be placed in advance and the starting time of the booking of each area would have to be considered as a preference rather than a requirement, to allow for the solution of potential conflicts. This makes it impossible to give the distributor a confirmation of the reservation at the time it is placed. While this might seem like a little price to pay to achieve the benefit of a lower average routing cost, it is worth noting that, in practice, distribution of goods is hardly limited to the aspects of routing in a city center and the booking of the stops at L/U areas.

Solving this problem, however, has the great value of providing the solution a booking system should aim for and will also be used as benchmark for the comparison of the solution found with the sequential booking or other approaches for the booking of L/U areas that could be devised.

Should a centralized system be implemented, the reservations, consisting of the required length of the stay in a selected subset of the L/U areas, could be collected up to a certain hour of the previous day, after which an optimization routine would run to reduce (and potentially remove) any potential overbooking of the L/U areas.

Once the reservation deadline is passed, an optimization problem is solved to find the best solution with no capacity violations. We call this problem the multiple Traveling Salesman with node scheduling (mTSPns). The problem is formally introduced in Section 5.7.1, where a formulation is also proposed.

5.7.1 The multiple Traveling Salesman Problem with node scheduling

The mTSPns is defined as follows. Let $G = (V, A)$ be a directed graph. The set of vertices V is composed of the set U of L/U areas and the set of vertices $\{v_k\}_{k \in K}$ representing the depots of all vehicles. Let $U_k \subseteq U$ be the set of L/U areas in which vehicle k has to stop and s_u^k be the length of the stop of the vehicle in the L/U area

u, then $U_k := \{u \in U | s_u^k > 0\}, k \in K$. Furthermore, let Q_u be the capacity of L/U area u, meaning the maximum number of vehicles that can be parked at the area at the same time. The mTSPns is the problem of finding a route for each vehicle starting and ending at its depot such that the total cost of the routes is minimum while not violating the capacity of each L/U area. For modeling purposes a sink depot is added to the graph for each vehicle. The return depot of vehicle k is denoted as $v_k + 1$. Furthermore, we let the stop lengths of vehicle k be defined for all $i \in \{0\} \cup U_k$ with stop length at the depot set to zero, i.e., $s_0^k = 0$. The formulation relies on the following variables:

- $x_{ij}^k = \begin{cases} 1 \text{ if the route of vehicle } k \text{ travels through edge } (i,j) \in A, \\ 0 \text{ otherwise,} \end{cases}$
- $T_i^k \geq 0$, the arrival time of vehicle k at vertex i,
- $a_{u,k}^t = \begin{cases} 1 \text{ if the vehicle } k \text{ has already reached the area } u \text{ at time } t, \\ 0 \text{ otherwise,} \end{cases}$
- $b_{u,k}^t = \begin{cases} 1 \text{ if the vehicle } k \text{ has already left area } u \text{ at time } t, \\ 0 \text{ otherwise,} \end{cases}$

Time is assumed to be discrete, meaning that $t_{ij}, s_u^k, T_u^k, t \in \mathbb{N}$, for all $i, j \in V \cup \{v_k + 1\}_{k \in K}, u \in U, k \in K$. The proposed formulation for the mTSPns is provided below.

$$\min \sum_{k \in K} T_{v_k+1}^k \tag{5.12}$$

s.t.

$$\sum_{j \in U_k \cup \{v_k+1\}} x^k_{v_k j} = 1 \qquad k \in K, \tag{5.13}$$

$$\sum_{j \in \{v_k\} \cup U_k} x^k_{j(v_k+1)} = 1 \qquad k \in K, \tag{5.14}$$

$$\sum_{j \in \{v_k\} \cup U_k} x^k_{ju} = \sum_{j \in U_k \cup \{v_k+1\}} x^k_{uj} = 1 \qquad u \in U_k, k \in K, \tag{5.15}$$

$$\left(T^k_i + t_{ij} + s^k_i - T^k_j\right) \leq M(1 - x^k_{ij}) \qquad i \in \{v_k\} \cup U_k, j \in U_k \cup \{v_k + 1\}, k \in K, \tag{5.16}$$

$$T^k_u \geq t + 1 - M a^t_{u,k} \qquad u \in U, k \in K, t \in [0, M], \tag{5.17}$$

$$T^k_u \leq t + M(1 - a^t_{u,k}) \qquad u \in U, k \in K, t \in [0, M], \tag{5.18}$$

$$T^k_u + s^k_u \geq t - M b^t_{u,k} \qquad u \in U, k \in K, t \in [0, M], \tag{5.19}$$

$$T^k_u + s^k_u \leq t - 1 + M(1 - b^t_{u,k}) \qquad u \in U, k \in K, t \in [0, M] \tag{5.20}$$

$$\sum_{k \in K} a^t_{u,k} - b^t_{u,k} \leq Q_u \qquad u \in U, t \in [0, M], \tag{5.21}$$

$$x^k_{ij} \in \{0, 1\} \qquad i, j \in \{v_k, v_k + 1\} \cup U, k \in K, \tag{5.22}$$

$$T^k_i \geq 0 \qquad i \in \{v_k, v_k + 1\} \cup U, k \in K, \tag{5.23}$$

$$a^t_{u,k} \in \{0, 1\} \qquad u \in U, k \in K, t \in [0, M] \tag{5.24}$$

$$b^t_{u,k} \in \{0, 1\} \qquad u \in U, k \in K, t \in [0, M] \tag{5.25}$$

The objective function (5.12) minimizes the route duration of the vehicles. Constraints (5.13) and (5.14) impose that each vehicle departs and returns to the depot and constraints (5.15) enforce flow conservation. Constraints (5.16) regulate the arrival time of each vehicle at the vertices it must visit. Constraints (5.17) and (5.19) impose $w^t_{u,k} = 1$ if the vehicle k is at L/U area u at time t. The capacity of each L/U area is imposed with constraints (5.21). Finally, (5.22)-(5.25) are domain constraints. Since no upper bound is imposed on the working horizon in the areas, constraints (5.19)-(5.21) exist for $t \in [0, M]$. The value of the large constant M is a crucial factor in the order of magnitude of the number of constraints in the formulation.

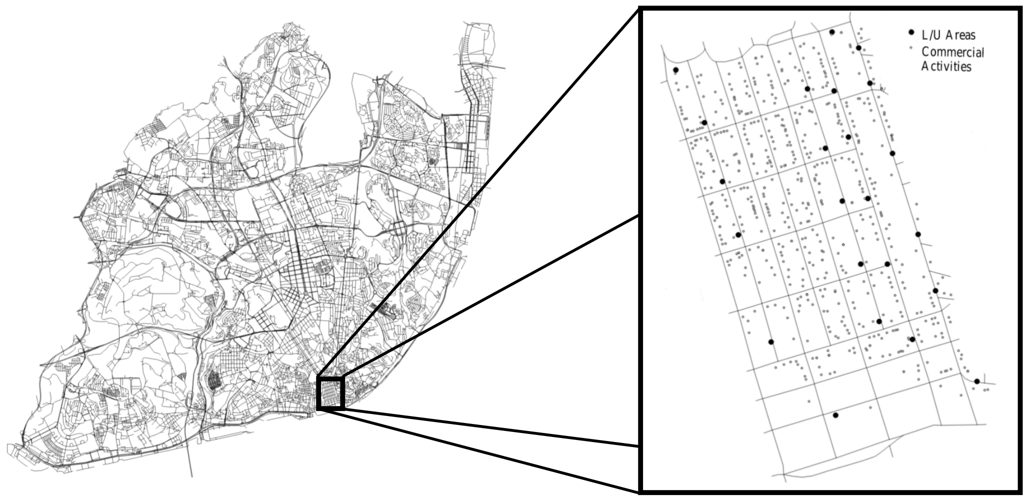
5.8 Computational experiments

In this section computational results are presented to compare the three proposed approaches. Firstly, the instance generation process is presented, then the solution method for each approach is discussed and, finally, results are presented.

5.8.1 Instance generation

The proposed solution methods have been tested on the layout of the Lisbon city center, in particular, in the neighborhood called Baixa. As the exact borders of the area are unclear we call Baixa the area delimited by Praa Dom Pedro IV and Praa da Figueira to the north, Praa do Comrcio to the south, Rua da Madalena to the east, Rua urea to the north-west and Praa do Municipio to the south-west (see Figure 5.1). In such neighborhood, 23 L/U areas have been placed over the years by the municipality, with capacity spanning from a minimum of 1 to a maximum of 8 spots, with an average of 3.26 spots per area. According to the 2010 census of commercial activities, 584 activities are located in the Baixa. For reference, the average walking distance from the closest L/U area is 39.45 seconds. Traveling by car between the L/U areas in the Baixa requires between 1 and 5 minutes.

The data has been obtained through the open data websites of the city of Lisbon, in particular, the data from the commercial census is available at `http://dados.cm-lisboa.pt/en/dataset/recenseamento-comercial-2010`, and that of the L/U areas location is available at `http://dadosabertos.cm-lisboa.pt/dataset/emel`. The latter includes all on-street parking. L/U areas can be highlighted by selecting only the records with attribute "Tipologia" equal to "Cargas e Descargas".



The road network of Lisbon The L/U areas in the Baixa

Figure 5.1: The layout of the instances.

The instance generation process takes two kinds of inputs, the number of L/U areas and the number of vehicles, and works as follows. As customers and L/U areas are not placed uniformly in Baixa, different areas have different booking requests, i.e., some areas are more requested than others. To create non-trivial instances that would allow for an evaluation of the benefits and downsides of the three presented approaches, this difference in the demand of the L/U areas is reproduced as follows. Firstly, the customers in Baixa are matched to the closest L/U area by walking distance. The probability of a vehicle requesting a stop in an L/U area is then computed as the ratio of the number of customers matched to that area over the total number of customers, normalized so that the area with the lowest ratio has probability 0.50 of being visited and the one with the highest ratio has probability 0.75 of being visited by a vehicle. Once the L/U areas to be visited by each vehicle have been sampled, the length of the stay for each vehicle in each L/U areas is sampled in the interval $[5, 10]$ minutes. As the routing from and to the depot does not involve any capacity constraint on the time of the visit, we consider, without loss of generality, only the routing and parking time spent in the Baixa, meaning that the cost of the arcs leaving the depot and returning to the depot is zero.

5.8.2 Solution method

The framework to solve the three approaches has been implemented in Java with CPLEX 12.6 API and run using a machine with 3.5 GHz Intel Xeon E5-1650v2 processor and 64 GB of RAM. The solution of each of the three approaches has been carried out as follows.

Independent TSPs

The solution of each of the TSPs in the independent TSPs approach, reflecting the current state of the distribution, has been obtained with the Lin-Kernighan heuristic (see Lin and Kernighan (1973)), in the implementation provided by Helsgaun (2000).

Sequential booking

The solution of the sequential booking approach is obtained by solving as many TSPmTW as vehicles, in sequence. Before the solution of the TSPmTW of the k^{th} distributor, the time windows of each area are obtained by fetching the arrival and departure times of all the $k-1$ distributors that have booked a stop in that area and computing the occupancy level of the area in time and the consequent time intervals in which the area is not fully booked. An example of the occupancy level of an area over time is shown in Figure 5.2. In the example, the area has a capacity of two vehicles and three distributors have already placed a reservation. The arrival time of a distributor is denoted as “A” and the departure as “D”. The time windows in which the area can be visited are those in which it is not shown to be fully booked.

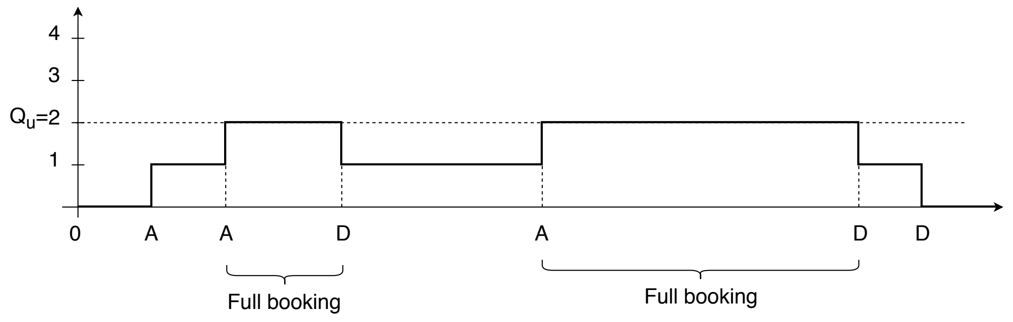


Figure 5.2: The step function of the occupancy leval of a L/U area with the full booking intervals highlighted.

The solution of each TSPmTW in the sequential booking approach has been obtained by solving the formulation reported in Section 5.6.1 with CPLEX, with time limit set to ten minutes.

Centralized booking

The solution of the centralized approach is obtained by solving a mTSPns. The solution of the problem is obtained by solving the formulation presented in Section 5.7.1 with CPLEX, with the time limit set to one hour. In particular, because of the high number of constraints required to enforce the capacity of each area at each time, CPLEX is initially provided with the formulation with the exclusion of constraints (5.17)-(5.21), which are added as lazy constraints every time a solution for the current model is found. The separation of the violated constraints works similarly to the computation of the time windows in the sequential approach. Specifically, once a solution is obtained, the occupancy level over time of an L/U area is computed based on the arrival and departure times of the solution found by CPLEX. An example of the evolution of the occupancy level is shown in Figure 5.3, highlighting the time interval in which the area is overbooked. For each L/U area, constraints (5.17)-(5.21) are added for each time t in which the area is overbooked.

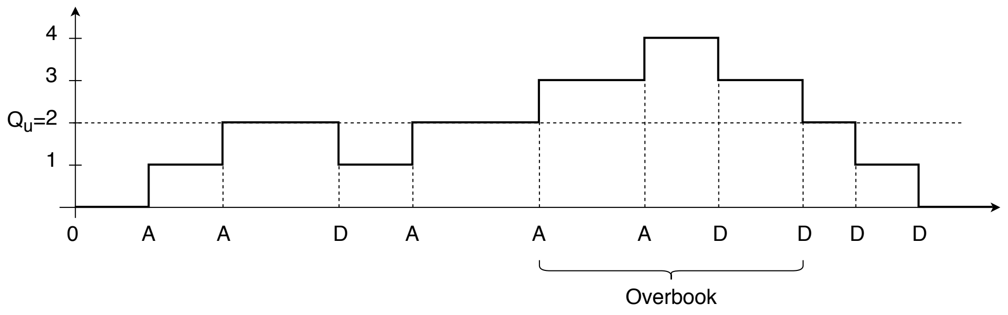


Figure 5.3: The function of the occupancy level of a L/U area with the overbooked interval highlighted.

5.8.3 Results

The computational experiments have been designed to assess the largest instances for which CPLEX finds an optimal solution to the formulation proposed for the mTSPns and to allow for a comparison of the three presented approaches.

To understand the maximum size of instances for which a mTSPns could be solved within one hour, instances have been tested with increasing number of L/U areas and number of distributors. The results are reported in Table 5.1 where "OPT" means that the optimal solution was found, "FEAS" means that a solution was found within the time limit but the optimality could not be proven, and "OOT" means that the execution ran out of time without finding any feasible solution. The results highlight how instances with $|U| + |K| \leq 8$ can be solved to optimality within one hour.

A comparison of the three approaches with respect to the routing costs of each distributor is reported in Table 5.2 and shown in Figure 5.4 for the instance with $|K| = 6$ and $|U| = 2$. The instance has been chosen as it is the one with the highest distributors to areas ratio, meaning that it is the one in which the L/U areas are

	$\|U\| = 2$	$\|U\| = 3$	$\|U\| = 4$	$\|U\| = 5$	$\|U\| = 6$	$\|U\| = 7$	$\|U\| = 8$	$\|U\| = 9$	$\|U\| = 10$
$\|K\| = 2$	**OPT**	**OPT**	**OPT**	**OPT**	**OPT**	**OPT**	**OPT**	FEAS	OOT
$\|K\| = 3$	**OPT**	**OPT**	**OPT**	**OPT**	FEAS	OOT	OOT	OOT	OOT
$\|K\| = 4$	**OPT**	**OPT**	FEAS	OOT	OOT	OOT	OOT	OOT	OOT
$\|K\| = 5$	**OPT**	**OPT**	FEAS	OOT	OOT	OOT	OOT	OOT	OOT
$\|K\| = 6$	**OPT**	FEAS	OOT	OOT	OOT	OOT	OOT	OOT	OOT
$\|K\| = 7$	FEAS	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT
$\|K\| = 8$	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT

Table 5.1: The results of the solution of the mTSPns different size of the instance.

more scarce with respect to the demand. It must be reported that the first area has capacity $Q_1 = 1$ and the second area has capacity $Q_2 = 2$. It is observed how, in comparison with the centralized system, the sequential booking of the areas is shown to provide a suboptimal solution, with an optimality gap of 15.84%. As expected, as reservations are placed, the routing of the distributors is increasingly worse with respect to the cost of the TSP, i.e., the solution they would have achieved disregarding the capacity of the L/U areas. The worse performance is also confirmed by higher standard deviation of the routing costs in the sequential approach compared to the centralized approach.

5.9 Conclusions

The conclusions drawn from results highlighted in the computational result section for the centralized approach are manifold.

To increase the size of the instances that could be solved to optimality, different approaches could be devised to express the capacity constraints of the L/U areas. For instance, each L/U area could be represented with as many dummy L/U areas

	IndepTSPs	Sequential	Centralized
$k = 0$	100	100	966.66
$k = 1$	100	100	133.33
$k = 2$	100	400	566.66
$k = 3$	100	766.66	100
$k = 4$	100	1033.33	1233.33
$k = 5$	100	1500	366.66
Avg.	100	650	561.11
S.Dev.	0	508.36	418.29

Table 5.2: A comparison of the routing costs for the instance with $|U| = 2$ and $|K| = 6$.

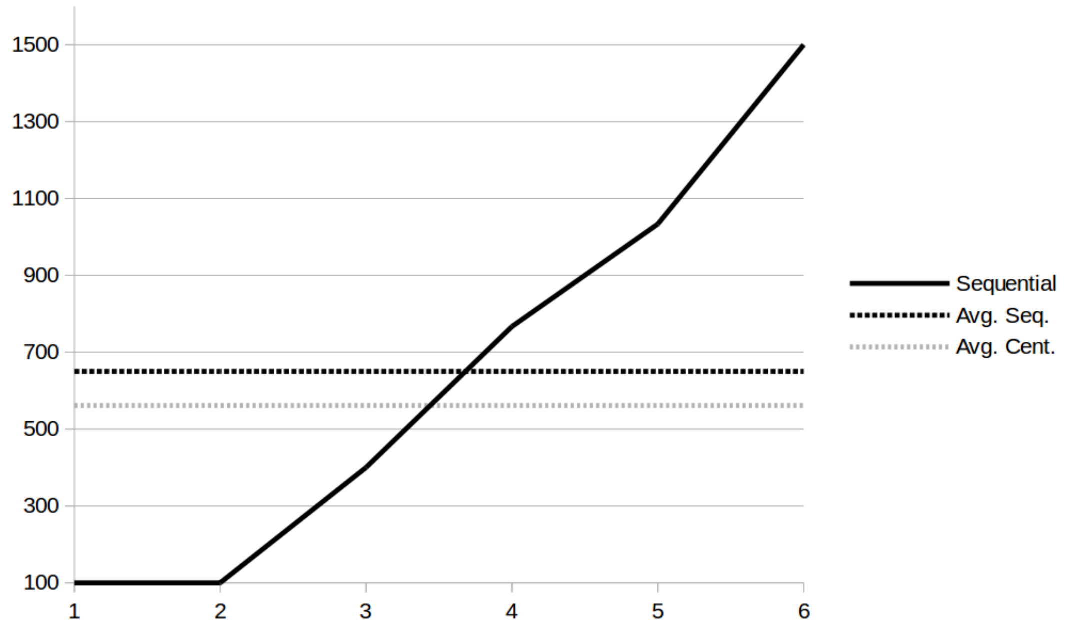


Figure 5.4: A comparison of the routing costs for the instance with $|U| = 2$ and $|K| = 6$.

with capacity one as its capacity and scheduling constraints could be enforced on each dummy L/U area. Other approaches could be considered drawing from the literature on resource constrained scheduling problems or adapting to the problem of routing with L/U areas the ones proposed in Froger et al. (2017) for the electric VRP with capacitated charging stations. The problem could also benefit from a tailor-made exact method.

The main problem that this paper aims to study, however, is not the one of finding optimal solution to the routing with L/U areas, but the one of finding an approach for the booking management system of L/U areas to solve the problem of double parking of commercial vehicles in the road network of a city. With this aim, providing the distributors with a good solution within a short time is a crucial factor in the adoption rate of the booking system. Various strategies could be tested to improve the results of the sequential approach to reduce the gap from the solution found in the centralized approach, such as dynamic pricing to smooth peaks in demand and time restricted access. The approach presented in the paper could also be used to highlight whether the L/U areas infrastructure is over or undersized either spatially or temporally and to assess whether the system could benefit from a dynamic allocation of parking spots as L/U areas.

Finally, to better simulate the real world dynamics of the L/U areas usage and obtain more realistic results, various factors could be incorporated in the instances. The most prominent is perhaps to include daily trend in the cost of routing and in the number of booking requests. The generation of such trends could benefit from the use of data from connected devices.

5.10 Acknowledgements

The authors wish to acknowledge the help provided by Vasco Damas Mra of the Cmara Municipal de Lisboa in the processing of the GIS data upon which the instances of this work are built.

Chapter 6

Conclusions

This thesis can be seen as a contribution to the study of relevant routing issues in the urban environment. The first part of the dissertation has been devoted to the study of the logistics of distribution centers and, in particular, to studying the implication of considering the arrival time of the parcels at the distribution centers, called their release dates, when planning for the delivery routes. The routing problem with release dates have been framed within the literature of routing problems by providing a classification based on the decisions to be taken when planning for the distribution. The class of routing problems in which timing decisions must be considered in addition to the characteristics of the classic Vehicle Routing Problems has been presented and the literature of the problems discussed within this class has been surveyed. To assess the benefits of considering the release dates when planning the delivery from the distribution center, the Traveling Salesman Problem with release dates has been studied under different settings. Firstly, the release dates have been considered as static and deterministic, to allow for a better understanding of the way release dates shape distribution planning. To better represent the impact of the information available in a real case scenario, the problem has then been studied in the setting where release dates are characterized by stochasticity and dynamically updated. This reflects the uncertainty in the arrival time of the vehicles delivering to the distribution center and their ability to provide updated information about the estimated time of arrival.

The second part of the thesis focused on the delivery phase of the distribution in urban areas. City distribution requires vehicles to temporarily stop at loading and unloading areas to perform the last leg of the delivery by foot. If a spot is not available, vehicles resort to double parking, which is a known cause of road congestion. The introduction of a booking management system has been studied as a solution for the problem. Two booking management systems have been discussed, and the arising routing problem presented. The solutions provided by the two systems have been compared with the current state of the distribution.

The urban environment and the needs and behavior of its population are constantly changing. City logistics must quickly adapt to these changes. Within the field of vehicle routing problems a considerable aid for a better decision-making can be envisioned from two aspects: integration and information.

The benefit of integration of different aspects of freight transportation and city logistics has already been proven in various aspects of routing. Prominent examples are the integration of inventory and routing decisions studied in the Inventory Routing Problem, discussed in Chapter 2, and the growing literature on rich vehicle routing, considering the complex characteristics of real-life VRPs. Scientific and technological advancements allow to tackle more and more complex problems, narrowing in this way the gap between scientific literature and real life applications.

One of the most promising research subjects, not only in the fields of city logistics and vehicle routing, arises from the wide adoption of connected devices, both personal and vehicular, and the evolution of urban areas towards smart cities. The processing of the huge amount of data generated by such devices is one of the main challenges of the present times and of the years to come. Considering this information will make for a better decision-making at the operational, tactical and strategical level.

Bibliography

M.K.I. Abdul Rahim, Y. Zhong, E.-H. Aghezzaf, and T. Aouam. Modelling and solving the multiperiod inventory-routing problem with stochastic stationary demand rates. *International Journal of Production Research*, 52:4351–4363, 2014.

N. Aiura and E. Taniguchi. Planning on-street loading-unloading spaces considering the behaviour of pickup-delivery vehicles. *Journal of the Eastern Asia Society for Transportation Studies*, 6:2963–2974, 2005.

D. Aksen, O. Kaya, F.S. Salman, and Ö. Tüncel. An adaptive large neighborhood search algorithm for a selective and periodic inventory routing problem. *European Journal of Operational Research*, 239:413–426, 2014.

A.R. Alho, J. de Abreu e Silva, J. Pinho de Sousa, and E. Blanco. Improving mobility by optimizing the number, location and usage of loading/unloading bays for urban freight vehicles. *Transportation Research Part D: Transport and Environment*, 61: 3–18, 2018.

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

H. Andersson, M. Christiansen, and G. Desaulniers. A new decomposition algorithm for a liquefied natural gas inventory routing problem. *International Journal of Production Research*, 54:564–578, 2016.

C. Archetti and M.G. Speranza. Vehicle routing problems with split deliveries. *International Transactions on Operations Research*, 19:3–22, 2012.

C. Archetti and M.G. Speranza. The inventory routing problem: the value of integration. *International Transactions in Operational Research*, 23:393–407, 2016.

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

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

C. Archetti, M.G. Speranza, and D. Vigo. Chapter 10: Vehicle routing problems with profits. In *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, pages 273–297. SIAM, 2014b.

C. Archetti, D. Feillet, and M.G. Speranza. Complexity of routing problems with release dates. *European Journal of Operational Research*, 247:797–803, 2015a.

C. Archetti, O. Jabali, and M.G. Speranza. Multi-period vehicle routing problem with due dates. *Computers & Operations Research*, 61:122–134, 2015b.

C. Archetti, E. Fernández, and D.L. Huerta-Muñoz. The flexible periodic vehicle routing problem. *Computers & Operations Research*, 85:58–70, 2017.

C. Archetti, D. Feillet, A. Mor, and M.G. Speranza. An iterated local search for the traveling salesman problem with release dates and completion time minimization. *Computers & Operations Research*, 98:24–37, 2018.

A. Azadeh, S. Elahi, M.H. Farahani, and B. Nasirian. A genetic algorithm-taguchi based approach to inventory routing problem of a single perishable product with transshipment. *Computers & Industrial Engineering*, 104:124–133, 2017.

N. Azi, M. Gendreau, and J.-Y. Potvin. An exact algorithm for a single-vehicle routing problem with time windows and multiple routes. *European journal of operational research*, 178:755–766, 2007.

N. Azi, M. Gendreau, and J.-Y. Potvin. An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles. *European Journal of Operational Research*, 202:756–763, 2010.

N. Azi, M. Gendreau, and J.-Y. Potvin. An adaptive large neighborhood search for a vehicle routing problem with multiple routes. *Computers & Operations Research*, 41:167–173, 2014.

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

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

L. Bertazzi and M.G. Speranza. Inventory routing problems: an introduction. *EURO Journal on Transportation and Logistics*, 1:307–326, 2012.

L. Bertazzi and M.G. Speranza. Inventory routing problems with multiple customers. *EURO Journal on Transportation and Logistics*, 2:255–275, 2013.

L. Bertazzi, A. Bosco, and D. Laganà. Managing stochastic demand in an inventory routing problem with transportation procurement. *Omega*, 56:112–121, 2015.

L. Bertazzi, D. Laganà, L.C. Coelho, and A. De Maio. *The Multi-depot Inventory Routing Problem: An Application of Vendor-management Inventory in City Logistic*. CIRRELT, 2017.

L. Bettencourt and G. West. A unified theory of urban living. *Nature*, 467:912, 2010.

K. Braekers, K. Ramaekers, and I. Van Nieuwenhuyse. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99: 300–313, 2016.

A.M. Campbell and J.H. Wilson. Forty years of periodic vehicle routing. *Networks*, 63:2–15, 2014.

R. Cantu-Funes, M.A. Salazar-Aguilar, and V. Boyer. Multi-depot periodic vehicle routing problem with due dates and time windows. *Journal of the Operational Research Society*, 69:296–306, 2018.

D. Cattaruzza, N. Absi, and D. Feillet. The multi-trip vehicle routing problem with time windows and release dates. *Transportation Science*, 50:676–693, 2016a.

D. Cattaruzza, N. Absi, and D. Feillet. Vehicle routing problems with multiple trips. *4OR*, 14:223–259, 2016b.

D. Cattaruzza, N. Absi, D. Feillet, and J. González-Feliu. Vehicle routing problems for city logistics. *EURO Journal on Transportation and Logistics*, 6:51–79, 2017.

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

J.C. Chu, S. Yan, and H.-J. Huang. A multi-trip split-delivery vehicle routing problem with time windows for inventory replenishment under stochastic travel times. *Networks and Spatial Economics*, 17:41–68, 2017.

L.C. Coelho and G. Laporte. Improved solutions for inventory-routing problems through valid inequalities and input ordering. *International Journal of Production Economics*, 155:391–397, 2014a.

L.C. Coelho and G. Laporte. Optimal joint replenishment, delivery and inventory management policies for perishable products. *Computers & Operations Research*, 47:42–52, 2014b.

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

L.C. Coelho, J.-F. Cordeau, and G. Laporte. Heuristics for dynamic and stochastic inventory-routing. *Computers & Operations Research*, 52:55–67, 2014.

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

Y. Crama, M. Rezaei, M. Savelsbergh, and T. Van Woensel. Stochastic inventory routing for perishable products. *Transportation Science*, 52:526–546, 2018.

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

M. Drexl. Synchronization in vehicle routinga survey of vrps with multiple synchronization constraints. *Transportation Science*, 46:297–316, 2012.

B. Eksioglu, A.V. Vural, and A. Reisman. The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, 57:1472–1483, 2009.

Y.Y. Fan, R.E. Kalaba, and J.E. Moore II. Arriving on time. *Journal of Optimization Theory and Applications*, 127:497–513, 2005.

M. Figliozzi and C. Tipagornwong. Impact of last mile parking availability on commercial vehicle costs and operations. In *Supply Chain Forum: An International Journal*, volume 18, pages 60–68. Taylor & Francis, 2017.

V. François, Y. Arda, Y. Crama, and G. Laporte. Large neighborhood search for multi-trip vehicle routing. *European Journal of Operational Research*, 255:422–441, 2016.

A. Froger, J.E. Mendoza, O. Jabali, and G. Laporte. *A Matheuristic for the Electric Vehicle Routing Problem with Capacitated Charging Stations*. PhD thesis, CIRRELT, 2017.

B. Gavish and S.C. Graves. The travelling salesman problem and related problems. Technical report, Massachusetts Institute of Technology, Operations Research Center; OR 078-78, 1978.

M. Gendreau, O. Jabali, and W. Rei. 50th anniversary invited articlefuture research directions in stochastic vehicle routing. *Transportation Science*, 50:1163–1173, 2016.

K. Helsgaun. An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126:106–130, 2000.

C. Hempsch and S. Irnich. Vehicle routing problems with inter-tour resource constraints. In *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 421–444. Springer, 2008.

G. Iassinovskaia, Sa. Limbourg, and F. Riane. The inventory-routing problem of returnable transport items with time windows and simultaneous pickup and delivery in closed-loop supply chains. *International Journal of Production Economics*, 183: 570–582, 2017.

Divya Jayakumar N., H. Grzybowska, D. Rey, and V. Dixit. Food rescue and delivery: Heuristic algorithm for periodic unpaired pickup and delivery vehicle routing problem. *Transportation Research Record: Journal of the Transportation Research Board*, 2548:81–89, 2016.

A.A. Juan, S.E. Grasman, J. Caceres-Cruz, and T. Bektaş. A simheuristic algorithm for the single-period stochastic inventory-routing problem with stock-outs. *Simulation Modelling Practice and Theory*, 46:40–52, 2014.

I. Kaparias, M.G.H. Bell, and H. Belzner. A new measure of travel time reliability for in-vehicle navigation systems. *Journal of Intelligent Transportation Systems*, 12: 202–211, 2008.

M.A. Klapp, A.L. Erera, and A. Toriello. The one-dimensional dynamic dispatch waves problem. *Transportation Science*, 52:402–415, 2016.

M.A. Klapp, A.L. Erera, and A. Toriello. The dynamic dispatch waves problem for same-day delivery. *European Journal of Operational Research*, 2018.

P. Lacomme, C. Prins, and W. Ramdane-Chérif. Evolutionary algorithms for periodic arc routing problems. *European Journal of Operational Research*, 165:535–553, 2005.

D. Laganà, F. Longo, and F. Santoro. Multi-product inventory-routing problem in the supermarket distribution industry. *International Journal of Food Engineering*, 11:747–766, 2015.

R. Lahyani, M. Khemakhem, and F. Semet. Rich vehicle routing problems: From a taxonomy to a definition. *European Journal of Operational Research*, 241:1–14, 2015.

K. Li, B. Chen, A.I. Sivakumar, and Y. Wu. An inventory–routing problem with the objective of travel time minimization. *European Journal of Operational Research*, 236:936–945, 2014.

X. Li, P. Tian, and S.C.H. Leung. Vehicle routing problems with time windows and stochastic travel and service times: Models and algorithm. *International Journal of Production Economics*, 125:137–145, 2010.

S. Lin and B.W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21:498–516, 1973.

S. Liu, S. Qin, and R. Zhang. A branch-and-price algorithm for the multi-trip multi-repairman problem with time windows. *Transportation Research Part E: Logistics and Transportation Review*, 116:25–41, 2018.

S.-C. Liu, M.-C. Lu, and C.-H. Chung. A hybrid heuristic method for the periodic inventory routing problem. *The International Journal of Advanced Manufacturing Technology*, 85:2345–2352, 2016.

H R. Lourenço, O C. Martin, and T. Stützle. Iterated local search: Framework and applications. In M. Gendreau and J.-Y. Potvin, editors, *Handbook of Metaheuristics*, pages 363–397. Springer US, Boston, MA, 2010.

C. Malandraki and M.S. Daskin. Time dependent vehicle routing problems: Formulations, properties and heuristic algorithms. *Transportation science*, 26:185–200, 1992.

M. Mes, M. Schutten, and A.P. Rivera. Inventory routing for dynamic waste collection. *Waste management*, 34:1564–1576, 2014.

M. Mirabi. A hybrid electromagnetism algorithm for multi-depot periodic vehicle routing problem. *The International Journal of Advanced Manufacturing Technology*, 71:509–518, 2014.

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

R. Montagné, M. Gamache, and M. Gendreau. A shortest path-based algorithm for the inventory routing problem of waste vegetable oil collection. *Journal of the Operational Research Society*, pages 1–12, 2018.

P.K. Nguyen, T.G. Crainic, and M. Toulouse. A hybrid generational genetic algorithm for the periodic vehicle routing problem with time windows. *Journal of Heuristics*, 20:383–416, 2014.

P.K. Nguyen, T.G. Crainic, and M. Toulouse. Multi-trip pickup and delivery problem with time windows and synchronization. *Annals of Operations Research*, 253:899–934, 2017.

F. Niakan and M. Rahimi. A multi-objective healthcare inventory routing problem; a fuzzy possibilistic approach. *Transportation Research Part E: Logistics and Transportation Review*, 80:74–94, 2015.

P.C. Nolz, N. Absi, and D. Feillet. A bi-objective inventory routing problem for sustainable waste management under uncertainty. *Journal of Multi-Criteria Decision Analysis*, 21:299–314, 2014a.

P.C. Nolz, N. Absi, and D. Feillet. A stochastic inventory routing problem for infectious medical waste collection. *Networks*, 63:82–95, 2014b.

N.M. Noor and A. Shuib. Multi-depot instances for inventory routing problem using clustering techniques. *Journal of Industrial and Intelligent Information*, 3, 2015.

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

M. Nourinejad, A. Wenneman, K.N. Habib, and M.J. Roorda. Truck parking in urban areas: Application of choice modelling within traffic microsimulation. *Transportation Research Part A: Policy and Practice*, 64:54–64, 2014.

T. Öncan, İ. K. Altınel, and G. Laporte. A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, 36: 637–654, 2009.

D.J. Papageorgiou, M.-S. Cheon, G. Nemhauser, and J. Sokol. Approximate dynamic programming for a class of long-horizon maritime inventory routing problems. *Transportation Science*, 49:870–885, 2014a.

D.J. Papageorgiou, A.B. Keha, G.L. Nemhauser, and J. Sokol. Two-stage decomposition algorithms for single product maritime inventory routing. *INFORMS Journal on Computing*, 26:825–847, 2014b.

Y.-B. Park, J.-S. Yoo, and H.-S. Park. A genetic algorithm for the vendor-managed inventory routing problem with lost sales. *Expert systems with applications*, 53: 149–159, 2016.

M.L. Pinedo. *Scheduling: theory, algorithms, and systems.* Springer, 2016.

L. Qin, L. Miao, Q. Ruan, and Y. Zhang. A local search method for periodic inventory routing problem. *Expert Systems with Applications*, 41:765–778, 2014.

M. Rahimi, A. Baboli, and Y. Rekik. Multi-objective inventory routing problem: A stochastic model to consider profit, service level and green criteria. *Transportation Research Part E: Logistics and Transportation Review*, 101:59–83, 2017.

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

G. Reinelt. TSPLIB - a traveling salesman problem library. *ORSA journal on computing*, 3:376–384, 1991.

D. Reyes, A.L. Erera, and M.W.P. Savelsbergh. Complexity of routing problems with release dates and deadlines. *European Journal of Operational Research*, 266:29–34, 2018.

J.-P. Riquelme-Rodríguez, A. Langevin, and M. Gamache. Adaptive large neighborhood search for the periodic capacitated arc routing problem with inventory constraints. *Networks*, 64:125–139, 2014.

U. Ritzinger, J. Puchinger, and R.F. Hartl. A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research*, 54:215–231, 2016.

M. Roca-Riu, E. Fernández, and M. Estrada. Parking slot assignment for urban distribution: Models and formulations. *Omega*, 57:157–175, 2015.

M. Roca-Riu, J. Cao, I. Dakic, and M. Menendez. Designing dynamic delivery parking spots in urban areas to reduce traffic disruptions. *Journal of Advanced Transportation*, 2017, 2017.

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

R.A. Russell and T.L. Urban. Vehicle routing with soft time windows and erlang travel times. *Journal of the Operational Research Society*, 59:1220–1228, 2008.

M. Savelsbergh and T. Van Woensel. 50th anniversary invited articlecity logistics: Challenges and opportunities. *Transportation Science*, 50:579–590, 2016.

H. Shaabani and I.N. Kamalabadi. An efficient population-based simulated annealing algorithm for the multi-product multi-retailer perishable inventory routing problem. *Computers & Industrial Engineering*, 99:189–201, 2016.

B.C. Shelbourne, M. Battarra, and C.N. Potts. The vehicle routing problem with release and due dates. *INFORMS Journal on Computing*, 29:705–723, 2017.

M.M. Solomon. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35:254–265, 1987.

M. Soysal, J.M. Bloemhof-Ruwaard, R. Haijema, and J.G.A.J. van der Vorst. Modeling a green inventory routing problem for perishable products with horizontal collaboration. *Computers & Operations Research*, 89:168–182, 2018.

D. Taş, N. Dellaert, T. Van Woensel, and T. De Kok. Vehicle routing problem with stochastic travel times including soft time windows and service costs. *Computers & Operations Research*, 40:214–224, 2013.

E.B. Tirkolaee, A. Goli, M. Bakhsi, and I. Mahdavi. A robust multi-trip vehicle routing problem of perishable products with intermediate depots and time windows. *Numerical Algebra, Control & Optimization*, 7:417–433, 2017.

E.B. Tirkolaee, M. Alinaghian, A.A.R. Hosseinabadi, M.B. Sasi, and A.K. Sangaiah.

An improved ant colony optimization for the multi-trip capacitated arc routing problem. *Computers & Electrical Engineering*, 2018.

P. Toth and D. Vigo. *Vehicle Routing: Problems, Methods, and Applications*, volume 18. SIAM, 2014.

M.W. Ulmer, B.W. Thomas, and D.C. Mattfeld. Preemptive depot returns for a dynamic same-day delivery problem. *Technical Report, Technische Universität Braunschweig*, pages 1–39, 2016.

M.W. Ulmer, B.W. Thomas, A.M. Campbell, and N. Woyak. The restaurant meal delivery problem: Dynamic pick-up and delivery with deadlines and random ready times. 2017.

R.G. Van Anholt, L.C. Coelho, G. Laporte, and I.F.A. Vis. An inventory-routing problem with pickups and deliveries arising in the replenishment of automated teller machines. *Transportation Science*, 50:1077–1091, 2016.

S.A. Voccia, A.M. Campbell, and B.W. Thomas. The same-day delivery problem for online purchases. *Transportation Science*, 2017. DOI https://doi.org/10.1287/trsc.2016.0732.

N. Wassan, N. Wassan, G. Nagy, and S. Salhi. The multiple trip vehicle routing problem with backhauls: formulation and a two-level variable neighbourhood search. *Computers & Operations Research*, 78:454–467, 2017.

E. Yadollahi, E.-H. Aghezzaf, and B. Raa. Managing inventory and service levels in a safety stock-based inventory routing system with stochastic retailer demands. *Applied Stochastic Models in Business and Industry*, 33:369–381, 2017.

H. Yahyaoui, I. Kaabachi, S. Krichen, and A. Dekdouk. Two metaheuristic approaches for solving the multi-compartment vehicle routing problem. *Operational Research*, pages 1–24, 2018.

Y. Zhang, Y. Mei, K. Tang, and K. Jiang. Memetic algorithm with route decomposing for periodic capacitated arc routing problem. *Applied Soft Computing*, 52:1130–1142, 2017.