The Dynamic Vehicle Routing Problem

Allan Larsen
Technical University of Denmark

The Dynamic Vehicle Routing Problem

Allan Larsen

LYNGBY 2001
IMM-PHD-2000-73

IMM

ISSN 0909-3192

Printed by IMM, DTU
Bookbinder Hans Meyer

Preface

This Ph.D. thesis entitled "*The Dynamic Vehicle Routing Problem*" has been prepared by Allan Larsen during the period January 1997 to June 2000 at the Department of Mathematical Modelling (IMM) at the Technical University of Denmark (DTU).

The thesis is submitted as a partial fulfillment of the requirement for obtaining the degree of doctor of philosophy (Ph.D.) at the Technical University of Denmark.

The work was fully supported by The Danish Transport Council (Transportrådet). The subject of the thesis is to investigate the dynamic version of the well-known vehicle routing problem in order to improve the performance of existing algorithms as well as to develop new algorithms. I would like to express my gratitude to Transportrådet for their belief and interest in this project.

Next, I would like to thank my thesis advisor, professor Oli B.G. Madsen, for his support and interest through this project. I have now served as Oli's master student, teaching/research assistant and lately as his Ph.D.-student for more than five years and I have enjoyed working with Oli from the very first day.

I would also like to thank professor Marius M. Solomon of Northeastern University in Boston, Massachusetts, for our collaboration on the technical report first presented at the Tristan III conference in Puerto Rico in 1998 and for providing good comments on and ideas for chapter 4 and 5.

The professors Richard C. Larson and Amadeo Odoni from Massachusetts Institute of Technology (M.I.T.) are thanked for their support and advice during my stay at the Operations Research Center at M.I.T. Also the students at the OR Center at M.I.T., the 52 Irving Street gang and the European Club at M.I.T. are thanked for making my stay in the US such a memorable experience.

I also wish to thank my old colleague Karsten Lund for his persistency and devotion while getting his old code tuned for my new data. Along the same line Rene Munk Jørgensen is thanked for writing the software for finding the coordinates of the real-life problem presented in the thesis. Søren Andersen and his colleagues at United Parcel Services are thanked for answering my endless number of questions as well as allowing me to use their data for a real-life data analysis.

Also, the staff - the academic as well as the administrative - at the Department of Mathematical Modelling (IMM) should be thanked for their friendly attitude. I have indeed enjoyed being a student and an employee at the Department from day one. Especially, my fellow members of the Operations Research Cake Group (ORCG) are thanked for providing more than the calories needed in order to perform the research of this thesis.

Lisbeth Mohr is thanked for her big effort in listing the numerous typos and bad structured sentences in the preliminary version of the thesis. Finally, I would like to thank my family, friends and last but certainly not least my girlfriend Camilla for all their love and support throughout the three years it took to prepare this thesis.

Lyngby, June 2000

Allan Larsen

Summary

During recent years distribution systems have become increasingly complex. This development is partly due to the high number of company mergers which leave distribution planners with ever bigger and complex problems. Another fact complicating distribution is the increased focus on timeliness in the distribution chains, as intelligent planning offers potential savings in capital bindings in costs related to stock and distribution. In other words time has become an extremely valuable resource. Nowadays most distribution systems must operate under strict temporal restrictions. This fact has caused an increasing interest in dynamic transportation models and systems in which data are considered to be time-dependent.

In this thesis the dynamic counterpart of the conventional vehicle routing problem will be studied. The traditional vehicle routing problem (VRP) consists of constructing minimum cost routes for the vehicles to follow so that the set of customers are visited exactly once. The VRP is an important subproblem in a wide range of distribution systems and a lot of effort has been devoted to research on various aspects of the VRP. However, the vast part of this research has been dealing with static environments, which in this sense means that the problem data are assumed to be static and not subject to change during the planning horizon. The increased focus on just-in-time logistics has together with the rapid development within telecommunications and computer hardware implied that the study of the far more complex dynamic versions of the VRP has received increasing interest from the scientic community as well as from the potential users of these methods.

The thesis begins by introducing the dynamic vehicle routing problem and discusses the differences between static and dynamic VRPs as well as provides some examples of real-life examples of DVRP. The existing literature dealing with the dynamic vehicle routing problem and related problems is reviewed in order to provide an overview of the richness of problems that have been investigated within this field.

The concept of measuring the dynamism within a dynamic vehicle routing problem is investigated and a framework for classifying dynamic routing applications according to their level of dynamism is proposed.

The Dynamic Traveling Repairman Problem (DTRP) proposed by Bertsimas and Van Ryzin is extended to embrace advance request customers as well as immediate request customers. Empirical analysis shows that the performance of the resulting problem has a linear relationship with the level of dynamism of the problem instances in question.

Next, the capacitated vehicle routing problem in the presence of time windows (DVRPTW) is examined under varying levels of dynamism. The performance of two simple batching strategies is examined in relation to the performance of a pure re-optimization strategy.

The A-priori Dynamic Traveling Salesman Problem with Time Windows (ADTSPTW) is introduced as an extension of the dynamic version of the well-known TSPTW. The extension consists in that a-priori information on future requests is included into the model.

Furthermore, a real-life instance of the dynamic vehicle routing problem originating from the pick-ups and deliveries of long-distance courier mail is examined using the algorithm proposed to solve the ADTSPTW.

Finally, the thesis discusses the present state of various DVRP methodologies and a number of different ideas for further research in this area are provided. The thesis concludes that more research on measures for the level of dynamism in a dynamic vehicle routing context is needed in order to provide more descriptive measures. Furthermore, the thesis concludes that using a-priori information in order to be able to reposition the vehicles expecting to receive new requests does not seem to offer significant performance improvements with respect to the lateness experienced by the customers.

Resumé (in Danish)

Distributionssystemer er i de senere år blevet stadigt mere komplekse. Denne udvikling er blandt andet forårsaget af det store antal virksomhedsfusioner, der medfører stadigt større og mere komplekse problemer for planlæggerne. En anden årsag hertil er den stigende interesse for det tidsmæssige aspekt i distributionskæden, idet god planlægning kan afstedkomme væsentlige besparelser i kapitalbindinger i forbindelse med lagring og distribution. Tid er med andre ord blevet en yderst vigtig ressource. Flertallet of distributionssystemer skal således i dag arbejde under stramme tidsmæssige restriktioner. Dette faktum har afstedkommet en forøget interesse for dynamiske transportmodeller og systemer, der arbejder med tidsafhængige data.

Denne afhandling omhandler det dynamiske modstykke til det konventionelle ruteplanlægningsproblem. Det traditionelle ruteplanlægningsproblem (VRP) består i at konstruere omkostningsminimale ruter for en flåde af køretøjer, således at mængden af kunder besøges netop én gang. Ruteplanlægningsproblemet er et vigtigt subproblem i en lang række distributionssystemer, og problemet er da også blevet genstand for stor interesse blandt forskere verden over. Hovedparten af forskningen har dog omhandlet statiske problemer, hvilket i denne forbindelse betyder, at problemets data forudsættes at være statiske og derfor ikke ændrer sig i løbet af planlægningshorisonten. Den forøgede fokus på just-in-time logistik har sammen med den rivende teknologiske udvikling inden for telekommunikation og computer hardware betydet en stigende interesse blandt såvel forskere som potentielle brugere af disse systemer for det langt mere komplicerede dynamiske ruteplanlægningsproblem.

I afhandlingens første kapitel introduceres det dynamiske ruteplanlægningsproblem, og forskellene mellem det statiske og dynamiske VRP bliver diskuteret. Endvidere gives der en række eksempler på praktiske dynamiske ruteplanlægningsproblemer. Dernæst gennemgås den eksisterende litteratur omhandlende det dynamiske ruteplanlægningsproblem og relaterede problemer, således at læseren er i stand til at danne sig et overblik over disse problemer. Udmåling af niveauet af dynamik er en blandt mange metoder til at klassificere og beskrive et dynamisk ruteplanlægningsproblem. Der gives en diskussion af hvorledes niveauet af dynamik kan udmåles. Dernæst introduceres et forslag til klassifikation af dynamiske ruteplanlægningsproblemer.

Den dynamiske rejsende reparatørs problem (DTRP), der blev introduceret af Bertsimas and Van Ryzin, udvides til at kunne omfatte statiske såvel som dynamiske kunder. Den efterfølgende empiriske analyse viser, at der eksisterer en lineær sammenhæng mellem den kørte distance og niveauet af dynamik for de undersøgte problemer.

Ruteplanlægningsproblemet med kapacitetsrestriktioner og tidsvinduer undersøges under varierende niveauer af dynamik. Endvidere introduceres to enkle batchingstrategier, og performance af disse sammenlignes med performance af en konventionel reoptimeringsstrategi.

Det dynamiske a-priori handelsrejsendes problem med tidsvinduer (ADTSPTW) introduceres som en udvidelse af den dynamiske version af det velkendte handelsrejsendes problem med tidsvinduer (TSPTW). Udvidelsen består i, at der i modellen inkluderes a-priori information om opkaldsintensiteterne af de enkelte delområder. Dernæst undersøges et virkeligt dynamisk ruteplanlægningsproblem omhandlende afhentning og udbringning af internationale kurérpostforsendelser. Problemet kan modelleres som et ADTSPTW og den udviklede metode til løsning af sådanne problemer testes på virkelige data.

Afhandligen afrundes med en kort diskussion af status for en række forskellige metodikker til løsning af dynamiske ruteplanlægningsproblemer. Der gives endvidere en række idéer til det videre arbejde inden for dynamisk ruteplanlægning. Afhandlingen konkluderer, at der er behov for mere forskning inden for udmåling af niveauet af dynamik således, at der kan opnås en bedre beskrivelse af niveauet af dynamik for konkrete probleminstanser. Endvidere konkluderes det på baggrund af afhandlingens empiriske analyser, at inkludering af a-priori information om opkaldsintensiteterne kun ser ud til at give meget små forbedringer i performance med hensyn til den totale forsinkelse oplevet af kunderne.

List of abbreviations

Abbreviations	Description	Definition (page)
ADTSPTW	The A-priori Dynamic Traveling Salesman Problem with Time Windows.	114
DARP	The Dial-a-Ride Problem.	37
DTRP	The Dynamic Traveling Repairman Problem.	30
DTSP	The Dynamic Traveling Salesman Problem.	28
DTSPTW	The Dynamic Traveling Salesman Problem with Time Windows.	108
DVRP	The Dynamic Vehicle Routing Problem	4
DVRPTW	The Dynamic Vehicle Routing Problem with Time Windows.	81
FCFS	First Come First Serve.	67
GIS	Graphical Information Systems.	14
GPS	Global Positioning System.	12
IP	Idle Point.	108
NN	Nearest Neighbor.	67
PART	The partitioning policy.	67
PDTRP	The Partially Dynamic Traveling Repairman Problem.	66
PTSP	The Probabilistic Traveling Salesman Problem.	22
PVRP	The Probabilistic Vehicle Routing Problem.	24
TSP	The Traveling Salesman Problem.	3
TSPTW	The Traveling Salesman Problem with Time Windows.	108
SQM	Stochastic Queue Median.	67
SVRP	The Stochastic Vehicle Routing Problem.	26
VRP	The Vehicle Routing Problem.	3
VRPTW	The Vehicle Routing Problem with Time Windows.	4

List of symbols

Symbol	Description	First occurence
b_i	The time the service begins at the i'th request.	84
dod	The degree of dynamism.	56
$edod$	The effective degree of dynamism.	58
$edod - tw$	The effective degree of dynamism for problems with time windows.	60
e_i	The earliest time for service to begin at the i'th request.	84
λ	The arrival rate in the Poisson process generating the immediate request customers.	44
l_i	The latest time for service to begin at the i'th request.	84
m	The number of vehicles.	83
n_{adv}	The number of advance requests (static customers).	58
n_{imm}	The number of immediate requests (dynamic customers).	58
n_{tot}	The total number of requests, i.e. $n_{tot} = n_{adv} + n_{imm}$	58
n_{subreg}	The number of subregions.	67
r_i	The reaction time of the i'th request, i.e. the time elapsed from the request was received to the time window ends.	59
$\overline{r}$	The average reaction time for a specific dataset.	93
t_i	The time the i'th request is recevied by the dispatcher.	58
$t_{i,j}$	The travel time between the i'th and the j'th request.	84
T	The latest time for which customers can call in to request service.	44

Contents

Chapter 1

Introduction

"The meaning of life is to move things -
correct me if I am not mistaken"
- anonymous.

The Vehicle Routing Problem (VRP) has been studied with much interest within the last three to four decades. The majority of these works focus on the static and deterministic cases of vehicle routing in which all information is known at the time of the planning of the routes. In most real-life applications though, stochastic and/or dynamic information occurs parallel to the routes being carried out. Real-life examples of stochastic and/or dynamic routing problems include the distribution of oil to private households, the pick-up of courier mail/packages and the dispatching of busses for the transportation of elderly and handicapped people. In these examples the customer profiles (i.e. the time to begin service, the geographic location, the actual demand etc.) may not be known at the time of the planning or even when service has begun for the advance request customers. Two distinct features make the planning of high quality routes in this environment much more difficult than in its deterministic counterpart; firstly, the constant change, secondly, the time horizon. A growing number of companies offer to service the customers within a few hours from the time the request is received. Naturally, such customer service oriented policies increase the dynamism of the system and therefore its complexity.

During the past decade the number of published papers dealing with dynamic transportation models has been growing. The dynamic vehicle routing problem is only a subset of these models. Psaraftis [37] examines the main issues in this area and provides a survey of the results found for various dynamic vehicle routing problems. Psaraftis mentions that only very few general results for some simple versions of the dynamic vehicle routing problem have been obtained. This fact may indicate that it is extremely difficult to obtain other general results for more advanced problems.

1.1 Outline of the Thesis

This thesis discusses various issues concerning the Dynamic Vehicle Routing Problem (DVRP). The presentation will be focusing on providing the reader with a theoretical basis for studying the DVRP as well as on providing the practioner with guidelines for implementing solution methods for solving the DVRP.

In this present chapter the conventional vehicle routing problem (VRP) and its dynamic counterpart - the DVRP - are introduced and we give a discussion of the differences between the conventional static VRP and the dynamic VRP. Next, the relevance of the DVRP and the technologies necessary are discussed. The chapter closes with a listing of a few real-life examples of the DVRP.

In chapter 2 we provide a survey of the existing literature dealing with the dynamic vehicle problem as well as related problems such as the stochastic vehicle routing problem (SVRP). The main goal of the chapter is to provide the reader with a consistent overview of the work on the DVRP and the progress made within this area throughout the past decades.

A substantial portion of the problem data in a DVRP are subject to stochasticity. In chapter 3 we discuss how to deal with this stochasticity when analyzing DVRP scenarios using simple simulation techniques.

Chapter 4 discusses how to measure the dynamism of a dynamic vehicle routing scenario. A framework based upon the degree of dynamism proposed by Lund et al. [29] is provided.

In chapter 5 the Partially Dynamic Traveling Repairman Problem (PDTRP) is introduced and the performance of a number of simple online dynamic policies is examined for various dynamic systems.

The dynamic version of the capacitated Vehicle Routing Problem with Time Windows (DVRPTW) is examined in chapter 6. The performance of a pure re-optimization strategy is compared to the performance of two simple strategies collecting the immediate requests into batches before the re-optimization is performed.

In chapter 7 the PDTRP problem of chapter 5 is extended to embrace time window constraints. Furthermore, the A-priori Dynamic Traveling Salesman Problem with Time Windows (ADTSPTW) is introduced.

A case-study of a real-life application of the Dynamic Traveling Salesman Problem with Time Windows (DTSPTW) is presented in chapter 8. The problem is solved using the ADTSPTW framework described in chapter 7.

In chapter 9 we suggest some ideas for further research in this field.

Finally, in chapter 10 we give our conclusions in a brief summary of the discussions of this thesis and list the scientific contributions of this thesis.

1.2 The Conventional Vehicle Routing Problem

The most fundamental and well-studied routing problem is without doubt the Traveling Salesman Problem (TSP) , in which a salesman is to visit a set of cities and return to the city he started in. The objective for the TSP is to minimize the total distance traveled by the salesman.

The Vehicle Routing Problem (VRP) is a generalization of the TSP in that the VRP consists in determining m vehicle routes, where a route is a tour that begins at the depot, visits a subset of the customers in a given order and returns to the depot. All customers must be visited exactly once and the total customer demand of a route must not exceed the vehicle capacity. The objective of the VRP is to minimize the overall distribution costs. In most real-life distribution contexts a number of side constraints complicate the model. These side constraints could for instance be time constraints on the total route time and time windows within which the service must begin.

The latter problem is referred to as the Vehicle Routing Problem with Time Windows (VRPTW) . Furthermore, having to deal with aspects such as multiple depots and commodities complicates the models further. Solution methods include exact methods such as mathematical programming, but custom designed heuristics and meta heuristics such as tabu search and simulated annealing have also been applied to the VRP.

Fisher [14] and Desrosiers et al. [11] provide excellent surveys on the VRP.

1.3 The Dynamic Vehicle Routing Problem

In this section the dynamic vehicle routing problem (DVRP) will be verbally defined and a simple example of a DVRP scenario will be presented.

Psaraftis [36] uses the following classification of the static routing problem; *"if the output of a certain formulation is a set of preplanned routes that are not re-optimized and are computed from inputs that do not evolve in real-time"*. While he refers to a problem as dynamic if *"the output is not a set of routes, but rather a policy that prescribes how the routes should evolve as a function of those inputs that evolve in real-time"*.

In the above definition by Psaraftis the temporal dimension plays a vital role for the categorizing of a vehicle routing problem. As will be demonstrated throughout this thesis the time of when relevant information is made known to the planner distinguishes a dynamic from a static vehicle routing problem.

In definition 1.1 we verbally define what we mean when we talk about a static vehicle routing problem.

Definition 1.1 The Static Vehicle Routing Problem.

1. *All information relevant to the planning of the routes is assumed to be known by the planner before the routing process begins.*

2. *Information relevant to the routing does not change after the routes have been constructed.*

The information which is assumed to be relevant, includes all attributes of the customers such as the geographical location of the customers, the on-site service time and the demand of each customer. Furthermore, system information as for example the travel times of the vehicle between the customers must be known by the planner.

The dynamic counterpart of the static vehicle routing problem as defined in definition 1.1 could then be formulated as:

Definition 1.2 The Dynamic Vehicle Routing Problem.

1. *Not all information relevant to the planning of the routes is known by the planner when the routing process begins.*
2. *Information can change after the initial routes have been constructed.*

Obviously, the DVRP is a richer problem compared to the conventional static VRP. If the problem class of VRP is denoted $\mathcal{P}(\mathcal{VRP})$ and the problem class of DVRP is denoted $\mathcal{P}(\mathcal{DVRP})$, then $\mathcal{P}(\mathcal{VRP}) \subset \mathcal{P}(\mathcal{DVRP})$.

The dynamic vehicle routing problem calls for online algorithms that work in real-time since the immediate requests should be served, if possible. As conventional static vehicle routing problems are $NP-hard$, it is not always possible to find optimal solutions to problems of realistic sizes in a reasonable amount of computation time. This implies that the dynamic vehicle routing problem also belongs to the class of $NP-hard$ problems, since a static VRP should be solved each time a new immediate request is received.

Psaraftis [35] refers to the solution x_t of the current problem p_t as a *tentative* solution. The tentative solution corresponds to the current set of inputs and only those. If no new requests for service are received during the execution of this solution, the tentative route is said to be optimal.

In Figure 1.1 a simple example of a dynamic vehicle routing situation is shown. In the example, two un-capacitated vehicles must service both advance and immediate request customers without time windows. The advance request customers are represented by black nodes, while those that are immediate requests are depicted by white nodes. The solid lines represent the two routes the dispatcher has planned prior to the vehicles leaving the depot. The two thick arcs indicate the vehicle positions at the time the dynamic requests are received. Ideally, the new customers should be inserted into the already planned routes without the order of the non-visited customers being changed and with minimal delay. This is the case depicted on the right hand side route. However, in practice, the insertion of new customers will usually be a much more complicated task and will imply a re-planning of the non-visited part of the route system. This is illustrated by the left hand side route where servicing the new customer creates a large detour.

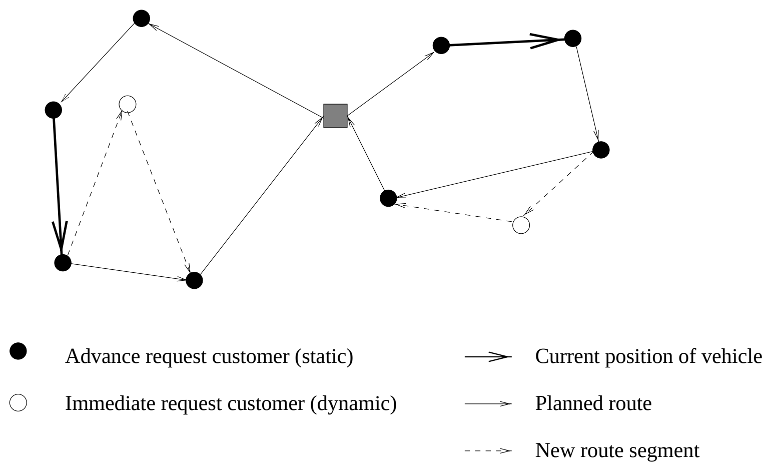

Advance request customer (static)
Immediate request customer (dynamic)
Current position of vehicle
Planned route
New route segment

Figure 1.1: A dynamic vehicle routing scenario with 8 advance and 2 immediate request customers.

Generally, the more restricted and complex the routing problem is, the more complicated the insertion of new dynamic customers will be. For instance, the insertion of new customers in a time window constrained routing problem will usually be much more difficult than in a non-time constrained problem. Note that in an online routing system customers may even be denied service, if it is not possible to find a feasible spot to insert them. Often this policy of rejecting customers includes an offer to serve the customers the following day of operation. However, in some systems - as for instance the pick-up of long-distance courier mail - the service provider (distributor) will have to forward the customer to a competitor when they are not able to serve them.

When investigating dynamic vehicle routing systems, randomly generated data rather than real-life data are often used. There are two main reasons for this. Firstly, the use of randomly generated data often enables more in-depth analyses, since the datasets can be constructed in such a way that other issues could be addressed. Secondly, the vast majority of real-life dynamic vehicle routing problems do not at the present capture all data needed for in-depth analyses of the specific routing problem. The geographical locations of all vehicles each time new immediate requests are received are one of the most commonly missing data items in the study of real-life problems.

If one chooses to use randomly generated data, several highly relevant issues need to be addressed. Below we briefly discuss three of these issues bearing in mind that several other issues must be considered.

First, the method used for generating the arrival times of the immediate request customers must be considered. This is an important issue, since system designers use such information to develop solution methods. Traditionally, a Poisson process is assumed for this purpose. The arrival rate parameter, generally denoted λ, then describes the "congestion" of the system. To emulate scenarios in more complex systems, the arrival process could be a mix of several different stochastic processes.

Another important modeling issue is the distributions that characterize the demand of the customers and the on-site service times at the customer locations. Uniform and Gaussian distributions as well as constant values have been employed. The latter is convenient, but also unrealistic in most cases. As an example, one could think of the pick-up of long-distance courier mail. In this context, the on-site service time at each customer will often vary greatly depending on a wide range of factors, as for instance the parking facilities, the physical layout of the buildings to be served and the efficiency of the pick-up itself (does the courier have to wait or is the customer set to hand over the mail?).

The last aspect to be mentioned in this introductory stage is the system reaction time. This is defined as the elapsed time between the receipt of the request and the end of the time window defining the latest feasible time for service to begin. The reaction time of each customer could be seen as a measure for the urgency in serving this customer. The overall reaction time of a system is strongly related to the level of dynamism present in the system. Generally, the more dynamic a system is, the more difficult or costly it is to generate a fast reaction. Therefore, the dynamic make-up of a system plays a major role in choosing appropriate models and algorithms to support decisions in different scenarios.

1.4 Static versus Dynamic Vehicle Routing

In this section the differences between the conventional static and the dynamic vehicle routing problem as described in the sections above will be discussed.

Psaraftis [36], [37] lists 12 issues on which the dynamic vehicle routing problem differs from the conventional static routing problem. Below we give a brief summary of these issues as they are indeed very central to our discussion of static versus dynamic routing.

1. **Time dimension is essential.**
   In a static routing problem the time dimension may or may not be important. In the dynamic counterpart time is always essential. The dispatcher must as a minimum know the position of all vehicles at any given point in time and particularly when the request for service or other information is received by the dispatcher.[1]

2. **The problem may be open-ended.**
   The process is often temporally bounded in a static problem. The routes start and end at the depot. In a dynamic setting the process may very well be unbounded. Instead of routes one considers paths for the vehicles to follow.

3. **Future information may be imprecise or unknown.**
   In a static problem all information is assumed to be known and of the same quality. In a real-life dynamic routing problem the future is almost never known with certainty. At best probabilistic information about the future may be known.

[1]In many real-life routing problems the dispatcher does not know the exact position of the vehicle at any given point in time, since online communication with all vehicles can be quite costly (see section 1.6.1 for a discussion of this issue).

4. **Near-term events are more important.**
   Due to the uniformity of the information quality and lack of input updates all events carry the same weight in a static routing problem. Whereas in a dynamic setting it would be unwise immediately to commit vehicle resources to long-term requirements. The focus of the dispatcher should therefore be on near-term events when dealing with a dynamic routing problem.

5. **Information update mechanisms are essential.**
   Almost all inputs to a dynamic routing problem are subject to changes during the day of operation. It is therefore essential that information update mechanisms are integrated into the solution method. Naturally, information update mechanisms are not relevant within a static context.

6. **Re-sequencing and reassigning decisions may be warranted.**
   In dynamic routing new input may imply that decisions taken by the dispatcher become suboptimal. This forces the dispatcher to reroute or even reassign vehicles in order to respond to the new situation.

7. **Faster computation times are necessary.**
   In static settings the dispatcher may afford the luxury of waiting for a few hours in order to get a high quality solution, in some cases even an optimal one. In dynamic settings this is not possible, because the dispatcher wishes to know the solution to the current problem as soon as possible (preferably within minutes or seconds). The *running-time* constraint implies that rerouting and reassignments are often done by using local improvement heuristics like insertion and k-interchange.

8. **Indefinite deferment mechanisms are essential.**
   Indefinite deferment means the eventuality that the service of a particular demand be postponed indefinitely because of that demands unfavorable geographical characteristics relative to the other demands. This problem could for instance be alleviated by using time window constraints or by using a nonlinear objective function penalizing excessive wait.

9. **Objective function may be different.**
   Traditional static objectives such as minimization of the total distance traveled or the overall duration of the schedule might be meaningless

in a dynamic setting because the process may be open-ended. If no information about the future inputs is available, it might be reasonable to optimize only over known inputs. Some systems also use nonlinear objective functions in order to avoid undesirable phenomena such as the above mentioned indefinite deferment.

10. **Time constraints may be different.**
    Time constraints such as latest pickup times tend to be softer in a dynamic routing problem than in a static one. This is due to the fact that denying service to an immediate demand, if the time constraint is not met, is usually less attractive than violating the time constraint.

11. **Flexibility to vary vehicle fleet size is lower.**
    In static settings the time gap between the execution of the algorithm and the execution of the routes usually allows adjustments of the vehicle fleet. However, within a dynamic setting the dispatcher may not have instant access to backup vehicles. Implications of this may mean that some customers receive lower quality of service.

12. **Queueing considerations may become important.**
    If the rate of customer demand exceeds a certain threshold, the system will become congested and the algorithms are bound to produce meaningless results. Although vehicle routing and queueing theory are two very well-studied disciplines, the effort to combine these has been scant.

Psaraftis [37] also proposes a taxonomy used for characterizing attributes of the information forming the input for the vehicle routing problem. The taxonomy consists of the following concepts:

- **Evolution of information.** In static settings the information does not change, nor is the information updated. In dynamic settings the information will generally be revealed as time goes on.

- **Quality of information.** Inputs could either; 1) be known with certainty (deterministic), 2) be known with uncertainty (forecasts) or 3) follow prescribed probability distributions (probabilistic). Usually, the quality of the information in a dynamic setting is good for near-term events and poorer for distant events.

- **Availability of information.** Information could either be local or global. One example of local information is when the driver learns of the precise amount of oil the current customer needs, while a globally based information system would be able to inform the dispatcher of the current status of all the customers' oil tanks. The rapid advances within information technologies increase the availability of information. This fast growth in the amount of information available raises the issue of when to reveal/make use of the information. For instance, the dispatcher may choose to reveal only the information that is needed by the drivers although she might have access to all information.

- **Processing of information.** In a *centralized* system all information is collected and processed by a central unit. In a *decentralized* system some of the information could for instance be processed by the driver of each truck.

Powell et al. [34] distinguish between dynamism within a problem, a model and the application of a model. They argue that:

- A problem is dynamic if one or more of its parameters are a function of time. This includes models with dynamic data that change constantly as well as problems with time-dependent data which are known in advance.

- A model is dynamic if it explicitly incorporates the interaction of activities over time. Here one should distinguish between deterministic dynamic models and stochastic models.

- An application is dynamic if the underlying model is solved repeatedly as new information is received. Consequently, solving models within dynamic applications require huge computational resources.

1.5 Relevance of the DVRP's

Along with the recent years' increased focus on just-in-time logistics and the advances within telecommunications and computer hardware the importance of being able to effectively make use of the huge amount of online information made available by these new technologies has become extremely important for a wide range of distribution oriented companies throughout the world. Cost efficient routing of vehicles play an important role to a wide range of industries. As distribution costs amount to approximately 10-15 % of a nation's gross national product even a relatively modest improvement in the distribution costs can be extremely important for the industries.

It could also be argued that a relatively high number of the routing problems being modeled as static problems do in fact include dynamic elements in a real-life situation. Examples of this phenomenon is the fact that on-site service times as well as travel times despite extensive empirical data are often filled with noise. This implies that the preplanned routes collapse because of the new temporal conditions of the system. I.e. what seemed to be an optimal solution (or at least a good quality solution) might turn out to be a sub-optimal solution.

1.6 Technical Requirements

In this section we provide a brief introduction to some of the most essential technologies when dealing with real-life applications of vehicle routing problems within a dynamic environment.

1.6.1 Communication and Positioning Equipment

The communication between the drivers of the vehicles and the dispatching center is essential in order to feed the most up-to-date information into the routing system. The equipment for determining the current position of the vehicles and the communication equipment for passing information on between the dispatching center and the drivers in the vehicles will be introduced below.

- Naturally, *positioning equipment* like the GPS (Global Positioning System) is essential to a dynamic vehicle routing system. The GPS is a constellation of 24 satellites orbiting Earth that constantly send out signals giving their positions and time. Signals from three or four different satellites at any given time can provide receivers on the ground with enough information to calculate their precise location within a few meters depending on which version of the GPS system is used. For further reading on the GPS system the text book by Collins [10] should be consulted. The prices of a GPS receiver has dropped from several thousand dollars few years ago to as low as 100 dollars for the most basic models used for hiking trips in remote areas etc.

- The *communication equipment* between the vehicle and the dispatching center is essential for the structure of the routing system. Mobile telephone communication systems are one example of a technology capable of providing this information. Another technology is a dedicated radio based communications system. The main difference in these technologies is the differences in initial and operating costs. A mobile telephone communications system is relatively costly to operate, but has low initial costs because the basic technology is provided by the telephone companies and the GSM system today offers almost full coverage in most western industrialized countries. In Denmark for example, sending an SMS (Short Message System) costs 0.50 dkr. (approximately 0.07 $). This means that the annual mobile telephone communications costs amount to approximately 30000 dkr. per vehicle if the positions have to be sent every second minute during an 8-hour operation day. The initial costs in implementing a radio based communications system are on the other hand very high, because transmission masts will have to be put up and relatively expensive radio equipment must be installed in every vehicle. In all, a radio based communication system has very high initial costs, while the operating costs are almost negligible. Furthermore, the radio based system does not offer the same flexibility compared to the mobile telephone communications system.

In Figure 1.2 the basic information flows between the vehicle and the dispatching center are shown.

Ideally, the dispatching center will know in which state the vehicle and the driver are at any given point in time. However, as the above description indicates, this may prove to be infeasible for some applications due to the operating costs of this method. However, within a real-life setting the positioning information is transmitted at fixed intervals and an interpolation scheme is employed in order to estimate the positions of the vehicles.

Alternatively, the driver sends a message about his current status and position to the dispatch center, each time he finishes the service at a customer.

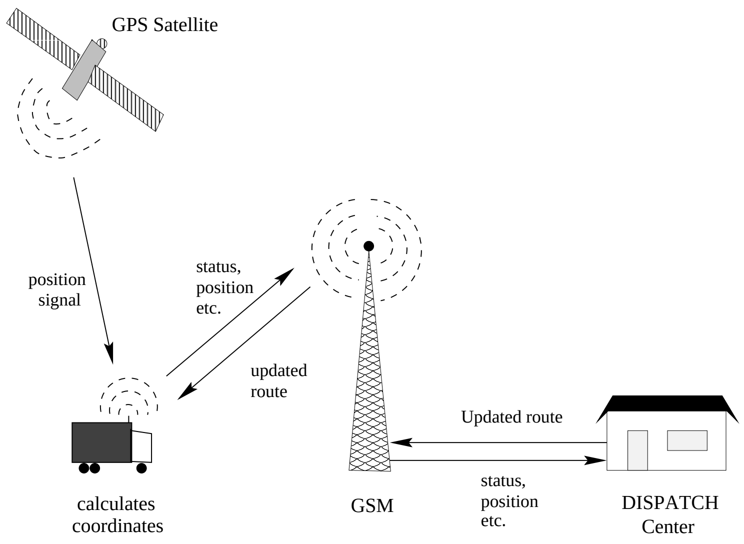


Figure 1.2: Sketch of the information flow in a GPS based vehicle routing system.

Obviously, this approach does not offer the same level of information for the dispatcher to support her decision as to which vehicle to dispatch to the next customer to be served. If the new information provided by the now idle driver/vehicle makes the dispatcher change her mind on the current planned route, she will have to call the other drivers manually to inform them about the changes in the current routes.

However, the conclusion of this discussion must be that a careful analysis will have to show which approach to choose when designing the system. Of course, history shows that the prices of communication decrease rapidly over the years. This could be the motivation to go for a telecommunications based system.

1.6.2 Geographical Information Systems

The advances within digital road maps and geographical information systems (GIS) have also been considerable through the 90's. Most western industrialized countries now have almost fully detailed road network databases. In Denmark, DAV (Dansk Adresse og Vejdatabase - Danish Road and Address Database) offers a digital road database which is connected to detailed information on every address in the country. The DAV database includes information on the zip codes, official street names, route numbers, road classification, highest and lowest street number on both sides of the streets. However, for the time being DAV still needs to include restrictions on turning and information on one-way streets. The price of a commercial single user license of DAV covering all areas of Denmark is approximately 40.000 dkr.

Naturally, in a real-life application it is vital that the solution algorithm chosen is capable of processing large amounts of geographical information fast enough to solve the problem online. However, issues related to computation of the shortest paths in a road network etc. will not be treated in this present thesis, although these issues become extremely important when implementing end user online routing applications.

1.7 Real-life Examples of DVRP's

In this section we list a number of real-life applications of dynamic vehicle routing problems.

1.7.1 The Traveling Repairman

Consider the situation that arises when for instance a bank teller machine breakes down and must be repaired by a service technician. The route to be followed by the technician may be determined using a distance based objective or it might take the urgency of the call (is the teller-machine located in a high intensity area or is it located in a remote area?) into consideration. This problem is often referred to as the "Dynamic Traveling Repairman Problem" (DTRP) and is one of the most well-studied dynamic vehicle routing problems. The DTRP will be described in detail in section 2.3.2. A similar example is the repairman from the electric power company traveling from house to house to repair sudden break-downs in the electric power supply. Recently, Weintraub et al. [43] published a paper dealing with this problem.

1.7.2 Courier Mail Services

Courier mail service companies throughout the world offer to pick-up mail and/or packages at one location and deliver the goods safely at another location within a certain time limit. Often, the mail/packages to be delivered are not local, but shipped from other cities or countries. Hence, the deliveries are shipped to a hub and then distributed from this hub to the delivery trucks. The deliveries form a static routing problem, because all recipients are known by the driver (and the dispatcher) before the vehicle leaves the depot. However, the pick-ups to be handled during the deliveries has the effect that the problem become dynamic in the sense that the driver and the dispatcher do not have all information on when and where the pick-ups are going to take place. The routing and dispatching issues in dynamic pick-up and delivery of courier mail and packages will be investigated in further detail in chapter 8 of the present thesis.

1.7.3 Distribution of Heating Oil

The distribution of heating oil to private households is often based on the so-called *degree-days* which is a simple measure of the accumulated outdoor temperature. The oil companies use the degree-day measure to keep track of how much oil the customers have used for heating their houses. Whenever the database tells the routing system that a customer is running low on heating oil, the customer is included in the pool of customers waiting to be served. Eventhough the degree-day customers could be thought of as static customers, the routing problem often becomes dynamic due to the fact that a subset of the customers might run out of oil before the degree-day database includes the customers in the replenishing list. Reasons for this situation could for instance be that a sudden change in the weather causes a high use of oil just before replenishment was to take place or a general higher use of oil due to changes in the behavior of the customer (for example when the house owner invites people to stay and therefore has to heat rooms which are normally left unheated). Experiences show that approximately 20 % of the customers visited by the oil company in a day are dynamic customers calling in during the day requesting immediate service. Another element which makes this problem different from - and much more difficult to solve than - the conventional static and deterministic routing problems is the fact that the degree-day measure is not a precise measure for the actual use of heating oil, but merely an estimate. This implies that the problem becomes stochastic in relation to the demand.

1.7.4 Dynamic Dial-A-Ride Systems

Dial-a-ride transportation systems are one application of the general pick-up and delivery vehicle routing problem, in which one or more types of commodities must be picked-up at one location and brought to another location where the goods are delivered. One example of a dynamic dial-a-ride system is the transportation of elderly and handicapped people. In Copenhagen, Denmark, the local urban bus companies also provide a service for elderly and handicapped people. At present time customers are supposed to call in for service one day before the requested trip is going to take place. This policy of course makes the system static, but in the future the bus company might offer the service as an online service for instance via a world wide web based booking system. In section 2.3.3 we will give a brief summary of the work on the dynamic version of the general pick-up and delivery VRP and the dynamic dial-a-ride problem.

1.7.5 Taxi Cab Services

Managing taxi cabs is yet another example of a real-life dynamic routing problem. In most taxi cab systems the percentage of dynamic customers is very high, i.e., only very few customers are known to the planner before the taxi cab leaves the taxi central at the beginning of its duty. A special attribute of the taxi cab routing and dispatching problem is that the state of the taxi can either be for hire or it can be engaged by one or more passengers. When the taxi is free for hire, the driver often repositions the vehicle to a centrally located taxi rank where the probability of being hailed is higher than it would have been if the driver had chosen to wait at the destination of the last customer. The location of the taxi ranks could either be based on extensive empirical data of calls or it could simply be based upon the intuition and experience of the driver. The policies for assigning customers to taxis differ from country to country and from company to company. The larger taxi cab companies in Denmark are owned by relatively few contractors, each of whom might have between one and 100 taxis. The contractors share a central call and dispatch center. The customers are then assigned to the taxi according to the number of taxi cabs owned by each contractor. This implies that the taxi cab routing and dispatching system will have to use a load balancing strategy when assigning the customers to taxis. This policy means that the contractors could be sure that they will get the best service from the company.

1.7.6 Emergency Services

The dispatching of emergency services (police, fire and ambulance services) resembles the dynamic vehicle routing problem through the fact that requests for service arrive in real-time and that the system resembles a geographical based queueing system. In most situations though, routes are not formed, because the requests are usually served before a new request appears. The problem then is to assign the best vehicle (for instance the nearest) to the new request. Methods for designing emergency service dispatch are therefore often based on location analysis for deciding where to locate the vehicles and crews .This area has been studied mostly from queueing oriented approaches. Recently, Gendreau et al. [19] published a paper dealing with ambulance location in the city of Montreal, Canada. For further reading in the area of dispatching emergency services the text book [28] written by Larson and Odoni should be consulted.

Chapter 2

Literature

In this chapter the existing literature covering the dynamic vehicle routing problem and related problems will be investigated. During the past 15-20 years the number of published papers dealing with stochastic and/or dynamic vehicle routing problems has been growing. Generally, the literature on dynamic vehicle routing problems covers a wide range of different applications and methodological approaches. The routing policies for auto-guided vehicles within a manufacturing context are one example of a problem related to the problem we will be examining in the present chapter. Naturally, this chapter cannot cover all aspects of vehicle routing problems with dynamic or stochastic elements. The goal of this chapter is rather to provide a brief introduction to the literature on these subjects. Also, this chapter intends to provide the reader with an overview of the methodological approaches to these problems. A number of different papers using various techniques have been chosen to be briefly discussed as they are estimated to be of specific importance to the present thesis. The study of the literature ended October 1st 1999 and work published after this date will not be treated in this thesis. This chapter is organized as follows. First, we give a brief discussion on some of the most interesting survey papers. In section 2.2 we discuss the different problem types using an a-priori based optimization approach. In section 2.3 we turn to problems using real-time optimization. In section 2.5 we provide a chronological overview of some of the most important works on vehicle routing problems dealing with stochastic and/or dynamic elements. Finally, in section 2.6

we give an assessment of the literature listed in this chapter.

2.1 Survey Papers

During the past decade a number of survey papers have appeared in various journals and books dealing with dynamic and/or stochastic vehicle routing problems. These papers provide the reader with a broad introduction to this subject and we will therefore list some of the most interesting among these papers before turning to the more in-depth literature in these areas.

Psaraftis [36] and [37] was among the first to study dynamic versions of the VRP. As mentioned in the previous chapter he outlined the status and prospects for future research within dynamic vehicle routing problems in these papers.

Powell et al. [34] concentrate on stochastic programming based models, but do also provide an excellent survey on various dynamic vehicle routing problems such as the dynamic traffic assignment problem which consists in finding the optimal routing of some goods from origin to the destination through a network of links which for instance could have time-dependent capacities. The authors also discuss how to evaluate the solutions since it is an important issue that distinguishes static from dynamic models. They note that in static models finding an appropriate objective function is fairly easy and that the objective function is usually a good measure for evaluating the solution. Whereas for dynamic models the objective function used to find the solution over a rolling horizon often has little to do with the measures developed to evaluate the overall quality of a solution.

Bertsimas and Simchi-Levi [5] provide a survey of deterministic and static as well as dynamic and stochastic vehicle routing problems for which they examine the robustness and the asymptotic behavior of the known algorithms. Bertsimas and Simchi-Levi argue that analytical analysis of the vehicle routing problem offers new insights into the algorithmic structure and it makes performance analysis of classical algorithms possible and it leads to a better understanding of models that integrate vehicle routing with other issues like inventory control. The authors conclude that a-priori optimization is an attractive policy if intensive computational power is not present. Furthermore, they point out that dealing with stochasticity in the VRP provides insights that can be useful when constructing practical algorithms for the VRP within a dynamic and stochastic environment.

Gendreau and Potvin [20] is the most recent survey on the DVRP. They note that the work on local area vehicle routing and dispatching still leaves a number of questions to be answered. Especially, research on demand forecasting used for constructing routes with look-ahead is needed in the future. Furthermore, the authors point out that it is relevant to consider several sources of uncertainty like cancellation of requests and service delays rather than just to focus on uncertainty in the time-space occurrence of service requests. Gendreau and Potvin also note that the issue of diversion deserves more attention. Due to the large amount of online information it has become possible to redirect the vehicle while on-route to the new customer. Finally, the authors advocate further research on parallel implementations and worst-case analysis in order to be able to assess the loss in not having full information available at the time of planning.

2.2 A-priori Optimization Based Methods

When dealing with optimization problems, which include uncertain elements, several approaches can be chosen. One way of dealing with problems like these is to determine an a-priori solution. By an a-priori solution we mean that the solution is based on probabilistic information on future events. Within the vehicle routing context a-priori based solutions mean that the planner determines one or more routes based on probabilistic information on future requests for service, customers demands, travel times etc.

Bertsimas et al. [4] argue that using re-optimization based methods for solving vehicle routing problems with random demands might not be without difficulties. They mention that even if the right amount of computer resources are available, it might be too time-consuming to solve the problem every time new information is received. Furthermore, redesigning the routes might create confusion for drivers and company policies concerning having the same driver service the same customers every day might be spoiled by redesigning the routes. These disadvantages do not apply when an a-priori strategy for designing the routes is chosen.

The Probabilistic Traveling Salesman Problem (PTSP) and the Probabilistic Vehicle Routing Problem (PVRP) as well as the Stochastic Vehicle Routing Problem (SVRP) are examples of problems solved by using a-priori optimization based methods. In the following sections these problems will be discussed.

2.2.1 The Probabilistic TSP

The Probabilistic Traveling Salesman Problem (PTSP) was first introduced by Jaillet [22] and [23]. The PTSP is defined on a graph $G = (N, A)$, where N is the set of nodes and A is the set of arcs. Each node is present with probability p_i. Nodes present with a probability of 1 are referred to as *black nodes*, whereas all other nodes are denoted *white nodes.* A distance (cost), c_{ij}, is associated to all of the arcs in E. Solving the problem consists of finding a tour of minimum expected length. When the tour is about to be carried out, it is revealed whether each white node is present or not. The nodes that are not present can be skipped and the salesman can travel directly to the next node requiring service. A simple example of this is shown in Figure 2.1.

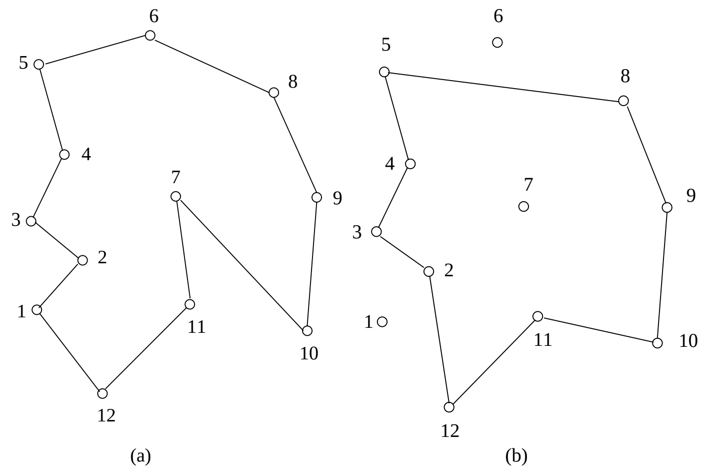


Figure 2.1: (a) The a-priori tour. (b) The actual tour when nodes 1, 6 and 7 do not need to be visited.

The applications of the PTSP may be in delivery contexts where a set of customers have to be serviced on a daily basis, but all customers do not require service every day. Bartholdi et al. [1] describe how they implemented a minimal technology system for routing vehicles delivering hot meals to elderly people. The list of clients changes at a rate of about 14 % each month. The "meals on wheels" programme is managed by a single full-time employee who is responsible for managing the budget, planning menus, monitoring quality and supervising part-time staff besides managing the delivery of the meals. In other words the manager is overworked and managing a computer-based routing system would only make things worse. Therefore, the authors decided to go for a minimal technology system (meaning no computers). The routing system is based on a traveling salesman heuristic, which is extremely simple, but does provide high quality tours on the average. The main idea is to use a *"space filling curve"* which should be imagined to be an infinitesimal thin curve which visits all points within a unit square. The relative position, Θ, of each client's location along the space filling curve is calculated. The route is then found when sorting the Θ values. The underlying structure of the meals on wheels problem resembles the PTSP, as all locations within the space filling curve are considered possible client locations. The new system improved the travel times by 13 % and the distributor was able to consolidate five routes into four. Another example of a PTSP application is the route of a post delivery man. The set of customers is fixed, but not all households receive mail every day and some can therefore be skipped by the post man.

Jaillet [22] formulates the PTSP as an integer nonlinear programming model and transforms this formulation first into a mixed integer linear programming problem and finally into an integer linear program. The final solution is found by using a branch-and-bound scheme similar to the methods used for the traditional TSP. Jaillet also shows that although dynamic programming (DP) might seem a natural choice for solving the PTSP exactly, it turns out that DP cannot provide an exact solution. Jaillet presents a number of heuristics inspired by *tour construction* and by *tour improvement* procedures known from the traditional TSP. Among these is the *"Supersavings Algorithm"* based on the Clarke-Wright [9] *"savings"* approach in which the central idea is to look at the savings in the expected length. Another tour construction heuristic is the *"Almost Nearest Neighbor Algorithm"* which similarly works on the increase in expected length. With respect to the tour improvement heuristics Jaillet mentions that a modified version of the l-opt method can be used to solve the PTSP. Finally, Jaillet discusses partitioning algorithms in which the main idea is to partition the service region into a set of smaller subregions and then solve the resulting PTSP in these subregions. Jaillet concludes that although exact methods seem a little too ambitious for the general PTSP, the proposed mathematical programming based method could be appropriate for solving reasonable-sized problems which have p_i's close to 1 or a dominant number of black nodes. Furthermore, he concludes that some of the tour construction procedures seem promising and if low computation times are important, partitioning algorithms should be considered. Jezequel [24] defines a *good PTSP tour* in the Euclidean case to be a compromise between a tour of short length and a tour including zigzag patterns. Jezequel shows that the performance of the heuristics proposed by Jaillet generally is very poor. Jezequel shows that a manual solution procedure was very efficient for coverage probabilities greater than or equal to 0.5. For instances with coverage probabilities less than 0.5 a new heuristic based on the savings idea produces very good results. Bertsimas and Howell [3] sharpen the best known bounds for the PTSP, derive several asymptotic results for various heuristics and the PTSP with the re-optimization strategy. Bertsimas and Howell also mention that one motivation for examining probabilistic versions of well-known deterministic models is to test the robustness (with respect to optimality) of the known methods within a stochastic environment. Laporte et al. [26] formulate the PTSP as an integer linear program and solve it by using a branch-and-cut approach. The authors solve problems ranging from 10 to 50 nodes with varying proportions of black and white nodes. The computation times range between less than a second and up to several hundreds of seconds.

2.2.2 The Probabilistic VRP

Bertsimas et al. [4] describe the Probabilistic Vehicle Routing Problem (PVRP) as a standard VRP, but with demands which are probabilistic in nature rather than deterministic. The PVRP is an extremely difficult problem to solve. Bertsimas [2] provides a recursive expression for finding the objective value which is also a very hard problem. Bertsimas also provide bounds and asymptotic analysis and several re-optimization policies for the PVRP.

Bertsimas et al. [4] propose two different strategies for serving the customers. Under *Strategy A* the vehicle visits all the customers in the same fixed order as under the a-priori sequence. However, the vehicle serves only

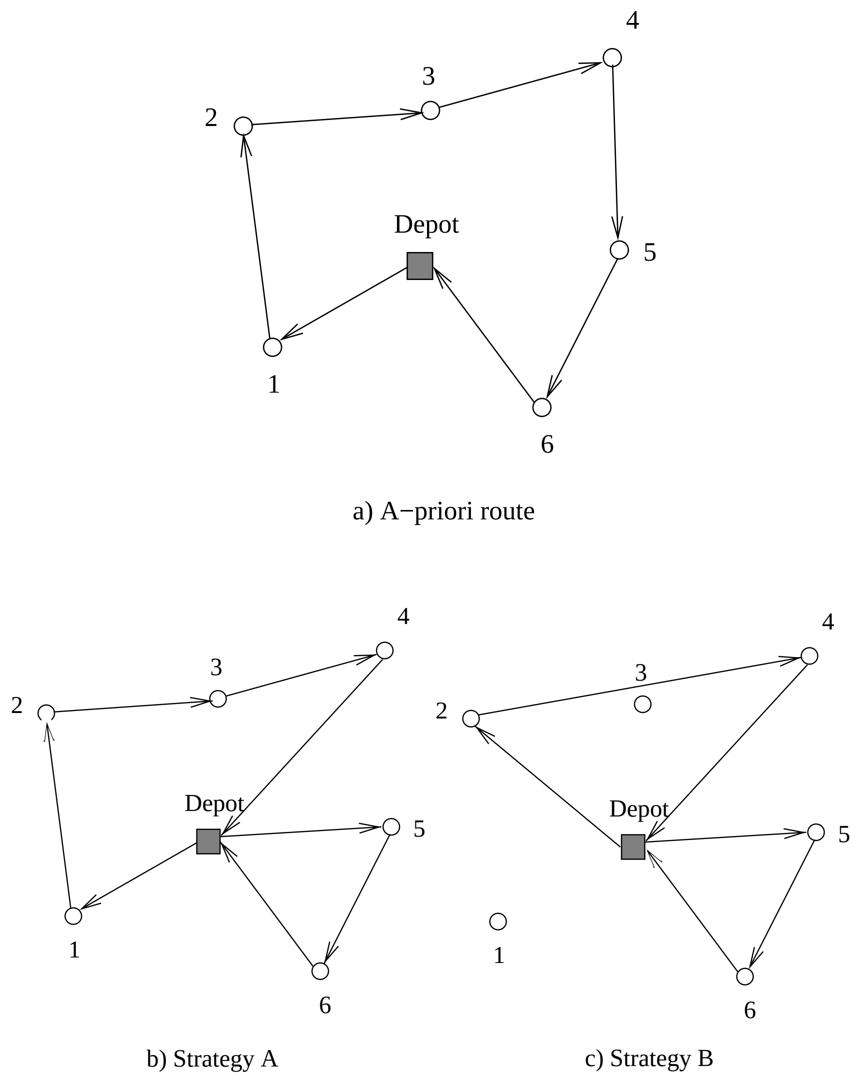


Figure 2.2: (a) The a-priori route. (b) The actual route when *Strategy A* is used. (c) The actual route when *Strategy B* is used.

the customers who require service that day. The total expected distance traveled corresponds to the fixed length of the a-priori route plus the expected length of the additional detours originating from visits to the depot when the capacity is exceeded. Using *Strategy B* the customers with no demand are simply skipped. Figure 2.2 shows a simple example of the PVRP in which a vehicle of capacity 2 has to serve 6 potential customers each with a demand of 0 or 1 unit). In the example customer 1 and 3 turns out to have a null demand and while *Strategy B* simply skips these customers *Strategy A* visits all 6 customers. Bertsimas and Simchi-Levi [5] note that these two methods differ with respect to the fact that *Strategy A* models situations in which the demands become known, when the vehicle arrives at the customers, while, for *Strategy B* the actual demand is known before the vehicle arrives at the customer. Gendreau et al. [16] examine a less restrictive formulation of the PVRP in that it allows for full or split deliveries. The PVRP is solved by finding the routes of minimum expected length. Seguin [40] and Gendreau et al. [17] propose an exact algorithm for the PVRP based on the integer L-shaped method. They solve instances with up to 46 vertices. They also show that stochastic customers are far more complex to handle compared to stochastic demands.

2.2.3 The Stochastic VRP

The Stochastic Vehicle Routing Problem (SVRP) arises when some of the elements of the problem are stochastic. The stochasticity could for instance be uncertain travel times, unknown demands and/or the existence of the customers. Hence, the SVRP could be thought of as a generalization of the PVRP described in the previous section. Gendreau et al. [18] provide an excellent survey paper on the SVRP.

Usually, the SVRP's are two-stage recourse problems. In the first stage a planned or a-priori route is designed and then a *recourse* is used in the second stage to accommodate problems like for instance exceeded capacity. Usually, the recourse generates a cost or a saving that should be considered when the first stage solution is designed. Gendreau illustrates the two-stage methodology by considering a VRP with stochastic demands. The first stage solution consists of m vehicle routes visiting each customer exactly once. After the first stage solution has been designed, the actual demands are disclosed. The disclosure implies that the first stage solution becomes infeasible because the total demands of the customers may exceed the vehicle capacity. In this case a simple second stage policy would be to follow the designed routes until the route fails and then return to the depot to unload/replenish and then resume the service of the customers by returning to the customer where the route failed. In this example the recourse action is defined as the cost associated in the return trip to the depot. Another example of a more clever recourse policy is to return to the depot, whenever the vehicle is near the depot and the residual capacity is below a certain threshold. Dror et al. [13] discuss different recourse policies for this problem.

The two basic formulations of stochastic programs are *chance constrained programs* (CCP) and *stochastic programs with recourse* (SPR). For the CCP the probability of failure in the first stage is constrained to be below a given threshold. The CCP solution does not take the costs associated with failure into consideration. In the SPR one seeks a first stage solution that minimizes the expected costs of the second stage solution plus the expected costs of the recourse. The SPR approach might seem the most intuitively correct of the two formulations. However, SPR's are often much harder to solve than the CCP's.

Gendreau et al. [18] divide the SVRP's into six categories. Below we provide a brief summary of these problem types.

- **TSP with Stochastic Customers (TSPSC)**
  The customers are present with probability p_i. The problem is also known under the name the Probabilistic Traveling Salesman Problem (PTSP) and was described in section 2.2.1.
- **TSP with Stochastic Travel Times (TSPST)**
  In this problem the customers are known, but the length of the arcs varies, i.e., the c_{ij} cost coefficients representing the travel times are random variables. The objective is usually to maximize the probability of completing the tour within a given deadline.
- **m-TSP with Stochastic Travel Times (m-TSPST)**
  This is the m vehicle version of the TSPST, where all the routes start and end in the same depot. Each vehicle is often associated with a cost in order to minimize the number of vehicles in the solution.
- **VRP with Stochastic Demands (VRPSD)**
  The demands of the customers are random variables. The VRPSD is

according to Gendreau et al. [18] without doubt the most studied of the SVRPs. They list references to more than 20 papers dealing with the VRPSD.

- **VRP with Stochastic Customers (VRPSC)**
  This is an extension of the TSPSC - the customers are present with probability p_i, but have deterministic demands. All work done on the VRPSC except for one paper considers the case of unit demands.

- **VRP with Stochastic Customers and Demands (VRPSCD)**
  The VRPSCD is a combination of the VRPSC and the VRPSD. This problem is also known under the name the Probabilistic Vehicle Routing Problem (PVRP) and was described in section 2.2.2.

Generally, for all of the above mentioned problem classes the instances solved to optimality vary from approximately 10 to 50 customers. Gendreau et al. [18] note that it is difficult to asses the quality of the heuristics due to the fact that it is very hard to find exact solutions to most of the SVRPs and because the quality of the bounds is usually quite poor. They conclude that the development of exact algorithms based on the Integer L-Shaped method as well as the construction of tabu search heuristics seem promising.

2.3 Real-time Optimization Methods

In the previous section all methods presented used stochastic/probabilistic information on future events to construct the routes. Within this setting routes will be planned before the vehicle leaves the depot in the morning. In this section, we describe methods that construct routes during the day of operation, i.e. in real-time while the vehicle is on-route. In section 2.3.1 we discuss the dynamic version of the well-known Traveling Salesman Problem. Section 2.3.2 provides a thorough discussion of the Dynamic Traveling Repairman Problem, as this problem will be examined further in the remaining part of this thesis. Finally, in section 2.3.3 we give a brief introduction to the dynamic dial-a-ride problem.

2.3.1 The Dynamic Traveling Salesman Problem

The classical Traveling Salesman Problem (TSP) is definitely one of the most studied problems in operations research. The vast majority of these works concentrate on deterministic and static applications of the TSP. In 1988 Psaraftis [36] introduced a dynamic version of the TSP - the so-called Dynamic Traveling Salesman Problem (DTSP). The motivation for introducing a dynamic TSP was that the classical static TSP is considered to be the "archetypal" (static) vehicle routing problem due to the fact that most other routing problems are extensions and generalizations of the TSP. Psaraftis defines the DTSP as follows: G is the complete graph consisting of n nodes. The demands for service are independently generated at each node of the graph. The generation process could for example be the Poisson process with the intensity parameter λ. The demands must be serviced by the salesman who travels at a constant velocity from node i to node j in the time t_{ij}. The service time of each of the demands is denoted t_0. The problem now consists of finding an "optimal" routing policy for the salesman to follow. Psaraftis identifies the following issues as being interesting and good research issues:

- **Performance measures**
  In communication networks the performance measures are either based on throughput or delay measures. In a vehicle routing context these measures mean that the "optimal" policy should be the policy that either maximizes the expected number of serviced demands per unit time or the policy that minimizes the average expected waiting time for the demands.

- **Light traffic**
  If the demand rate λ is "relatively low", the vehicle will be able to keep up with the demand and the throughput will be $n \cdot \lambda$, independent of the policy, whereas the waiting time will indeed depend on the policy used. The policy of *"service the (probably sole) demand as soon as it appears and then wait"* results in less waiting time compared to the policy *"service demands as you go according to a [node 1 $\rightarrow$ node 2 $\rightarrow$ ... $\rightarrow$ node n $\rightarrow$ node 1 $\rightarrow$ ...] scheme"* for low values of λ.

- **Heavy traffic**
  If the demand rate on the other hand takes on higher values, the situation becomes more complicated. In some cases the demand rate gets so high that the vehicle cannot keep up with demands. In many cases the performance of the routing policies decides whether or not the system can keep up with the demand. In very heavily loaded systems demands might even need to be rejected.

- **Repositioning**
  In cases where the demand rate is low, it may make sense to reposition the vehicle to a strategically located node. This problem resembles a facility location problem and Psaraftis suggests further research on the combination of dynamic vehicle routing and facility location.

Naturally, there exist a wide variety of different versions of the DTSP. The graph G could for example be incomplete, symmetric or Euclidean. Furthermore, the arrival process generating new demands does not have to be the Poisson process and the demand rate could vary for different nodes. Finally, Psaraftis suggests research on when a myopic policy is optimal and whether or not it makes sense to accumulate (batch) demands before going to the next node.

2.3.2 The Dynamic Traveling Repairman Problem

The Dynamic Traveling Repairman Problem (DTRP) was introduced by Bertsimas and Van Ryzin in the paper entitled *"A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane"* [6] and was initially motivated by Psaraftis's [36] work on the dynamic version of the TSP. Bertsimas's and Van Ryzin's work on the DTRP is by far the most extensive and mathematically concise work on dynamic vehicle routing problems. Bertsimas and Van Ryzin mention that in real distribution systems orders/demands arrive randomly in time and the dispatching of vehicles is a continuous process of collecting demands, forming tours and dispatching vehicles. In dynamic settings the waiting time is often more important than the travel cost. Examples of applications where the waiting time is the important factor include the replenishment of stocks in a manufacturing context, the management of taxi cabs, the dispatch of emergency services, geographically dispersed failures to be serviced by a mobile repairman. Bertsimas and Van Ryzin define the DTRP as follows:

- A repairman (or a vehicle/server) travels at unit velocity in a bounded convex region $\mathcal{A}$ of area A.
- All demands are dynamic and arrive in time according to a Poisson process with the intensity parameter λ. The locations of the demands are independently and uniformly distributed in $\mathcal{A}$.

- Each demand requires an independently and identically distributed amount of on-site service time with mean duration $\bar{s}$ and second moment $\bar{s^2}$. The fraction of time that the server spends on on-site servicing the demands is denoted ρ. For stable systems $\rho = \lambda \bar{s}$.

- The system time, T_i, of demand i is defined as the elapsed time between the arrival of demand i and the time the server completes the service of the demand. The steady-state system time denoted by T is defined by $T = lim_{i \to \infty} E[T_i]$.

- The waiting time, W_i, of demand i is defined as the time elapsed from the demand arrived until the service starts. Thus, $T_i = W_i + s_i$. The steady-state waiting time, W, is defined as $W = T - \bar{s}$.

The problem is to design a routing policy that minimizes T. The optimal value of T is denoted T^*. Bertsimas and Van Ryzin stress that although the DTRP resembles a traditional queueing system, queueing theory does not apply due to the fact that the system time T includes the travel times which cannot not be regarded as independent variables. The approach of the authors is to derive lower bounds for all policies for the average system time T. After that Bertsimas and Van Ryzin analyze several policies and compare their performance to the lower bounds. To obtain these results the authors use techniques from combinatorial optimization, queueing theory, geometrical probability and simulation.

The analysis is divided into two cases; one for the light traffic case (i.e. $\lambda \to 0$) and one for the heavy traffic case (i.e. $\rho \to 1$).

For the light traffic case, Bertsimas and Van Ryzin establish the lower bound for T by dividing the system time into three components; the waiting time originating from the travel to the demands prior to demand i, the waiting time originating from the service of the customers prior to demand i and finally demand i's own on-site service time.

$$T^* \geq \frac{E[\| X - x^* \|]}{1-\rho} + \frac{\lambda \bar{s}^2}{2(1-\rho)} + \bar{s} \tag{2.1}$$

If $\mathcal{A}$ is a square $E[\| X - x^* \|] = 0.383\sqrt{A}$. The bound grows with larger values of ρ.

For the heavy traffic case, Bertsimas and Van Ryzin prove that there exists a constant $\gamma = \frac{2}{3}\sqrt{2\pi} \approx 0.266$ so that:

$$T^* \geq \gamma^2 \frac{\lambda A}{(1-\rho)^2} - \frac{1-2\rho}{2\lambda} \tag{2.2}$$

This means that the system time grows by a factor of $(1-\rho)^{-2}$. In traditional queueing systems system time normally grows with a factor of $(1-\rho)^{-1}$. This difference is caused by the geometry of the system. Bertsimas and Van Ryzin [6] mention that the γ value is probably not tight and conjecture that the heavy traffic bound will remain true for larger values of γ.

Bertsimas and Van Ryzin analyze a wide range of routing policies for the DTRP. A brief description of these policies is given below.

- **First Come First Served (FCFS).**
  The demands are served in the order in which they are received by the dispatcher.
- **Stochastic Queue Median (FCFS-SQM).**
  The FCFS-SQM policy is a modification of the FCFS policy. According to the FCFS-SQM policy the server travels directly from the median of the service region to the location of the demand. After the service has been completed, the server returns to the median and waits for the next demand.
- **Nearest Neighbor (NN).**
  After completing service at one location the server travels to the nearest neighboring demand.
- **Traveling Salesman Problem (TSP).**
  The demands are batched into sets of size n. Each time a new set of demands has been collected, a Traveling Salesman Problem is solved. The demands are served according to the optimal TSP tour. If more than one set exists at the same time, the sets are served in an FCFS manner.
- **Space Filling Curve (SFC).**
  The demands are served, as they are encountered during repeated clockwise sweeps of a circle $\mathcal{C}$ that covers the service region.

- **Partitioning policy (PART).**
  The service region $\mathcal{A}$ is partitioned into m^2 smaller subregions in which the demands are served using an FCFS policy. When no more demands are left in a subregion, the server travels to the next subregion.

The FCFS-SQM policy can be shown to give asymptotically optimal performance in light traffic. The conclusions of the simulation tests are that SFC and NN proved to be efficient in both light and heavy traffic. Furthermore, the SFC and NN policies are non-parametric which means that these policies are more stable in a system with a heavy variation in traffic conditions.

Generalizations - Multiple Capacitated Vehicles

In the paper *"Stochastic and Dynamic Vehicle Routing in the Euclidean Plane with Multiple Capacitated Vehicles"* [7], Bertsimas and Van Ryzin generalize their findings of the first paper. They provide bounds and propose a number of new policies for both the un-capacitated m server case and for the case with m servers each with a capacity of q.

For the un-capacitated case, Bertsimas and Van Ryzin extend the lower bounds shown above to the m vehicle case. The vehicle travels at constant velocity, v.

For the light traffic case the authors establish the following lower bound:

$$T^* \geq \frac{1}{v} E[min_{x_0 \in \mathcal{D}^*} \parallel X - x_0 \parallel] + \bar{s} \tag{2.3}$$

The bound can be interpreted as the expected travel time from the closest server to the location of the demand plus the on-site service time.

For the heavy traffic case the authors establish the following lower bound:

$$T^* \geq \gamma^2 \frac{\lambda A}{m^2 v^2 (1-\rho)^2} - \frac{\bar{s}(1-2\rho)}{2\rho} \tag{2.4}$$

The system time still grows by a factor of $(1-\rho)^{-2}$ as in the un-capacitated single-server case. It should also be noted that if the number of servers or the velocity is doubled, the first part of the bound is reduced by a factor of 4.

The SQM policy of the un-capacitated single-server case can be extended to the m server case simply by locating the servers at the m median locations for the region $\mathcal{A}$. When new demands arrive, they are assigned to the nearest median location and its corresponding server. After service has been completed, the server should return to its median. Bertsimas and Van Ryzin show that the mSQM policy is optimal in light traffic. The authors propose several policies for the heavy traffic case. Among these is the simple Random Assignment policy (RAμ), in which the Poisson process is divided into m Poisson subprocesses. Each vehicle is assigned to a subprocess and the demands are served using a heavy-traffic policy μ. The RAμ policy has a constant factor guarantee, but the system time increases by m. Another heavy-traffic policy is the G/G/m version of the TSP policy. Here, the demands are collected into sets of size n. An optimal TSP tour is found each time a new set is collected. The optimal tours are served according to the FCFS scheme. The authors show that for a modified version of this TSP policy the system time could be made independent of m.

In the capacitated version of the multiple-server DTRP, a depot with a fixed location is associated to each server and the servers are only allowed to visit their designated depots. It is allowed for the depots to be coincident. The capacity of each server is denoted q and indicates how many customers a single vehicle is able to serve before returning to the depot. Bertsimas and Van Ryzin provide the following lower bound for the heavy traffic case:

$$T^* \geq \frac{\gamma^2}{9}\frac{\lambda A(1+1/q)^2}{m^2v^2(1-\rho-2\lambda\bar{r}/mqv)^2} - \frac{\bar{s}(1-2\rho)}{2\rho} \tag{2.5}$$

where $\bar{r}$ is the expected distance from an arbitrary customer in $\mathcal{A}$ to the nearest depot. It should be noted that for the case where $q \rightarrow \infty$ the bound reduces to the bound in equation 2.4, although the present constant is weaker. In accordance with the intuition the system time is seen to increase for decreasing values of the capacity, q.

Bertsimas and Van Ryzin propose three policies for the capacitated case. In the qRP (capacitated Region Partitioning) policy the service region $\mathcal{A}$ is divided into k smaller subregions and a TSP tour is constructed each time q consecutive demands have arrived in a subregion. A starting demand is selected from the optimal tour and the server travels to this demand. After completion of the tour the server returns to the depot. The tours should be served in an FCFS order and the number of subregions k should be optimized.

In the qTP policy sets of n demands are collected and optimal tours are performed. The tour is then split into $l = \lceil n/q \rceil$ segments each to be served by a server. As with the qRP policy the optimal tours are served in FCFS order.

Finally, in the modified qTP policy $\mathcal{A}$ is divided into k equally sized subregions using radial cuts centered at the depot. Sets of size n/k are formed in each region and the corresponding optimal tours are constructed. The tours are deposited in an FCFS queue and are to be served by the first available vehicle.

Bertsimas and Van Ryzin prove that all of the above mentioned policies are within a constant factor of the optimal policies in heavy traffic. Bertsimas and Van Ryzin conclude that the stability condition is independent of any characteristics of the service region for the un-capacitated case. However, in the capacitated case, the depot location and the system geometry strongly influence the stability conditions. Furthermore, the authors establish that the expected system time in heavy traffic is $\Theta(\lambda A/(m^2v^2(1-\rho)^2))$ for the un-capacitated system and $\Theta(\lambda A/(m^2v^2(1-\rho-2\lambda\bar{r}/mvq)^2))$ for the capacitated system. The authors also provide a discussion of the relevance of the objective function used for the DTRP. It is argued that in a real-life situation the objective function often consists of a mixture of waiting time costs and travel costs. The analysis is based on an examination of a general objective function which both includes the system time and the travel costs. The analysis confirms that there is a trade-off between travel costs and system time in a dynamic routing system. An example shows that the travel costs can be reduced in return for an increase in the system time.

Further generalizations - General Demand and Inter-arrival Time Distributions

In the paper *"Stochastic and Dynamic Vehicle Routing with General Demand and Inter-arrival Time Distributions"* [8] Bertsimas and Van Ryzin use different techniques to extend their results to describe the more realistic case where the demand locations have an arbitrary continuous distribution and the arrivals follow a general renewal process. The authors also obtain significant improvements of the best known lower bounds. Bertsimas and Van Ryzin conclude that static vehicle routing methods when properly adapted can yield near optimal or in some cases even optimal policies within a dynamic setting. This means that the methods - exact as well as heuristics - developed for static problems are not irrelevant in a dynamic environment.

Recent Work

Papastavrou [33] proposes a new DTRP policy called the **generation policy**. The generation policy initially positions the server at the median of the service region. The server then travels as soon as the next demand for service is received. After the service is completed, the server returns to the median. If there are no waiting demands when the server returns to the median, it should wait for the next demand to be received. Otherwise, the server should serve the waiting demands according to the optimal TSP tour. After completing the TSP tour the server should return to the median and wait for the next "generation" of demands. The motivation for proposing this policy was that the policies described above either perform well in light *or* heavy traffic, but not in both. The numerical results show that the generation policy performs very well in light as well as in moderate and heavy traffic conditions. Papastavrou stresses the importance of a policy that performs well under all conditions, because changing policies can be a costly process and in some situations also difficult to implement. Furthermore, in a constantly changing environment changing policies might not be possible.

Pick-up and Deliveries

Swihart and Papastavrou [42] extend the DTRP into a pick-up and delivery problem. They refer to this problem as the single-vehicle Dynamic Pick-Up and Delivery Problem (DPDP). The objective of DPDP is to minimize the expected system time for the demands. Unit-capacity as well as multiple-capacity variations of the DPDP is considered. It should be noted that by multiple-capacity the authors understand a vehicle with infinite capacity, i.e. the vehicle can carry an infinite number of demands. Swihart and Papastavrou provide a lower bound for the performance for both the unit-capacity and the multiple-capacity cases in light and heavy traffic conditions. For the unit-capacity case three policies were examined; a) the sectoring policy in which the service region is divided into m^2 sectors, b) the stacker crane policy in which demands are grouped into continuous sets as they arrive and an optimal tour is constructed whenever a new set is complete and c) the well-known nearest neighbor policy. The Nearest Neighbor policy performed best for high values of λ. For the multiple-capacity case Swihart and Papastavrou propose the following two policies.

1. The dual TSP policy in which demands are grouped into sets of n demands. An optimal tour through the pick-up locations of the demands is constructed for each set. Then, an optimal tour is constructed through the delivery locations for each set.
2. The multiple-capacity nearest neighbor policy in which the vehicle proceeds to the nearest pick-up or valid delivery location. A valid delivery location is defined as the delivery location for demands that have been picked up but not delivered.

The multiple-capacity nearest neighbor policy performed better than the dual traveling salesman policy. The authors suggest further research on bounds and policies for cases with limited capacities. Swihart and Papastavrou also note that the bounds presented could be tightened further.

2.3.3 Dynamic Dial-A-Ride Systems

The Dial-A-Ride Problem (DARP) consists in finding the cost optimal route through a number of customers; each of which has an origin location (pick-up location) and a destination location (delivery location). The objective to be minimized is usually a function of the number of vehicles in the solution, the distance driven and the level of service provided to the customers. The conventional DARP could be divided into the following three broad categories.

- **Many-to-one (MTO)**
  MTO is the simplest type of the DARP to provide and it is relatively

easy to construct high quality routes. As an example of a real-life application of the MTO one could mention the transport of customers from a residential area to for instance the local shopping mall.

- **Many-to-few (MTF)**
  MTF is harder to provide due to the fact that this system services more than one attraction center. However, the quality/cost ratio is relatively high in an MTF, because a considerable amount of the transit goes from home to some attraction center. Again, a real-life example could for instance be three different stops at the same large shopping mall.

- **Many-to-many (MTM)**
  The MTM service could be thought of as a taxi cab system in which multiple passengers could be in the vehicle at the same time. Passengers are being picked up at several different locations and dropped off at several different locations. The MTM problem is therefore similar to transportation of elderly and handicapped people.

A number of additional side constraints might be present, as is the case when dealing with conventional VRP. Madsen et al. [30] examine a Dial-A-Ride routing and scheduling Problem with Time Windows (DARPTW). They solve the real-life problem faced by the Copenhagen Firefighting Service (CFFS) when transporting elderly and handicapped people. CFFS does not have all requests for service at the beginning of the day of operation. In other words customers might call in for service during the day of operation. The authors deal with the online requests by re-optimization of the routes each time a new request is received by the dispatchers. One of the system requirements was that new requests should be added and scheduled within a maximum of 2 seconds.

Dial [12] introduces the ADART (Autonomous Dial-A-Ride Transit) system which is a modernized version of the DARP. The ADART system has fully automated dispatching and autonomously managed vehicles servicing both customers subscribing to the ADART service and immediate request customers calling in for service. Dial argues that “ADART is an idea whose time has come” and that autonomous dial-a-ride may be the answer for transit service in a low-density area.

2.4 Related Problems

In this section a number of papers addressing the DVRP or related problems will be examined.

Real-Time Vehicle Routing and Scheduling

Gendreau et al. [15] describe a dynamic routing problem motivated by a courier service application. The customers appear in real-time and must be serviced within a given soft time window. The problem is modeled as a dynamic version of the Vehicle Routing Problem with Time Windows (DVRPTW). The tabu search heuristic used in this work was originally designed for the static version of this problem and was therefore modified in order to deal with dynamic version. The method is then implemented on a parallel platform. The authors finally present numerical results for both light and heavy traffic and compare the tabu search method with other heuristic methods.

Time-dependent TSP and VRP

In the Time-Dependent Traveling Salesman Problem (TD-TSP) and the Time-Dependent Vehicle Routing Problem (TD-VRP) the objective is to minimize the total time of the routes. The travel times between customers and between the depot and the customers depends on the distance between the points and the time of the day. The time-dependency accounts for variations in travel speed caused by congestion etc. All other data are static and deterministic. Malandraki [31] and Malandraki and Daskin [32] introduce and develop several heuristic algorithms based on the nearest-neighbor method. Malandraki and Daskin also propose a mathematical-programming-based heuristic for the TD-TSP which uses cutting planes. The authors test their algorithms on randomly generated test problems with 10 to 25 nodes and with travel times represented by step functions of two or three periods per element in the travel time matrix. The nearest-neighbor heuristics solve the instances within one second while the cutting plane algorithm is relatively computationally expensive even for small instances. The gaps between the value of the best feasible solution and the solution of the final LP relaxation are also large. The authors conclude that more research is needed within this area in order to develop algorithms for time dependent problems.

The MORSS Algorithm

In 1985 Psaraftis et al. [38] developed the MORSS (M.I.T. Ocean Routing and Scheduling System) to route cargo ships for the U.S. Military Sealift Command (MSC) in emergency situations. The objective for the MSC is to allocate cargo ships to cargoes so that all cargoes arrive at their destinations within a certain time window. A number of additional constraints such as ship capacity and compatibility between ships, cargoes and ports have to be considered. Furthermore, three criteria have to be satisfied; the cargoes should be delivered within time, the ship utilization must be high and port congestion must be avoided. Psaraftis argues that the problem is dynamic in nature, because the environment conditions change in real-time. The proposed algorithm uses a rolling planning horizon principle in which only cargoes within the temporal "front end" of the horizon are considered for permanent assignment. The algorithm pays less attention to cargoes further away into the future due to the fact that these future cargoes are subject to changes.

2.5 Chronological Overview

Figure 2.3 gives a chronological overview of some of the most important work on dynamic and stochastic vehicle routing problems. A virtual line through the middle of the illustration divides the literature in two sides; the left-hand one embracing a-priori optimization based models and the right-hand one embracing the real-time optimization based models. Horizontally, the illustration is organized so that the basic models as for instance the DTRP and PTSP are located in the middle of the illustration corresponding to the axis of complexity shown at the bottom of the figure. Similarly the complex DVRPTW studied by Gendreau et al. is placed at the very right to represent the complexity of this problem.

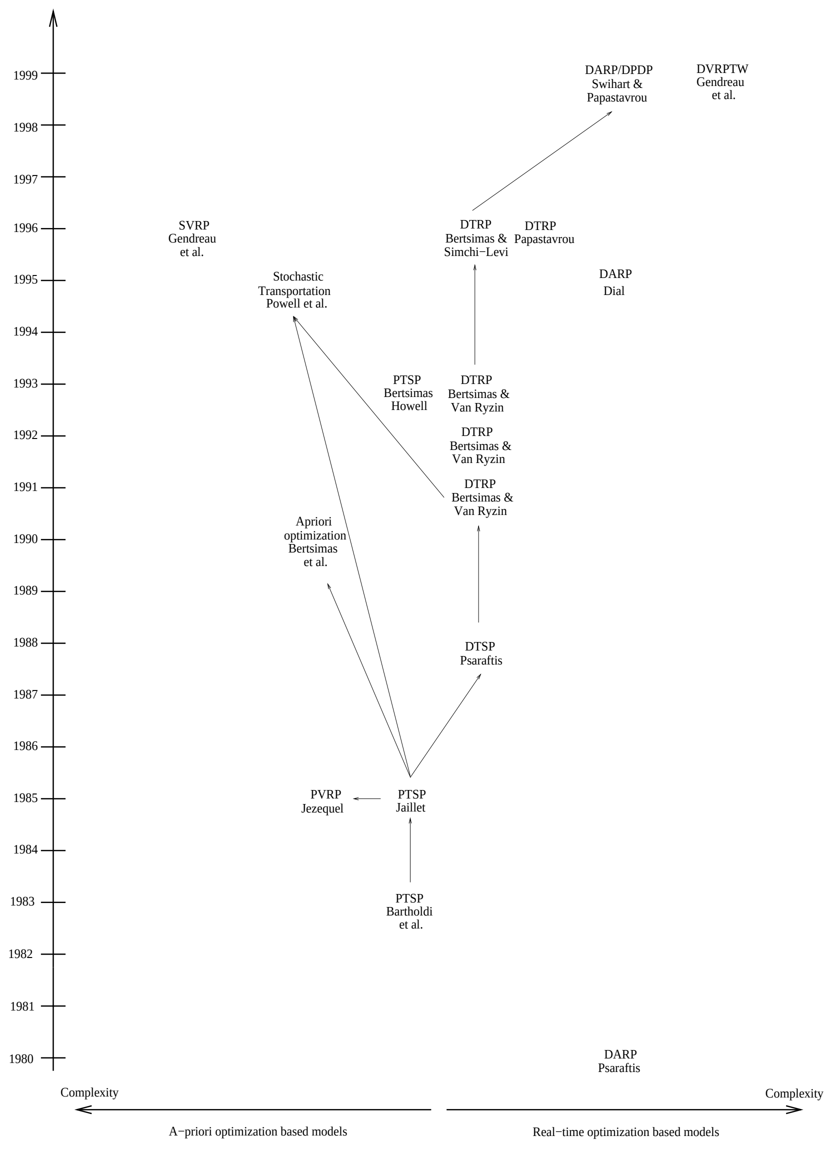

1999
1998
1997
1996
1995
1994
1993
1992
1991
1990
1989
1988
1987
1986
1985
1984
1983
1982
1981
1980
DARP/DPDP Swihart & Papastavrou
DVRPTW Gendreau et al.
SVRP Gendreau et al.
DTRP Bertsimas & Simchi−Levi
DTRP Papastavrou
Stochastic Transportation Powell et al.
DARP Dial
PTSP Bertsimas Howell
DTRP Bertsimas & Van Ryzin
DTRP Bertsimas & Van Ryzin
DTRP Bertsimas & Van Ryzin
Apriori optimization Bertsimas et al.
DTSP Psaraftis
PVRP Jezequel
PTSP Jaillet
PTSP Bartholdi et al.
DARP Psaraftis
Complexity
Complexity
A−priori optimization based models
Real−time optimization based models

Figure 2.3: Chronological overview of important literature.

2.6 Summary

Below a brief summary of the literature reviewed in this chapter will be given.

The a-priori optimization based methods, i.e., models based on mathematical programming, do not seem to be applicable in a real-life online DVRP context, since these problems are static by nature, and using such methodology in a dynamic environment leads to very high computation times which should be avoided in real-life applications. Furthermore, it may not be worth the computational effort to try to find an optimal or near-optimal solution in a real-time setting, because new requests may render the solution sub-optimal. Finally, PVRP and SVRP based models require extensive a-priori information such as the probability of a certain customer requiring service upon a certain day and time. In most cases such detailed information will not be available.

The real-time optimization based methods, like the dynamic version of the TSP and the DTRP, seem much more promising when solving real-life problems, since these models are dynamic by nature. However, despite the extensive theoretical work on the DTRP by especially Bertsimas and Van Ryzin, experiences with real-life applications of the DTSP as well as the DTRP seem to be almost non-existing. This may be due to the fact that the DTSP as well as the DTRP in their basic versions are too simple to be found in any real-life distribution contexts.

For real-life DVRP problems methods like the parallel implementation of the tabu search proposed by Gendreau et al. [15] seems to be an interesting approach, since the feasibility check and insertion of new immediate requests could be done with very little computational effort. This approach also uses the idle periods when no new immediate requests have been received for a certain amount of time to improve on the current best solution. In this way the computational power is exploited during idle periods. Naturally, given the right amount of information and a reasonable long idle period an ideal method should be able to foresee possible scenarios and initiate new calculations.

Chapter 3

The Simulation of Data

As mentioned in section 1.3 randomly generated data rather than real-life datasets are often used when analyzing and designing dynamic vehicle routing systems. In this chapter we will discuss some of the important issues that should be addressed when generating random test data.

The times of new immediate requests calling in for service, the geographical locations, the on-site service times, and the demands of the customers are all parameters subject to stochasticity within a dynamic vehicle routing problem. We discuss these parameters in order to provide some guidelines for dealing with stochasticity when simulating DVRP scenarios.

Throughout the chapter an empirical case study will be examined. The data of this study originate from a real-life scenario of a long-distance courier mail and packages service provider. The routing problem faced by such companies is examined in further detail in chapter 8.

3.1 Request Times of New Customers

Naturally, the arrival process for generating the time when the immediate request customers call in for service is essential when analyzing a dynamic routing system. In this section this arrival process is analyzed. Firstly, we provide a brief study on the commonly used Poisson process. Then, the call times of the long-distance courier mail case study are presented. Finally, we propose a time-dependent Poisson process for generating call times in a constantly changing environment.

3.1.1 The Poisson Process

Traditionally, the main reason for using the Poisson process is that this process is very simple to work with as well as the fact that it possesses some nice statistical characteristics which means that it is possible to derive analytical expressions describing the system in question.

The most important statistical properties of the Poisson process could be summarized as follows: The arrival intensity parameter of the Poisson process, usually denoted λ, denotes the number of requests per time unit, i.e., if the time unit is 1 hour, the Poisson process with $\lambda = 2.5$ generates 2.5 requests per hour in average. The expected number of requests during the period of T hours is therefore λT. The time interval between two requests is exponentially distributed, i.e., the Poisson process has no memory of old events.

3.1.2 Pick-up of Courier Mail - Case Study

In Figure 3.1 the relation between the number of calls received by the call center of a long-distance courier mail service provider is shown as a function of the time of the day. The X-axis has been sampled at 15 minutes' intervals so that all calls received within each 15-minute interval were accumulated before printing. This was done to remove noise from outliers and hereby making the trend in the data more clear. The figure is based on all immediate requests received during a two-week (Monday-Friday) period in four almost adjacent service areas. The number of immediate requests total 283 calls for service during this period. The figure shows extensive variations in the number of calls throughout the day. The call center opens at 8:00 and closes at 18:00. The morning hours are relatively slow, but the number of calls increase towards noon. During lunch breaks (approximately 12:00 - 13:00) the number of calls is relatively low. The number of calls increase during the first part of the afternoon and reach its maximum at approximately 14:30.

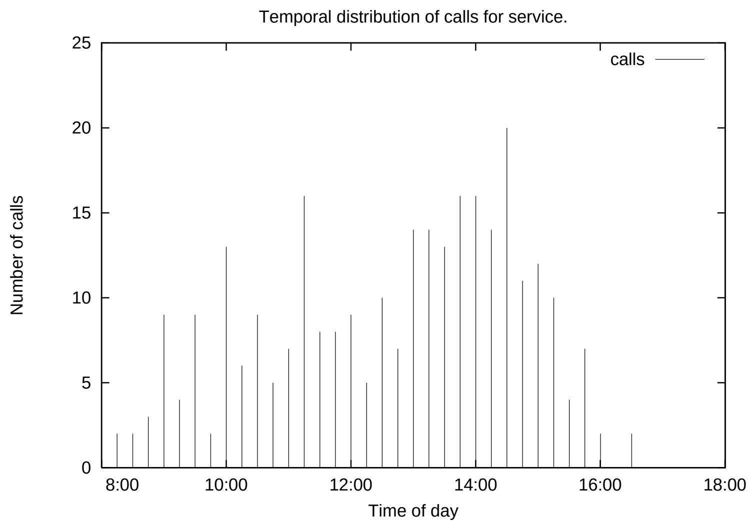


Figure 3.1: The number of calls in a real-life instance of a long-distance courier mail service provider as a function of the time of the day.

3.1.3 The Time-dependent Poisson Process

The previous section clearly indicates that the call intensity varies during the day. Naturally, the actual shape of the distribution depends heavily on the application. In the above scenario the majority of the customers call in during the afternoon, because most offices close between 16:00 and 17:00. In other scenarios such as the distribution of oil to private households one would expect the majority of the customers to call in early during the morning hours after a cold night without heating. Obviously, this means that using a distribution with a constant intensity parameter does not give an adequate description of the actual real-life situation. Alternatively, other distributions for arrivals could be used. The peak hour phenomenon could for instance be modeled by using a time-dependent Poisson process. In Figure 3.2 an example of a continuous time-dependent Poisson process is shown.

The generation of time-dependent Poisson distributed numbers could for instance be done by using the so-called *rejection method* (Larson and Odoni

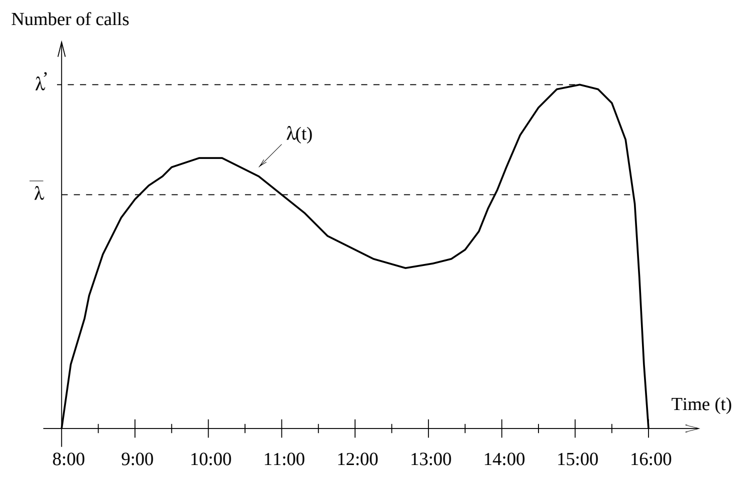


Figure 3.2: An example of a time-dependent Poisson distribution.

[28]) which is based on a geometrical interpretation of the probability density function. With reference to the example shown in Figure 3.2 the rejection method could be explained as follows:

1. Estimate the average arrival intensity rate, $\overline{\lambda}$, over the time period T.

$$\overline{\lambda} = \frac{1}{T}\int_0^T \lambda(t)dt \tag{3.1}$$

2. Find the number of requests to be generated, k, by sampling the Poisson process with the arrival intensity of $\overline{\lambda}$ over the time period T.

3. Generate two uniformly distributed random numbers, r_1 and r_2, where $0 \leq r_1 \leq T$ and $r_2 = max_t\lambda(t) = \lambda'$. If $r_2 \leq \lambda(r_1)$, accept r_1 as the i'th time instant τ_i. Repeat this step until k time instants have been accepted.

4. Sort the time instants τ_i, $i = 1, 2, ..., k$ into an ordered set.

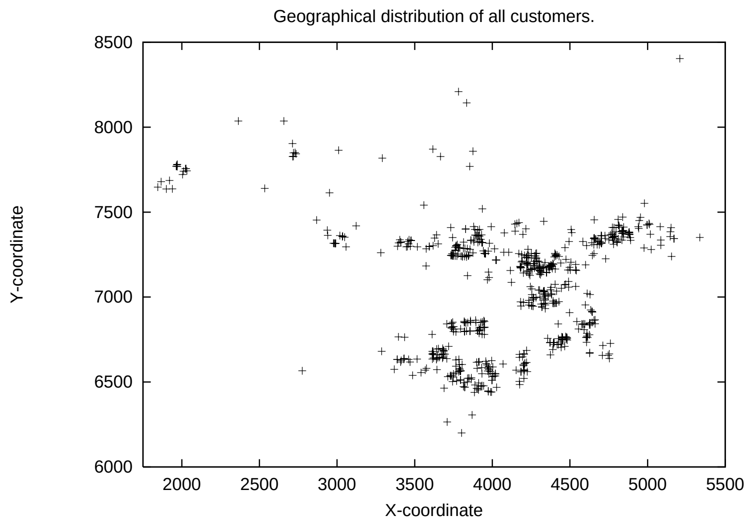
We believe that new generation methods should consider using time-dependent arrival processes since the arrival intensities within a dynamic distribution context are often seen to be subject to relatively heavy variations. However, the time-dependent Poisson process will not be considered in the remaining part of this thesis. The reason for not considering this aspect is that including such information would introduce yet more parameters which increase the risk of blurring the general picture.

3.2 The Geographical Location of Customers

The spatial distribution of the customers' geographical location is one of the most essential parameters in both static and dynamic vehicle routing problems. The major part of the work on static vehicle routing uses Solomon's test dataset [41] for reference. These datasets are divided into 6 groups, denoted R1, R2, C1, C2, RC1 and RC2. Each group consists of 8 to 12 different instances. The R1 and R2 datasets are randomly generated according to a uniform distribution. The C1 and C2 datasets are clustered, while the RC1 and RC2 datasets are a mixture of being clustered and randomly distributed. All problems consist of 100 customers located in a 100 x 100 square area.

3.2.1 Courier Mail Services - Case Study

Figure 3.3 shows the geographical distribution of the customers from the long-distance courier mail service application. It can be seen from the figure that the locations of the customers are strongly correlated. Furthermore, it should also be noted that a number of the customers are located outside the most congested areas. The strong tendency towards clustering could be explained by the type of the application. The spatial distribution of the customers in Figure 3.3 seems natural, because the long-distance courier mail services are mostly used by big corporations which tend to be located in "industrial neighborhoods" or even in office buildings with a number of different corporations in the same building. The outliers are either companies located in a remote area or private households only receiving/sending long-distance mail once in a while.



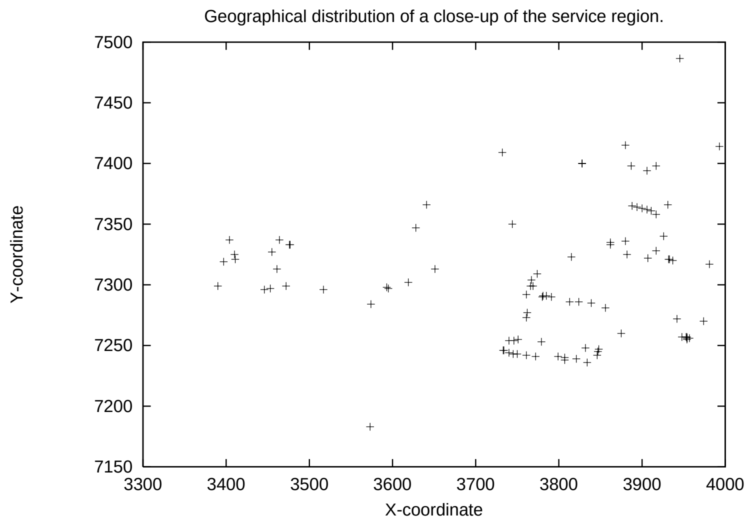


Figure 3.3: The geographical distribution of customers of a real-life instance for the long-distance courier mail problem.

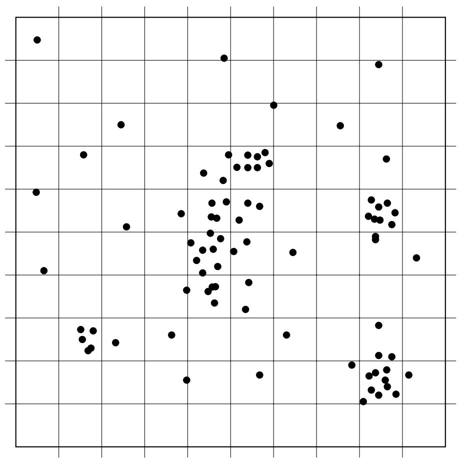


Figure 3.4: Illustration of the process of generating a combination of clustered and randomly distributed locations of customers using a 10 x 10 grid.

3.2.2 Generating Geographical Locations

In order to be able to capture the spatial distribution, the service region can be divided into n smaller subregions, in which the arrival intensity in the customer generation process varies from subregion to subregion.

In Figure 3.4 an example of this method is shown. The customer locations were generated by dividing the 1000 x 1000 sized service region into 100 subregions. In each subregion a uniform distribution generated the geographical locations. The number of customers in each subregion were generated by using a Poisson process with intensity parameters which depended on the subregion.

The generation of customers could of course also be performed in several other ways; for instance by using more sophisticated probabilistic distributions.

3.3 On-site Service Times

In the major part of the research on time constrained vehicle routing problems (like for instance the VRPTW) constant values are used for describing the on-site service times. However, in real-life and especially within a dynamic setting the on-site service times are subject to stochasticity. In most distribution contexts the on-site service time involves loading/un-loading physical items. In most cases this means that the service requires human interaction of some form. An evident example of this could be the situation that occurs when the mail man delivers a parcel and the receiver must sign a document. The time spent on this situation could of course vary from few seconds to several minutes. These factors suggest that using only constant service times might not be the most appropriate approach.

3.3.1 The Log-normal Distribution

The log-normal distribution is a transformation of the well-known normal distribution. A stochastic variable X is said to be log-normally distributed, if $log(X)$ is normally distributed, i.e.:

$$X \in LN(\alpha, \beta^2) \Longleftrightarrow logX \in N(\alpha, \beta^2) \tag{3.2}$$

Hence, the log-normal distribution possesses the desirable characteristic of only generating positive numbers. In Figure 3.5 10,000,000 samples of the log-uniform distribution are shown.

The expected value $E(X)$ and the variance $V(X)$ are defined as:

$$\begin{aligned} E(X) &= e^{\alpha+\frac{1}{2}\beta^2} && (3.3)\\ V(X) &= e^{2\alpha+\beta^2}(e^{\beta^2}-1) && (3.4) \end{aligned}$$

and the parameters α and β are defined as:

$$\begin{aligned} \alpha &= log_e\mu - \frac{1}{2}log_e\mu(\frac{\sigma^2}{\mu^2}+1) && (3.5)\\ \beta^2 &= log_e\mu(\frac{\sigma^2}{\mu^2}+1) && (3.6) \end{aligned}$$

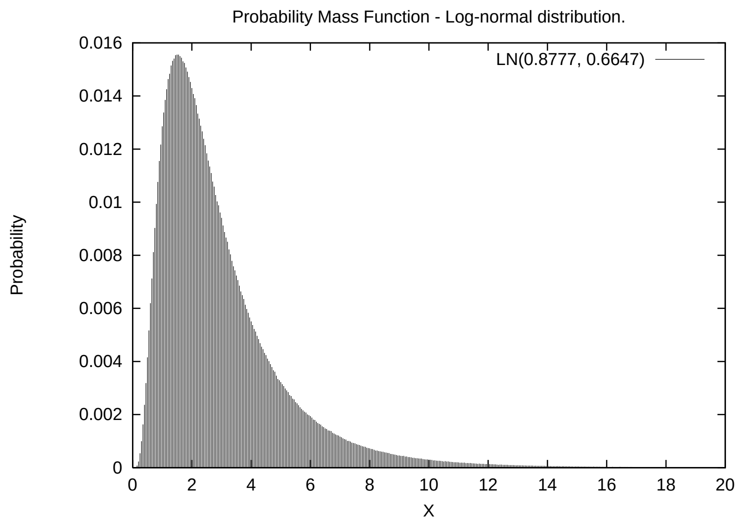


Figure 3.5: Simulation of 10,000,000 randomly generated numbers in the log-normal distribution.

where $\mu = E(X)$ and $\sigma^2 = V(X)$. This means that the expected value and the variance should be 3.0 and 5.0 respectively. A simple method for generating samples in the log-normal distribution can be found in section 5.3 of Ross [39].

In the following we will use the log-normal distribution to generate on-site service times, as we believe that this distribution will give a fair description of the on-site service times within the above sketched distribution contexts. Before using this distribution in a real-life context an empirical analysis should be considered in order to test whether or not the distribution gives a fair description of the real-life data. Also, this analysis should provide the planners with the α and β parameters of the distribution.

3.4 Customer Demands

The actual demand of customers - advance request as well as immediate request customers - may also be subject to strong variations. This problem is usually referred to as the vehicle routing problem with stochastic customers and demands (please see section 2.2.3). As was the case with the previously mentioned parameters, several different statistical distributions could be used for generating suitable customer demands. The actual application will have to show which distribution is the most realistic one to use.

The subject of stochastic customer demands will not be studied in further detail in this thesis.

3.5 Travel Times

The final system parameter subject to stochasticity to be considered in this chapter is the travel time between the customers. The uncertainty in this parameter is due to for instance reduced travel speed caused by congestion in the traffic network or it could be caused by unforeseen events such as road construction and traffic accidents. The lateness caused by such events often results in considerable delays. Hence, a true dynamic routing system should be able to deal with stochastic travel times and the routing system must be able to deal with these situations.

Jørgensen [25] analyzes the implications of a noise-filled travel time matrix for a static vehicle routing problem with time windows. The noise is determined by using a random generator which multiplies each element in the travel time matrix by a parameter, p_{noise}. Three different scenarios are considered. In the first scenario $p_{noise} \in [0.95, 1.05]$, in the second $p_{noise} \in [0.90, 1.10]$ and in the third $p_{noise} \in [0.75, 1.25]$. This means that the average level of the noise is 0! The conclusion is that the total distance driven by the vehicle is only subject to very small changes (from less than 1 % to approximately 5 %) for the first two scenarios, while the total distance driven might be up to 25 % longer for the last scenario. However, in real-life situations the noise would usually only be in the direction of longer travel times so that the average noise level is greater than 0. Therefore, one might expect that the stochasticity in the travel time matrix means that the total distance driven actually increases for increasing values of travel times.

The subject of stochastic travel times will not be studied in further detail in this thesis.

3.6 Summary

In this chapter a number of parameters potential to be subject to stochasticity were discussed. In general, when using simulation as an analytical tool to assess the characteristics of a system the risk of misuse and wrong conclusions are present if the right data are not used. Therefore, it is of paramount importance that the analyst tries to identify which distributions to use for generating the test instances in order to be able to give a consistent and precise description of the system. However, in most real-life dynamic vehicle routing scenarios the use of very simple simulation techniques should be sufficient in order to provide the model with the necessary information.

Chapter 4

The Degree of Dynamism

Measuring the performance of a dynamic vehicle routing system is not a trivial assignment. In contrast to a deterministic and static vehicle routing problem the performance of the dynamic counterpart is assumed to be dependent on not only the number of customers and the spatial distribution of these, but also the number of dynamic events and the time when these events actually take place. Therefore, a single measure for describing the system's "dynamism" would be very valuable when one wants to examine the performance of a specific algorithm under varying conditions.

As we will see in the following, measures that might seem promising for the use of describing dynamism for one system might turn out to be inadequate for describing the dynamism of other systems. Therefore this chapter is divided into two sections where the first section discusses measures for dynamism in systems without time windows while the second section examines systems with the presence of time windows.

The intention of this chapter is to examine the existing system measures with respect to describing the dynamism of a vehicle routing system and to introduce new measures for this type of systems. Furthermore, we propose a framework for dynamic vehicle routing systems based on their *degree of dynamism*.

4.1 Dynamism without Time Windows

In this section we examine measures which try to describe the dynamism of a dynamic vehicle routing system without time windows. In a system without time windows only three parameters are relevant: The number of static customers, the number of dynamic customers and the arrival times of the dynamic customers.

4.1.1 The Degree of Dynamism

In many dynamic routing systems information is not always only received while the system is on-line. Often, some level of information is gathered before the day of operation begins. The extent of new information emerging during the operational phase of the system can be used to assess the dynamism of the system in question. The most basic measure for this in a routing context is the number of dynamic requests relative to the total. Following Lund et al. [29], we define this ratio as *the degree of dynamism* of the system considered and denote it *dod* . Hence:

$$dod = \frac{Dynamic\, requests}{Total\, requests} \tag{4.1}$$

In the example shown in Figure 1.1 on page 6 the degree of dynamism is therefore 20 % (2 out of 10 customers are dynamic).

However, this measure does not take the arrival times of the dynamic requests into account. This means that two systems, one in which the dynamic requests are received at the beginning of the planning horizon and the other in which they occur late during the day, are perceived as equivalent. Naturally, in real world applications these two scenarios are however very different. Figure 4.1 illustrates two DVRP scenarios in which the times for receiving immediate requests differ considerably. In `Scenario A` all six immediate requests are received relatively early during the planning horizon. In `Scenario B` the requests are distributed almost evenly throughout the planning horizon. We suggest that the planner would prefer the latter scenario to the first, since `Scenario B` gives her time to react to the dynamic requests as opposed to the situation sketched in `Scenario A` in which she may not have enough time to find a suitable reaction to the dynamic requests received at the very end of the planning horizon.

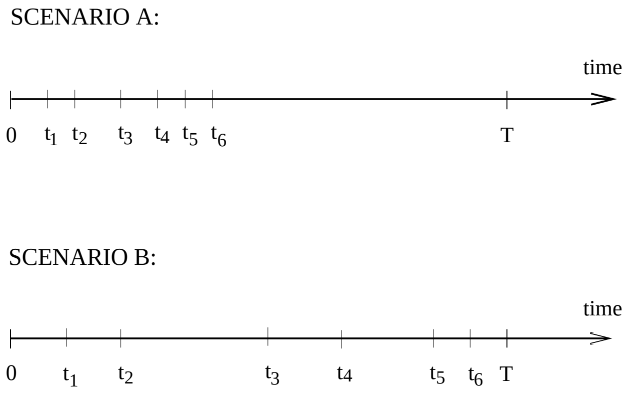


Figure 4.1: Arrival time of dynamic requests.

Furthermore, from a performance point of view it is clear that having the highest number of requests in the pool of waiting requests improves the solution quality with respect to the objective of minimizing the total distance driven. Hence, in the systems illustrated by the two scenarios in Figure 4.1 the expected length of the route would be shorter in `Scenario A` than in `Scenario B` due to the fact that the planner in the former scenario from time t_6 has all information on the locations of the requests which means that she from that point in time could form an optimal TSP tour through the pool of waiting customers.

In section 4.1.2 we extend the above defined measure to include the times when the immediate requests are received.

Before turning to the extended degree of dynamism measure, a final comment on the scenarios illustrated in Figure 4.1 should be made. Assuming that a number of advance requests are already in the pool of waiting requests to be served, the system described in `Scenario A` is likely to be the most difficult to manage since if the planner is already busy taking care of the advance request customers during the early stages of the planning horizon, she would be likely to prefer to have to deal with the immediate requests later during the day after the advance requests have been serviced. Using this reasoning one might classify `Scenario A` as the most dynamic of the two systems illustrated. However, we chose to go with the first classification since this seems to be the most intuitively correct with respect to the performance of the system.

4.1.2 Effective Degree of Dynamism - EDOD

Let us now consider a scenario in which the planning horizon starts at time 0 and ends at time T. The advance requests are received *before* the beginning of the planning horizon *or* at time 0 at the latest. The time the i'th immediate request is received is denoted t_i, i.e. $0 < t_i \leq T$. The number of immediate requests received during the planning horizon is denoted n_{imm} and the number of advance requests is denoted n_{adv}. The total number of requests, n_{tot} is therefore $n_{adv} + n_{imm}$. We now define the following measure as the effective degree of dynamism, denoted *edod*:

$$edod = \frac{\sum_{i=1}^{n_{imm}} (\frac{t_i}{T})}{n_{tot}} \tag{4.2}$$

The effective degree of dynamism then represents an average of how late the requests *are received* compared to the latest possible time the requests *could be received*.

It can easily be seen that:

$$0 \leq edod \leq 1 \tag{4.3}$$

where $edod = 0$ in a pure dynamic system and $edod = 1$ in a pure static system in which all the requests are received at time 0 and time T respectively. It is also obvious that

$$\lim_{t_i \to T \ \forall i} edod = 1 \tag{4.4}$$

4.2 Dynamism and Time Windows

In several vehicle routing applications the service of the customers must commence within a given interval of time - usually referred to as Time Windows (TW). The time the i'th immediate request is received is denoted t_i and the earliest time that service can begin (i.e. the start of the time window) is denoted e_i while the latest possible time that service should begin is denoted l_i. In applications with time windows the *reaction time* is a very important issue. The reaction time is defined as the temporal distance between the time the request is received and the latest possible time at which the service of the requests should begin. In Figure 4.2 the reaction time of the i'th immediate request is denoted r_i, i.e. $r_i = l_i - t_i$.

In this section the effective degree of dynamism is extended so that the reaction times are included in the measure.

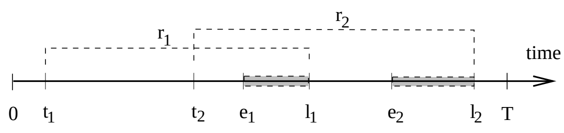


Figure 4.2: The reaction times of two dynamic customers in a DVRP with time windows.

4.2.1 Effective Degree of Dynamism - EDOD-TW

Consider the two scenarios sketched in Figure 4.3 - in both of these scenarios we have two immediate requests with time windows. In `Scenario A` the width of the time windows of the immediate requests is relatively wide compared to the width of the time windows in `Scenario B`. Furthermore, the reaction times in `Scenario A` are relatively long compared to the reaction times in `Scenario B`. This means that `Scenario A` would be preferred by the planner because this situation gives her much more room to insert the dynamic requests into the routes.

In general the planner would prefer to have a relatively long reaction time for the immediate requests. The following measure therefore uses the relation between the reaction time and the remaining part of the planning horizon as the key component.

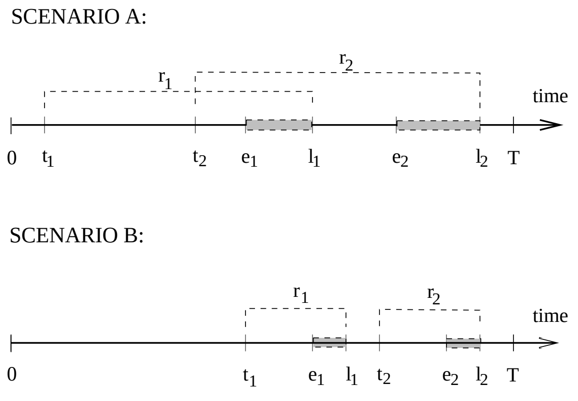


Figure 4.3: Two scenarios with two dynamic requests.

The effective degree of dynamism measure can then be extended to:

$$\begin{aligned} edod - tw &= \frac{1}{n_{tot}} \sum_{i=1}^{n_{tot}} \left(\frac{T - (l_i - t_i)}{T}\right) \quad (4.5) \\ &= \frac{1}{n_{tot}} \sum_{i=1}^{n_{tot}} \left(1 - \frac{r_i}{T}\right) \quad (4.6) \end{aligned}$$

Again, it is easy to see that

$$0 \leq edod - tw \leq 1 \quad (4.7)$$

because

$$l_i - t_i \leq T, \quad i = 1, 2, ..., n_{tot} \quad (4.8)$$

4.3 Framework for Dynamic Systems

In this section we propose a general framework for the dynamic routing systems based on their degree of dynamism. The degree of dynamism measure gives rise to a continuum of dynamic levels. However, we believe it is possible to categorize the vast majority of routing systems found in practice by using three echelons. Henceforth, we only discern between weakly, moderately, and strongly dynamic systems. We discuss these next.

4.3.1 Weakly Dynamic Systems

Routing environments with a weak degree of dynamism include the distribution of heating oil or liquid gas to private households. In this example, most customers (more than 80 %) are known at the time of the construction of the routes. These are "automatic replenishment" customers for whom the company uses their demand profile and "degree days" to schedule deliveries. Requests may also be received during the day from "on call" customers. They get serviced as a function of their level of inventory, the available time, and unused tank capacity. The reaction time is considerably longer in such a problem compared to that in a taxi dispatching system. Other examples include residential utility repair services, such as cable television and telephone. The transportation of elderly and handicapped people - usually referred to as the dynamic dial-a-ride problem - is yet another example of a routing system that can be classified as weakly dynamic (cf. sections 1.7.4 and 2.3.3).

The traditional approach for solving such problems has been based on adaptation of static procedures. A static vehicle routing problem is solved every time an input update occurs. In chapter 6 the relationship between the solution quality of such an approach and the degree of dynamism is examined in detail.

4.3.2 Moderately Dynamic Systems

Practical examples of such systems include long-distance courier mail services and appliance repairs. In the latter setting, for instance, scheduled customers are interspersed with dynamic ones that need immediate attention due to the gravity of their request (e.g. a broken refrigerator will take precedence over an already scheduled partly broken stove top).

Compared to weakly dynamic systems, in moderate ones the number of immediate requests account for a substantial part of the total number of customers who have to be serviced. For a wide range of applications, researchers have also included stochastic elements, such as unknown customer demands, and used stochastic programming models. However, these tend to become extremely hard to solve due to their combinatorial nature. Solution strategies have often been based on deferring decisions until the latest possible moment. Ideally, this will improve the quality of the decisions made because the level of uncertainty decreases as time elapses. Yet, the frequent updates of the available information severely limit computation time. As an alternative approach, one could use a model which utilizes the time between input updates to perform improvements of potential routes. This, of course, requires detailed information on future requests.

4.3.3 Strongly Dynamic Systems

Emergency services, such as police, fire and ambulance departments exhibit strong dynamic behavior. Another example is taxi cab services in which only a negligible number of the customers have ordered their ride in advance. The importance of especially the first of these problems has motivated numerous strategic and tactical analyses of their associated costs and quality of service since the early 1970s. In particular, ways to decrease response times have been studied with continued interest. The work of Larson and Odoni [28] captures many of the key issues.

Strongly dynamic systems are characterized by the fast pace of changes in the data and the urgency of almost all requests received. Furthermore, the quality of a-priori information is often relatively poor with regards to data such as the locations of the customers, their demands, etc. If, on the other hand, a-priori information or even expectations are available, it would be evident to try to incorporate them into the algorithms used. For example, this could involve moving an idle vehicle currently situated in a low traffic area to a central location.

Another characteristic of these systems is that queueing begins to occur in relatively heavy traffic. Therefore, handling aspects related to it often plays a central role. This is especially true given the type of systems found in practice and the importance of reaction time. Examples of queueing-based algorithms include those by Bertsimas and Van Ryzin [6], [7] and [8].

4.3.4 System Classification

The objective of a dynamic routing system is often closely related to the response time of the service provided. Figure 4.4 illustrates the relationship between the degree of dynamism and the objectives for different problem classes.

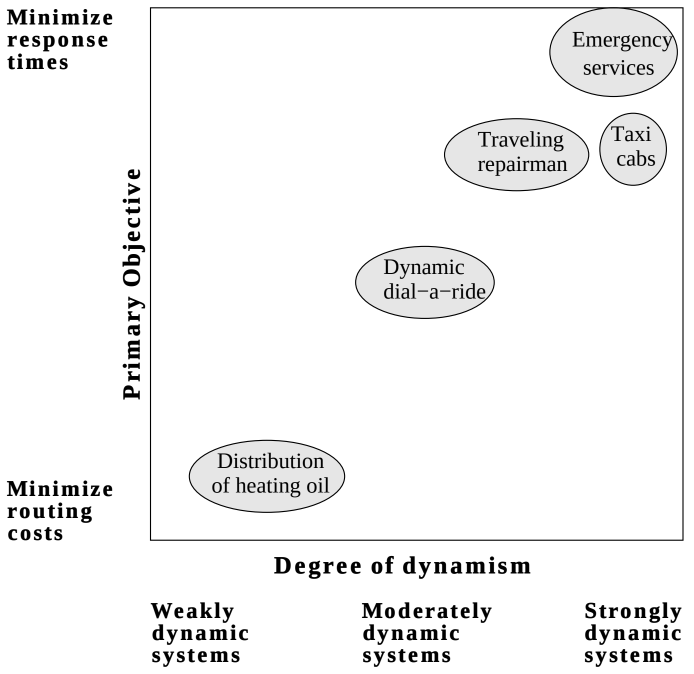


Figure 4.4: Illustration of the framework for classifying dynamic routing problems by their degree of dynamism and their objective.

Problems placed in the upper-right corner of the figure are characterized by a strong dynamism and an objective which seeks to minimize the response time of the system. This would be the case in for instance emergency services like police, fire and ambulances services where almost all customers are immediate requests and the time for the emergency service personnel to arrive at the incident is sought to be minimized. Problems placed in the lower-left corner are characterized by few immediate requests and the wish to minimize the routing costs.

Of course, using such strict labelling often means that some elements get misplaced and therefore cause misunderstandings. Therefore, the relation scheme illustrated in Figure 4.4 should be handled with much care and not misused to try to force a problem into a misplaced position in the figure. The dynamic dial-a-ride problem is one example of an application which is found in many different versions. The so-called *tele-bus* version of the dial-a-ride problem (DARP) is highly likely to be more dynamic than the transportation of the elderly and handicapped people due to the fact that the latter group would usually have planned their trip in advance, while the former group would be more likely to make instant travel decisions.

4.4 Conclusions

In this chapter the degree of dynamism measure was examined for systems both with and without time window constraints. Furthermore, a framework for dynamic vehicle routing systems based on their degree of dynamism was proposed. The framework should prove useful in selecting appropriate models and algorithms according to the dynamic characteristics of the system examined.

Chapter 5

Partially Dynamic Vehicle Routing

In this chapter we further explore the *degree of dynamism* measure by introducing the Partially Dynamic Traveling Repairman Problem (PDTRP). A number of simple online dynamic policies to minimize the routing costs and/or the the waiting time experienced by the customers are described. The results of the computational study indicate that an increase in the dynamic level results in a linear increase in route length for all policies studied. Furthermore, for relatively congested systems, a Nearest Neighbor policy performed uniformly better than the other dispatching rules studied.

We examine the impact of the *degree of dynamism* measure on solution methodology and quality. To give practical meaning to this concept, in section 5.1, we introduce the Partially Dynamic Traveling Repairman Problem (PDTRP). In section 5.2 we discuss how to simulate the PDTRP. The results of the simulation are discussed in section 5.3. Finally, in section 5.4 we discuss how the PDTRP could be extended and then we offer our conclusions in section 5.5.

The work presented in this chapter is also described in Larsen et al. [27].

5.1 The Partially Dynamic Traveling Repairman Problem (PDTRP)

In the Dynamic Traveling Repairman Problem introduced by Bertsimas and Van Ryzin [6] all demands are dynamic, i.e. all customers are immediate request customers. The Partially Dynamic Traveling Repairman Problem (PDTRP) is a variant of this problem involving both advance and immediate request customers. We will define the PDTRP as follows:

- A repairman (or a vehicle/server) travels at constant velocity in a bounded region $\mathcal{A}$ of area A.
- A subset of demands are dynamic. These arrive in time according to a Poisson process with intensity parameter λ. The locations of the demands are independently and uniformly distributed in $\mathcal{A}$.
- Each demand requires an independently and identically distributed amount of on-site service time that becomes known once the service is completed.
- The route is updated only at customer locations. That is, a vehicle cannot change its destination while traveling.
- The objective is to minimize the repairman routing cost.

The PDTRP differs from the DTRP in two major aspects. First, the geographical locations of a subset of the customers to be serviced are already known by the dispatcher before the server leaves the depot. Note also that the service times of advance as well as immediate requests are not known to the dispatcher, until the service at the respective customers is completed.

Second, the problems seek to optimize different objective functions. In a pure dynamic model it makes good sense to seek to minimize the overall system time. This has been defined as the sum of the waiting time and the on-site service time of the demands. However, in a partially dynamic setting the dispatcher may be more interested in minimizing the distance traveled by the repairman. Nevertheless, the immediate request customers should be served in some prioritized order - so that customers calling in late would not be serviced before customers that have been waiting for a relatively long period of time. Such scenarios include residential appliance repair services and courier mail pick up where customers are not offered a specific time window for service. For example, the advance request customers could have their appliances fixed in the morning and the immediate request customers in the afternoon.

5.1.1 Routing Policies

We consider several heuristics originally analyzed by Bertsimas and Van Ryzin [6] for the DTRP. We give below a brief description of these policies:

- **First Come First Serve (FCFS).**
  The demands are served in the order in which they are received by the dispatcher.
- **Stochastic Queue Median (SQM).**
  This is a modification of the FCFS rule where the server travels directly from the median of the service region to the next demand location. After the service has been completed, the server returns to the median and waits for the next demand.
- **Nearest Neighbor (NN).**
  After completing service at one location the server travels to the nearest neighboring demand.
- **Partitioning policy (PART).**
  The service region is partitioned into a number of smaller subregions, n_{subreg}, in which the demands are served using an FCFS policy.

We chose these policies because they include routing cost based policies - NN, waiting time based policies -FCFS, FCFS-SQM, and a mixture of these - PART.

The FCFS policy schedules the advance requests first without considering the locations of the requests. The immediate requests are added to the schedule as they are received by the dispatcher. In scenarios with more than one advance request a more elegant version of the FCFS would take the locations of the advance requests into consideration. A natural way of including this information into the policy is to construct a tour through the advance requests for instance by using the nearest neighbor method.

This gives another version of the FCFS policy which we will refer to as the **Nearest Neighbor - FCFS** (NN-FCFS) policy.

Similarly one could argue that for the PART policy advance requests should be visited according to an NN rule rather than at random within each subregion. Hence, the PART policy examined in the remaining part of the this chapter will visit all requests which were received by the dispatcher at the same time according to an NN rule. However, to customer whose requests for service are received at unique points in time the FCFS rule still applies.

The FCFS-SQM rule has been shown to give asymptotically optimal system time in light traffic. However, from a routing perspective, in environments where the triangle inequality holds, this policy will always produce longer distances than FCFS. It should be noted that the number of subregions used in the PART policy parameterizes this - i.e. the number of subregions should be found by optimization. We next consider the behavior of the above policies under conditions where the number of dynamic requests vary relatively to the number of static requests.

5.2 Simulation of the PDTRP

We have simulated the PDTRP to examine the empirical behavior of certain system parameters and solution methodologies. Specifically, we have focused on the impact of the degree of dynamism on the distance traveled and the relative performance of the heuristics described above. To explore these issues, we have selected the geographic locations of the customers to be uniformly distributed in the square [0,10000] x [0,10000], with the depot located at the median, i.e. at (5000,5000). The vehicle speed was set at 40 km/h. Note that we used a scaling factor of 100 units per km, i.e. the service region is equivalent to 10 km x 10 km. The service times were generated according to the log-normal distribution $LN(0.8777, 0.6647)$ (cf. section 3.3 for details). This means that the average service time was 3 minutes, while the variance was 5. These values were chosen to resemble the service times of pick-ups and deliveries in long-distance courier mail services.

5.2.1 Generation of Datasets

In order to be able to examine scenarios with varying degrees of dynamism comprehensive data material was generated. In the following the process for generating the datasets will be described.

Prior to the actual data generation process the expected number of customers, $E\{n_{tot}\}$, was chosen. This parameter should be seen as the number of customers we expect to generate on average over the entire dataset. The generation process of each single instance could be seen as a four step process:

1. The number of advance request customers, n_{adv}, is randomly generated in the interval $[0, E\{n_{tot}\}]$.
2. The geographic locations and the service times of n_{adv} advanced request customers are generated.
3. A stochastic number of immediate request customers, n_{imm}, is created using a conventional Poisson process with the length of the time horizon, T, equal to eight hours. The intensity parameter of the Poisson process, λ, is calculated from the following expression:
$$\lambda = \frac{E\{n_{tot}\} - n_{adv}}{T} \tag{5.1}$$
4. The geographic locations and the service times of n_{imm} immediate request customers are generated.

In order to ensure that scenarios with the same degree of dynamism are represented in the dataset by the same number of instances we continue to generate new instances, until we have exactly n_{rep} instances with $dod = \{0, 5, 10, ..., 95, 100\}\%$. All other instances will be rejected.

The process of generating datasets could be summarized as shown in the algorithm 1. The data generation algorithm was implemented in the ANSI-C programming language.

Three different settings were considered. In `Scenario A` $E\{n_{tot}\}$ was set at 20. In `Scenario B` we let $E\{n_{tot}\} = 30$ and in `Scenario C` $E\{n_{tot}\} = 40$. Given these values, all three scenarios resulted in medium to heavy

Algorithm 1 Generation of datasets

```
Choose $n_{rep}$
Choose $E\{n_{tot}\}$
**repeat**
  Find $n_{adv}$ by sampling $U(0, E\{n_{tot}\})$
  Generate customer data for $n_{adv}$ advance request customers.
  Find $\lambda$ according to equation 5.1.
  **repeat**
    Find the request time for the $i$'th immediate customer by sampling
    $P(\lambda)$.
    Generate customer data for the $i$'th immediate request customer.
  **until** $(time = T)$
  Calculate the *dod*.
  **if** $(dod\ MOD\ 5 = 0)$ & $(|D(dod)| < n_{rep})$ **then**
    Accept instance and increment $|D(dod)|$ by 1.
  **end if**
**until** $(|D(dod)| = n_{rep} \forall i)$
```

traffic situations - with only little idle time for the repairman. For all three scenarios, we used $n_{rep} = 50$, i.e. 50 different instances were generated for $dod = \{0, 5, 10, ..., 95, 100\}\%$. This means that each scenario consisted of 21 x 50 = 1050 instances in total.

In Appendix A an example of a randomly generated data file is shown.

5.3 Computational Results

Our computational experiments are summarized in the figures 5.1 through 5.5. Figures 5.1 through 5.3 depict the average total traveled distance by the repairman as a function of the number of immediate requests for `Scenarios A, B` and `C`, respectively.

In Appendix B the log files and the actual routes of a single instance of the `Scenario B` dataset produced by the NN-FCFS and the PART policies are shown. Immediate requests begin to be generated at 8:00 and new customers are generated according to equation 5.1 at rate $\lambda = 1.125$ requests/hour until the clock turns 16:00. In this case a total of 30 customers were generated, 21 of these were advance requests and the remaining

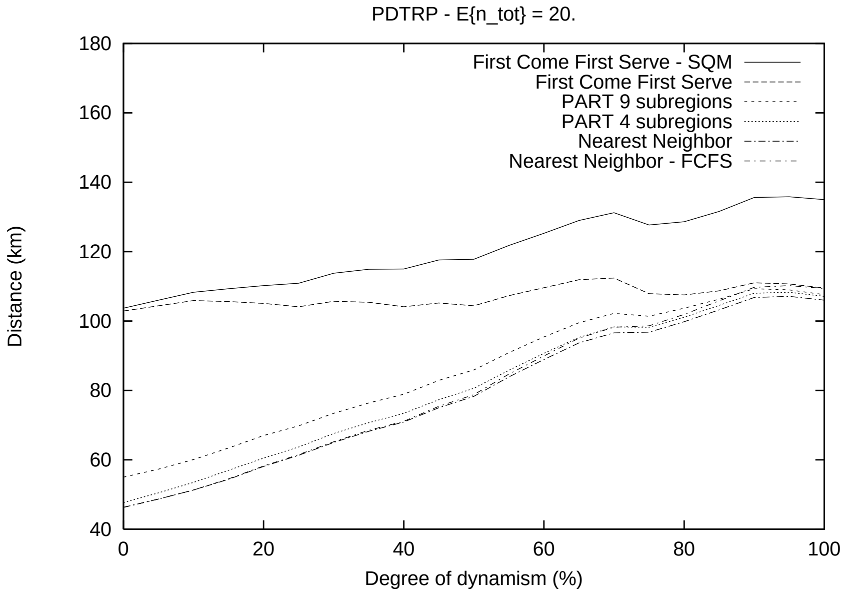


Figure 5.1: `Scenario A`: The distance traveled as a function of the degree of dynamism in the system.

9 were immediate requests (please note that customer number 22 is an immediate customer calling in at 8:00:02 - two seconds after the calling center opens). Using the terminology mentioned above we have $n_{tot} = 30$, $n_{adv} = 21$ and $n_{imm} = 9$. The degree of dynamism then is $dod = \frac{n_{imm}}{n_{tot}} = 0.3$. This means that 30 % of the customers are immediate request customers. The log files tell us that the NN-FCFS policy produces the shortest route with a length of 88.91 km, whereas the PART policy produces a slightly longer route with a distance of 89.61 km. In Figures B.1 and B.2 in Appendix B the actual routes produced by the two policies are shown. The routes are seen to differ quite a lot. Both routes start by serving `Customer 4`. Thereafter, the NN-FCFS policy chooses to go to the nearest customer, in this case `Customer 18` since `Customer 25` has not yet placed its request for service. The PART policy goes to `Customer 11`, since the first customer on the route was located in the upper-left region and the repairman therefore

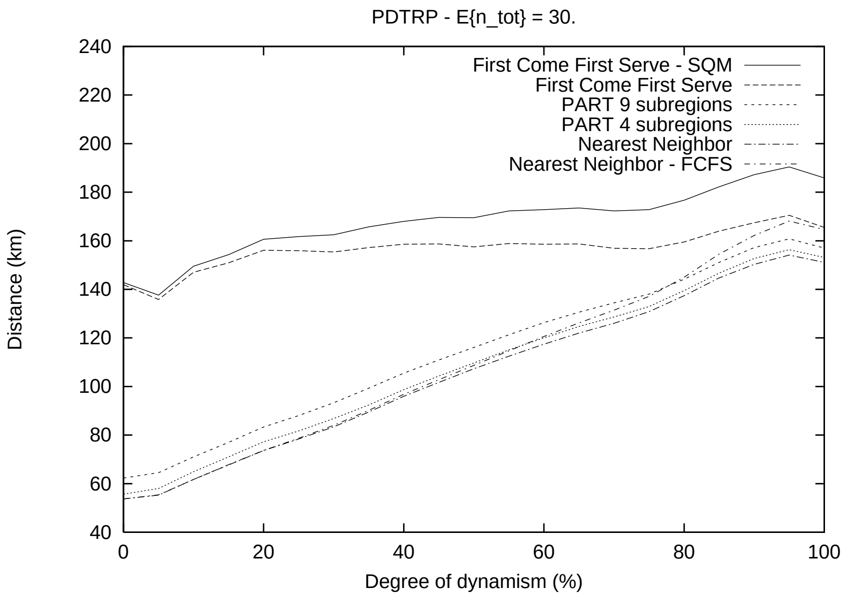


Figure 5.2: `Scenario B`: The distance traveled as a function of the degree of dynamism in the system.

has to serve this subregion, until no more requests are waiting in line to be served. The PART policy then serves the lower-left, the lower-right, the upper-right and then again the upper-left subregions and this goes on until no more requests are waiting to be served.

Figure 5.4 shows the average number of minutes the repairman is busy serving or traveling to customers as a function of the degree of dynamism. The busy time is seen to increase considerably for increasing levels of dynamism.

In Figure 5.5 the distance is shown as the function of the degree of dynamism for four different values of the number of subregions. It can be seen that the best performance is obtained with 4 subregions for this scenario. Generally, one should expect to see good performance with respect to the distance driven when using a small number of subregions, since the few subregions mean that good routes can be formed within each subregion.

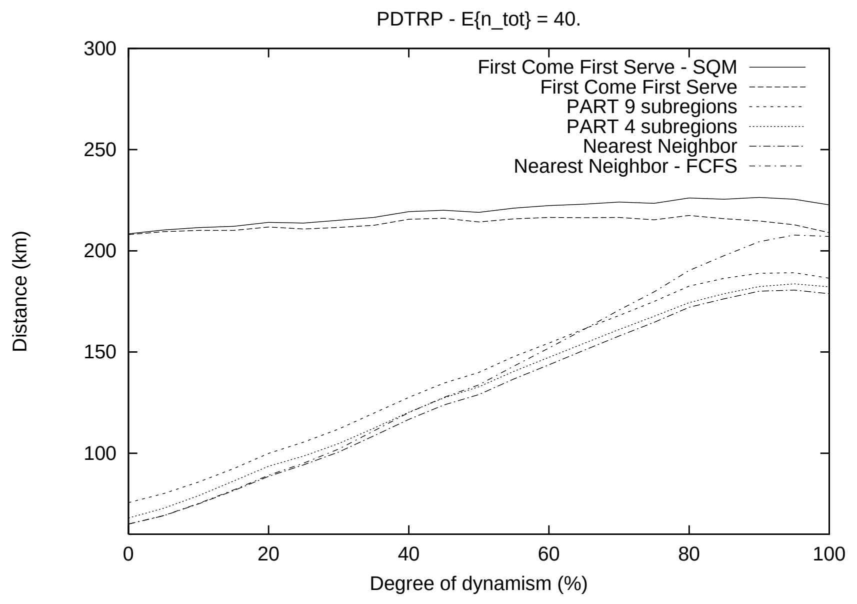


Figure 5.3: `Scenario C`: The distance traveled as a function of the degree of dynamism in the system.

Naturally, such good performance with respect to the distance is obtained at the expense of experiencing poor performance with respect to the waiting time of the customers. This could be explained, as the increase in the distance is due to the fact that the PART policy becomes the FCFS policy as $n_{subreg} \to \infty$, which could also be formulated as the service offered to the customers increases as $n_{subreg} \to \infty$.

The average waiting time of the immediate requests are shown as a function of the degree of dynamism in Figure 5.6 for `Scenario C`. Except for the FCFS and the FCFS-SQM policies the average waiting times are seen to be rather invariant to the level of dynamism. For the FCFS and the FCFS-SQM policies the average waiting time is seen to be considerably higher for weakly dynamic instances than for strongly dynamic instances. This may at first seem a bit surprising but should be explained as follows: The waiting time based policies (the FCFS and the FCFS-SQM) will start by

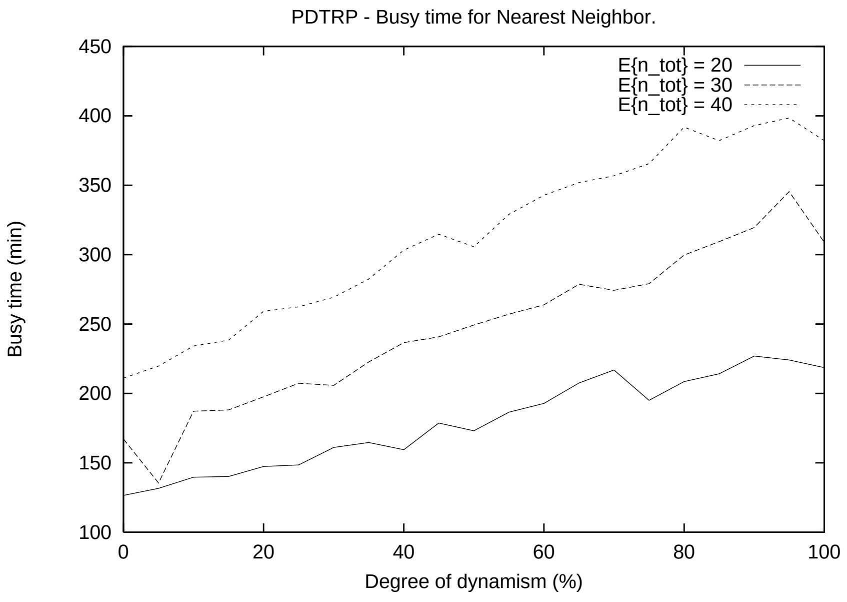


Figure 5.4: The busy time of the repairman for scenarios `A, B` and `C` as a function of the degree of dynamism in the system.

serving the advance request customers and let the (few) immediate request wait until the advance requests have been served. This means that the average waiting time of the immediate requests will be relatively high for instances with few immediate and many advance requests.

5.3.1 Effective Degree of Dynamism

In addition to the three scenarios we also considered a special instance of `Scenario C`. We will refer to this scenario as `Scenario C*`. As for the other scenarios we generated 50 different instances. However, instead of generating the instance using the basic *dod* measure we used the effective degree of the dynamism measure, *edod*, as defined in section 4.1.2. Even though the range of the *edod* measure is the same as the range of the *dod*

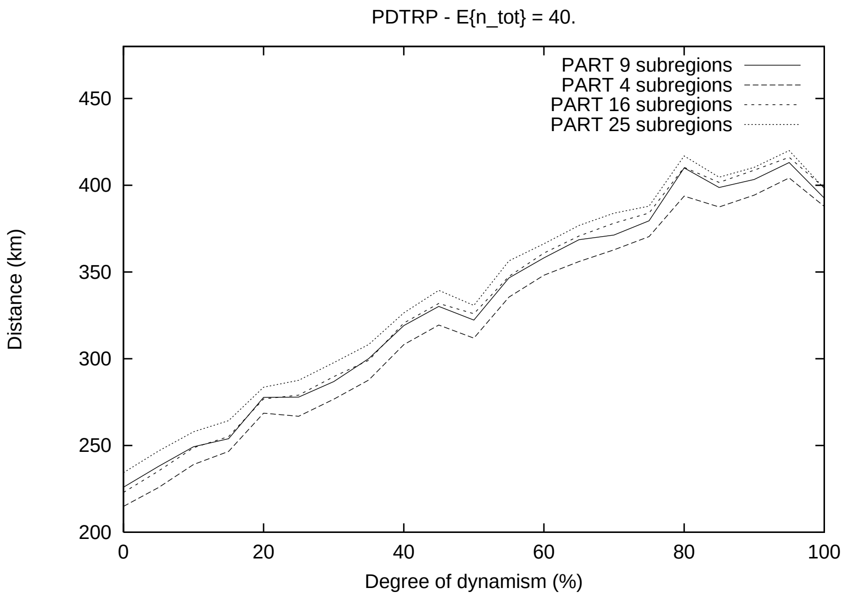


Figure 5.5: `Scenario C`: The distance as a function of the degree of dynamism for the different versions of the PART policy.

measure it is not realistic to generate instances with *edod* $= 100\%$, since this would mean that all requests would have to be received at the very end of the planning horizon. Therefore we chose to generate 50 instances, each with *edod* $= \{0, 3, 6, ..., 57, 60\}\%$ [1]. In Figure 5.7 the distance is shown as a function of the effective degree of the dynamism. The behavior of the distance is seen to be very similar to the behavior of Figure 5.3. However, for instances with *edod* > 50 % the distance seems to decrease for increasing values of *edod*. We see no explanation to this rather unexpected behavior. Judging from Figure 5.7, the *edod* measure does not seem to offer anything special not already captured by the more simple *dod* measure.

[1]During the tests we tried to generate 50 instances each with *edod* $= \{0, 4, 8, ..., 76, 80\}\%$. However, after more than 3000 minutes of computation time on an HP/9000-785 computer only instances with *edod* ≤ 68 % were complete and only 7 instances with *edod* $= 72$ and none with *edod* > 76 % were generated.

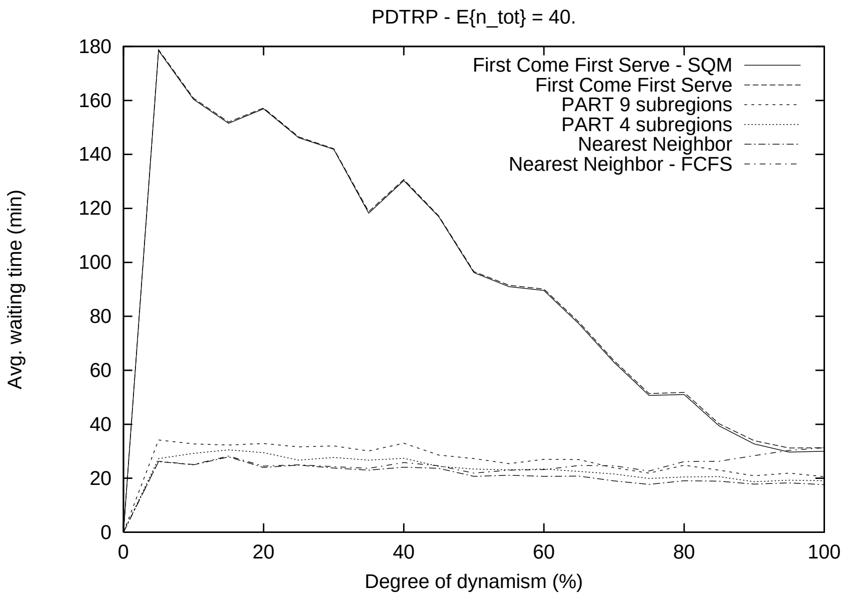


Figure 5.6: `Scenario C`: The average waiting time of the immediate requests as a function of the degree of the dynamism.

5.3.2 General Observations

For all three scenarios one also notes that as the level of dynamism becomes stronger, the behavior of NN increasingly resembles that of FCFS. This is because the presence of larger numbers of immediate request customers spreads their scheduling over time, implicitly decreasing the set of neighbors each customer has at any given point in time. At the higher end of dynamism the customer currently being serviced may only have one neighbor - the next immediate request customer. Hence, as the level of dynamism increases, we observe an expected transition from a geographical to a temporal emphasis. Furthermore, the figures illustrate that the average total distances for the FCFS and the FCFS-SQM policies were almost independent of the degree of dynamism. The difference was larger in `Scenario A` than in `B` and `C` since the repairman had slightly more idle

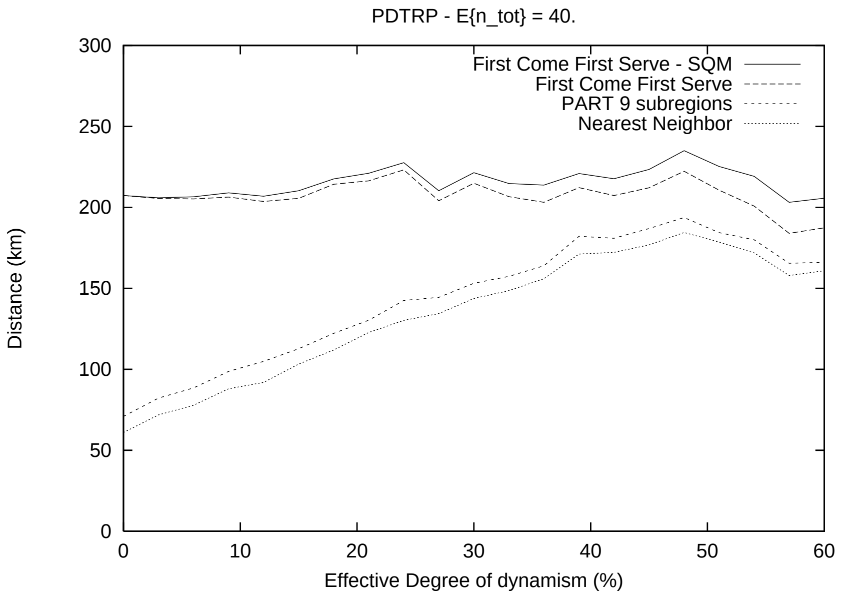


Figure 5.7: `Scenario C`: The distance as a function of the effective degree of dynamism.

time with only 20 expected customers to serve (ref. to Figure 5.4. Finally, the PART policy, based on a mixture of routing costs and waiting time, produced intermediate route lengths.

5.4 Extensions

In this section future research issues for refining the proposed PDTRP will be discussed.

The use of a-priori information on future requests seems to be an interesting issue for further research. However, such information may not be known for the length of the operational horizon or for the whole region. Nevertheless, information may be known for a shorter time period and clustered geographically. A simple algorithm would divide the service region into smaller subregions and the day of operation into time periods. The a-priori information on the arrival intensity of immediate requests could then be used to find and serve the subregion with the highest number of expected requests in the next time period. The basic idea of such an algorithm is outlined in algorithm 2.

Algorithm 2 "Go to the busiest" subregion

```
Choose $\Delta t$
Estimate $\lambda_i$
Initialize $time$
repeat
  for $i = 1$ to $n_{subreg}$ do
    $E\{n_{tot_i}\} = \Delta t \cdot \lambda_i + n_{adv_i}$
  end for
  $sub\_region = \max_i E\{n_{tot_i}\}$
  Serve $sub\_region$
  Update $time$
until ($time \geq T$) and (all requests are served)
```

where

λ_i	denotes the arrival rate of the Poisson process generating the immediate request customers in subregion i.
n_{adv_i}	denotes the number of advance requests in subregion i.
n_{tot_i}	denotes the number of total (advance + immediate) requests in subregion i.
Δt	denotes the time interval to look forward.

Each time a new set of demands has been collected in a region, a policy minimizing the route length could be used. If updates have to be done frequently, a simple policy like the Nearest Neighbor should be used, while in cases with few updates a more computational requiring method like the Traveling Salesman Policy should be used.

Another interesting issue for further research would be to examine other criteria such as the total schedule time. For example, the FCFS-SQM policy forces the repairman to return to a central position, when he is idle with the expectation of decreased waiting times of future requests. This aspect can only be captured by time-oriented objectives.

5.5 Conclusions

In this chapter the Partially Dynamic Traveling Repairman Problem was introduced and a number of different routing policies to minimize routing costs were examined. The empirical study conducted illustrated a linear relationship between the degree of dynamism and the route cost in fairly busy systems. The simulation test runs also suggested the use of a Nearest Neighbor policy over the other policies examined. Finally, an extension of the degree of dynamism measure was briefly examined. However, the computational study did not show any benefits obtained by using the extended measure over the basic one.

Chapter 6

The Capacitated Dynamic Vehicle Routing Problem with Time Windows

In this chapter we examine a dynamic version of the conventional vehicle routing problem in the presence of time window constraints. We will refer to this problem as the Dynamic Vehicle Routing Problem with Time Windows (DVRPTW). The work was originally motivated by the routing problem arising in the situation where oil companies must deliver oil to private households for heating. This problem has often been treated as a static routing problem, since it usually has been assumed that all information is known to the planner at the time the routing is being done. However, changes in the service offered to the customer by most major oil companies imply that customers are offered to be served during the day of operation. Still, the majority of the customers visited during a day of operation are advance request customers who are usually served based on the degree-day measure as described in subsection 1.7.3. However, the experience shows that approximately 10-20 % of the customers served run out of oil and therefore must be served as soon as possible and preferably during the same day. Moreover, in a real-life situation the demand of the customers is subject to stochasticity, since the estimates of the customers' consumption might be subject to strong variations and not be captured adequately by the degree-day measure. The stochasticity of the demand adds to the dynamism of the problem, since the dispatcher must also consider the risk of the vehicles running out of oil and therefore having to return to the depot to replenish. However, in this introductory investigation of this problem the stochasticity of the demand will not be considered. Readers interested in vehicle routing problem subject to stochasticity in the customer demands should consult the references of subsection 2.2.3.

The work presented in this chapter is an extension of the work done by Lund et al. [29]. As mentioned in chapter 4, Lund et al. were the first to consider the degree of problem dynamism and its influence on solution quality. They adapted the well-known insertion heuristic proposed by Solomon [41] for the static vehicle routing problem with time windows (VRPTW) and simulated its behavior for varying dynamic levels. The authors concluded that solution quality, as measured by a ratio of dynamic distance over static distance and capacity utilization of the overall system, was preserved relatively well, when only a limited number of dynamic requests were introduced.

We intend to examine how the insertion heuristic performs in the case of a more general dataset and also examine the effect of *the reaction time* of the immediate requests plays on the overall system performance. Furthermore, two simple strategies for batching the dynamic requests into groups are implemented and the performance of these is compared to the performance of conventional re-optimization whenever new information is received.

The chapter begins with a brief description of the simulation environment and the insertion heuristic implemented by Lund. The kernel of the program is with few exceptions the original one. However, extensions and modifications have been made to the interface and the main simulation control in order to handle the new datasets. These modifications and extensions are briefly described. For a more detailed description of the routing heuristic and the simulation environment the technical report by Lund et al. [29] should be consulted. Section 6.2 discusses the process of generating the datasets and compares the method used in the datasets of this thesis to the one used by Lund. In section 6.3 the computational results are presented and discussed. Finally, in section 6.4 some concluding remarks are made along with some guidelines for further research on this problem.

6.1 Simulation of the DVRPTW

In this section a brief description of the simulation environment implemented by Lund et al. will be given. Furthermore, the routing heuristic and the modifications made to this will be discussed. The structure of the simulator is sketched in Figure 6.1. As can be seen from the figure the simulator essentially consists of two basic modules: A system module and a routing module.

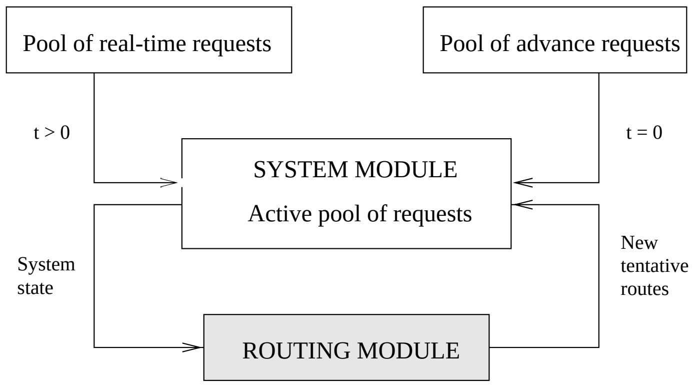


Figure 6.1: The simulator

6.1.1 The System Module

The system module controls the system of m vehicles (the geographial locations, the residual capacities, the states of vehicles etc.), while the routing module solves the current routing problem. Two different types of events can occur during the simulation; 1) a *request event* occurs, whenever a new request for service is received by the dispatchers and 2) a *vehicle event* occurs, whenever the vehicles enter a new state. The simulator works in three different vehicle states; driving, servicing and waiting.

Opposed to the simulation environment presented in chapter 5 for solving the PDTRP, Lund used the fixed-time-increment simulation technique (see Larson [28]) with a one-minute time increment. This means that the system module checks the state of the system each minute. If a request event or a vehicle event has occured, the system module updates the current system state. If a request event has occured at time t, the routing module is activated. The m state vectors $SV(n,t) = (loc_t, cap_t, veh_t), n = 1, 2, ..., m$ characterize the actual dispatching situation for each vehicle with regard to their current geographical location, residual-capacity and the vehicle state.

The routing module now solves the actual routing problem and passes the new solution vector on to the system module. It should be noted that the new solution should be thought of as a tentative solution which is subject to be changed, whenever a new request is received. Naturally, changes can only be made to the unvisited parts of the routes. Also, if a vehicle is already traveling towards the next customer on its route when a new request is received, the customer must proceed and commence the service when it arrives at the customer (assuming that the time window has already opened, otherwise it must wait). This means that the vehicle is not allowed to divert from the next customer, if the vehicle is already traveling towards this customer. This policy corresponds to the policy used in Gendreau et al. [15] (see section 2.4).

The *active pool of requests* holds the current set of requests. Initially, at time $t = 0$ the active pool of requests consists of all advance request customers. During the simulation this active pool of requests is enlarged if a new immediate request for service is received and reduced whenever the on-site service of a request is ended.

Following the ideas of the MORSS algorithm (see section 2.4) Lund stresses the fact that near-term events are more important than long-term events. It is therefore important not to allocate resources to a request that has to be serviced way into the future since other intermediate requests can render this decision suboptimal. The system module takes this in to account by first assigning a vehicle at the latest feasible point in time. I.e. let e_j denote the earliest time for service to begin at customer j and let $t_{i,j}$ denote the travel time between customers i and j, then the system module lets the vehicle wait at customer i until time $e_j - t_{i,j}$. For the sake of completeness the time the service at customer i begins is denoted b_i and the latest time for the service to begin at customer i is denoted l_i.

6.1.2 The Routing Module

Essentially there are two ways to find a new tentative route, if a new service request has occured, namely local or global heuristic procedures. A local procedure only examines possible insertions in the current sequence of stops, while a global procedure solves the current problem from *"scratch"* every time an input revision has occurred. A procedure based on local operations is suboptimal, because of the possibility that appearance of a new request can render the decisions made before the appearance of the new request suboptimal. This fact concerns both the sequence of stops on a particular route and the assignment of vehicles to customers. On the contrary, a global procedure that solves the actual problem from scratch includes the possibility of reassignment and re-sequencing, if necessary. The drawback of using such a global procedure is the computational burden, but relatively this is still a limited amount of time.

The routing module implemented by Lund et al. was a modified version of Solomon's [41] insertion heuristic in which a weighted sum of detour and delay is minimized to identify the best insertion place for each customer. Lund solves the current problem from scratch every time an input revision has occurred by using this insertion heuristic. The original procedure had to be modified in order to take the current state of the system into account (i.e. the geographical location of each vehicle, the residual capacities etc.). Lund notes that operating in a real-time environment makes the notion of depots void, which means that a new criteria for selecting seed customers must be found. Initially, at time $t = 0$ the routes are initialized by finding the customers furthest away from the depot. Hereafter, at time $t > 0$ the seed customer is dynamically selected as the customer that minimizes a weighted combination of the distance to the current vehicle position and the distance to the depot, where the weights are set at 1 and 2, respectively. In this way seed customers that are close to the current vehicle position but far from the depot are selected.

Lund notes that as with a real-life vehicle routing problem the appearance of a new immediate request may imply that the current routing problem becomes infeasible. The problem of infeasibility caused by new requests is particularly relevant, since the current problem is solved from scratch each time new requests are received. This implies that the new immediate request(s) may cause already scheduled requests to become infeasible. However, Lund decided that new immediate requests should not be included in the routes at the expense of already scheduled requests. The problem was solved by using the following two priorities when inserting new requests:

- `Priority 1`: Insert the new immediate request customer in the current solution (i.e. the solution is feasible).

- `Priority 2 (Forced insertion)`: Insert the new immediate request customer in the current solution by minimizing the deviation in time from the original time window of the customer.

This means that a `Priority 2` insertion allows for the routing module to violate the original time windows of the immediate request customers. After a new customer has been forced into the current solution, the customer is given a modified time window based on the new arrival time, b_i, which minimizes the deviation in time from the original time window, i.e. the beginning of the forced customers time window $e_i = b_i$. This new time window is not changed afterwards.

The forced insertion principle sketched above corresponds to the real-life policy for dealing with new immediate requests. In a real-life situation the dispatcher will suggest a new time window, if it was not possible to honour the customer's original request for service.

6.1.3 Modifications of the Routing Module

In the following the additions to and modifications of the original routing module implemented by Lund et al. will be described. All functions were implemented using the ANSI-C programming language. The work of Lund et al. consisted of approximately 4000 lines of code, while the modifications and extensions described in the following consist of approximately 500 lines of code.

Return Trips to the Depot

We allow for the vehicle to go back to the depot in case no further customers are on the current scheduled route. If the vehicle goes back to the depot we assume that the vehicle gets reloaded so that it will have full capacity when/if it leaves the depot later. It should be noted that the time spent on replenishment is not considered. However, this aspect could be included by introducing a ready time for the vehicles.

Batching Strategies

An alternative to re-optimization each time an input update occurs is to consider strategies for collecting a number of immediate request customers into a batch. In the following this type of methods will be referred to as *batching strategies*. The main idea of the batching strategy is to delay the planning of the routes, until either a certain number of requests have been received by the calling center or a certain amount of time has elapsed since the last planning was performed. This way the level of the available information at the time of the re-optimization should be higher than the corresponding level of information when re-optimization is performed immediately after each input update. Naturally, a wide range of possible criteria for deciding when, which and how many requests to collect into a batch exist. In this thesis the following two batching strategies will be considered:

- The *"size-driven"* batching strategy, for which the immediate requests are collected into sets of n requests. Whenever a set has been collected, the insertion heuristic will be called.
- The *"time-driven"* batching strategy is based upon re-optimization at fixed points in time (for instance every 10th minute).

The ideas of the batching strategies are illustrated in Figure 6.2. For the size-driven strategy re-optimization is performed after the receipt of three new immediate requests, while for the time-driven strategy re-optimization is performed at time $t = 0, 1, 2, ..., 5$. One notes that the last re-optimization for the size-driven strategy is performed at the time the call center closes, since no more immediate requests could be received at a later point in time. For both strategies deferring the planning of the routes until more requests have been received (hopefully) provides the planner with more detailed and up-to-date information and therefore might improve the quality of the solution.

The most obvious drawback of applying a simple batching strategy as described above is that in a real-life DVRPTW environment immediate request customers calling in for service must be provided with a time window for the service to commence. However, using a simple batching strategy may mean that it is not possible to inform the customer calling in whether or not the dispatcher will be able to meet her request for service until the next re-optimization has been performed. This would mean that the dispatcher will have to call back to inform the customer, if her request for service will be met. Naturally, this policy will not usually be applicable in a real-life DVRPTW situation for obvious reasons. Therefore, a mix

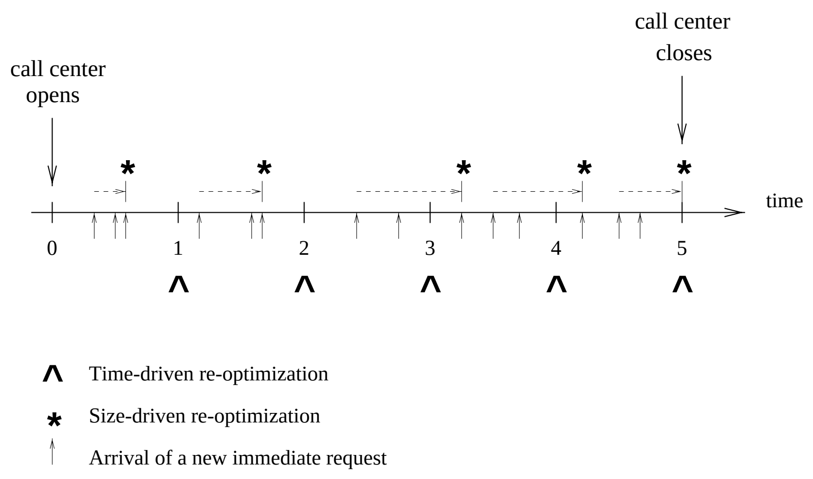

call center
opens
call center
closes
time
0
1
2
3
4
5
Time-driven re-optimization
Size-driven re-optimization
Arrival of a new immediate request

Figure 6.2: Illustration of the size-driven and the time-driven batching strategies.

between real-time insertion of new requests and re-optimization driven by batching strategies should be considered. Whenever a new request is received, the algorithm should try to find a feasible spot to insert the new customer without re-scheduling the customers already in the solution, while the re-optimization of all the customers accepted to be serviced by the dispatcher is to be driven by the batching strategy. Following this policy the dispatcher will instantly be able either to accept or reject new customers calling in for service.

6.2 The Generation of Datasets

In this section the datasets used for testing the implemented insertion heuristic will be described. Firstly, a brief discussion of the method for generating datasets used by Lund et al. [29] will be given. Thereafter, a description of the datasets used for the present analysis of the DVRPTW will be provided.

6.2.1 Original Datasets

In the original work on the DVRPTW by Lund, the `r101` dataset of the well-known Solomon test instances were used in all runs. In order to infer dynamism into the `r101` dataset a number of customers were chosen randomly to be immediate requests. The generation of the times of the customers calling in for service were done by using the time windows of the customers in the `r101` dataset. Lund controlled how early the immediate requests were received compared to the beginning of the time windows, e_i, by using the parameter $\alpha \in [0, 1]$. Let t_i denote the time request i was received, then $t_i = \alpha e_i$. The principle is illustrated in Figure 6.3. A small value of the α parameter meant that the request was received relatively early compared to the beginning of the time window, whereas a large value of the α parameter meant that the request was received relatively late.

Using this method the time of receiving a new request will be distributed in the interval: $[\alpha e_i - \Delta t, \alpha e_i + \Delta t]$, where Δt is a random number of approximately 10 minutes in order to infer some noise into the time variable.

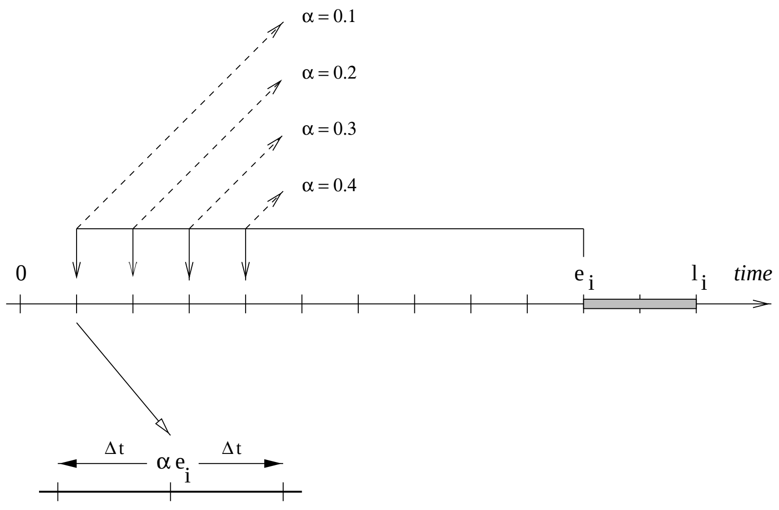


Figure 6.3: Generation of time of request for service.

In Figure 6.4 the time windows for two customers are shown. Using $\alpha = 0.5$ implies that `customer A`'s and `customer B`'s requests for service will be received during the time intervals $[0, 1]$ and $[2.5, 3.5]$ respectively. This means that the time elapsed between the request of `customer A` is received until the time window begins is 0 time units for the worst case, while the corresponding time for `customer B` will be at least 2.5 time units. Hence, using this way of generating the request times for the immediate requests imposes a strong correlation between the time of the receipt of the immediate requests and the time the dispatcher has to respond to these new requests.

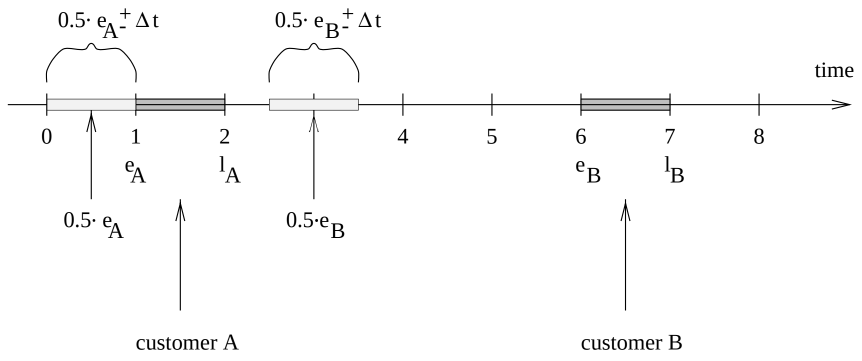


Figure 6.4: Illustration of the dependency between the time the immediate requests are received and the temporal placement of the time windows.

Furthermore, by using the same test instance for all test runs Lund did not test the temporal characteristics of the insertion heuristic very well. Finally, the number of customers was also constant in all runs (100 customers), although in a real-life DVRPTW situation the number of customers will usually be subject to stochasticity.

6.2.2 New Datasets

The datasets used for the present analysis of the DVRPTW were generated as described in section 5.2.1. The number of expected customers, $E\{n_{tot}\}$, was chosen to be 100 as in Solomon's `r101` dataset. The geographical locations of the customers were uniformly distributed in the square [0,100] x [0,100], with the depot located at the median, i.e. at (50,50). The vehicle speed was set at 60 km/h, which means that 1 geographical unit could be driven in 1 minute. The immediate requests were assumed to be able to call in for service during opening hours begining at 8:00 and ending at 16:00. I.e. the time horizon for receiving new calls was 8 hours, $T = 480$ minutes. The depot is assumed to open at 8:00, whereas the closing time for the depot is not fixed, as we allow for the late immediate requests to be served and for the vehicle to return to the depot. It is assumed that the dispatcher does not have the authority to reject customers - or in other words all customers must be served.

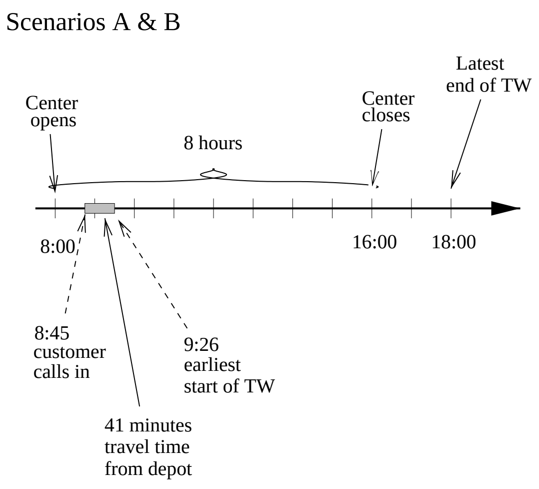
As the issue of stochasticity in the service times was not considered in the original insertion heuristic, it was decided to use a constant service time of 10 minutes per customer. The demand of the customers was set at 10 units, while the capacity of the vehicles was set to 200 units. This implies that 20 customers could be visited before going back to the depot to replenish. The maximum number of vehicles was set at 25, although this restriction never came into play.

The time windows for the advance request customers were generated so that the latest possible time for the time window to close would be $T +$ 120 minutes. I.e. the earliest and the latest possible times would then be 8:00 and 18:00 respectively, when $T = 480$ minutes.

Three different scenarios, `Scenarios A`, `B` and `C` were generated. Below a brief description of the scenarios is given.

For `Scenarios A` and `B` the time windows of the immediate request customers open at the time the request is received by, the dispatcher plus a default travel time from the depot to the customers' geographical location. For `Scenario C` all time windows of the immediate request customers open at the time when the request is received plus a random time of not less than 60 minutes which accounts for a "minimum reaction time" for the dispatcher to include the new request on a route. This policy is consistent with the policies of long-distance courier mail service providers. Figure 6.5 illustrates the temporal relations of the calling times and the time windows for the immediate requests of `Scenarios A`, `B` and `C` respectively.

`Scenario A` consisted of customers with relatively narrow time windows, whereas `Scenarios B` and `C` consisted of customers with wider time windows. All scenarios consisted of 100 instances with *dod* = {0, 5, 10, ..., 100} % which means that each scenario consisted of a total of 2100 instances.



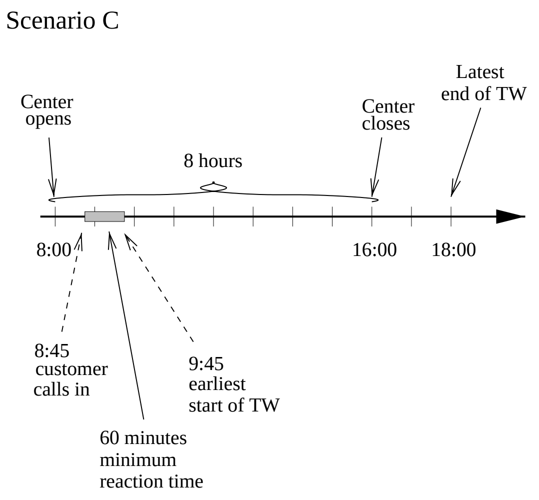


Figure 6.5: Time line illustrating the temporal relations between the calling times and the time windows of the immediate request customers.

Scenario A

The minimum and the maximum length of the time windows was set at 30 and 60 minutes respectively. This resulted in time windows with an average length of 44.8 minutes. The average number of customers was 100.0. The average value of the reaction times denoted $\overline{r}$ was 198.2 minutes, meaning that the dispatcher had well over three hours in average to initiate service of the customers before the time windows close. Remembering that the reaction time is defined as the time elapsed from the immediate request is recevied by the dispatcher until the time window closes, the $\overline{r}$ might seem surprisingly high, as the average length of the time windows is only 44.8 minutes. However, the high value is caused by the reaction times of the advance requests as these can be subject to a strong variation, since the reaction time is calculated as the time elapsed from the beginning of the day of operation until the time windows close.

Scenario B

The minimum and the maximum length of the time windows were set at 60 and 180 minutes respectively. This resulted in time windows with an average length of 114.5 minutes. The average number of customers was 100.1. For this scenario $\overline{r} = 260.6$ minutes.

Scenario C

Similar to `Scenario B` the minimum and the maximum length of the time windows were set at 60 and 180 minutes respectively. This resulted in time windows with an average length of 106.1 minutes. The average number of customers was 100.1. For this scenario $\overline{r} = 323.2$ minutes.

6.3 Computational Results

In this section the results obtained during the simulation will be presented. The discussion is divided into three sections. First, the performance of the heuristic with respect to the distance, lateness and the number of forced customers will be discussed. Next, the issue of reaction time of the customers will be examined. Thereafter, the performance of the batching strategies will be discussed. Finally, the performance as a function of the effective degree of dynamism will be analysed.

6.3.1 Distance, Lateness and Forced Customers

In the following the performance with respect to the distance, the lateness and the number of forced customers will be examined.

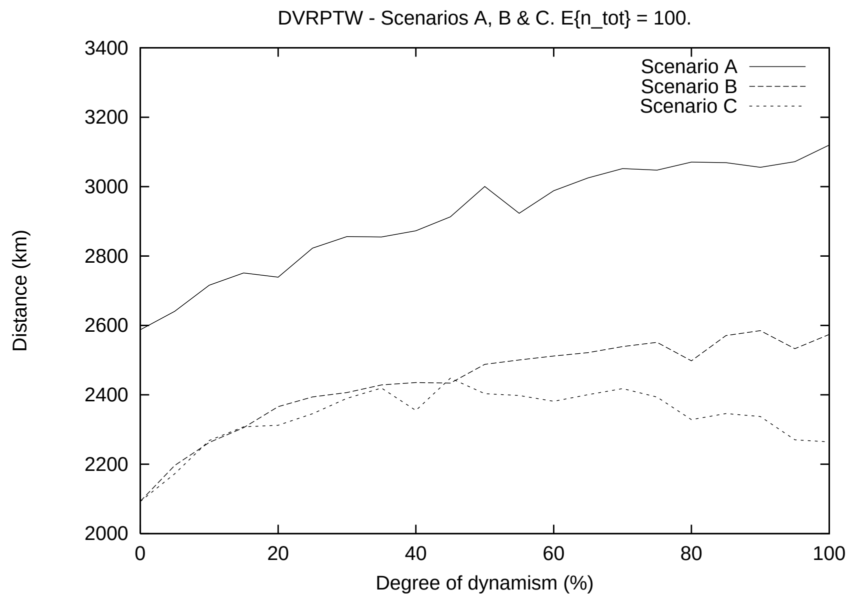


Figure 6.6: The distance traveled as a function of the degree of dynamism.

In Figure 6.6 the averages of the total distances traveled by the vehicles are shown as a function of the degree of dynamism.

For `Scenarios A` and `B` the distance increases with an increasing level of dynamism. Similar to the empirical analysis of the Partially Dynamic Traveling Repairman Problem (PDTRP) examined in chapter 5 the average distance traveled is seen to increase for an increasing level of dynamism. The average distances of `Scenario A` are seen to be considerably longer

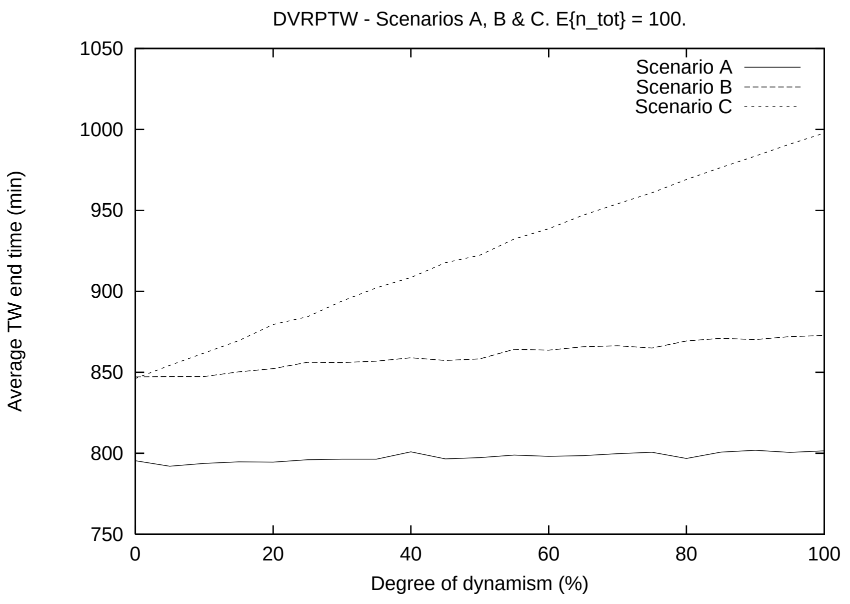


Figure 6.7: The average of the ending time of the time windows.

than the corresponding distances of `Scenario B`. This is due to the fact that the time windows are relatively narrow in `Scenario A` compared to `Scenario B`. The narrow time windows imply that the vehicles may have to visit the customers in an order that might be sub-optimal from the perspective of a distance based objective in order to maintain a solution that is temporally feasible.

The average distance traveled by the vehicles for `Scenario C` seems to reach its maximum value for systems with a moderate degree of dynamism, whereas for systems with a strong degree of dynamism the average distance decreases for increasing degrees of dynamism. This result might seem a bit surprising when compared to `Scenarios A` and `B`. However, in Figure 6.7 the average time for the time windows to end is shown as a function of the degree of dynamism. For `Scenario C` it is easy to see that the time windows end at a much later point in time. This implies that the dispatcher on average will have more time available to initiate service before the time windows end for strong dynamic systems as these are less temporally restrictive.

In Figure 6.8 the average lateness of the vehicles is shown as a function of the degree of dynamism. The lateness is seen to be very moderate for all three scenarios for instances with low levels of dynamism. The lateness is seen to increase considerably for systems with 100 % dynamism for `Scenario A`. The lateness of `Scenario A` is seen to be higher than for `Scenarios B` and `C`, since the time windows are wider for these scenarios.

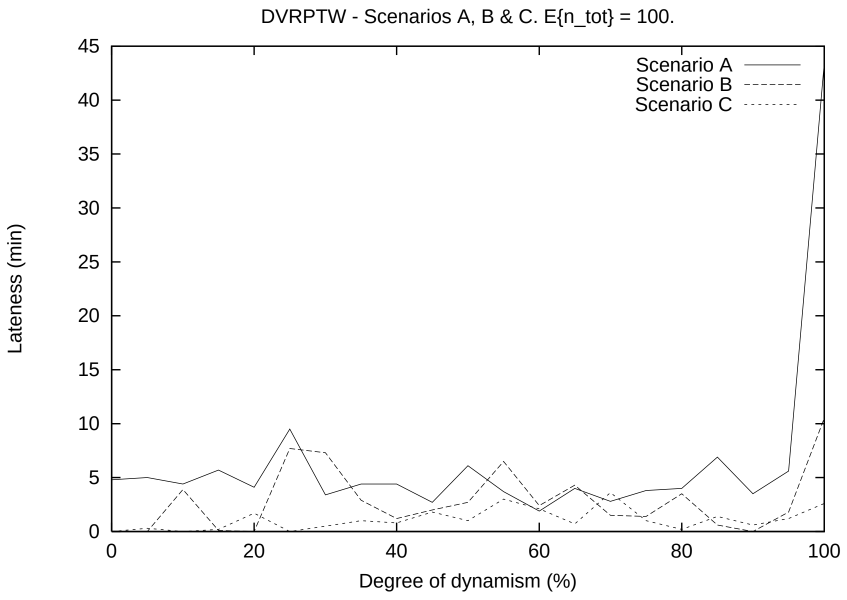


Figure 6.8: The average of the total lateness as a function of the degree of dynamism.

In Figure 6.9 the number of forced customers are shown as a function of the degree of dynamism. As in the case with the lateness, the number of customers that has to be forced into the solution is relatively low for systems with a low degree of dynamism whereas for `Scenario A` a relatively large number of customers have to be forced into the solution for systems with

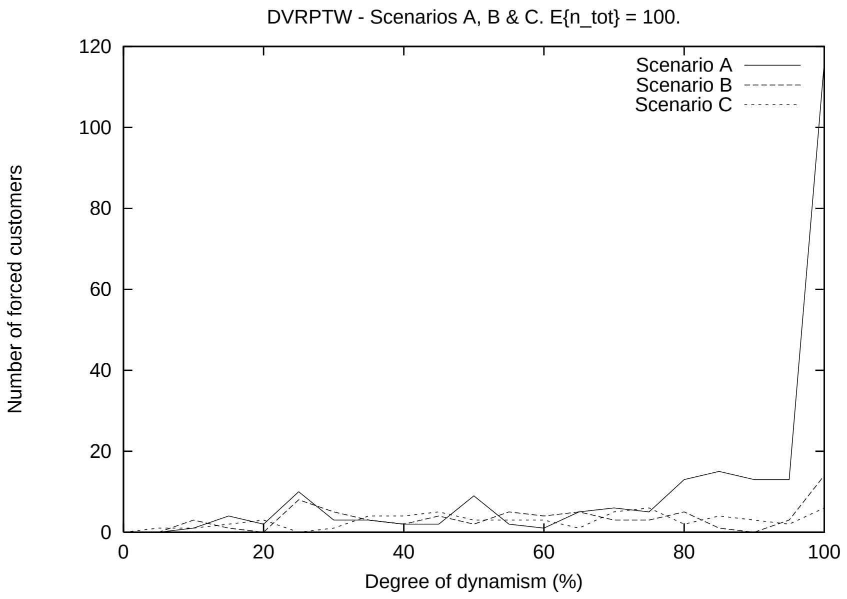


Figure 6.9: The average of the total number of forced customers as a function of the degree of dynamism.

a strong degree of dynamism as this scenario is more temporally restricted than the other two scenarios.

6.3.2 Reaction Time

In Figure 6.10, the average distance traveled by the vehicles is shown as a function of the average reaction times of the customers. For `Scenarios A` and `B` the performance is as one might expect seen to be inversely proportional to the average reaction time of the system. I.e. instances with low reaction time perform relatively poorly compared to systems with a relatively high reaction time. However, for `Scenario C` the average distance is seen to reach its maximum value for instances with a moderate reaction time whereas the performance is seen to be better for instances with a low reaction time. The explanation to this behavior corresponds to the discussion of the performance of `Scenario C` given above. That is, the average distance is relatively low due to the relatively long time horizon for serving the requests. This is verified by Figure 6.11 in which the average reaction time is shown as a function of the degree of dynamism. This figure shows that the reaction time of `Scenario C` decreases at a much slower pace compared to the decreases for `Scenarios A` and `B` for increasing values of the degree of dynamism.

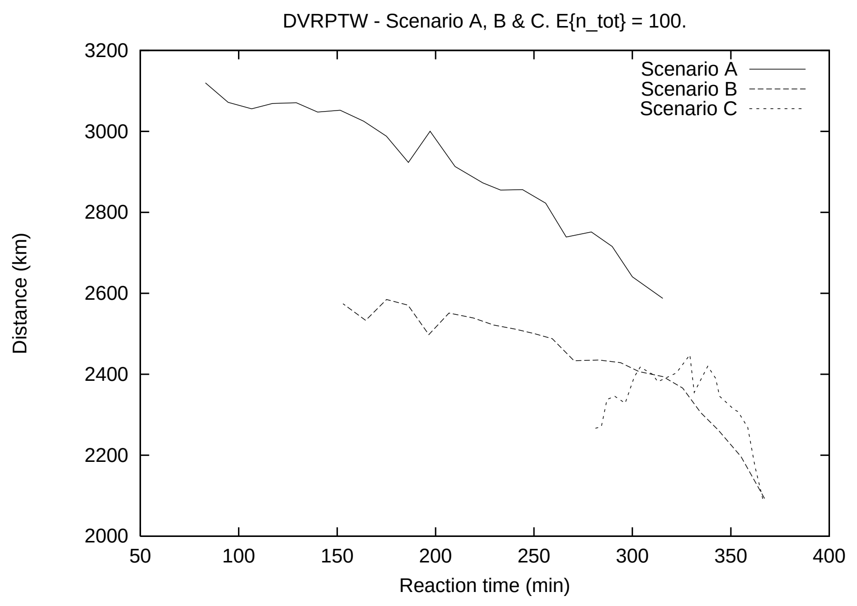


Figure 6.10: The distance traveled by the vehicles as a function of the reaction time.

6.3.3 Batching Strategies

In the following the performance of the batching strategies will be discussed. For the size-driven batching strategy re-optimization was performed each time a set of three immediate requests were received, whereas

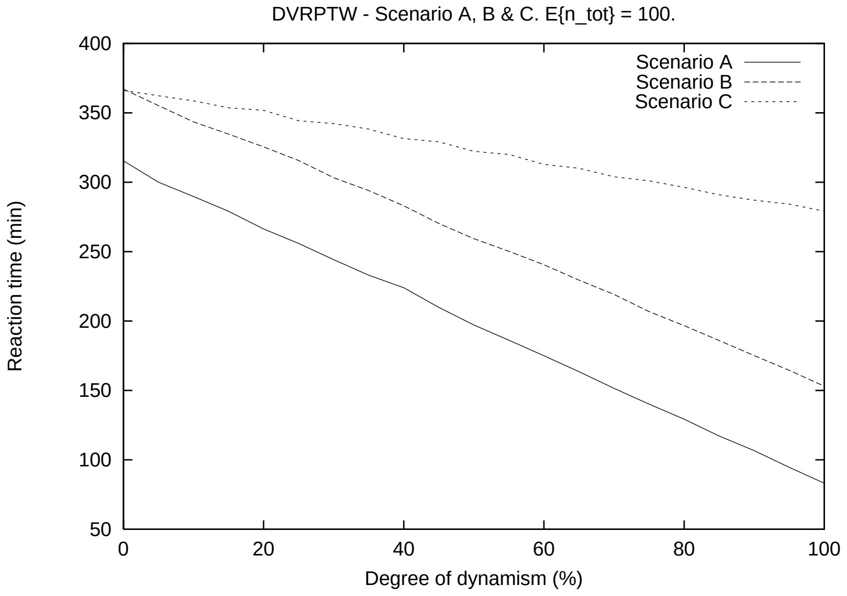


Figure 6.11: The average reaction time as a function of the degree of dynamism.

for the time-oriented batching strategy re-optimization was performed each 10th minute.

In Figure 6.12, the average distance for `Scenarios A` and `B` is shown as a function of the degree of dynamism when using the size-driven and the time-driven batching strategies relative to the pure re-optimization strategy. I.e. the average distance of the pure re-optimization strategy is plotted as index 100.

As can be seen from the figure only the size-based strategy seems to perform marginally better than the pure re-optimization strategy for instances with a low degree of dynamism and only for `Scenario B`. This slightly better performance could be explained by examining Figure 6.13 in which the lateness is shown as a function of the degree of dynamism for all three strategies in `Scenario B`. The lateness of the size-driven policy is seen to be

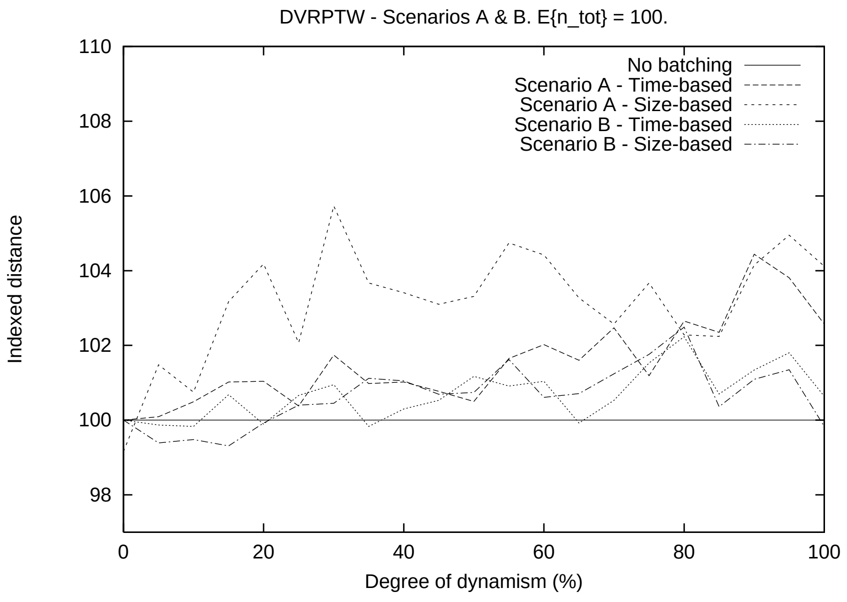


Figure 6.12: The average distance traveled as a function of the degree of dynamism for the size-driven and the time-driven batching relative to the pure re-optimization strategy.

very high compared to the pure re-optimization strategy for systems with a low level of dynamism. The reason for this behavior is that re-optimization is performed after the receipt of three immediate requests implying that for an instance with only 10 immediate requests re-optimization will be performed each $\frac{480}{10} = 48$ minutes on the average[1]. This behavior is verified in Figure 6.14, which shows the average lateness for the three strategies for `Scenario C`. Again, the size-driven strategy results in a relatively high lateness for systems with a low degree of dynamism. One also notes that the lateness on average is much lower for `Scenario C` compared to `Scenario B`, which is due to the wider time windows for the `Scenario C` instances.

[1] Assuming that the time between the recept of each immediate request is equi-distant - which of course is not the case since the immediate requests are generated by using a Poisson process.

For all other levels of dynamism the re-optimization strategy is seen to give the best performance with respect to the distance traveled.

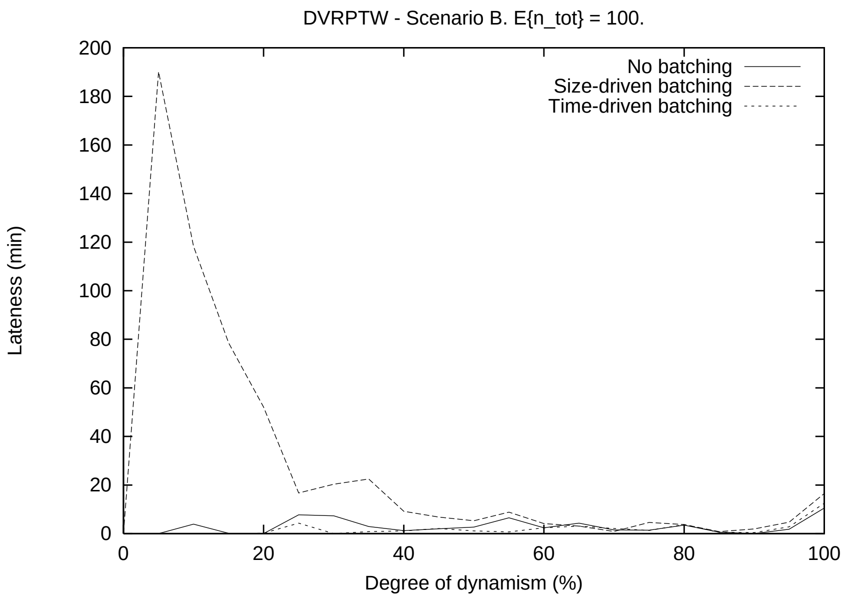


Figure 6.13: The average lateness as a function of the degree of dynamism for the size-driven and the time-driven batching strategies as well as for pure re-optimization.

In Figure 6.15 the average distance for `Scenario C` is shown for the two batching strategy approaches relative to the pure re-optimization strategy. The size-based strategy seems to improve the performance by up to 2 % for systems with degrees of dynamism below 40 %. This improvement is obtained, because the relatively wide time windows and the long reaction time of this scenario mean that it is easier to ensure the temporal feasibility of the routes and that shorter routes then can be constructed. However, as it was the case for `Scenario B`, the improved distance is obtained at the expense of increased lateness as shown in Figure 6.13.

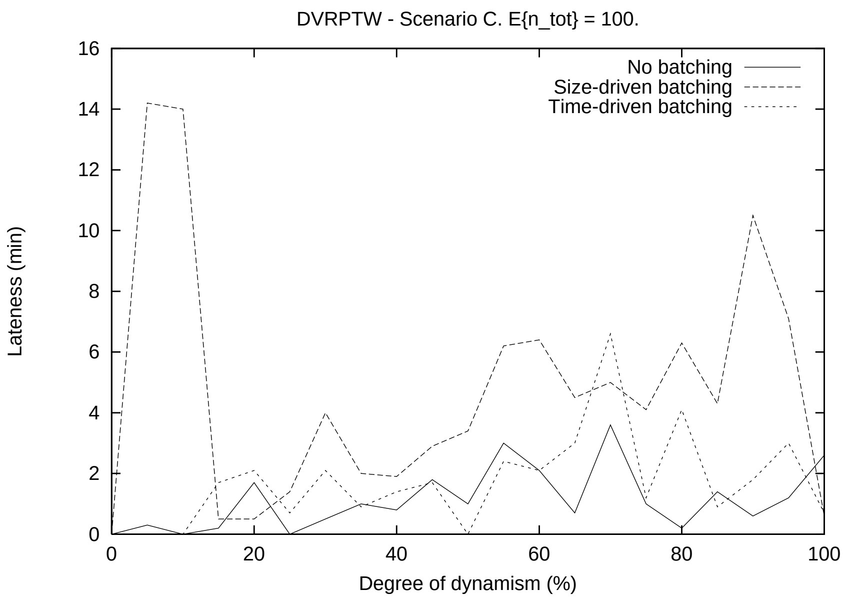


Figure 6.14: The average lateness as a function of the degree of dynamism for the size-driven and the time-driven batching strategies as well as for pure re-optimization.

6.3.4 The Effective Degree of Dynamism

In addition to the three scenarios described above an alternative scenario was generated in order to examine the measure referred to as the effective degree of dynamism for systems with time windows, $edod\!-\!tw$, as defined in section 4.2.1. This alternative scenario will be refered to as `Scenario B*`, since the parameters are the same as for `Scenario B`. The difference is that we generated 100 different instances for $edod - tw = \{15, 18, 21, ..., 57\}\%$. This means that `Scenario B*` consisted of 15 x 100 = 1500 instances in total. We did attempt to generate instances with $edod - tw > 57$, but did not succeed. However, it was indeed possible to generate instances with $edod - tw < 15$. This illustrates the shortcomings of the $edod - tw$ measure, since the range is somewhat more blurred than the range of the

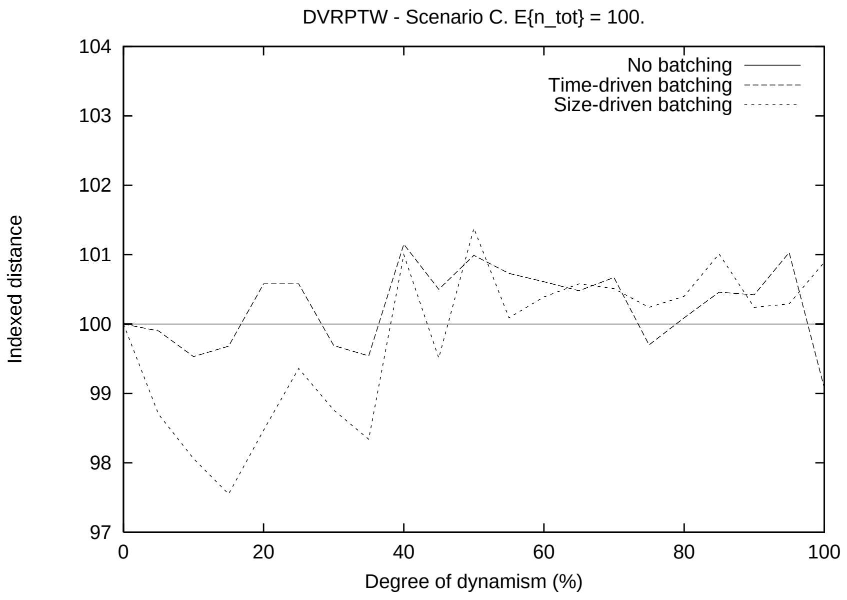


Figure 6.15: The average distance traveled as a function of the degree of dynamism for the size-driven and the time-driven batching relative to the pure re-optimization strategy.

basic *dod* measure which simply spans the interval [0,100] %. It is also easier to relate to the basic *dod* measure than to the $edod - tw$ measure.

In Figures 6.16 and 6.17 the average distance traveled by the vehicles and the average lateness are shown as a function of $edod - tw$. Firstly, it is noted that the distance increases for increasing values of $edod - tw$ for all three policies used. However, when $edod - tw$ reaches approximately 45 % the distance does not seem to increase any further for larger values of $edod - tw$. We do not see any logical explanation to this behavior. The size-driven batching policy is seen to give a slightly better performance than the pure re-optimization and the time-driven batching policies. This behaviour is caused by the lower frequence in re-optimization, which could be seen from Figure 6.17 in which the lateness of the size-driven batching policy is seen to be considerably higher for $edod - tw < 45\%$. The lateness

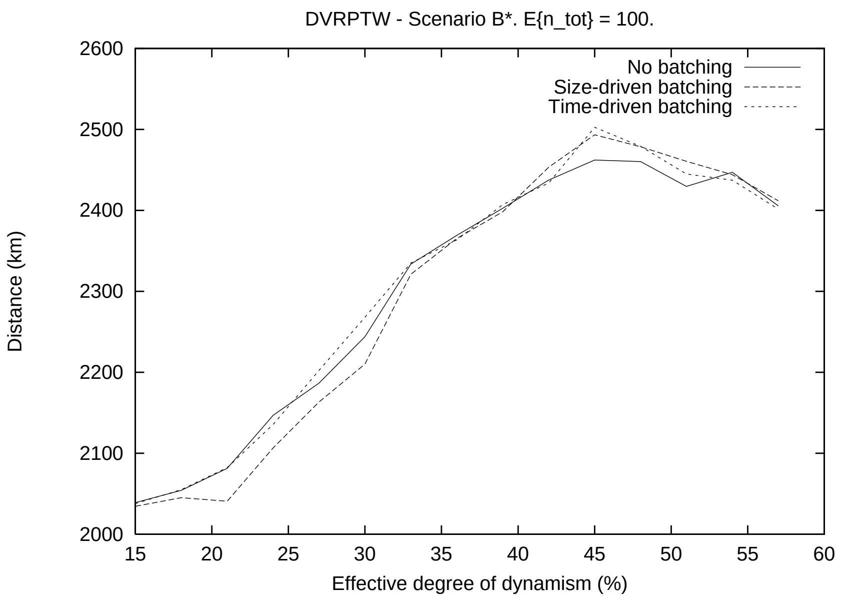


Figure 6.16: The distance traveled by the vehicles as a function of the effective degree of dynamism ($edod - tw$).

is seen to increase considerably for systems with $edod - tw$ greater than approximately 45 %. This gives us the impression that the systems with a very high *dod* value cover a wide range of the *edod* measure. In other words, systems with $dod > 90$ % might have $edod - tw$ values in the interval [40,60]. A comparison of Figures 6.16 and 6.17 to the figures previously described in this chapter does not seem to provide any new information on the structure of the scenario being examined.

6.4 Conclusions

The performance of a dynamic implementation of the well-known insertion heuristic proposed by Solomon was tested on various sets of test data.

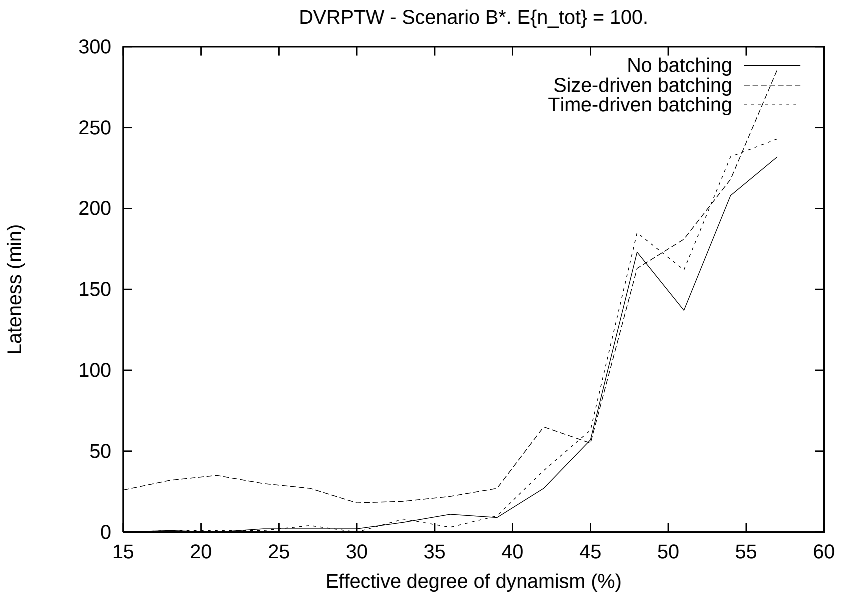


Figure 6.17: The average lateness as a function of the effective degree of dynamism $(edod - tw)$.

The average distance driven by the vehicles turned out to be highly dependent on both the degree of dynamism of the system in question and the structure of the data. Assuming that the distribution of the time windows is independent of the degree of the dynamism, the results indicated that the performance with respect to the distance increases for increasing values of the degree of dynamism.

The lateness as well as the closely related measure of how many customers have to be forced into the solution was shown to increase steeply for systems with a strong degree of dynamism.

Based on the above empirical analysis the batching strategies did not improve the performance of the heuristic. In a few cases the size-based strategy did improve the performance with respect to the distance driven by the vehicles. However, these improvements were achieved at the expense of a considerable increase in the lateness. Still, we believe that batching strategies might turn out to be a valuable approach if an improvement heuristic is added to the routing module so that at times when the system is running idle, the system will use the time to improve the solution found by the insertion heuristic.

A large number of other test runs were performed during the implementation of this heuristic and all data sets showed the same behavior as described in the previous section. This verifies the findings of this chapter.

Finally, the effective degree of dynamism measure for systems with time windows was examined. The conclusion to this extension of the basic degree of dynamism measure was that the extended measure did not provide more information than the more intuitive basic measure.

Chapter 7

The A-Priori Dynamic Traveling Salesman Problem with Time Windows

In this chapter we propose a general formulation of the dynamic version of the Traveling Salesman Problem with Time Windows (DTSPTW) including both advanced request and immediate request customers. The DTSPTW formulation is extended to include low level a-priori information based on knowledge of the arrival intensities of the service regions subregions. A set of simple on-line routing policies taking this a-priori information into consideration are introduced. The proposed policies are tested on a set of randomly generated test-instances motivated by the problem faced when picking up and delivering long-distance courier mail and packages.

The chapter is organized as follows: In section 7.1 the DTSPTW is formulated. Section 7.2 describes the on-line algorithm developed for solving the DTSPTW. In section 7.3 the DTSPTW model is extended to include a-priori information on future requests. Section 7.4 describes the generation of the test data used for evaluating the policies proposed. The results obtained are discussed in section 7.5. In section 7.6 we give a brief discussion on issues which seem to be interesting for further research. Finally, in section 7.7 the conclusions are summed up.

7.1 DTSPTW - Problem Formulation

In this section the Dynamic Traveling Salesman Problem with Time Windows (DTSPTW) is defined. Consider the service region, $\mathcal{A}$, in which customers are to be served. The customers can be divided into two groups; 1) advance request customers who are known by the dispatcher before she begins designing the routes and 2) immediate request customers who appear in real-time. The service of each customer must begin within a given time interval also known as the time window. In this chapter we examine a TSP with so-called soft time windows (TSPTW). This means that the time windows may be violated, but a penalty is then added to the objective function.

The service region, $\mathcal{A}$, is assumed to be a square and is divided into n_{subreg} equally sized subregions (see the example shown in Figure 7.1). This assumption was only made for the sake of convenience during the presentation, as the proposed models also apply to a non-square service region and to subregions of different size. A set of idle points, $\mathcal{IP}$, is defined. Each idle point (IP) serves as a possible "resting location" (as a taxi rank in taxi cab services) for the vehicle and its driver when the vehicle is idle. Idle point i will be refered to as IP_i. It should be noted that the idle points and the subregions are not necessarily related in that some subregions may have multiple IP's while other subregions may not have any IP's at all.

7.1.1 Advance Request Customers

The number of advance request customers is denoted n_{adv}. The earliest time for service to begin at customer i is denoted e_i, while the latest time for service to begin is denoted l_i. The time interval $[e_i, l_i]$ will be referred to as the time window of the i'th customer. The actual on-site service time is unknown to the dispatcher until the service is ended. However, the on-site service times are assumed to be distributed according to a log-normal distribution with parameters α and β. The log-normal distribution has been chosen, because the on-site service times in a real-life distribution context are assumed to have the shape of the log-normal distribution (refer to Figure 3.5). The time the service begins at customer i is denoted b_i. We assume that $b_i \geq e_i$ for all advance as well as immediate request customers, i.e. the service cannot begin before the time window opens. In case the vehicle would arrive before e_i, the vehicle is assumed to wait at its current position before going to the next customer. The advance request customers are represented by the black nodes in Figure 7.1.

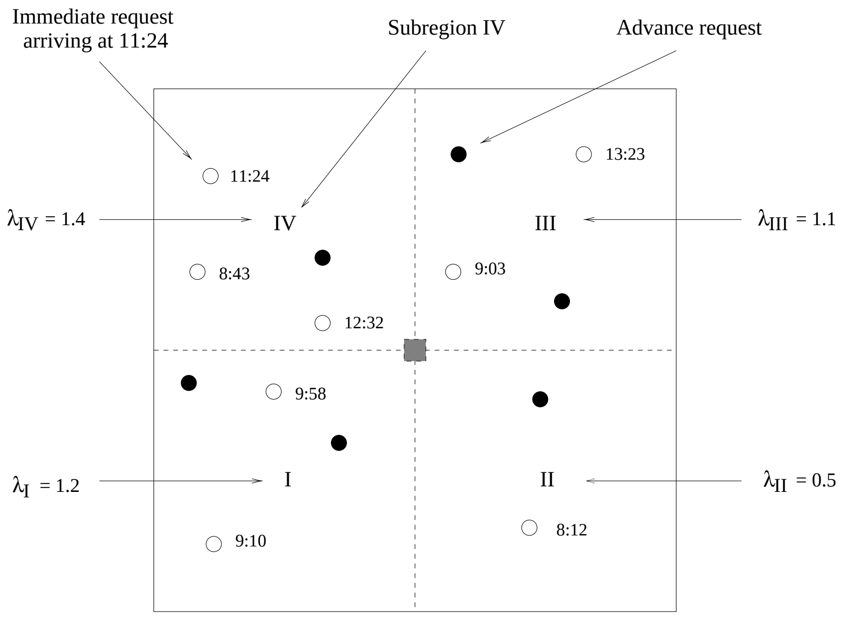


Figure 7.1: A simple example of a Dynamic Traveling Salesman Problem (DTSP) with 4 subregions with Poisson arrival intensities as shown in the figure. The scenario consists of 6 advance request customers (black nodes) and 8 immediate request customers (white nodes).

7.1.2 Immediate Request Customers

During the day of operation a stochastic number of customers call in for service. These customers are refered to as immediate request customers. We assume that the immediate request customers in subregion i arrive according to a Poisson process with the intensity λ_i. The time of the receipt of i'th immediate request is denoted, t_i. In Figure 7.1 the t_i's are indicated just next to the white nodes representing the immediate request customers. The same notation is used for the immediate request customers as used above for the advance request customers for the on-site serviced times as well as for the time windows. Note that the time window of an immediate request customer can either open at the time of the receipt of the call, i.e. $e_i = t_i$, or it can open at a later point in time, i.e., $e_i > t_i$.

7.1.3 Cost Function

Our goal is to find a cost optimal route (or at least a high quality route) for the vehicle to follow so that all customers are served within the specified time windows. Due to the dynamic nature of the problem the solution method will have to be an on-line solution method that works in real-time. The cost function consists of two parts. The first part represents the distribution costs, i.e. the physical costs of serving the customers. The cost of traveling from customer i to customer j is denoted c_{ij}.

The second part represents the penalty imposed in case the time windows are violated in the solution. The penalty, $\phi(i)$, for violating the time window for customer i is defined as:

$$\phi(i) = \begin{cases} \gamma(b_i - l_i), & when\ b_i > l_i \\ 0, & otherwise \end{cases} \tag{7.1}$$

where γ is a parameter. The $\gamma(b_i - l_i)$ part of the objective function can be referred to as a lateness penalty.

At time t the total number of customers in the system waiting to be served is denoted $n_{tot}(t)$, and is made up by the number of unserved advanced requests and by unserved immediate requests which have called in for service at time $t' < t$.

The cost function at time t is denoted, $cs(x,t)$, and can then be written as:

$$cs(x,t) = \theta \sum_{i,j}^{n_{tot}(t)} c_{ij}x_{ij} + \gamma \sum_{i=1}^{n_{tot}(t)} max(0, (b_i - l_i)) \tag{7.2}$$

where θ is a parameter and x_{ij} is the decision variables which are 1, if the vehicle travels to customer j after completion of service at customer i, otherwise x_{ij} is 0.

7.2 Simulation of the DTSPTW

In this section we sketch our framework for solving the DTSPTW introduced above. The basic idea of the algorithm is described in Algorithm 3 on page 112. The algorithm is basically an event-based simulation program also known as a "turning-the-clock" based simulation approach [28]. This approach was used in order to be able to run tests faster, i.e. by using this approach we "skip" all eventless points in time during the day of operation. However, using this approach also means that we do not include the computation time of the DTSPTW algorithm. However, all tests of the DTSPTW algorithm showed that the computation times never exceeded a limit of 2 seconds and it is therefore safe to say that the computation times are negligible.

As it can be seen in the description of the algorithm, the vehicle is forced to wait before it goes to a new customer, if the vehicle would arrive at the next customer before the time window opens. Alternatively, we could also have to chosen to go to the location of the next customer and wait at this position instead.

The simulation environment including all auxiliary functions was implemented in ANSI-C. The implemented program consisted of approximately 4500 lines of code.

7.2.1 Solving the TSPTW Heuristically

The solution method used to solve the DTSPTW has to be very fast in order to be embedded in the on-line routing algorithm. We implemented a very simple heuristic for the TSPTW. The heuristic is based on exchanges in the order the customers are served on the route. The basic idea is to test whether or not an exchange of two customers improves the objective function. If the exchange improves the objective, the exchange is executed otherwise it is not. The TSPTW heuristic is sketched in Algorithm 4 on page 113.

Algorithm 3 Event-based simulation of the DTSPTW model.

```
Solve the TSPTW for the $n_{adv}$ advance request customers
customer $i$ = first customer on the route
$time = t_0$
repeat
  if ($event = new\_customer$) then
    solve current TSPTW
    customer $i$ = first customer on the route
    if ($e_i > t_i$) then
      wait at current position
      $new\_event = end\_of\_forced\_wait$
    else
      $new\_event = start\_traveling$ to customer $i$
    end if
  else if ($event = start\_traveling$) then
    calculate arrival time at customer $i$
    $new\_event = begin\_service$ at customer $i$
  else if ($event = begin\_service$) then
    calculate end of service time
    $new\_event = end\_of\_service$
  else if ($event = end\_of\_service$) then
    if ($queue > 0$) then
      customer $i$ = next customer on route
      $new\_event = start\_traveling$
    else
      wait at current position
      $new\_event = end\_of\_forced\_wait$
    end if
  else if ($event = end\_of\_forced\_wait$) then
    $new\_event = start\_traveling$ to customer $i$
  end if
  $event = next\_event$
  $time = time$ of $next\_event$
until ($time \geq T$) & ($no\_more\_events$)
```

Algorithm 4 Outline of the TSPTW heuristic.

```
$iter = 0$
$current\_best\_route$ = customers sorted according to the median of the TW's.
**repeat**
  **for** $i = 0$ to $customers$ **do**
    **for** $j = 1$ to $max\_pos$ **do**
      $left = i$
      $right = (i + j)\ MOD\ customers$
      $temporary\_route$ = $left$ customer is exchanged with $right$ customer
      compute $cost(temporary\_route)$
      **if** $cost(temporary\_route) < cost(current\_best\_route)$ **then**
        $current\_best\_route = temporary\_route$
      **end if**
      $iter = iter + 1$
    **end for**
  **end for**
**until** $(iter = max\_iter)$
```

In the algorithm the parameter *max_pos* denotes the maximum number of customers between two exchange spots. I.e. if $max\_pos = 3$, it is allowed to exchange customer i with $i+1$, i with $i+2$ and i with $i+3$. The MOD operator denotes the modulus of the integer division. A number of 100 iterations was used as the maximum number of iterations in all runs. The parameter *max_pos* was chosen to 5 for all runs.

The heuristic was implemented using the ANSI-C programming language. The implemented algorithm consisted of approximately 300 lines of code.

It should be noted that the DTSPTW we solve with the above sketched algorithm could be seen as a stochastic problem due to the fact that the on-site service times are not known at the time of the planning. In order to be able to calculate the temporal information needed in the DTSPTW the expected value of the on-site service times is used. Naturally, this approach may (and often will) imply that the solutions to the DTSPTW problems differ from what would have been the solutions, if alternative values for the on-site service times had been used. However, it is assumed that the overall performance of the policies described in the next section is not affected by the evaluation of the on-site service times.

7.3 The A-priori DTSPTW

In the following we extend our on-line routing policies for the DTSPTW to include the a-priori information of the arrival intensities λ_i. The idea is that inclusion of this a-priori information will enable us to extend the on-line policies in order to improve the solution quality with respect to the distance driven by the vehicle and/or the total time window violation. We will refer to the a-priori information based DTSPTW as the ADTSPTW .

7.3.1 Routing Policies

When the vehicle finishes the service at the current customer, two situations can occur:

1. The time window of the next customer has already opened (or it will open before the vehicle reaches the location of this next customer).
2. Either the time window of the next customer to be served has not opened yet (the vehicle must wait before starting to travel to the next customer) or no customers are waiting to be served.

In the latter case, the vehicle could wait at the just served customer's location or it could go to one of the idle points either to wait for a new request to appear or just to remain idle, until the time window of the next customer opens.

Below we introduce three simple policies for choosing which idle point to go to when the vehicle becomes idle.

- NEAREST
  - The vehicle goes to the nearest idle point relative to the current position of the vehicle.
- BUSIEST
  - The vehicle goes to the idle point with the highest λ_i-value.

- HI-REQ (Highest expected number of immediate requests)
  - The vehicle goes to the idle point with the highest attractiveness score (AS). The attractiveness score for subregion i, AS_i, is defined as:

$$AS_i = \Delta T_i \cdot \lambda_i, i \in \mathcal{IP} \tag{7.3}$$

  where ΔT_i is the *idle time* of the vehicle, if the i'th idle point is chosen i.e. the time interval from the present moment until the time the vehicle will have to leave the idle point in question in order to arrive at the first customer on the remaining part of the route.

The attractiveness score of the HI-REQ policy could be interpreted as the exptected number of customers arriving within the time interval ΔT_i.

In order to be able to control when and when not to go to one of the idle points we will now define a threshold value, L_{thres}. The threshold value refers to the probability of receiving *at least* one new request during the time interval ΔT_i. The probability of receiving k customers during the time interval ΔT_i given that the customers arrive according to a Poisson process with intensity λ_i could be expressed as:

$$P\{X = k\} = \frac{1}{k!}(\lambda_i \Delta T_i)^k e^{-\lambda_i \Delta T_i}, \;\; k = 0, 1, 2, ... \tag{7.4}$$

Which means that the probability of receiving *at least* one customer during the time interval ΔT_i must be:

$$P\{X \geq 1\} = 1 - (\lambda_i \Delta T_i) e^{-\lambda_i \Delta T_i} \tag{7.5}$$

The vehicle should then go to idle point i if

$$P\{X \geq 1\} \geq L_{thres} \tag{7.6}$$

Otherwise the vehicle should rest at its current position.

Finally, the policy that does not allow for reposition when the vehicle is idle is referred to as the NO REPOS policy, i.e.;

- NO REPOS
  - The vehicle waits at its current position until the next customer should be visited.

7.4 Generation of Datasets

In this section we describe how the datasets used to test our policies are generated. The service region, $\mathcal{A}$, is bounded by the square [0,10000] x [0,10000], with the depot located at the median (i.e. at (5000,5000)). A scaling factor of 1000 units per km is used, i.e. the service region is equivalent to a 10 km x 10 km area. The vehicle is assumed to be driving at constant speed - 40 km/h. As mentioned in section 7.1, the on-site service times were generated according to a log-normal distribution with an average service time of 3 minutes and a variance of 5 minutes^2.

We generate two types of customers; the advance request and the immediate request customers. The immediate request customers call in for service during business hours of the courier mail service provider. The center opens at 8:00 and closes at 16:00, i.e. the calling period, T, is set at 8 hours.

We will examine four different scenarios - hereafter referred to as `Scenarios A - Narrow`, `A - Wide`, `B - Narrow` and `B - Wide`. For all scenarios we used $n_{subreg} = 9$. The geographical locations of the customers are assumed to be uniformly distributed within each subregion. The times for receiving the immediate requests are assumed to be Poisson distributed. The relation between the arrival rates of the Poisson process in the subregions is refered to as *arrival intensity weights*. Figure 7.2 shows the arrival intensity weights of all 9 subregions for `Scenarios A - Narrow`, `A - Wide` and `Scenarios B - Narrow` and `B - Wide` respectively. The distribution of arrival intensity weights is seen to vary. For the `A` scenarios the highest intensity weight is in the middle of the service region, i.e. in the same subregion as the depot. For the `B` scenarios the intensity volume is centered towards the upper right part of the service region, whereas `Subregions II`, `III` and `VII` are low intensity regions. These scenarios were chosen to test the performance of the policies under varying distributions of the arrival intensities.

The dashed line through the subregions indicates in which order these are visited. The arrival intensity weight of subregion i of `Scenario A -`

Narrow is denoted, $\tau_i^{A-Narrow}$. Hence $\underline{\tau^{A-Narrow}} = \{1,2,1,2,4,2,1,2,1\}$. This means that the arrival rate of the Poisson process will be four times as high for Subregion V as it will for Subregion I.

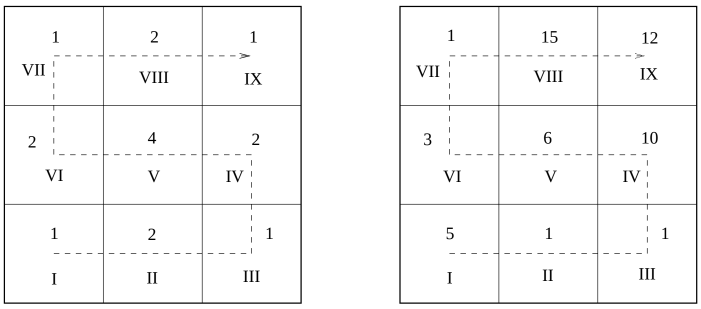


Figure 7.2: Arrival intensity weights for Scenarios A - Narrow/A - Wide and B - Narrow/B - Wide respectively.

The time windows for the advance request as well as for the immediate request customers were generated so that the latest possible time for the time window to close would be $T + 120$ minutes. I.e. the earliest and the latest possible times for the time windows to open and close respectively would then be 8:00 and 18:00 when $T = 480$ minutes.

The minimum and maximum lengths of the time windows of the advance request customers are shown in Table 7.1 on page 119.

The generation of the time windows of the immediate request customers differs from the generation of the advance request customers, since the time window are assumed to open at the time the request is received by the dispatcher plus a default time travel time from the depot to the customers' geographical location. In practice, the time windows are generated as the advance requests, following the minimum and maximum lengths as specified in Table 7.1. After the generation of the time windows the time the windows open, e_i, is adjusted so that the time windows open at the time the call is received, t_i, plus the time it takes for the vehicle to reach the new customer i from the depot, i.e. $e_i' = t_i + t_{depot,i}$. This generation scheme implies that the time windows of the immediate requests on average become wider than the time windows of the advance requests. The reason for using this rather special way of generating the time windows is that we want to emulate a real-life situation, where almost all immediate request customers are ready to receive service at the time the order is placed with the distributor. Furthermore, by following this generation method we hope to provide the optimal conditions for the repositioning policies by enabling immediate service of the immediate requests. Figure 7.3 illustrates the temporal relations of the calling times and the time windows of the immediate request customers. The above mentioned parameters related to the generation of the time windows were all chosen to resemble the actual parameters used by long-distance courier mail service providers.

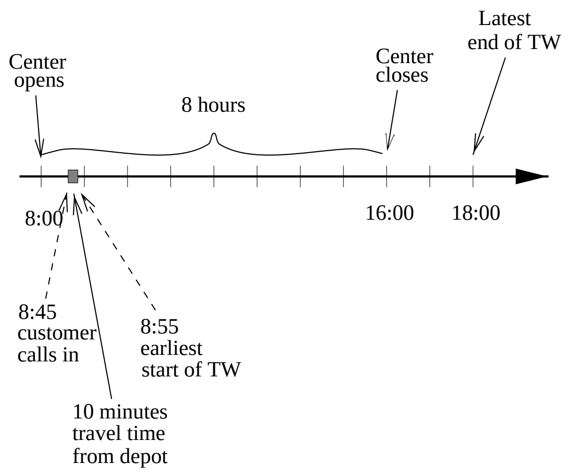


Figure 7.3: Time line illustrating the temporal relations between the time windows of the immediate request customers and the time the request for service is received.

Finally, it should be noted that we assume that the dispatchers do not have the authority to reject customers - or in other words all customers must be served.

7.4.1 Scenarios

The four scenarios constructed for the analysis were generated by using the two sets of arrival intensity weights as shown in Figure 7.2 combined with customers with relatively narrow and relatively wide time windows

Parameter / Scenario	`A - Narrow`	`A - Wide`	`B - Narrow`	`B - Wide`
Average number of cust.	40.6	40.5	40.5	40.7
Min. TW length (min.)	30	60	30	60
Max. TW length (min.)	90	180	90	180
Average TW width (min.)	89.1	143.7	89.1	143.7

Table 7.1: Parameters for `Scenarios A - Narrow, A - Wide, B - Narrow` and `B - Wide`.

respectively. All scenarios consisted of 100 instances with $dod = \{0, 5, 10, ..., 100\}$ %, which means that each scenario consisted of a total of 2100 instances.

In Table 7.1 the average number of customers, the average width of the time windows and the minimum as well as the maximum lengths of the time windows are shown for all four scenarios.

7.4.2 Parameter Values

As mentioned earlier the present formulation of the DTSPTW includes soft time windows. In all runs we did not accept that the time windows were violated without a penalty being incurred. If the vehicle would arrive too early at a customer, the vehicle would have to wait at the preceeding customer before leaving. We used two different setups of the parameter values of the cost function. In the first series of runs - hereafter referred to as `Series I` we used $\theta = 100$ and $\gamma = 1$, while in the second series - `Series II` - we used $\theta = 1$ and $\gamma = 100$. In other words for the `Series I` our goal was to minimize the distance traveled by the vehicle. While for `Series II` the goal was to minimize the lateness. $\bar{n}_{tot}$ and $\bar{tw}_{len}$ denote the average number of customers and the average length of the time windows, respectively.

The following four values were used as the threshold parameter, $L_{thres} \in \{0.2, 0.4, 0.6, 0.8\}$

7.5 Computational Results

In this section we discuss the results obtained, when testing the three repositioning policies, i.e. the NEAREST, BUSIEST and HI-REQ policies. The pure re-optimization policy NO REPOS is used as reference. NO REPOS simply solves a new TSPTW problem, each time a new immediate request is received by the dispatcher.

The results of the test runs are shown in Appendix C in Table C.1 through Table C.8.

We begin our discussion by examining the results of the `Series I` runs in which the distance driven is minimized. Here, the pure re-optimization policy is seen to perform better than the repositioning policies with respect to the distance for all four scenarios. Naturally, this oberservation was expected since the distance traveled caused by the repositioning will increase the total distance driven. Even though, it is not possible to find a single case in which one of the repositioning policies performs better than the NO REPOS policy on average. The tables show in a number of cases for which the repositioning policies did indeed manage to find a better solution than the NO REPOS solution. This on the other hand also means that some really poor solutions were found by the repositioning policies in order to keep the average distance driven when using the repositioning policies higher than when the NO REPOS policy is used. This fact is verified by the standard deviation which is generally higher for the repositioning policies than for the NO REPOS policy. When comparing the results of the `A` scenarios to the results of the `B` scenarios, it is seen that the average distances driven are marginally shorter for the `B` scenarios than for the `A` scenarios. This is naturally due to the concentration of the customers in the upper right part of the service region. However, the performance of the repositioning policies for the `B` scenarios does not seem to be better than the performance of the repositioning policies for the `A` scenarios.

Next, we examine the results of the `Series II` runs in which the lateness is minimized. The tables show that the performance of the repositioning policies generally are very close to the performance of the NO REPOS policy. However, it should be noted that the number of best performances are higher for all three repositioning policies (for the lowest L_{thres} parameter value) compared to the NO REPOS policy. The average lateness figures show that only very small performance improvements over the NO REPOS policy were made. When comparing the number of best instances, it is seen that the HI-REQ policy performs the best for all four scenarios and for both series of runs. The standard deviation of the lateness is seen to be relatively high compared to the absolute values of the lateness. This indicates that some really poor solutions with respect to the lateness were found. A comparison between the average lateness for the `A` and `B` scenarios indicates no difference.

Appendix C also includes the actual routes constructed by the NO REPOS, the HI-REQ and the NEAREST policies for a single instance of the `Scenario A` dataset. In addition to the routes Appendix C also holds the log files of this particular instance. The routes and the log files for the HI-REQ and the NEAREST policies are found by using $L_{thres} = 0.2$. For this instance the NO REPOS policy is seen to perform best with respect to the distance traveled. However, the HI-REQ policy performs approximately 8 % better than the NO REPOS policy with respect to the lateness. The reduced lateness of the HI-REQ policy is obtained by an increase of 9.4 % in the distance traveled. From the log file and the route it can be seen that the HI-REQ policy causes the vehicle to visit four idle points during the day of operation. IP-5 which is identical to the depot is visited three times, while IP-2 is visited once.

In the following we discuss the performance of the policies implemented as a function of the degree of dynamism. For simplicity, only the results of `Scenario A - NARROW` dataset will be discussed. However, the three other scenarios verified the behavior which is described in the following.

In Figure 7.4 the average distance is shown as a function of the degree of dynamism when distance is minimized. The distance is seen to be relatively constant for scenarios with less than approximately 80-85 % dynamism. For scenarios with more than 80 % dynamism the distance is seen to increase considerably for increasing values of the degree of dynamism. Furthermore, it seems like the average distance reaches its minimum value for systems with a moderate degree of dynamism (approximately 50 %). This behavior is rather unexpected in a comparison to the corresponding figures of the PDTRP and DVRPTW analyses of the previous chapters. Figure 7.4 also shows that the repositioning policies for all scenarios perform worse than the pure re-optimization policy.

Figures 7.5 and 7.6 both show the average lateness as a function of the degree of dynamism when the lateness and the distance repectively are minimized in the objective function. The average lateness is seen to decrease for increasing values of the degree of dynamism and for a pure dynamic

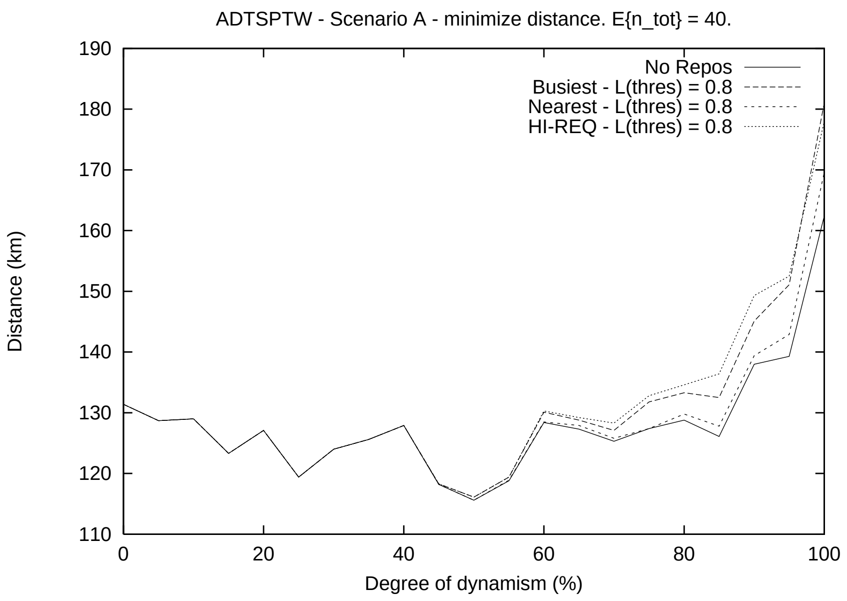


Figure 7.4: The average distance as a function of the degree of dynamism when the distance is minimized.

system the average lateness is seen to be close to 0. This behavior at first seems illogical, since a number of immediate requests ought to imply that the problem becomes temporally infeasible. However, the explanation to this unexpected beavior should be found in the generation of the time windows of the immediate requests. As mentioned in section 7.4 the average length of the time windows of the immediate request customers is longer than the length of the time windows of the advance request customers. Therefore, the heavily dynamic scenarios become less temporal restricted than the weakly dynamic scenarios.

We also tested the implemented policies on other datasets where the average length of the time windows did not depend on the degree of dynamism. For these datasets the behavior of the average lateness was as expected, i.e. the lateness increased with increasing values of the degree of dynamism.

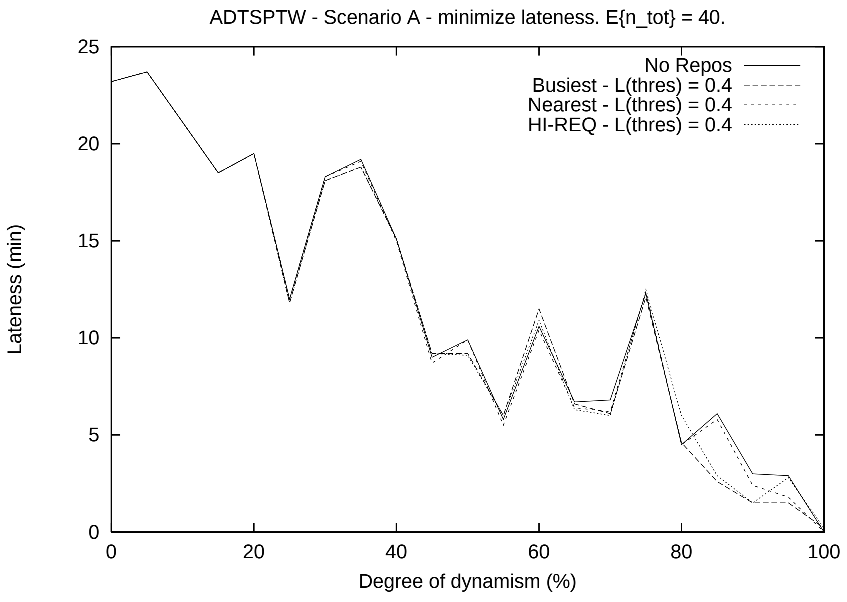


Figure 7.5: The average lateness as a function of the degree of dynamism when the lateness is minimized.

For this particular scenario `- Scenario A - Narrow -` the repositioning policies are seen to perform better than the pure re-optimization policy for instances with *dod* > 80 % when the lateness is minimized. Especially, the BUSIEST and the HI-REQ polcies are seen to outperform the NO REPOS policy. This behavior is unfortunately not charateristic for all scenarios. In fact, the level of dynamism for which the repositioning policies outperform the NO REPOS policy varies greatly for each scenario.

Figures 7.5 and 7.6 also show the difference between minimizing the lateness and the distance. The average lateness is approximately 23 minutes and 450 minutes for pure static scenarios when minimizing the lateness and the distance respectively.

In Figure 7.7 the number of repositionings is shown as a function of the degree of dynamism for the three repositioning policies. The number of

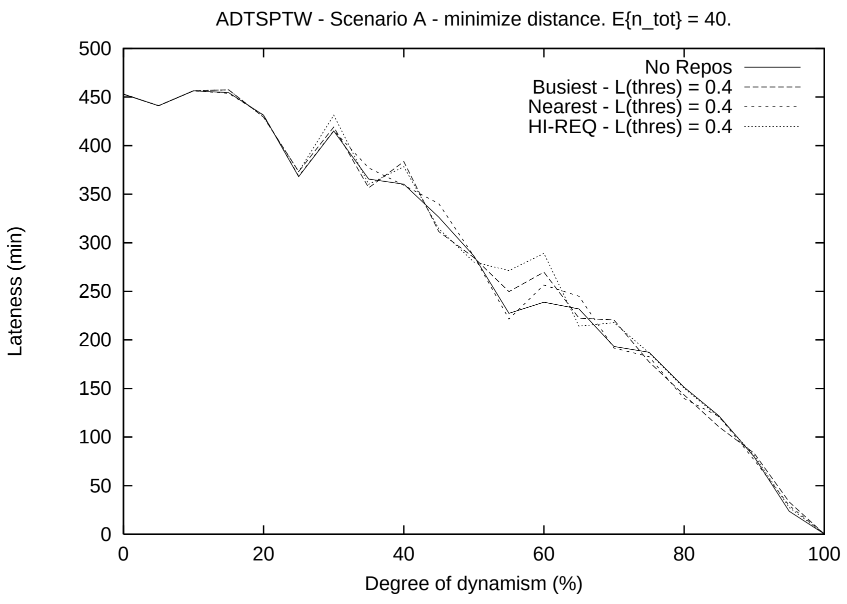


Figure 7.6: The average lateness as a function of the degree of dynamism when the distance is minimized.

repositionings are seen to increase considerably when the degree of dynamism grows. This behavior is expected since we assume that the degree of dynamism and the expected number of customers are known for each instance. This implies that the probability of receiving new requests given by equation 7.5 grows for increasing values of the degree of dynamism. The high number of repositions is indeed desirable, since it may be beneficial to reposition the vehicle to an idle point in a highly dynamic environment. However, the relative increase in the number of repositions for highly dynamic environments may be subject for further research. The L_{thres} parameter could for instance be extended to be made dependent on the degree of dynamism.

Figure 7.7 also shows that the HI-REQ policy causes the vehicle to reposition most frequently, whereas the fewest repositions are seen for the NEAREST policy.

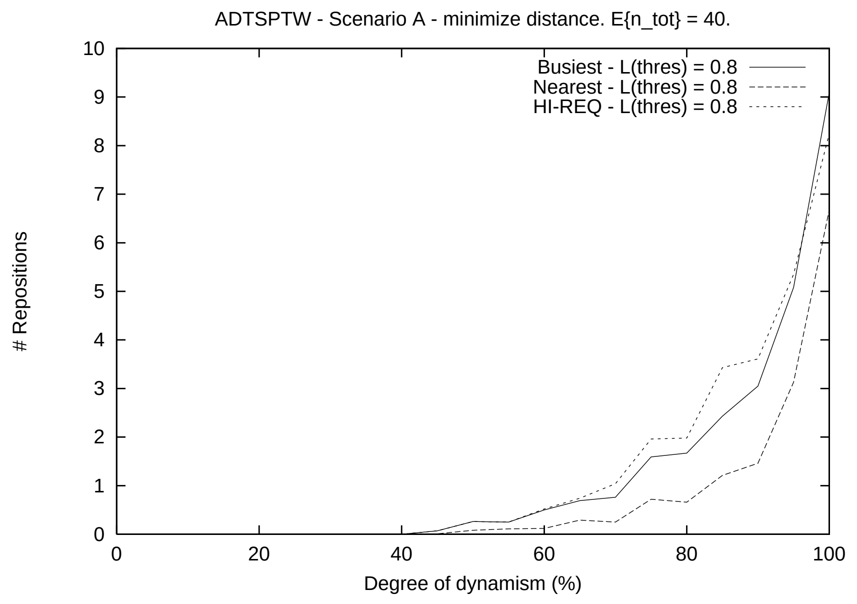


Figure 7.7: The average number of repositions as a function of the degree of dynamism when the distance is minimized.

In Figure 7.8 the number of repositionings is shown as a function of the degree of dynamism for the HI-REQ strategy for the four different values of the threshold parameter, L_{thres}. For the present scenario it is seen that the vehicle is repositioned more than six times as often for $L_{thres} = 0.8$ as for $L_{thres} = 0.2$ for systems with $dod = 50$ %.

7.6 Further Work

A natural extension of the model described in this paper is a generalization into dealing with multiple vehicles. The extension of course leads to other strategic decisions on for example the issue of how to assign the requests to the vehicles.

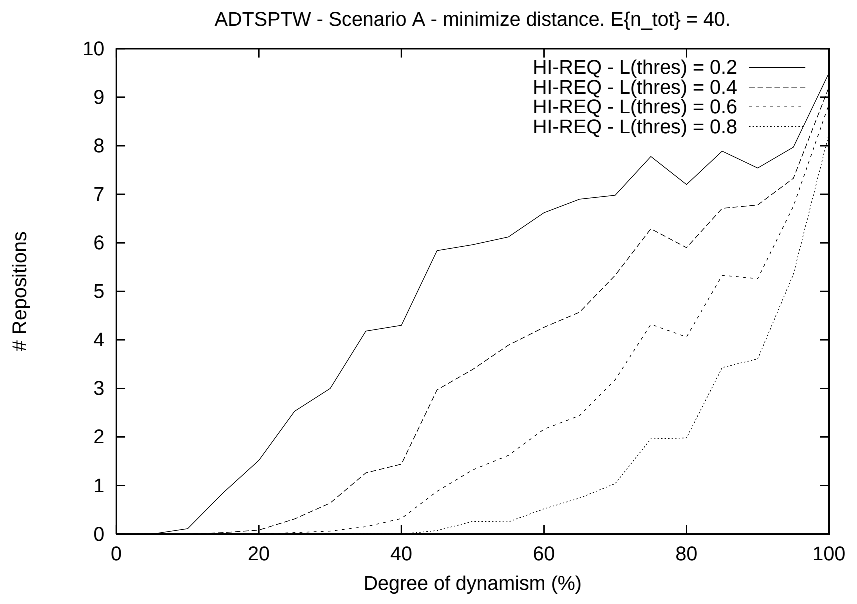


Figure 7.8: The average number of repositionings as a function of the degree of dynamism when the distance is minimized.

Another important issue to investigate is the potential savings in allowing the vehicle(s) to divert from the current route when a new and promising request appears in the vicinity of the current position of the vehicle. In relation to the present work diversion could be implemented in two different situations. Firstly, the most obvious situation to allow the vehicle to divert in is the situation where the vehicle is repositioning to an idle point. The other situation is when the new immediate request appears, while the vehicle is on-route to serve the next planned customer. Recently, the diversion issue has gained increased popularity amongst OR researchers. Interesting work on diversion can be found in Yang et al. [44] and in Ichoua et al. [21].

The HI-REQ policy could be extended in several ways. A natural extension would be to take the neighboring arrival intensities into consideration when calculating the attractiveness score. To illustrate this we have included Figure 7.9. In the first case the HI-REQ policy would probably choose one of the subregions with a $\lambda = 2$ as the potential idle point to reposition to although the central subregion might be the best choice due to the short distance to all the high intensity neighbors. In the case illustrated by the right-hand figure another example of the same problem is shown. The HI-REQ would probably choose the top-right subregion (with $\lambda = 3$) as the potential idle point. In other words, the HI-REQ policy would not include the bottom-left subregions. In both cases one could argue that a location based analysis could improve the quality of the choice of subregion.

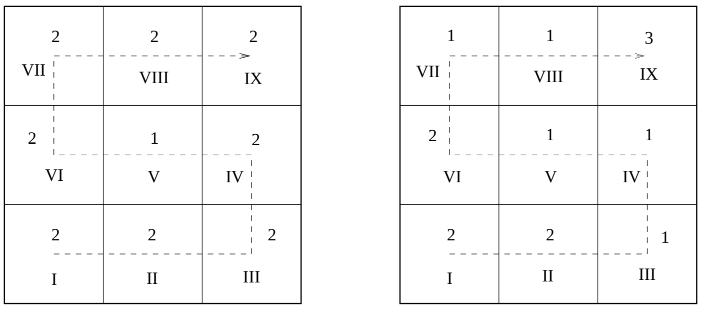


Figure 7.9: Two ADTSPTW scenarios with the 9 subregions' arrival intensity weights.

7.7 Conclusions

In this chapter we formulated a model for the Dynamic Traveling Salesman Problem with Time Windows (DTSPTW) including advance request as well as immediate request customers. This model was motivated by the pick-ups and deliveries of long-distance courier mail and packages. We extended the DTSPTW to also include a-priori information on future requests. We proposed three simple repositioning policies based on the a-priori information. Finally, these policies were tested on a set of test data resembling the aforementioned long-distance courier mail service application.

The results indicated some potential for reducing the lateness of the vehicle when using an a-priori based repositioning policy at the expense of an increase in the distance driven by the vehicle. The HI-REQ reposititioning policy seemed to have the highest number of best performances with respect to the lateness. However, no clear conclusions on which repositioning policy should be used could be drawn from the empirical analysis.

Further research within this field should consider allowing diversion during repositioning or even during traveling from one customer to the next on the route. Furthermore, in order to get the desired effect from the idle points more sophisticated methods for locating the idle points should be considered.

Chapter 8

A Real-life Scenario

In this chapter we examine a real-life example of the dynamic vehicle routing problem, namely the problem arising when people and companies all over the world each day make use of long-distance courier mail services to send mail as well as packages across the world. Naturally, the underlying distribution system is very large as well as rather complex. Figure 8.1 illustrates a simplified version of this distribution system. In this chapter the routing of the truck delivering and picking-up the mail and/or packages is considered. As can be seen from the figure this problem arises in both "ends" of the distribution system. With most long-distance courier mail service providers both the delivery and the pick-up problem is handled by the same vehicle. Usually, the deliveries are made during the morning hours, whereas the pick-ups mainly take place during the afternoon. This rhythm is natural since the customers in a long-distance courier mail distribution systems are to a large extent offices and other similar businesses wishing to receive their materials as early as possible and have their mail picked-up as late as possible. This means that each courier mail truck and its driver both acts as a delivery as well as a pick-up truck. This combined problem of delivering and picking-up courier mail can be considered as an application of the dynamic vehicle routing problem, since some unknown fraction of the pick-up customers call in for service, while the trucks are on-route.

Firstly, the underlying routing problem will be presented in further detail. Secondly, some key figures and the composition of the real-life dataset used

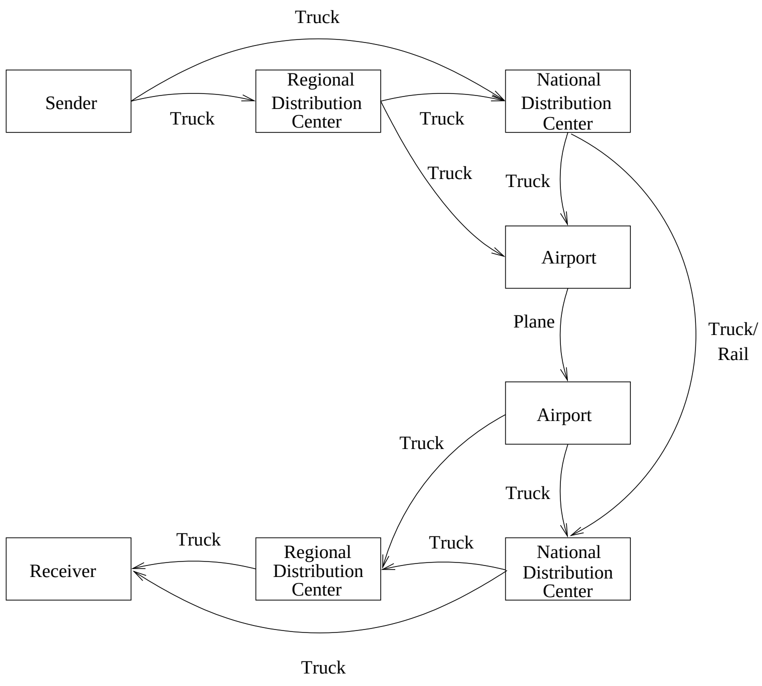

Truck
Sender
Truck
Regional Distribution Center
Truck
National Distribution Center
Truck
Truck
Airport
Plane
Truck/ Rail
Airport
Truck
Truck
Receiver
Truck
Regional Distribution Center
Truck
National Distribution Center
Truck

Figure 8.1: Simplified illustration of the distribution system for a long-distance courier mail service provider.

in this chapter will be described. Next, we describe how the heuristic that was developed to solve the A-priori Dynamic Traveling Salesman Problem with Time Windows (ADTSPTW) introduced in chapter 7 had to be modified in order to solve this specific application. Finally, we present the results of the test runs and point to possible further research in this field.

8.1 Problem Description

The pick-up and delivery problem faced by long-distance courier mail service providers (as for instance UPS, FedEx, DHL etc.) differs from the dial-a-ride problems (DARP) described in chapter 2.3.3, since in the long-distance courier mail problem the goods are *either* to be picked up at a customer and then transported back to the depot for further processing *or* to be transported from the depot to the delivery location. In other words a customer in the courier mail problem could either be a delivery customer *or* a pick-up customer. The objective of the courier mail service provider is to distribute all mail and packages safely within the given time restrictions while keeping the operating costs as low as possible.

The following example illustrates the structure of the distribution system for a fictive long-distance courier mail service provider.

1. Mr. Jones of London, England wishes to send his lawyer, Mr. Jakobsen of Copenhagen, Denmark a small parcel holding some urgent documents concerning his will. Mr. Jones telephones his long-distance courier mail service provider (hereafter referred to as FedUPS) at 13:43 and asks them to pick-up the parcel at his office during the afternoon the same day. The FedUPS call center kindly accepts Mr. Jones request and offers to pick-up the parcel during the time interval of 15:00 to 17:00 the same day. Mr. Jones happily complies and goes back to his office to finish the documents.
2. At 13:46 the call center operator passes Mr. Jones request on to a FedUPS dispatcher who calls the driver operating in Mr. Jones's neighborhood and informs him about Mr. Jones request for service.
3. At 16:33 the FedUPS driver arrives at Mr. Jones's office in London and is handed the parcel - some paper work related to billing is also taking place at the same time.
4. At 17:52 the FedUPS driver arrives at the regional FedUPS distribution center and begins to unload his truck.
5. At 22:00 a large FedUPS truck arrives at the regional distribution center and a number of large containers are loaded onto the truck - one of them holding Mr. Jones documents.
6. At 22:30 the truck arrives at the London Heathrow International Airport and the containers are loaded onto a FedUPS aircraft.
7. At 00:00 the FedUPS aircraft takes off at Heathrow Airport.

8. At 02:20 (local time) the FedUPS aircraft lands at Copenhagen Airport, Denmark and the containers are unloaded from the plane to a big FedUPS truck.
9. At 04:00 the FedUPS truck arrives at the regional distribution center and unloads a number of containers bound for this specific region in Denmark.
10. At 05:00 the FedUPS crew starts sorting the parcels into designated areas.
11. At 07:00 the FedUPS drivers begin their duties by sorting the parcels into their preferred order of delivery. Hereby a set of delivery routes are constructed.
12. At 9:34 Mr. Jakobsen receives the urgent documents from Mr. Jones.
13. At 11:52 The FedUPS driver finishes his route of deliveries and prepares for the first pick-ups.
14. At 12:48 The FedUPS driver is called by the FedUPS dispatching center - he is now assigned to pick up a new parcel...

Usually the following three different types of customers are considered in respect of this problem:

Type	Description	Known in advance
Delivery customers	Ordinary delivery customers with time windows	Yes
Regular pick-up	Customers requiring pick-ups every day of the week (or some specific weekdays) within a pre-agreed time window	Yes
On-call pick-up	Customers calling during the day of operation ordering pick-up - negotiates a time window.	No

Table 8.1: Customer characteristics in the long-distance courier mail problem.

The delivery and the regular pick-up customers are all known in advance to the routing procedure, which means that these customers should be perceived as advance request customers. However, the on-call pick-up customers - such as Mr. Jones - who calls in for service during the day of operation (while the vehicle is on-route) must be treated as dynamic customers.

8.2 Collection of Data

The data were collected at one of United Parcel Services' (UPS) more than 3000 regional distribution centers. Geographically, the data were made up by four almost adjacent areas. The period chosen for investigation was two weeks (Monday to Friday) in March 1999. These weeks were assumed to be representative for the operation, since they did not conflict with public holidays. This means that a total of ten days of operation were considered. Each area is served by a single pick-up and delivery truck. Naturally, the four areas could have been aggregated to a single service region served by multiple vehicles. However, since the ADTSPTW model introduced in the previous chapter considers single vehicles, four separate areas were considered. The total volume of the datasets therefore sums up to 40 separate datasets.

On average each vehicle serves 61.3 customers per day. The on-call pick-up customers comprise approximately 11.5 % of the total number of customers. The remaining part of the customers are delivery customers and regular pick-up customers (customers with an agreement of daily pick-ups). In Table 8.2 a number of key figures are listed for the four areas in question. The temporal distribution of the on-call pick-up customers' time for requesting service is shown in Figure 3.1 on page 45. As mentioned in chapter 3 the figure indicates that the intensity of the calls varies during the day. The morning hours are relatively slow, whereas the number of on-call requests increases during the afternoon reflecting normal office hours.

8.2.1 Postprocessing of the Data

All data came as text and had to be typed into text files. After this tedious assignment had been completed, a simple program that was able to scan

Area Code	Average customers	*dod* (%)	Deliveries	Total Pick-ups	On-call Pick-ups
A	59.1	9.6	528	63	57
B	63.3	9.8	569	64	63
C	59.2	10.6	522	70	61
D	63.8	15.8	520	105	102

Table 8.2: Key figures for real-life data collected in March 1999. All numbers are for the entire 10-day period.

and parse the text files was implemented. This program created a database of different addresses and passed these on to the road data base DAV [1] in order to find the exact coordinates of the customers. It should be noted that approximately 10 % of the addresses were erroneous in some sense either due to inconsistency between the official names of the streets and the driver's route log or due to errors in the database. The total number of addresses including delivery as well as pick-up customers summed up 686. The service region was divided into 729 subregions; each covering an area of 1 km x 1 km. The service region therefore spans an area of 729 km^2.

8.2.2 Estimation of Service Times

Unfortunately, the service times of the customers were not available from the driver's log files. This implies that the service times had to be estimated. As mentioned in chapter 3 one way of simulating on-site service times is to use the log-normal distribution. The parameters of the log-normal distribution could for instance be tuned by noting the time of the departure from the depot and the time the first x customers were serviced. This gives us the time spent on traveling from one customer to the next as well as the service time of these customers. Subtracting the travel times taken from the travel time matrix leaves us with the desired service times. The average and variance of these data could then be found and used as parameters in the log-normal distribution to generate service times for all customers. However, as the times the on-site service ended nor were available for all customers, we simply decided to use 3 minutes and 5 minutes[2]

[1]DAV - Dansk Adresse- og Vejdatabase - please refer to section 1.6.2 for further information on DAV.

as the average and the variance respectively.

8.2.3 Estimation of the Distance Matrix

The initial intention of this case study was to use real-life data all the way through the analysis. However, as described above the on-site service times had to be estimated. Furthermore, after the customer data had been processed, it turned out that due to problems with the conversion of the geographical coordinate coding system from the DAV system to the MapInfo system[2] it was not possible to generate the right input for a shortest path algorithm. It was therefore decided to proceed using Euclidean distances using a correction factor of 15 %, since it was estimated that the true distance between two addresses is approximately 15 % longer than the Euclidean distance. Naturally, this is very dependent on the density of the road network in question. However, for most of the service region dealt with in this case study the 15 % seems to be a fair correction factor, since this value has been used for other similar projects.

8.2.4 Estimation of Arrival Intensities

In order to estimate the arrival intensities of the subregions the pick-ups were plotted by use of built-in functions within the MapInfo framework. MapInfo also provides functions to compute the number of observations within a certain geographical bounded region. In Table 8.3 the total number of pick-ups are shown for dataset A for the entire 10-day period. The cumulated number of pick-ups are used as the arrival intensity weights.

8.2.5 Location of Idle Points

It was decided to create a set of idle points to serve as potential "resting locations" for the vehicles when idle time occurs. The positions of the idle point were chosen as customer addresses having 3 or more pick-ups during the ten day observation period. This resulted in a total of 37 idle points distributed over 25 different subregions. Naturally, in a real-life situation

[2]MapInfo is a software package which allows the users to perform location analyses. For further information please see MapInfos web-site at www.mapinfo.com.

Subregion	390	391	392	417	418	420	421	423	445
# Pick-ups	5	15	7	2	10	5	1	1	7

Subregion	487	541	549	553	580	581	582	661	689
# Pick-ups	1	2	1	1	1	1	1	1	1

Table 8.3: The total number of pick-ups within each subregion for the entire 10 day period for dataset A.

the idle points chosen should not be the customer addresses, but dedicated idle points located in a high intensity area.

8.3 Routing method

As described above the pick-up and delivery problem faced by a long-distance courier mail service provider could be formulated as a dynamic version of the TSPTW. The main idea of this case-study was to test the performance of the routing policies proposed in chapter 7 on a dataset originating from a real-life dynamic vehicle routing application. The input files produced by the scanner and parser routines mentioned above resembled the randomly generated data files used in the previous chapters. This meant that the data reading routines did not have to be changed, nor did the travel distance function since the distance was computed as the Euclidean distance between the customers using the correction factor mentioned above. In case a shortest path based travel distance matrix had been constructed, the computation of the distance obviously would have had to be changed.

In order to minimize the distance traveled and the lateness respectively two sets of the α and β parameters were considered. In the first series of runs we used $\theta = 100$ and $\gamma = 1$ while in the second series $\theta = 1$ and $\gamma = 100$ were used. The following four values of the threshold parameter, $L_{thres}(\Delta T_i, i)$, were used: $L_{thres}(\Delta T_i, i) \in \{0.05, 0.10, 0.20, 0.30\}$. The values were chosen, since they resulted in a good variation in the number of repositionings for the various policies. The vehicle was assumed to be driving at constant speed - 40 km/h.

Subregion	Area A	Area B	Area C	Area D
94				1
147				3
175				1
176				2
201				2
203				1
228				1
251			1	
283				1
288			1	
341			1	
361		5		
362		1		
390	1			
391	2			
392	1			
393		1		
397			1	
394		1		
413		2		
418	1			
420	1			
445	1			
451			3	
452			1	
Total	7	10	8	12

Table 8.4: The number of idle points located in each subregion.

8.4 Computational Results

The computational results of the real-life dataset are shown in Table D.1 through Table D.8 in appendix D. The HI-REQ and the BUSIEST IP policies are seen to perform worse with respect to the distance compared to the pure re-optimization policy for all areas and threshold levels. The NEAREST IP policy is seen to have performance almost identical to the pure re-optimization policy with respect to the distance. However, in a few cases (mainly for the `Area C` dataset) the NEAREST IP outperforms the re-optimization policy with respect to the distance driven. With respect to the lateness measure the picture is even more blurry. However, it seems that the repositioning based policies mostly perform at least as well as the the pure re-optimization based policy. It can be seen from the figures that the absolute values of the lateness range between 1.35 and 86.62 minutes which are relatively small values compared to the number of customers on each route. The relative lateness then ranges between 0.02 and 1.41 minutes per customer [3].

Unfortunately, the actual routes driven by the trucks cannot be compared to the routes found using the ADTSPTW algorithm due to a number of facts. Firstly, the on-site service times used in the simulation are rough estimates of the real service times. Furthermore, the distance matrix used is based on Euclidean distances using a correction factor of 1.15. Last but not least the velocity of the trucks is also based on a rough estimate, whereas in a real-life situation congestion, road construction and accidents may influence the actual velocity of the trucks.

In Figure 8.2 the route produced by the ADTSPTW heuristic when using the HI-REQ policy with a threshold of 0.1 while seeking to minimize the distance traveled is shown. The distribution of the customers in this particular instance can roughly be divided into three groups; a) the group of customers surrounding the depot located in the southern part of the figure, b) the largest group of customers located near the median of the service region and c) the three-customer group located in the north-western part of the service region. In Appendix D the log files of this instance are listed for the HI-REQ and the pure re-optimization based policies. It is seen that the distance traveled is 98.75 km for the pure re-optimization policy, whereas it is 104.59 km for the HI-REQ policy. The additional distance of the HI-REQ policy is due to the fact that the truck repositions three times

[3]When the average of 61.3 customers per route is used.

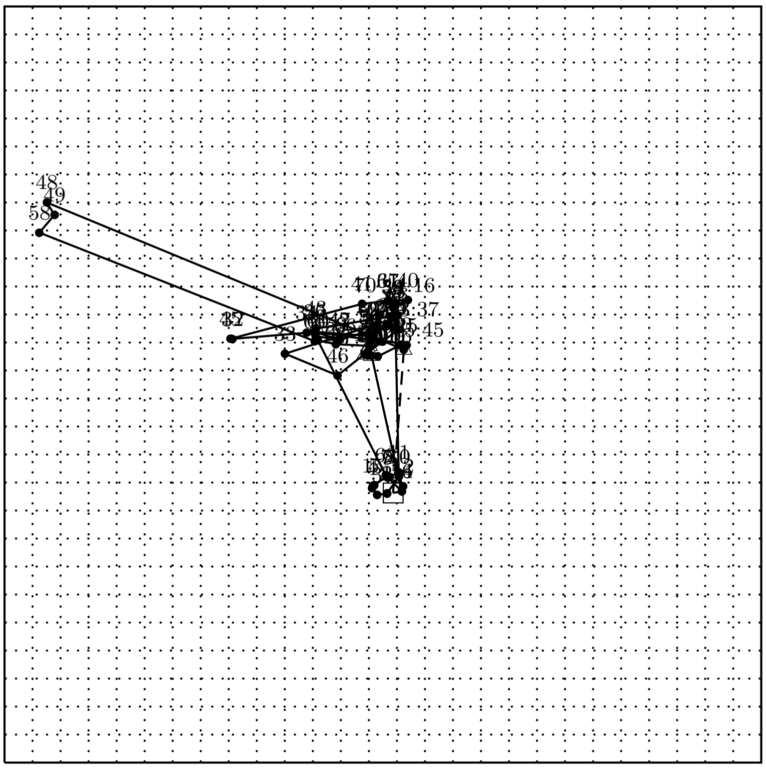


Figure 8.2: ADTSPTW scenario.

(to idle points 5, 1 and 1) when using the HI-REQ policy with $L_{thres} = 0.1$. There is no lateness for neither of the policies.

8.5 Conclusions

In this chapter we analyzed a real-life dynamic vehicle routing problem originating at a long-distance courier mail service provider. The analysis of this routing problem showed that the structure of subproblems dealing with the pick-ups and deliveries differs in that the deliveries with few exceptions are made before the pick-up begins. This structure suggests that the routing is divided into a morning delivery subproblem and an afternoon pick-up subproblem. The former subproblem is an almost 100 % static problem, while the latter subproblem is mainly a dynamic one. Therefore a dedicated routing system should a) construct a set of delivery routes using the data of the delivery customers and b) provide an on-line routing / dispatching algorithm for the afternoon pick-up problem. In case of urgent on-call requests appearing during the morning the routing algorithm should be able to make a simple insertion of the new request into the delivery routes.

The temporal restrictiveness of the present real-life dataset is relatively small and this means that the chance of reducing the lateness by using repositioning policies is relatively low since it is likely that the majority of the on-call requests could be served within the specified time windows even though the on-call requests may be scattered. Therefore, the present dataset does not provide the ideal basis for testing the performance with respect to the lateness of the repositioning policies compared to the pure re-optimization policy.

As was described above, the location of idle points were selected on a rather naive basis, since only the number of pick-ups during the 10-day period was considered. Naturally, in a real-life situation methods combining location analysis and extensive data material should be used.

During the test-runs of the present dataset, all the computations have been performed while the system was off-line. By off-line we mean that the computations were performed while the simulation clock was stopped. However, all computations in the present form are performed within a second and therefore we feel it is fair to say that the computation time is negligible. It should also be noted that in a real-life situation the computation of the distances between the customers will have to be made using a shortest path method on a large-scale road network. Naturally, this will affect the computation times considerably.

Chapter 9

Further Research Perspectives

In this chapter we will give a brief discussion on the state-of-the-art within the DVRP methodology. Furthermore some directions for further research on the DVRP will be provided.

9.1 Assessment of the Methodology

In this section we give a brief discussion on how the various solution methodologies can or cannot be used to solve real-life DVRP applications.

9.1.1 Mathematical Programming Based Methods

Even though the rapid development in computer hardware along with a long line of improvements in optimization software has pushed the boundaries of what is possible to solve in a reasonable amount of time, it is still computationally hard to solve real-life static VRP's. This is mainly due to the wide variety of side constraints that are often present in a real-life routing problem. Naturally, this fact makes it almost absurd to encompass a mathematical programming based VRP model within a real-time environment. Furthermore, seeking after optimal solutions to dynamic problems may also be pointless, as new input may render an optimal solution at stage t suboptimal at stage $t+1$. Finally, the dynamism implies that stochastic elements such as uncertain customer demands are introduced. Therefore, it may be necessary to make the model stochastic, which means that it becomes even harder to find an optimal solution (or even just a feasible one) within a reasonable time.

Therefore, we do not believe that DVRP systems based on mathematical programming methods will be able to solve realistic problems within the time requirements in the near future.

9.1.2 Meta-heuristics

Meta-heuristics usually also require relatively long computation times in order to provide good quality solutions. However, in some real-world applications meta-heuristics like tabu search and simulated annealing might prove to be a good choice of method. This is due to the fact that these methods for the most part will be able to find a feasible solution within few seconds. This is an important issue since a real-life DVRP system should be able to provide the dispatcher with either a feasible solution or a message saying that the request is being rejected due to infeasibility so that the customer could be informed whether or not her request for service will be accepted.

The real benefit from applying meta-heuristics to solve the DVRP comes when the temporal distribution of the immediate requests for service allows the meta-heuristic to gradually improve the initial solution where after the improved solution is actually implemented. On the other hand, if an immediate request is received before an optimal or a near-optimal solution is being implemented, the computation time spent on finding this solution could be seen as wasted, but since the computer is assumed to be running idle during this period, this will mean nothing in reality. Finally, if the arrival intensities of the immediate requests are relatively high, the meta-heuristic will most likely not be able to improve on the initial solution before a new immediate request is received, but in this situation all methods will have difficulties.

We believe that meta-heuristics applied to DVRPs might be well-suited for improving the performance of real-life applications. As argued above, the success must be assumed to be heavily dependent on the temporal distribution of the immediate requests.

9.1.3 Simple A-priori On-line Policies

In chapters 7 and 8 simple a-priori information based on-line policies were discussed. The empirical analyses did not convince us that simple a-priori based methods are the best choice when implementing DVRP. One major problem is that these methods tend to include numerous problem specific parameters as can be seen from the previous chapters. This usually means that relatively extensive empirical data are required in order to be able to customize the policies to fit the specific problem. Furthermore, the simple on-line policies examined in the previous chapters did not use the idle time of the computer to improve the initial solution.

The degree of dynamism was relatively low in the case study presented in chapter 8, i.e. approximately 10 %. For other applications such as for the taxi cab services the degree of dynamism is considerably higher - close to 100 %. In such systems re-positioning policies may be a good choice, as the empirical analysis in chapter 7 points to the best performance with respect to the lateness being obtained for systems with high degrees of dynamism. We believe that for systems with a high degree of dynamism simple repositioning policies may be a good choice when seek to minimize the lateness in the system.

9.1.4 Fast Insertion Method

During the collection of the data for the case study of the courier mail service presented in chapter 8, the business people explained that the dispatchers are often able to guess when a specific company calls in with a request for service, since a high number of the pick-up customers follow the same demand pattern, i.e. they call in requesting a pick-up at specific times during the day. Naturally, this information is a very application dependent characteristic, which is due to the nature of the DVRP in question. If this information it would be obvious to build a repository containing the historic information on immediate requests.

The algorithm briefly outlined in the following builds on inserting non-existing dummy customers into the route based on using the historic information kept in the repository.

The algorithm should first construct a number of routes visiting all advance request customers and a number of dummy customers. After an initial feasible solution is found, an improvement heuristic should be run. Each time an immediate request is received, the algorithm should remove one (or perhaps more than one) dummy customer and the algorithm should then try to insert the new immediate request customer into a route by either inserting the customer into the now empty slot of the dummy customer or by re-shuffling the customers already on the route.

Naturally, the location and the number of dummy customers to be inserted (and removed) are crucial factors in the implementation of such a system. The number of customers should be calculated using the estimates of the arrival intensity of the immediate requests, whereas the geographical locations could be calculated using the information in the repository.

This method could be implemented in several ways. In the following we sketch just one possible implementation. For the long-distance courier mail service problem the day of operation could be divided into a delivery phase, which begins when the vehicles leaves the depot during the morning hours and ends around noon, and a pick-up phase, which begins approximately at 14:00 and ends at 17:00. The routes for the delivery phase should probably be constructed without inclusion of dummy customers, since only very few of the time windows of the immediate requests close during the delivery phase. The geographical location of the idle points could serve as the locations of the dummy customers for the pick-up phase.

This method is naturally very application specific and it might only be applicable for a very limited number of applications. As for the application of routing the distribution of heating oil, service must be assumed to be very infrequent and not to follow at precise pattern. The same could be said about the repair of teller machines.

9.1.5 Summary and Assessment

In this section we firstly argued that mathematical programming based methods do not seem to be the best choice of method for solving the DVRP. Next, we argued that using meta-heuristic methods to improve on the current solution while the system is idle waiting for the next immediate request to be received may be a good way to go. Thereafter, it was argued that simple a-priori on-line policies may be worth a close look when systems with a high degree of dynamism are dealt with. In section 9.1.4 we outlined an algorithm for a system which is based on fast insertion of new immediate requests, an improvement heuristic and the insertion of dummy customers.

Furthermore, we believe that future research should consider allowing the vehicles to divert from the current route in case an immediate request appears in the vicinity of one of the vehicles' current position.

9.2 Further Improvements

In this section we provide some directions which we believe are relevant to future research on the DVRP.

9.2.1 Measure for Dynamism

In chapter 4 the degree of dynamism and some extensions to this very simple measure for dynamism were discussed. It was argued that the basic measure does not characterize all situations, since it does not consider at what time the immediate requests are received. In chapters 5 and 6 the extensions to the basic degree of dynamism measure were analyzed empirically. The results did not show any increased descriptiveness in using the extended versions of the measures. Therefore, we believe that further research on simple but descriptive measures for describing the dynamism of a DVRP is still needed.

9.2.2 Analytical Improvements

In the present work by Bertsimas et al. only immediate request customers are considered. The PDTRP presented in chapter 5 also includes advance requests. The PDTRP could be examined by using the methodology of Bertsimas et al. In this way upper and lower bounds on the performance could be established taking the degree of dynamism into account. Also, Bertsimas et al. did not examine applications with time windows. Naturally, adding this type of side constraints increases the complexity of the theoretical analysis considerably.

9.2.3 Multiple Vehicles

Extending the ADTSPTW presented in chapter 7 to the multiple vehicle case and later on also to the capacitated case is a very natural next step for further research on the a-priori based models. Extending the DTSPTW to the multiple vehicle case leads to considerations on the strategy for dispatching the vehicles. One possibility would be to assign the customers to a vehicle according to a sweep method and then solve each problem with a simple TSPTW heuristic which is able to solve problems in real-time, for instance the DTSPTW algorithm presented in chapter 7.

Chapter 10

Conclusions

In this thesis various aspects of the Dynamic Vehicle Routing Problems were discussed. In this final section we give a short summary of our findings and list the scientific contributions we believe have been provided in this thesis.

First, during our discussion of the existing DVRP literature we argued that methods purely based on stochastic mathematical programs are not ready to be included in a real-time environment. Instead we believe that on-line policies based on either simple heuristics or meta-heuristics are be the best choice of methods for the present moment.

The degree of dynamism measure was extended to embrace the times the immediate requests are received by the dispatcher. The empirical analysis performed in chapters 5 and 6 for the Partially Dynamic Traveling Repairman Problem (PDTRP)and the Capacitated Vehicle Routing Problem with Time Windows (DVRPTW) did not convince us that the proposed extensions of the degree of dynamism measure provide the improved descriptiveness wished for. However, we still believe that the measure of the degree of dynamism is important for describing DVRP instances and therefore we believe that the measure should be further refined.

The DVRP covers a wide range of practical applications and the taxonomy introduced in chapter 4 represents only a first step towards an attempt to categorize these according to the degree of dynamism and the objective function.

The Partially Dynamic Traveling Repairman Problem (PDTRP) introduced in chapter 5 illustrated a linear relationship between the degree of dynamism and the route length. Simulation runs suggested that the best performance was obtained when the Nearest Neighbor policy was used.

The Capacitated Dynamic Vehicle Routing Problem with Time Windows (DVRPTW) was examined in chapter 6 and the results indicated that the performance with respect to the distance driven by the vehicle turned out to be highly dependent on both the level of dynamism and the structure of the data for the system in question. However, the lateness experienced by the customers was shown to increase for increasing levels of dynamism.

A number of simple on-line repositioning policies were proposed for examining the A-priori Dynamic Traveling Salesman Problem with Time Windows (ADTSPTW) in chapter 7. The results indicated that only very small performance improvements could be achieved by using the proposed a-priori information based repositioning policies. However, we still believe that the availability of a-priori information on future requests could prove very beneficial for designing more sophisticated methods.

A real-life instance of the dynamic routing problem originating when a long-distance courier mail service provider has to design routes to pick-up or deliver mail was examined in chapter 8. The analysis of this problem showed that this particular instance of the vehicle routing problem could be seen as two separate sub-problems. The delivery problem could be seen as a conventional static VRPTW performed during the morning hours of the day of operation. However, the pick-up problem, which is mainly occured during the afternoon, is a highly dynamic problem, as only the minority of these customers are known in advance. The proposed repositioning policies of chapter 7 only showed marginal improvements in the performance with respect to the lateness.

Finally, in chapter 9 we discussed the present state of the art of the methodology applied to the DVRP. Also, a number of improvements and directions for further research were provided.

We believe that the applications examined in this thesis show how difficult it is to build general and generic algorithms for solving DVRPs. In other words methods that will work well for one problem might perform rather poorly for other problems.

10.1 Scientific Contributions

Two new methods of measuring the degree of dynamism taking into consideration the time the requests are received by the dispatchers were proposed in chapter 4. A taxonomy for categorizing practical applications based on the degree of dynamism and the objective function was proposed.

In chapter 5 the Partially Dynamic Traveling Repairman Problem (PDTRP) was introduced as an extension of the Dynamic Traveling Repairman Problem embracing advance requests as well as immediate request customers.

Furthermore, in chapter 6 we turned the attention towards using batching strategies as an alternative to pure re-optimization each time new information is received.

In chapter 7 the A-priori Dynamic Traveling Salesman Problem with Time Windows (ADTSPTW) was introduced and a number of on-line policies for repositioning the vehicle were proposed and examined empirically.

In chapter 8 a real-life instance of the A-priori Dynamic Traveling Salesman Problem with Time Windows was studied and a number of interesting key figures were provided.

Finally, in chapter 9 we outlined an insertion heuristic using a-priori information to insert dummy customers at locations, where the immediate requests are most likely to appear.

Bibliography

[1] John J. Bartholdi III, Loren K. Platzman, R. Lee Collins, and William H. Warden III. A Minimal Technology Routing System for Meals on Wheels. *Interfaces*, 13:1–8, 1983.

[2] Dimitris Bertsimas. A Vehicle Routing Problem With Stochastic Demand. *Operations Research*, 40:574–585, 1992.

[3] Dimitris Bertsimas and Louis H. Howell. Further Results on the Probabilistic Traveling Salesman Problem. *European Journal of Operational Research*, 65:68–95, 1993.

[4] Dimitris Bertsimas, Patrick Jaillet, and Amedeo R. Odoni. A Priori Optimization. *Operations Research*, 38(6):1019–1033, 1990.

[5] Dimitris Bertsimas and David Simchi-Levi. A New Generation of Vehicle Routing Research: Robust Algorithms, Addressing Uncertainty. *Operations Research*, 44:286–304, 1996.

[6] Dimitris Bertsimas and Garrett Van Ryzin. A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane. *Operations Research*, 39:601–615, 1991.

[7] Dimitris Bertsimas and Garrett Van Ryzin. Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane with Multiple Capacitated Vehicles. *Operations Research*, 41:60–76, 1993.

[8] Dimitris Bertsimas and Garrett Van Ryzin. Stochastic and Dynamic Vehicle Routing with General Demand and Interarrival Time Distributions. *Applied Probability*, 25:947–978, 1993.

[9] G. Clarke and J.W. Wright. Scheduling of Vehicles from a Depot to a number of Delivery Points. *Operations Research*, 12:568–581, 1964.

[10] J. Collins. *Global Positioning System*. Springer Verlag, Wien, New York, 1992.

[11] Jacques Desrosiers, Yvan Dumas, Marius M. Solomon, and Francois Soumis. *Handbooks in Operations Research and Management Science*, volume 8, chapter Time Constrained Routing and Scheduling, pages 35–140. Elsevier Science, Amsterdam, 1995.

[12] Robert B. Dial. Autonomous Dial a Ride Transit - Introductory Overview. *Transportation Research - Part C*, 3:261–275, 1995.

[13] Moshe Dror, Gilbert Laporte, and Pierre Trudeau. Vehicle Routing with Stochastic Demands: Properties and Solution Frameworks. *Transportation Science*, 23:166–176, 1989.

[14] Marshall L. Fischer. *Network Routing*, volume 8, chapter Vehicle Routing, pages 1–33. Elsevier Science, Amsterdam, 1995.

[15] Michel Gendreau, Francois Guertin, Jean-Yves Potvin, and Éric Taillard. Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching. *Transportation Science*, 33(4):381–390, 1999.

[16] Michel Gendreau, Gilbert Laporte, and René Séguin. A Tabu Search Heuristic for the Vehicle Routing Problem with Stochastic Demands and Customers. Technical report, Centre de Recherche sur les Transport, Universite de Montreal, 1994.

[17] Michel Gendreau, Gilbert Laporte, and René Séguin. An Exact algorithm for the Vehicle Routing Problem with stochastic Customers and Demands. *Transp. Sci.*, 29:143–155, 1995.

[18] Michel Gendreau, Gilbert Laporte, and René Séguin. Stochastic Vehicle Routing. *European Journal of Operational Research*, 88:3–12, 1996.

[19] Michel Gendreau, Gilbert Laporte, and Francois Semet. Solving an Ambulance Location Model by Tabu Search. *Location Science*, 5(2):75–88, 1997.

[20] Michel Gendreau and Jean-Yves Potvin. *Fleet Management and Logistics*, chapter Dynamic Vehicle Routing and Dispatching, pages 115–126. Kluwer Academic Publishers, 1998.

[21] Soumia Ichoua, Michel Gendreau, and Jean-Yves Potvin. Diversion Issues in Real-Time Vehicle Dispatching. Technical report, Centre de Recherche sur les Transports, Univerity of Montreal, April 1999.

[22] Patrick Jaillet. *Probabilistic Traveling Salesman Problems.* PhD thesis, Massachusetts Institute of Technology, Cambridge, Mass., 1985.

[23] Patrick Jaillet. A Priori Solution of a Traveling Salesman Problem in which a Random Subset of the Customers are Visited. *Operations Research*, 36:929–936, 1988.

[24] Antoine Jezequel. Probabilistic Vehicle Routing Problems. Master's thesis, Massachusetts Institute of Technology, Cambridge, Mass., 1985.

[25] Michael Jørgensen. Distribution of Everyday Necessities - Minimization of the Fleet Size (in Danish). Master's thesis, The Institute of Mathematical Statistics and Operations Research, The Technical University of Denmark, DK-2800 Lyngby, 1984.

[26] Gilbert Laporte, Francois V. Louveaux, and Hélène Mercure. A priori Optimization of the Probabilistic Traveling Salesman Problem. *Operations Research*, 42:543–549, 1994.

[27] Allan Larsen, Oli B.G. Madsen, and Marius M. Solomon. Partially Dynamic Vehicle Routing - Models and Algorithms. Technical report, The Department of Mathematical Modelling, The Technical University of Denmark, 1999.

[28] Richard C. Larson and Amadeo R. Odoni. *Urban Operations Research.* Prentice Hall, Englewood Cliffs, New Jersey, 1980.

[29] Karsten Lund, Oli B. G. Madsen, and Jens M. Rygaard. Vehicle Routing Problems with Varying Degrees of Dynamism. Technical report, IMM, The Department of Mathematical Modelling, Technical University of Denmark, 1996.

[30] Oli B.G. Madsen, Hans F. Ravn, and Jens M. Rygaard. A Heuristic Algorithm for a Dial-a-Ride Problem with Time Windows, Multiple Capacities and Multiple Objectives. *Ann. of Oper. Res.*, 60:193–208, 1995.

[31] Chryssi Malandraki. *Time Dependent Vehicle Routing Problems: Formulations, Solution Algorithms and Computational Experiments.* PhD thesis, Nortwestern University, 1989.

[32] Chryssi Malandraki and Mark S. Daskin. Time Dependent Vehicle Routing Problems: Formulations, Properties and Heuristic Algorithms. *Transportation Science*, 26:185–200, 1992.

[33] Jason D. Papastavrou. A Stochatic and Dynamic Routing Policy using Branching Processes with State Dependent Immigration. *European Journal of Operational Research*, 95(1):167–177, 1996.

[34] Warren B. Powell, Patrick Jaillet, and Amadeo Odoni. *Network Routing*, volume 8, chapter Stochastic and Dynamic Networks and Routing, pages 141–295. Elsevier Science, Amsterdam, 1995.

[35] Harilaos N. Psaraftis. A Dynamic Programming Solution to the single Many-to-many Immediate Request Dial-a-Ride Problem. *Transportation Science*, 14:130–154, 1980.

[36] Harilaos N. Psaraftis. *Vehicle Routing: Methods and Studies*, chapter Dynamic Vehicle Routing Problems, pages 223–248. Elsevier Science Publishers B.V. (North Holland), 1988.

[37] Harilaos N. Psaraftis. Dynamic vehicle routing: Status and prospects. *Ann. of Oper. Res.*, 61:143–164, 1995.

[38] Harilaos N. Psaraftis, James B. Orlin, Daniel Bienstock, and Paul M. Thompson. Analysis and Solution Algorithms of Sealift Routing and Scheduling Problems: Final Report. Technical report, Sloan School of Management, M.I.T., 1985.

[39] Sheldon M. Ross. *A Course in Simulation.* Maxwell Macmillan International Editions, 1990.

[40] Réne Séguin. *Problémes stochastiques de tournées de véhicules.* PhD thesis, Départment d'informatique et de reserche opérationelle, Centre de reserche sur les transports, Université de Montréal, 1994. CRT-979.

[41] Marius Solomon. Algorithms for the Vehicle Routing and Scheduling Problem with Time Window Constraints. *Operations Research*, 32:254–265, 1987.

[42] Michael R. Swihart and Jason D. Papastavrou. A Stochastic and Dynamic Model for the Single-Vehicle Pick-Up and Delivery Problem. *European Journal of Operational Research*, 114:447–464, 1999.

[43] Andres Weintraub, J. Aboud, C. Fernandez, G. Laporte, and E. Ramirez. An Emergency Vehicle Dispatching System for an Electric Utility in Chile. *Journal of the Operational Research Society*, 50:690–696, 1999.

[44] Jian Yang, Patrick Jaillet, and Hani S. Mahmassani. On-Line Algorithms for Truck Fleet Assignment and Scheduling under Real-Time Information. Technical report, Department of Management Science and Information Systems, The University of Texas at Austin, 1999.

Appendix A

Data Files

In this appendix we provide an example of the randomly generated data used for the empirical analyses throughout this thesis. The code for generating the data instances can be obtained by contacting the author of this thesis.

The syntax of the data files is inspired by the well-known test instances provided by Solomon [41]. For the sake of convenience we used the syntax for the analyses of both the PDTRP, the DVRPTW and the ADTSPTW. Naturally, the time windows and the capacities were not used for the PDTRP analysis and the capacities were not used for the study of the ADTSPTW.

The first two numbers denote the number of advance and immediate requests respectively. Hence, the example below consists of 26 advance and 11 immediate requests. The next number denotes the maximum number of vehicles that could be used to serve the requests. This problem should therefore be served by a maximum of 20 vehicles. Finally, the fourth number in the header denotes the capacity of each vehicle.

The data section is divided into two sections. The first section consists of a single line which shows the data for the depot. While the other section consists of one line for each request. The first column denotes the number of the request, the second and third columns denote the X and Y coordinate of the request. The fourth column denotes the time the request was received by the dispatchers. The fifth and sixth columns denote the time window of the request. The seventh column denotes the on-site service time in minutes. The last column denotes the demand of the request.

In the example below the depot is located at (5000.00, 5000.00) and opens at 480.00 and closes at 1500.00. The request time (which is 0 for the depot) is not used. The first request is an advance request customer and it is located at (1253.82, 968.18). The time window is [547.80, 598.62] while the on-site service time is 1.73 minutes and the demand is 10 units. The 27th request is the first immediate request customer and the request was received at 591.63.

```
 26
 11
 20
200
  0   5000.00   5000.00      0.00    480.00   1500.00    0.00    0.00
  1   1253.82    968.18    480.00    547.80    598.62    1.73   10.00
  2   1384.72   1762.99    480.00    687.39    745.19    2.73   10.00
  3   3094.20   2550.53    480.00    506.85    549.59    5.91   10.00
  4   9576.48   4654.48    480.00    764.75    812.89    1.51   10.00
  5   9573.84   6350.61    480.00    875.76    911.06    3.64   10.00
  6   6993.13   3839.67    480.00    975.30   1020.90    2.59   10.00
  7   9552.93   5550.80    480.00    603.02    646.86    1.78   10.00
  8   7212.07   3837.57    480.00    737.06    775.18    2.14   10.00
  9   3805.87   5575.32    480.00   1011.97   1069.17    2.34   10.00
 10   5516.84   6275.46    480.00    782.70    819.61    6.94   10.00
 11   5595.78   6238.85    480.00    974.53   1031.19    0.85   10.00
 12    311.19   5292.99    480.00    635.03    666.21    2.59   10.00
 13   4637.64   9521.33    480.00    973.20   1027.55    1.99   10.00
 14   4795.22   8859.28    480.00    527.39    573.75    3.69   10.00
 15   5385.45   8476.86    480.00    729.19    784.72    1.94   10.00
 16   6403.13   7349.17    480.00    880.61    911.55    1.21   10.00
 17   4504.05   9115.44    480.00    737.87    780.39    2.99   10.00
 18   6240.91   8419.07    480.00    546.58    578.69    0.75   10.00
 19   4706.14   7269.89    480.00    756.15    795.58    2.61   10.00
 20   3478.20   9216.48    480.00    936.46    987.90    0.33   10.00
 21   8675.88   9018.62    480.00    691.93    725.07    5.25   10.00
 22   9133.04   9311.02    480.00    995.99   1054.25    2.12   10.00
 23   9991.62   8832.03    480.00    923.05    972.82    0.48   10.00
 24   9089.24   9620.14    480.00    948.02   1000.75    1.07   10.00
 25   9458.15   8200.92    480.00    814.47    846.23    2.07   10.00
 26   9269.72   6840.59    480.00   1031.76   1064.38    1.00   10.00
 27   8894.78   9036.10    591.63    729.46    786.25    1.75   10.00
 28   9570.26   8845.63    638.76   1020.16   1077.74    1.39   10.00
 29   9250.81   8697.04    675.70    909.96    940.39    0.55   10.00
 30   7536.03   9437.90    720.38    924.85    973.60    2.30   10.00
 31   9103.49   7900.10    798.86    876.22    926.37    3.22   10.00
 32   7532.72   9217.40    814.76    953.34    985.51    5.18   10.00
 33   9809.07   9348.79    864.98   1021.56   1059.58    5.90   10.00
 34   9132.22   6711.71    865.68    949.31   1006.84    4.99   10.00
 35   7275.50   9101.40    886.26    949.70    991.06    3.60   10.00
 36   7262.64   7725.94    921.37    999.45   1059.06    2.35   10.00
 37   9661.63   7808.42    949.56   1019.44   1064.12    1.70   10.00
```

Appendix B

Partially Dynamic Traveling Repairman Problem - Simulation Results

In this appendix the log files of a single instance of the PDTRP comprising of 30 customers, of which 22 were advance requests and 8 immediate requests will be shown. The tables list the customer number, the time the request was received (8:00:00 represents an advance request customer), the time the service was begun at the customer for the policy, the on-site service time, the waiting time (for the immediate requests) and the geographical location of the customer, respectively.

Policy: Nearest Neighbor - FCFS

```
        Request     Start       On-site      Waiting
        time        service     service      time
Cust.   (hh:mm:ss)  (hh:mm:ss)  time (min)   (hh:mm:ss)  Location
-----------------------------------------------------------------

Depot                                                   [5000.0,5000.0]
```

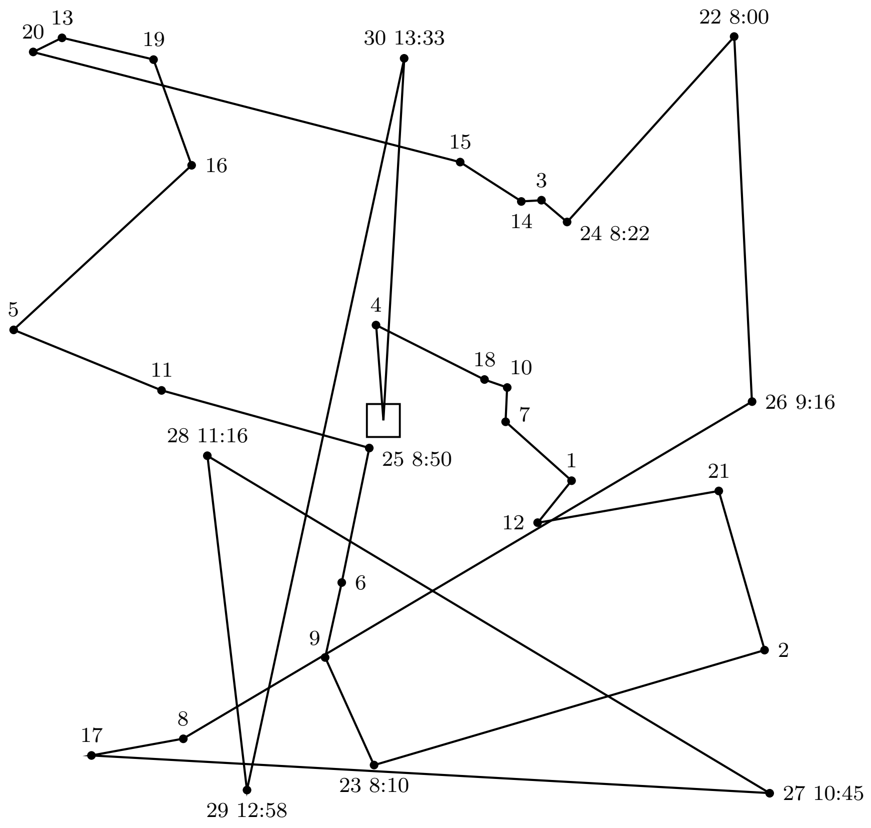


Figure B.1: A Nearest Neighbor - FCFS policy scenario.

```
  4     8:00:00    8:01:51    1.05       Static   [4910.9,6230.7]
 18     8:00:00    8:05:09    4.74       Static   [6236.2,5525.7]
 10     8:00:00    8:10:20    3.30       Static   [6513.4,5424.6]
  7     8:00:00    8:14:17    2.72       Static   [6494.2,4983.0]
  1     8:00:00    8:18:40    1.70       Static   [7300.5,4221.1]
 12     8:00:00    8:21:24    2.03       Static   [6886.9,3677.3]
 21     8:00:00    8:26:49    2.10       Static   [9102.3,4087.6]
  2     8:00:00    8:32:06    2.48       Static   [9662.7,2031.5]
 23     8:09:43    8:42:05    2.28      0:32:22   [4885.3, 546.3]
  9     8:00:00    8:46:38    2.16       Static   [4288.3,1937.7]
  6     8:00:00    8:50:17    1.59       Static   [4489.5,2903.8]
 25     8:50:23    8:54:32    8.79      0:04:08   [4826.6,4643.8]
 11     8:00:00    9:07:17    3.46       Static   [2286.2,5388.3]
```

```
  5      8:00:00     9:13:42      0.97        Static   [ 475.6,6169.9]
 16      8:00:00     9:19:15      4.14        Static   [2654.3,8295.4]
 19      8:00:00     9:25:33      5.74        Static   [2188.2,9661.7]
 13      8:00:00     9:33:01      1.99        Static   [1069.9,9942.6]
 20      8:00:00     9:35:36      4.05        Static   [ 715.6,9758.0]
 15      8:00:00     9:47:47      1.27        Static   [5939.0,8335.7]
 14      8:00:00     9:50:24      2.52        Static   [6688.0,7829.2]
  3      8:00:00     9:53:18      1.35        Static   [6932.8,7843.1]
 24      8:21:43     9:55:16      2.40       1:33:33   [7246.7,7562.4]
 22      8:00:02    10:02:24      3.02       2:02:21   [9290.6,9957.0]
 26      9:16:17    10:12:30      3.50       0:56:12   [9507.2,5241.5]
  8      8:00:00    10:28:18      2.96        Static   [2551.1, 888.2]
 17      8:00:00    10:32:59      1.08        Static   [1428.6, 671.3]
 27     10:45:15    10:57:42      1.11       0:12:27   [9723.0, 185.1]
 28     11:15:36    11:27:49      0.99       0:12:12   [2848.2,4542.5]
 29     12:58:09    13:04:40      1.49       0:06:31   [3330.7, 223.2]
 30     13:32:49    13:47:18      2.33       0:14:28   [5255.1,9677.2]

Depot               13:56:39                           [5000.0,5000.0]


Degree of dynamism:            30

Total route length (km):     88.91
Total waiting time (min):  1709.19

Avg. waiting time for imm.req. cust. (min):    39.37
```

Policy: PART - 4 subregions

```
        Request     Start       On-site      Waiting
        time        service     service      time
Cust.   (hh:mm:ss)  (hh:mm:ss)  time (min)   (hh:mm:ss)  Location
-----------------------------------------------------------------

Depot                                                  [5000.0,5000.0]

  4      8:00:00     8:01:51      1.05        Static   [4910.9,6230.7]
 11      8:00:00     8:07:02      3.46        Static   [2286.2,5388.3]
  5      8:00:00     8:13:27      0.97        Static   [ 475.6,6169.9]
 16      8:00:00     8:18:59      4.14        Static   [2654.3,8295.4]
 19      8:00:00     8:25:17      5.74        Static   [2188.2,9661.7]
 13      8:00:00     8:32:45      1.99        Static   [1069.9,9942.6]
 20      8:00:00     8:35:21      4.05        Static   [ 715.6,9758.0]
  6      8:00:00     8:51:08      1.59        Static   [4489.5,2903.8]
  9      8:00:00     8:54:12      2.16        Static   [4288.3,1937.7]
 23      8:09:43     8:58:38      2.28       0:48:55   [4885.3, 546.3]
  8      8:00:00     9:04:27      2.96        Static   [2551.1, 888.2]
```

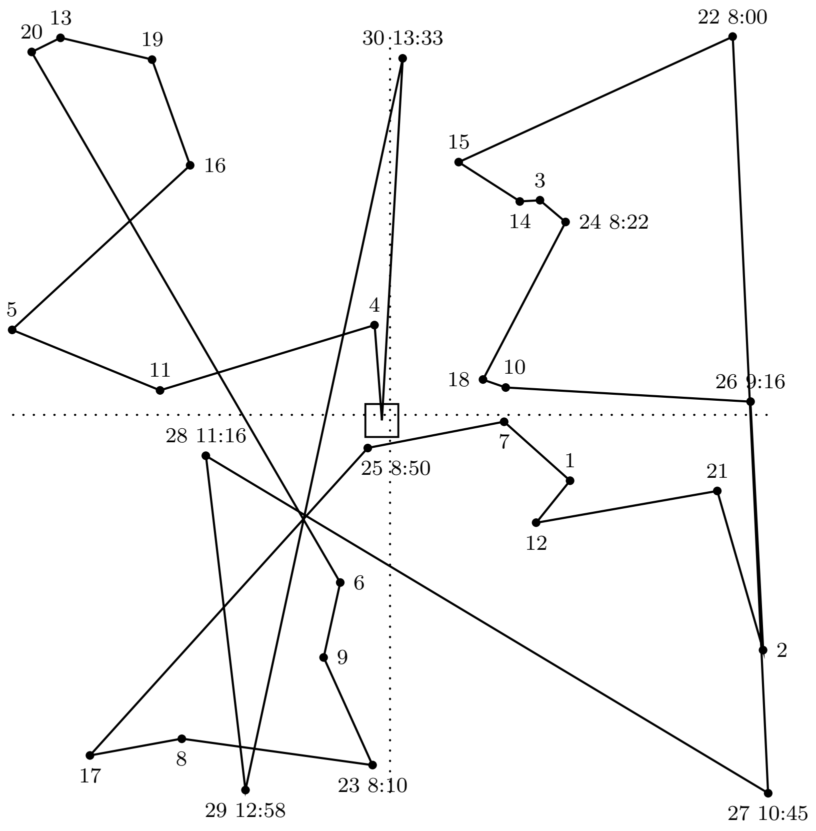


Figure B.2: A PART policy with 4 subregions.

```
 17      8:00:00     9:09:08       1.08         Static   [1428.6, 671.3]
 25      8:50:23     9:18:03       8.79        0:27:39   [4826.6,4643.8]
  7      8:00:00     9:29:23       2.72         Static   [6494.2,4983.0]
  1      8:00:00     9:33:46       1.70         Static   [7300.5,4221.1]
 12      8:00:00     9:36:30       2.03         Static   [6886.9,3677.3]
 21      8:00:00     9:41:55       2.10         Static   [9102.3,4087.6]
  2      8:00:00     9:47:12       2.48         Static   [9662.7,2031.5]
 26      9:16:17     9:54:30       3.50        0:38:13   [9507.2,5241.5]
 10      8:00:00    10:02:30       3.30         Static   [6513.4,5424.6]
 18      8:00:00    10:06:15       4.74         Static   [6236.2,5525.7]
 24      8:21:43    10:14:24       2.40        1:52:40   [7246.7,7562.4]
  3      8:00:00    10:17:26       1.35         Static   [6932.8,7843.1]
 14      8:00:00    10:19:09       2.52         Static   [6688.0,7829.2]
 15      8:00:00    10:23:01       1.27         Static   [5939.0,8335.7]
 22      8:00:02    10:29:53      3.02       2:29:50   [9290.6,9957.0]
 27     10:45:15    10:59:55      1.11       0:14:40   [9723.0, 185.1]
 28     11:15:36    11:27:49      0.99       0:12:12   [2848.2,4542.5]
 29     12:58:09    13:04:40      1.49       0:06:31   [3330.7, 223.2]
 30     13:32:49    13:47:18      2.33       0:14:28   [5255.1,9677.2]

Depot               13:56:39                           [5000.0,5000.0]


Degree of dynamism:             30

Total route length (km):     89.61
Total waiting time (min):  1956.12

Avg. waiting time for imm.req. cust. (min):    47.25
```

Appendix C

A-priori Traveling Salesman Problem with Time Windows - Simulation Results

In this appendix the results of the simulation of the four scenarios of the chapter on the ADTSPTW will be presented. First the results will be presented in tables. Then, the actual routes constructed by the ADTSPTW policies and corresponding log files of a single instance of the `Scenario A - NARROW` dataset will be presented.

C.1 Simulation Results

In the tables C.1 through C.8 the results of the simulation of the ADT-SPTW are shown. For all combinations of the policies implemented and the L_{thres} parameter values used the average distance, the standard deviation of the distance, the number of instances where the policy performed uniformly best, the average lateness, the standard deviation of the lateness and finally the number of instances where the policy performed uniformly best are given.

A - NARROW		SERIES I					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	128.12	19.55	344	276.66	211.92	21
NEAREST	0.2	132.02	20.31	67	278.98	214.97	106
NEAREST	0.4	130.43	20.39	48	278.30	214.17	17
NEAREST	0.6	129.54	20.39	35	276.48	213.44	9
NEAREST	0.8	128.90	20.21	18	276.69	212.45	3
BUSIEST	0.2	137.50	21.54	32	279.98	215.91	100
BUSIEST	0.4	134.57	21.76	17	279.68	213.17	20
BUSIEST	0.6	132.65	21.74	15	278.61	214.39	6
BUSIEST	0.8	130.95	21.64	7	277.85	213.33	2
HI-REQ	0.2	144.35	22.50	18	288.92	221.48	177
HI-REQ	0.4	138.37	22.36	22	282.42	213.90	64
HI-REQ	0.6	134.44	21.91	14	277.53	214.00	46
HI-REQ	0.8	131.44	21.54	24	276.52	213.46	21

Table C.1: `Scenario A - NARROW`. `SERIES I` - min. of the distance.

A - NARROW		SERIES II					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	138.39	19.37	317	11.80	24.02	3
NEAREST	0.2	142.16	20.11	52	11.56	23.59	18
NEAREST	0.4	140.54	20.17	36	11.63	23.90	1
NEAREST	0.6	139.73	20.05	19	11.69	23.90	0
NEAREST	0.8	139.13	19.90	9	11.76	24.03	1
BUSIEST	0.2	148.82	20.66	7	11.54	23.78	14
BUSIEST	0.4	144.99	21.46	4	11.46	23.56	1
BUSIEST	0.6	142.77	21.67	1	11.70	23.95	0
BUSIEST	0.8	140.89	21.36	1	11.78	24.01	0
HI-REQ	0.2	149.15	20.46	23	11.83	24.20	41
HI-REQ	0.4	144.63	21.03	12	11.57	23.80	1
HI-REQ	0.6	142.37	21.21	4	11.75	23.96	1
HI-REQ	0.8	140.60	20.92	9	11.82	24.03	0

Table C.2: `Scenario A - NARROW`. `SERIES II` - min. of the lateness.

B - NARROW		SERIES I					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	125.23	19.73	406	275.33	221.94	46
NEAREST	0.2	129.28	20.07	81	278.51	224.29	93
NEAREST	0.4	127.90	20.34	44	276.88	222.32	13
NEAREST	0.6	126.95	20.36	37	275.63	221.72	12
NEAREST	0.8	126.29	20.39	34	274.64	222.38	4
BUSIEST	0.2	135.87	21.63	31	277.99	221.69	84
BUSIEST	0.4	132.78	21.95	16	278.69	223.77	11
BUSIEST	0.6	130.56	22.07	13	277.27	222.23	16
BUSIEST	0.8	128.46	22.08	27	277.85	222.66	9
HI-REQ	0.2	142.90	22.63	16	278.32	220.09	164
HI-REQ	0.4	137.64	22.75	14	277.84	223.22	65
HI-REQ	0.6	133.31	22.75	16	277.23	223.05	34
HI-REQ	0.8	129.81	22.30	15	277.95	222.68	16

Table C.3: `Scenario B - NARROW`. `SERIES I` - min. of the distance.

B - NARROW		SERIES II					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	135.83	19.76	388	11.96	22.82	9
NEAREST	0.2	139.75	19.73	63	11.81	22.08	28
NEAREST	0.4	138.06	20.05	44	11.83	22.64	3
NEAREST	0.6	137.23	20.19	37	12.04	22.89	0
NEAREST	0.8	136.68	20.20	13	12.01	22.82	0
BUSIEST	0.2	147.96	21.17	6	11.81	22.59	22
BUSIEST	0.4	143.63	21.71	10	11.86	22.63	3
BUSIEST	0.6	140.87	22.05	5	11.99	22.76	0
BUSIEST	0.8	138.99	22.05	1	12.02	22.81	0
HI-REQ	0.2	149.34	21.35	12	11.97	22.36	49
HI-REQ	0.4	144.40	21.84	10	12.03	22.78	7
HI-REQ	0.6	141.17	22.15	8	12.04	22.75	3
HI-REQ	0.8	138.97	21.95	3	12.00	22.82	1

Table C.4: `Scenario B - NARROW`. `SERIES II` - min. of the lateness.

A - WIDE		SERIES I					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	126.40	19.66	320	122.91	142.00	18
NEAREST	0.2	130.00	20.20	77	125.36	145.67	53
NEAREST	0.4	128.69	20.37	49	122.27	143.81	8
NEAREST	0.6	127.90	20.36	30	121.34	142.31	3
NEAREST	0.8	127.27	20.24	37	121.94	141.67	2
BUSIEST	0.2	134.54	21.66	36	123.19	149.07	70
BUSIEST	0.4	132.45	21.75	17	123.10	147.46	10
BUSIEST	0.6	130.85	21.87	18	124.62	148.34	7
BUSIEST	0.8	129.46	21.88	10	123.35	143.64	4
HI-REQ	0.2	142.37	22.20	18	128.29	141.98	137
HI-REQ	0.4	136.77	22.32	13	124.80	145.78	47
HI-REQ	0.6	132.99	22.11	22	125.68	148.00	25
HI-REQ	0.8	129.98	21.59	12	123.20	143.41	16

Table C.5: `Scenario A - WIDE`. `SERIES I` - min. of the distance.

A - WIDE		SERIES II					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	130.85	19.65	292	5.51	13.21	2
NEAREST	0.2	134.30	20.29	89	5.60	13.21	8
NEAREST	0.4	132.85	20.48	54	5.64	13.59	0
NEAREST	0.6	132.01	20.45	34	5.64	13.41	0
NEAREST	0.8	131.54	20.22	23	5.50	13.12	0
BUSIEST	0.2	140.33	21.60	16	5.49	12.26	13
BUSIEST	0.4	137.29	22.00	13	5.70	13.70	1
BUSIEST	0.6	135.21	22.19	8	5.60	13.53	0
BUSIEST	0.8	133.49	21.99	1	5.46	13.11	0
HI-REQ	0.2	143.48	21.42	12	5.71	12.59	30
HI-REQ	0.4	138.05	21.69	13	5.67	13.62	1
HI-REQ	0.6	135.17	21.68	11	5.54	13.35	0
HI-REQ	0.8	133.22	21.49	7	5.47	13.12	0

Table C.6: `Scenario A - WIDE`. `SERIES II` - min. of the lateness.

B - WIDE		SERIES I					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	123.40	20.24	399	119.66	144.86	26
NEAREST	0.2	127.21	20.40	69	123.16	152.06	61
NEAREST	0.4	125.90	20.69	45	123.80	153.68	5
NEAREST	0.6	125.06	20.80	36	121.51	148.18	6
NEAREST	0.8	124.29	20.80	44	120.48	145.84	2
BUSIEST	0.2	133.09	22.09	23	122.37	148.26	64
BUSIEST	0.4	130.61	22.28	17	123.22	149.31	10
BUSIEST	0.6	128.76	22.54	15	119.95	146.36	5
BUSIEST	0.8	126.86	22.59	32	119.18	145.81	8
HI-REQ	0.2	141.57	23.12	20	124.27	156.85	139
HI-REQ	0.4	136.27	22.93	7	123.62	153.31	38
HI-REQ	0.6	132.13	22.97	10	120.11	148.07	36
HI-REQ	0.8	128.41	22.75	28	120.07	147.05	14

Table C.7: `Scenario B - WIDE`. `SERIES I` - min. of the distance.

B - WIDE		SERIES II					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	128.23	20.04	361	5.70	14.02	7
NEAREST	0.2	131.83	20.46	75	5.83	12.77	18
NEAREST	0.4	130.59	20.64	44	5.82	13.08	2
NEAREST	0.6	129.70	20.64	30	5.67	12.63	0
NEAREST	0.8	129.08	20.59	18	5.80	14.14	0
BUSIEST	0.2	138.43	22.05	25	6.20	13.56	18
BUSIEST	0.4	135.40	22.41	16	5.99	13.50	1
BUSIEST	0.6	133.29	22.49	15	5.75	12.75	1
BUSIEST	0.8	131.39	22.43	8	5.79	14.02	0
HI-REQ	0.2	142.90	22.36	15	6.35	14.45	30
HI-REQ	0.4	137.72	22.68	22	5.78	13.46	1
HI-REQ	0.6	134.33	22.58	16	5.72	12.72	1
HI-REQ	0.8	131.77	22.50	5	5.79	14.00	2

Table C.8: `Scenario B - WIDE`. `SERIES II` - min. of the lateness.

C.2 Actual Routes and Log Files.

In this section the actual routes constructed by the ADTSPTW policies implemented are shown along with a listing of the log files for the pure re-optimization policy, NO REPOS, the HI-REQ and the NEAREST policies respectively.

The dotted lines in Figure C.1 divide the service region into the 9 subregions. The longer dashed lines represent the beginning and the end of the route.

In Figures C.2 and C.3 the dotted lines going from a customer to an idle point represent reposition movements.

NO REPOS policy

```
        Request     Start       On-site                       Time            TW
        time        service     service                       Window          violation
Cust.   (hh:mm:ss)  (hh:mm:ss)  time (min)   Location      (hh:mm)         (hh:mm:ss)
------------------------------------------------------------------------------------------

Depot                                         [5000,5000]
  4       8:00:00    10:10:28      8.43       [6552,3015]   [10:10,10:58]          -
  1       8:00:00    10:26:01      0.73       [2255,1005]   [ 8:18, 9:30]     0:55:51
 19       8:00:00    10:38:32      0.69       [2769,8851]   [10:04,11:06]          -
 16       8:00:00    11:05:18      2.74       [2179,5158]   [11:05,11:54]          -
 17       8:00:00    11:10:00      3.34       [2846,6280]   [10:16,10:46]     0:23:56
 24       8:00:00    12:07:14      2.83       [8569,8815]   [12:07,12:41]          -
 20       8:00:00    12:13:16      2.67       [6490,9323]   [ 9:44,10:28]     1:45:34
 11       8:00:00    12:24:04      6.25       [3809,4613]   [ 9:51,10:59]     1:25:08
 13       8:00:00    12:32:58      2.59       [5419,5333]   [11:13,11:56]     0:36:44
  8       8:00:00    12:40:42      2.02       [8713,6273]   [11:09,11:57]     0:44:12
 26       8:28:31    12:48:00      3.99       [8909,2751]   [12:00,13:07]          -
  7       8:00:00    12:54:49      2.13       [7036,2946]   [12:21,13:13]          -
 22       8:00:00    13:03:54      5.84       [4569,6870]   [12:47,13:33]          -
 25       8:18:59    13:16:02      4.29       [ 604,5502]   [12:44,13:22]          -
  2       8:00:00    13:27:12      3.26       [2038,1150]   [12:35,13:41]          -
 34      12:17:19    14:29:35      1.63       [6554,1182]   [14:30,15:20]          -
  6       8:00:00    14:34:34      3.09       [8765, 839]   [13:05,13:59]     0:35:54
 32      11:28:17    14:42:39      2.76       [9672,4036]   [12:52,14:16]     0:26:20
  9       8:00:00    14:48:24      3.44       [9049,5938]   [13:25,13:59]     0:49:42
 27       9:00:40    14:55:05      2.31       [6925,5584]   [14:49,15:35]          -
 10       8:00:00    14:58:15      1.14       [6497,5202]   [13:03,13:47]     1:11:10
 33      11:35:51    15:01:28      1.93       [6744,3840]   [12:37,13:25]     1:36:36
  5       8:00:00    15:09:02      3.71       [4562, 778]   [15:01,15:36]          -
 12       8:00:00    15:18:40      1.61       [6004,4453]   [13:31,14:44]     0:34:40
 30      10:12:07    15:20:45      1.71       [5686,4470]   [14:54,15:45]          -
 31      11:07:07    15:25:10      1.34       [3898,4241]   [13:10,14:32]     0:52:51
 21       8:00:00    15:31:48      1.74       [5849,7179]   [13:43,14:17]     1:14:38
 29       9:57:39    16:23:20      5.77       [4328,7647]   [16:23,17:27]          -
 15       8:00:00    16:52:37      1.41       [4964,3720]   [16:53,17:57]          -
```

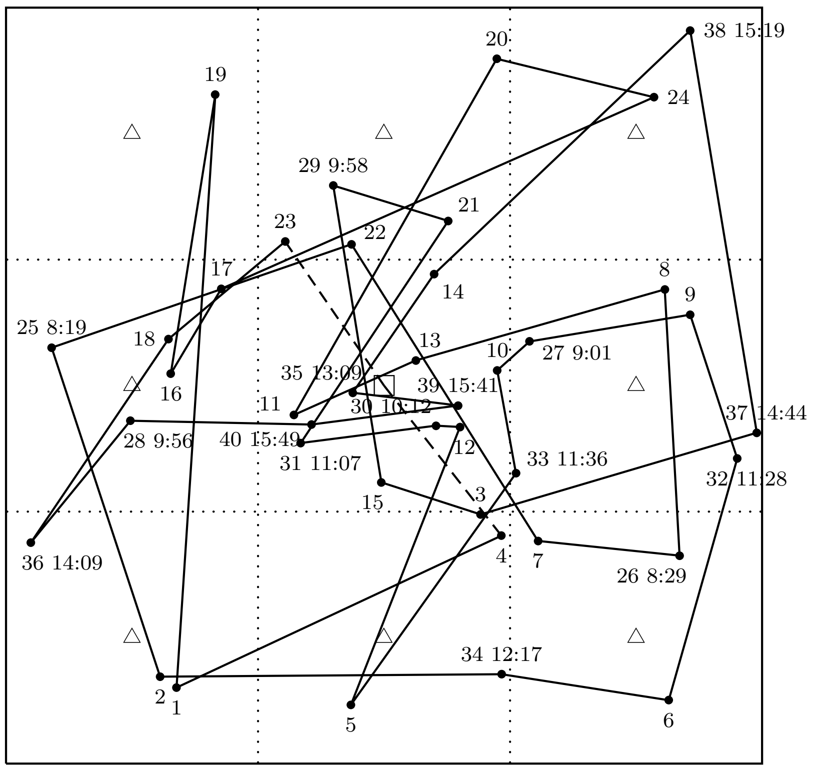


Figure C.1: ADTSPTW route found by the NO REPOS policy.

```
  3      8:00:00    16:56:06      2.86      [6277,3296]   [16:37,17:21]          -
 37     14:44:24    17:04:41      2.58      [9930,4377]   [15:59,16:32]    0:33:09
 38     15:19:07    17:15:21      1.71      [9050,9698]   [16:27,17:26]          -
 14      8:00:00    17:24:04      2.71      [5662,6476]   [17:11,17:50]          -
 35     13:09:02    17:29:38      7.58      [4586,4906]   [16:01,16:40]    0:49:28
 39     15:41:25    17:39:19      1.45      [5981,4737]   [16:50,17:43]          -
 40     15:49:00    17:43:42      1.97      [4040,4485]   [17:12,17:57]          -
 28      9:56:17    17:49:16      4.39      [1645,4535]   [14:53,16:20]    1:29:43
 36     14:08:37    17:56:47      1.68      [ 329,2924]   [16:58,17:58]          -
 18      8:00:00    18:03:20      2.53      [2149,5616]   [17:19,17:59]    0:04:18
 23      8:00:00    18:08:53      1.41      [3692,6906]   [16:49,17:19]    0:49:37
Depot               18:13:46                [5000,5000]

Total route length (km):       125.69
Total TW penalty time (min):  1019.67
```

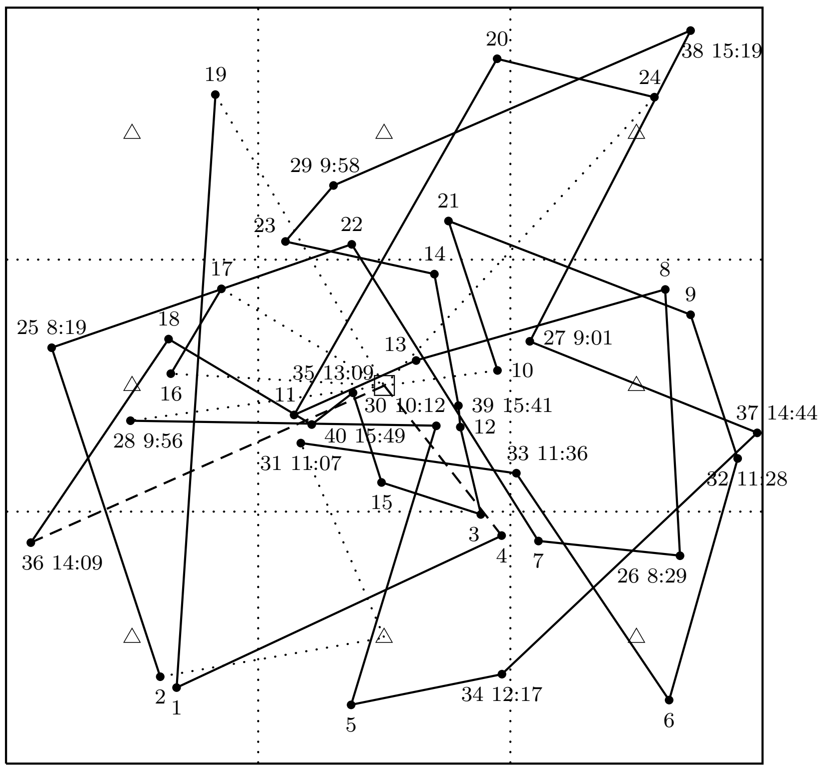


Figure C.2: ADTSPTW route found by the HI-REQ policy.

HI-REQ policy - $L_{thres} = 0.2$

```
        Request     Start       On-site                    Time            TW
        time        service     service                    Window          violation
Cust.   (hh:mm:ss)  (hh:mm:ss)  time (min)   Location      (hh:mm)         (hh:mm:ss)
------------------------------------------------------------------------------------

Depot                                      [5000,5000]
  4      8:00:00    10:10:28      8.43     [6552,3015]   [10:10,10:58]           -
  1      8:00:00    10:26:01      0.73     [2255,1005]   [ 8:18, 9:30]     0:55:51
 19      8:00:00    10:38:32      0.69     [2769,8851]   [10:04,11:06]           -
IP  5               10:47:37               [4999,4999]
 16      8:00:00    11:05:18      2.74     [2179,5158]   [11:05,11:54]           -
 17      8:00:00    11:10:00      3.34     [2846,6280]   [10:16,10:46]     0:23:56
IP  5               11:22:08               [4999,4999]
 24      8:00:00    12:07:14      2.83     [8569,8815]   [12:07,12:41]           -
 20      8:00:00    12:13:16      2.67     [6490,9323]   [ 9:44,10:28]     1:45:34
 11      8:00:00    12:24:04      6.25     [3809,4613]   [ 9:51,10:59]     1:25:08
 13      8:00:00    12:32:58      2.59     [5419,5333]   [11:13,11:56]     0:36:44
  8      8:00:00    12:40:42      2.02     [8713,6273]   [11:09,11:57]     0:44:12
 26      8:28:31    12:48:00      3.99     [8909,2751]   [12:00,13:07]           -
  7      8:00:00    12:54:49      2.13     [7036,2946]   [12:21,13:13]           -
 22      8:00:00    13:03:54      5.84     [4569,6870]   [12:47,13:33]           -
 25      8:18:59    13:16:02      4.29     [ 604,5502]   [12:44,13:22]           -
  2      8:00:00    13:27:12      3.26     [2038,1150]   [12:35,13:41]           -
IP  2               13:44:47               [4999,1666]
 31     11:07:07    14:12:49      1.34     [3898,4241]   [13:10,14:32]           -
 33     11:35:51    14:18:28      1.93     [6744,3840]   [12:37,13:25]     0:53:36
  6      8:00:00    14:25:49      3.09     [8765, 839]   [13:05,13:59]     0:27:09
 32     11:28:17    14:33:54      2.76     [9672,4036]   [12:52,14:16]     0:17:35
  9      8:00:00    14:39:40      3.44     [9049,5938]   [13:25,13:59]     0:40:57
 21      8:00:00    14:48:15      1.74     [5849,7179]   [13:43,14:17]     0:31:05
 10      8:00:00    14:53:06      1.14     [6497,5202]   [13:03,13:47]     1:06:02
IP  5               14:59:41               [4999,4999]
 28      9:56:17    15:46:30      4.39     [1645,4535]   [14:53,16:20]           -
 30     10:12:07    15:56:57      1.71     [5686,4470]   [14:54,15:45]     0:11:35
  5      8:00:00    16:04:27      3.71     [4562, 778]   [15:01,15:36]     0:28:07
 34     12:17:19    16:11:12      1.63     [6554,1182]   [14:30,15:20]     0:51:17
 37     14:44:24    16:19:49      2.58     [9930,4377]   [15:59,16:32]           -
 27      9:00:40    16:27:15      2.31     [6925,5584]   [14:49,15:35]     0:52:29
 38     15:19:07    16:36:30      1.71     [9050,9698]   [16:27,17:26]           -
 29      9:57:39    16:45:56      5.77     [4328,7647]   [16:23,17:27]           -
 23      8:00:00    16:53:10      1.41     [3692,6906]   [16:49,17:19]           -
 14      8:00:00    17:11:28      2.71     [5662,6476]   [17:11,17:50]           -
 39     15:41:25    17:16:50      1.45     [5981,4737]   [16:50,17:43]           -
 12      8:00:00    17:18:43      1.61     [6004,4453]   [13:31,14:44]     2:34:43
  3      8:00:00    17:22:06      2.86     [6277,3296]   [16:37,17:21]     0:00:59
 15      8:00:00    17:27:02      1.41     [4964,3720]   [16:53,17:57]           -
 35     13:09:02    17:30:19      7.58     [4586,4906]   [16:01,16:40]     0:50:09
 40     15:49:00    17:38:55      1.97     [4040,4485]   [17:12,17:57]           -
 18      8:00:00    17:44:12      2.53     [2149,5616]   [17:19,17:59]           -
 36     14:08:37    17:51:36      1.68     [ 329,2924]   [16:58,17:58]           -
Depot               18:00:57               [5000,5000]

Total route length (km):      138.75
Total TW penalty time (min):  937.29
```

NEAREST policy $L_{thres} = 0.2$

```
        Request     Start       On-site                   Time           TW
        time        service     service                   Window         violation
Cust.   (hh:mm:ss)  (hh:mm:ss)  time (min)   Location     (hh:mm)        (hh:mm:ss)
------------------------------------------------------------------------------------

Depot                                      [5000,5000]
  4      8:00:00    10:10:28      8.43     [6552,3015]   [10:10,10:58]           -
  1      8:00:00    10:26:01      0.73     [2255,1005]   [ 8:18, 9:30]     0:55:51
 19      8:00:00    10:38:32      0.69     [2769,8851]   [10:04,11:06]           -
 16      8:00:00    11:05:18      2.74     [2179,5158]   [11:05,11:54]           -
 17      8:00:00    11:10:00      3.34     [2846,6280]   [10:16,10:46]     0:23:56
 24      8:00:00    12:07:14      2.83     [8569,8815]   [12:07,12:41]           -
```

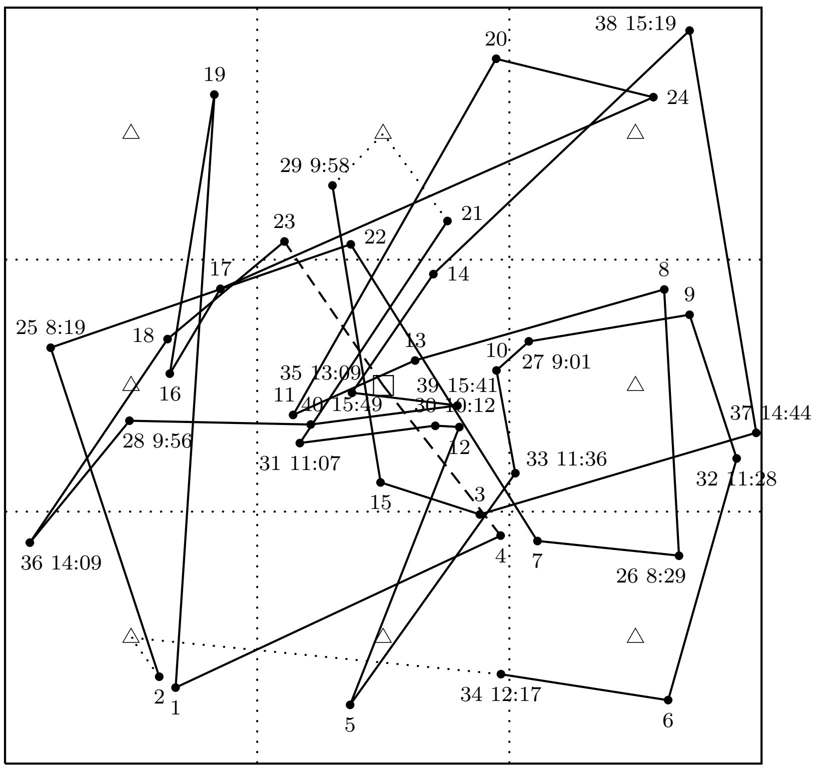


Figure C.3: ADTSPTW route found by the NEAREST policy.

20	8:00:00	12:13:16	2.67	[6490,9323]	[9:44,10:28]	1:45:34
11	8:00:00	12:24:04	6.25	[3809,4613]	[9:51,10:59]	1:25:08
13	8:00:00	12:32:58	2.59	[5419,5333]	[11:13,11:56]	0:36:44
8	8:00:00	12:40:42	2.02	[8713,6273]	[11:09,11:57]	0:44:12
26	8:28:31	12:48:00	3.99	[8909,2751]	[12:00,13:07]	-
7	8:00:00	12:54:49	2.13	[7036,2946]	[12:21,13:13]	-
22	8:00:00	13:03:54	5.84	[4569,6870]	[12:47,13:33]	-
25	8:18:59	13:16:02	4.29	[604,5502]	[12:44,13:22]	-
2	8:00:00	13:27:12	3.26	[2038,1150]	[12:35,13:41]	-
IP 1		13:31:25		[1666,1666]		
34	12:17:19	14:29:35	1.63	[6554,1182]	[14:30,15:20]	-
6	8:00:00	14:34:34	3.09	[8765, 839]	[13:05,13:59]	0:35:54
32	11:28:17	14:42:39	2.76	[9672,4036]	[12:52,14:16]	0:26:20
9	8:00:00	14:48:24	3.44	[9049,5938]	[13:25,13:59]	0:49:42
27	9:00:40	14:55:05	2.31	[6925,5584]	[14:49,15:35]	-
10	8:00:00	14:58:15	1.14	[6497,5202]	[13:03,13:47]	1:11:10

```
 33      11:35:51    15:01:28      1.93      [6744,3840]   [12:37,13:25]     1:36:36
  5       8:00:00    15:09:02      3.71      [4562, 778]   [15:01,15:36]           -
 12       8:00:00    15:18:40      1.61      [6004,4453]   [13:31,14:44]     0:34:40
 30      10:12:07    15:20:45      1.71      [5686,4470]   [14:54,15:45]           -
 31      11:07:07    15:25:10      1.34      [3898,4241]   [13:10,14:32]     0:52:51
 21       8:00:00    15:31:48      1.74      [5849,7179]   [13:43,14:17]     1:14:38
IP  8                15:35:41                [4999,8332]
 29       9:57:39    16:23:20      5.77      [4328,7647]   [16:23,17:27]           -
 15       8:00:00    16:52:37      1.41      [4964,3720]   [16:53,17:57]           -
  3       8:00:00    16:56:06      2.86      [6277,3296]   [16:37,17:21]           -
 37      14:44:24    17:04:41      2.58      [9930,4377]   [15:59,16:32]     0:33:09
 38      15:19:07    17:15:21      1.71      [9050,9698]   [16:27,17:26]           -
 14       8:00:00    17:24:04      2.71      [5662,6476]   [17:11,17:50]           -
 35      13:09:02    17:29:38      7.58      [4586,4906]   [16:01,16:40]     0:49:28
 39      15:41:25    17:39:19      1.45      [5981,4737]   [16:50,17:43]           -
 40      15:49:00    17:43:42      1.97      [4040,4485]   [17:12,17:57]           -
 28       9:56:17    17:49:16      4.39      [1645,4535]   [14:53,16:20]     1:29:43
 36      14:08:37    17:56:47      1.68      [ 329,2924]   [16:58,17:58]           -
 18       8:00:00    18:03:20      2.53      [2149,5616]   [17:19,17:59]     0:04:18
 23       8:00:00    18:08:53      1.41      [3692,6906]   [16:49,17:19]     0:49:37
Depot                18:13:46                [5000,5000]

Total route length (km):       127.52
Total TW penalty time (min):  1019.67
```

Appendix D

A Real-life Scenario - Simulation Results

In this appendix the results of the simulation of the real-life scenario treated in chapter 8 will be presented.

D.1 Simulation Results

In the tables D.1 through D.8 the results of the simulation of the ADT-SPTW are shown. For all combinations of the policies implemented and the L_{thres} parameter values used the average distance, the standard deviation of the distance, the number of instances where the policy performed uniformly best, the average lateness, the standard deviation of the lateness and finally the number of instances where the policy performed uniformly best are given.

D.2 Log Files

Below the log files for the first Monday instance of the `Area A` dataset are listed. This instance comprises a total of 71 customers, of which 64 are delivery customers (advance requests), a single request is an un-called

Area A		SERIES I					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	82.85	14.89	0	11.67	23.75	0
NEAREST	0.05	83.92	18.24	1	28.98	39.70	0
NEAREST	0.10	83.65	16.24	0	28.98	39.70	0
NEAREST	0.20	83.80	15.96	0	23.85	33.85	0
NEAREST	0.30	82.96	14.68	0	11.67	23.75	0
BUSIEST	0.05	85.20	17.72	0	16.47	26.25	0
BUSIEST	0.10	85.51	17.19	0	16.47	26.25	0
BUSIEST	0.20	84.94	17.25	0	15.58	26.72	0
BUSIEST	0.30	83.11	14.50	0	11.67	23.75	0
HI-REQ	0.05	84.88	17.92	0	21.61	34.80	0
HI-REQ	0.10	85.12	17.42	0	21.61	34.80	0
HI-REQ	0.20	84.88	17.32	0	15.58	26.72	0
HI-REQ	0.30	83.11	14.50	0	11.67	23.75	0

Table D.1: `Scenario Area A`. SERIES I - min. of the distance.

Area A		SERIES II					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	89.98	22.76	0	2.15	6.33	0
NEAREST	0.05	89.99	24.54	1	1.35	3.72	0
NEAREST	0.10	89.87	23.20	0	1.35	3.72	0
NEAREST	0.20	89.71	22.99	0	2.15	6.33	0
NEAREST	0.30	90.10	22.57	0	2.15	6.33	0
BUSIEST	0.05	91.16	23.84	0	2.80	6.42	0
BUSIEST	0.10	91.48	23.35	0	2.80	6.42	0
BUSIEST	0.20	90.82	23.30	0	2.15	6.33	0
BUSIEST	0.30	90.27	22.50	0	2.15	6.33	0
HI-REQ	0.05	91.23	24.02	0	2.00	4.04	0
HI-REQ	0.10	91.54	23.53	0	2.00	4.04	0
HI-REQ	0.20	90.76	23.38	0	2.15	6.33	0
HI-REQ	0.30	90.27	22.50	0	2.15	6.33	0

Table D.2: `Scenario Area A`. SERIES II - min. of the lateness.

Area B		SERIES I					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	48.50	8.28	1	28.27	53.89	0
NEAREST	0.05	47.43	8.18	2	27.12	53.29	0
NEAREST	0.10	48.41	8.27	0	27.12	53.29	0
NEAREST	0.20	48.55	8.25	0	28.26	53.89	0
NEAREST	0.30	48.51	8.28	0	28.26	53.89	0
BUSIEST	0.05	49.76	8.69	0	37.72	67.67	0
BUSIEST	0.10	49.76	8.69	0	37.72	67.67	0
BUSIEST	0.20	49.69	8.66	0	37.72	67.67	0
BUSIEST	0.30	49.72	8.62	0	37.72	67.67	0
HI-REQ	0.05	50.16	8.34	0	30.72	66.04	1
HI-REQ	0.10	50.16	8.34	0	30.74	66.02	0
HI-REQ	0.20	50.16	8.34	0	30.74	66.02	0
HI-REQ	0.30	49.62	8.32	0	30.74	66.02	0

Table D.3: `Area B`. SERIES I - min. of the distance.

Area B		SERIES II					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	48.10	8.23	2	17.43	54.64	0
NEAREST	0.05	47.25	8.54	1	17.43	54.64	0
NEAREST	0.10	47.70	8.17	0	17.43	54.64	0
NEAREST	0.20	48.14	8.21	0	17.43	54.64	0
NEAREST	0.30	48.10	8.23	0	17.43	54.64	0
BUSIEST	0.05	50.93	9.77	0	17.19	53.72	0
BUSIEST	0.10	50.93	9.77	0	17.19	53.72	0
BUSIEST	0.20	50.88	9.78	0	17.19	53.72	0
BUSIEST	0.30	50.36	9.60	0	17.19	53.72	0
HI-REQ	0.05	52.13	9.45	0	16.68	53.89	0
HI-REQ	0.10	52.13	9.45	0	16.68	53.89	0
HI-REQ	0.20	52.51	9.06	0	16.68	53.89	0
HI-REQ	0.30	51.44	9.17	0	16.68	53.89	0

Table D.4: `Area B`. SERIES II - min. of the lateness.

Area C		SERIES I					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	72.16	15.81	0	75.22	78.57	0
NEAREST	0.05	69.31	13.48	3	75.22	78.57	0
NEAREST	0.10	71.24	12.90	0	75.22	78.57	0
NEAREST	0.20	72.68	15.51	0	75.22	78.57	0
NEAREST	0.30	72.48	15.51	0	75.22	78.57	0
BUSIEST	0.05	74.07	11.00	0	72.29	81.19	0
BUSIEST	0.10	73.80	11.23	0	72.29	81.19	0
BUSIEST	0.20	74.86	13.98	0	72.29	81.19	0
BUSIEST	0.30	74.71	14.10	0	72.29	81.19	0
HI-REQ	0.05	78.34	11.33	0	72.90	79.63	0
HI-REQ	0.10	76.61	10.90	0	72.90	79.63	0
HI-REQ	0.20	76.73	12.48	0	72.90	79.63	0
HI-REQ	0.30	74.66	14.16	0	73.84	79.67	0

Table D.5: `Area C`. SERIES I - min. of the distance.

Area C		SERIES II					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	75.22	13.23	0	51.07	69.22	0
NEAREST	0.05	71.50	11.19	4	51.07	69.22	0
NEAREST	0.10	74.22	10.07	0	51.07	69.22	0
NEAREST	0.20	74.99	11.88	0	51.07	69.22	0
NEAREST	0.30	75.53	12.94	0	51.07	69.22	0
BUSIEST	0.05	75.99	8.63	0	51.07	69.22	0
BUSIEST	0.10	75.72	8.99	0	51.07	69.22	0
BUSIEST	0.20	76.09	10.71	0	51.07	69.22	0
BUSIEST	0.30	76.59	11.87	0	51.07	69.22	0
HI-REQ	0.05	78.32	10.87	0	51.07	69.22	0
HI-REQ	0.10	77.39	9.94	0	51.07	69.22	0
HI-REQ	0.20	77.65	10.71	0	51.07	69.22	0
HI-REQ	0.30	76.46	11.91	0	51.07	69.22	0

Table D.6: `Area C`. SERIES II - min. of the lateness.

Area D		SERIES I					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	73.84	27.48	1	53.56	60.39	0
NEAREST	0.05	73.01	26.35	0	46.53	46.01	0
NEAREST	0.10	72.80	26.29	0	46.53	46.01	0
NEAREST	0.20	72.62	26.36	0	46.53	46.01	0
NEAREST	0.30	73.87	27.46	0	53.56	60.39	0
BUSIEST	0.05	78.09	27.62	0	58.13	54.23	0
BUSIEST	0.10	77.39	25.18	0	49.43	52.67	0
BUSIEST	0.20	75.21	25.52	0	49.43	52.67	0
BUSIEST	0.30	74.67	26.03	0	39.31	40.52	1
HI-REQ	0.05	80.73	25.35	0	77.28	110.93	0
HI-REQ	0.10	79.09	24.06	0	46.70	36.79	0
HI-REQ	0.20	75.96	25.34	0	62.60	55.23	0
HI-REQ	0.30	75.96	25.34	0	62.60	55.23	0

Table D.7: `Area D`. SERIES I - min. of the distance.

Area D		SERIES II					
Policies	L_{thres}	Distance	Dev.	# Best	Lateness	Dev.	# Best
NO REPOS	-	76.81	27.40	1	2.41	5.93	0
NEAREST	0.05	78.28	27.46	0	1.86	5.72	0
NEAREST	0.10	77.52	27.49	0	1.86	5.72	0
NEAREST	0.20	76.88	27.39	0	2.41	5.93	0
NEAREST	0.30	76.86	27.40	0	2.41	5.93	0
BUSIEST	0.05	85.33	28.47	0	5.37	8.23	0
BUSIEST	0.10	84.46	27.67	0	5.37	8.23	0
BUSIEST	0.20	81.03	25.62	0	5.37	8.23	0
BUSIEST	0.30	80.05	25.97	0	5.37	8.23	0
HI-REQ	0.05	86.62	27.10	0	2.69	6.20	0
HI-REQ	0.10	83.18	27.47	0	2.67	6.21	0
HI-REQ	0.20	80.92	26.08	0	2.67	6.21	0
HI-REQ	0.30	79.90	26.23	2	3.33	6.25	0

Table D.8: `Area D`. SERIES II - min. of the lateness.

customer (also to be traeted as an advance request) and the remaining 6 customers are immediate requests customers.

Policy: NO-REPOS

Cust.	Request time (hh:mm:ss)	Start service (hh:mm:ss)	On-site service time (min)	Location	Time Window (hh:mm)	TW violation (hh:mm:ss)
Depot				[712844,6175782]		
14	8:00:00	8:00:33	1.30	[713152,6175834]	[8:00,10:30]	-
12	8:00:00	8:02:10	1.86	[713198,6176015]	[8:00,10:30]	-
10	8:00:00	8:04:42	2.64	[713058,6176364]	[8:00,10:30]	-
2	8:00:00	8:08:38	0.89	[712628,6175768]	[8:00,10:30]	-
1	8:00:00	8:09:32	3.40	[712629,6175758]	[8:00,10:30]	-
3	8:00:00	8:13:34	2.67	[712279,6175716]	[8:00,10:30]	-
4	8:00:00	8:16:46	2.61	[712091,6175952]	[8:00,10:30]	-
15	8:00:00	8:19:33	2.79	[712126,6176040]	[8:00,10:30]	-
5	8:00:00	8:22:25	4.21	[712173,6176054]	[8:00,10:30]	-
7	8:00:00	8:27:34	1.17	[712606,6176358]	[8:00,10:30]	-
9	8:00:00	8:28:50	1.79	[712657,6176324]	[8:00,10:30]	-
8	8:00:00	8:30:46	2.14	[712728,6176350]	[8:00,10:30]	-
11	8:00:00	8:33:32	3.06	[713044,6176519]	[8:00,10:30]	-
23	8:00:00	8:46:19	7.45	[712921,6182021]	[8:00,12:00]	-
33	8:00:00	9:01:03	2.38	[708991,6180751]	[8:00,12:00]	-
46	8:00:00	9:07:00	2.26	[710861,6179975]	[8:00,12:00]	-
18	8:00:00	9:11:41	1.56	[711940,6180828]	[8:00,12:00]	-
24	8:00:00	9:14:03	1.55	[712134,6181236]	[8:00,12:00]	-
22	8:00:00	9:16:02	2.65	[712377,6181196]	[8:00,12:00]	-
16	8:00:00	9:19:57	2.85	[711828,6180724]	[8:00,12:00]	-
17	8:00:00	9:23:00	2.04	[711938,6180692]	[8:00,12:00]	-
55	8:00:00	9:25:03	2.62	[711938,6180692]	[8:00,12:00]	-
19	8:00:00	9:28:18	1.37	[712298,6180656]	[8:00,12:00]	-
20	8:00:00	9:31:18	4.19	[713126,6181070]	[8:00,12:00]	-
25	8:00:00	9:35:52	2.46	[713332,6181058]	[8:00,12:00]	-
50	8:00:00	9:40:42	2.98	[712009,6181324]	[8:00,12:00]	-
51	8:00:00	9:43:43	1.90	[712029,6181322]	[8:00,12:00]	-
52	8:00:00	9:45:43	5.77	[712014,6181380]	[8:00,12:00]	-
35	8:00:00	9:54:59	1.23	[710042,6181534]	[8:00,12:00]	-
21	8:00:00	10:00:33	1.02	[712471,6181180]	[8:00,12:00]	-
27	8:00:00	10:02:22	2.51	[712302,6181604]	[8:00,12:00]	-
28	8:00:00	10:05:26	0.93	[712590,6181726]	[8:00,12:00]	-
42	8:00:00	10:09:25	2.72	[710925,6181252]	[8:00,12:00]	-
36	8:00:00	10:13:46	1.76	[710042,6181534]	[8:00,12:00]	-
43	8:00:00	10:15:45	1.70	[710091,6181650]	[8:00,12:00]	-
6	8:00:00	10:22:15	0.77	[712775,6182082]	[8:00,12:00]	-
31	8:00:00	10:23:09	5.00	[712850,6182061]	[8:00,12:00]	-
26	8:00:00	10:28:27	2.92	[713006,6182118]	[8:00,12:00]	-
13	8:00:00	10:31:36	5.57	[712921,6182021]	[8:00,12:00]	-
30	8:00:00	10:37:45	6.13	[712942,6181690]	[8:00,12:00]	-
29	8:00:00	10:44:19	11.94	[712706,6181768]	[8:00,12:00]	-
47	8:00:00	10:59:32	2.47	[710925,6181252]	[8:00,12:00]	-
44	8:00:00	11:03:30	3.30	[710087,6181378]	[8:00,12:00]	-
34	8:00:00	11:07:25	2.57	[709762,6181500]	[8:00,12:00]	-
32	8:00:00	11:14:40	4.03	[707114,6181280]	[8:00,12:00]	-

```
 45      8:00:00    11:18:49      1.47     [707043,6181284]   [ 8:00,12:00]       -
 41      8:00:00    11:28:52      2.44     [711737,6182530]   [ 8:00,12:00]       -
 37      8:00:00    11:32:57      1.19     [712662,6182645]   [ 8:00,12:00]       -
 40      8:00:00    11:35:24      0.75     [713373,6182666]   [ 8:00,12:00]       -
 39      8:00:00    11:37:11      1.87     [712834,6182424]   [ 8:00,12:00]       -
 38      8:00:00    11:39:03      5.84     [712834,6182424]   [ 8:00,12:00]       -
 54      8:00:00    11:44:54      1.67     [712834,6182424]   [ 8:00,12:00]       -
 53      8:00:00    11:46:34      2.74     [712834,6182424]   [ 8:00,12:00]       -
 64      8:00:00    11:50:37      1.48     [712942,6181690]   [ 8:00,18:00]       -
 63      8:00:00    11:53:14      7.39     [712302,6181604]   [ 8:00,18:00]       -
 57      8:00:00    12:01:23      7.94     [712103,6181220]   [ 8:00,18:00]       -
 66      9:36:00    12:09:43      3.59     [711993,6181031]   [ 9:36,16:00]       -
 59      8:00:00    12:15:24      3.31     [710808,6181090]   [ 8:00,18:00]       -
 60      8:00:00    12:19:52      3.28     [710162,6181227]   [ 8:00,18:00]       -
 62      8:00:00    12:32:39      0.65     [712581,6176420]   [ 8:00,18:00]       -
 56      8:00:00    12:34:45      2.06     [713152,6175834]   [ 8:00,18:00]       -
 67     10:06:00    12:45:33      3.83     [712079,6180678]   [10:06,16:30]       -
 68     13:06:00    13:06:26      6.86     [711838,6180722]   [13:06,16:00]       -
 61      8:00:00    13:16:59      1.05     [712662,6182645]   [ 8:00,18:00]       -
 69     13:37:00    13:38:56      4.63     [713044,6181614]   [13:37,16:00]       -
 70     14:16:00    14:17:31      3.08     [712900,6182471]   [14:16,16:15]       -
 65      8:00:00    15:15:00      0.82     [710048,6181199]   [15:15,15:45]       -
 58      8:00:00    15:34:27      0.88     [700226,6185074]   [ 8:00,18:00]       -
 49      8:00:00    15:36:48      1.57     [700766,6185708]   [ 8:00,18:00]       -
 48      8:00:00    15:39:17      3.54     [700491,6186150]   [ 8:00,18:00]       -
 71     15:45:00    16:09:18      1.65     [713230,6180898]   [15:45,17:00]       -
Depot               16:20:01               [712844,6175782]

Total route length (km):         98.75
Total TW penalty time (min):      0.00
Reposition to IP:                    0
```

Policy: HI-REQ - Threshold = 0.1

```
        Request     Start       On-site                    Time            TW
        time        service     service                    Window          violation
Cust.   (hh:mm:ss)  (hh:mm:ss)  time (min)   Location      (hh:mm)         (hh:mm:ss)
-----------------------------------------------------------------------------------

Depot                                      [712844,6175782]
 14      8:00:00     8:00:33      1.30     [713152,6175834]   [ 8:00,10:30]       -
 12      8:00:00     8:02:10      1.86     [713198,6176015]   [ 8:00,10:30]       -
 10      8:00:00     8:04:42      2.64     [713058,6176364]   [ 8:00,10:30]       -
  2      8:00:00     8:08:38      0.89     [712628,6175768]   [ 8:00,10:30]       -
  1      8:00:00     8:09:32      3.40     [712629,6175758]   [ 8:00,10:30]       -
  3      8:00:00     8:13:34      2.67     [712279,6175716]   [ 8:00,10:30]       -
  4      8:00:00     8:16:46      2.61     [712091,6175952]   [ 8:00,10:30]       -
 15      8:00:00     8:19:33      2.79     [712126,6176040]   [ 8:00,10:30]       -
  5      8:00:00     8:22:25      4.21     [712173,6176054]   [ 8:00,10:30]       -
  7      8:00:00     8:27:34      1.17     [712606,6176358]   [ 8:00,10:30]       -
  9      8:00:00     8:28:50      1.79     [712657,6176324]   [ 8:00,10:30]       -
  8      8:00:00     8:30:46      2.14     [712728,6176350]   [ 8:00,10:30]       -
 11      8:00:00     8:33:32      3.06     [713044,6176519]   [ 8:00,10:30]       -
 23      8:00:00     8:46:19      7.45     [712921,6182021]   [ 8:00,12:00]       -
 33      8:00:00     9:01:03      2.38     [708991,6180751]   [ 8:00,12:00]       -
 46      8:00:00     9:07:00      2.26     [710861,6179975]   [ 8:00,12:00]       -
 18      8:00:00     9:11:41      1.56     [711940,6180828]   [ 8:00,12:00]       -
 24      8:00:00     9:14:03      1.55     [712134,6181236]   [ 8:00,12:00]       -
 22      8:00:00     9:16:02      2.65     [712377,6181196]   [ 8:00,12:00]       -
 16      8:00:00     9:19:57      2.85     [711828,6180724]   [ 8:00,12:00]       -
 17      8:00:00     9:23:00      2.04     [711938,6180692]   [ 8:00,12:00]       -
 55      8:00:00     9:25:03      2.62     [711938,6180692]   [ 8:00,12:00]       -
 19      8:00:00     9:28:18      1.37     [712298,6180656]   [ 8:00,12:00]       -
 20      8:00:00     9:31:18      4.19     [713126,6181070]   [ 8:00,12:00]       -
 25      8:00:00     9:35:52      2.46     [713332,6181058]   [ 8:00,12:00]       -
 50      8:00:00     9:40:42      2.98     [712009,6181324]   [ 8:00,12:00]       -
 51      8:00:00     9:43:43      1.90     [712029,6181322]   [ 8:00,12:00]       -
 52      8:00:00     9:45:43      5.77     [712014,6181380]   [ 8:00,12:00]       -
 35      8:00:00     9:54:59      1.23     [710042,6181534]   [ 8:00,12:00]       -
 21      8:00:00    10:00:33      1.02     [712471,6181180]   [ 8:00,12:00]       -
 27      8:00:00    10:02:22      2.51     [712302,6181604]   [ 8:00,12:00]       -
 28      8:00:00    10:05:26      0.93     [712590,6181726]   [ 8:00,12:00]       -
 42      8:00:00    10:09:25      2.72     [710925,6181252]   [ 8:00,12:00]       -
 36      8:00:00    10:13:46      1.76     [710042,6181534]   [ 8:00,12:00]       -
 43      8:00:00    10:15:45      1.70     [710091,6181650]   [ 8:00,12:00]       -
  6      8:00:00    10:22:15      0.77     [712775,6182082]   [ 8:00,12:00]       -
 31      8:00:00    10:23:09      5.00     [712850,6182061]   [ 8:00,12:00]       -
 26      8:00:00    10:28:27      2.92     [713006,6182118]   [ 8:00,12:00]       -
 13      8:00:00    10:31:36      5.57     [712921,6182021]   [ 8:00,12:00]       -
 30      8:00:00    10:37:45      6.13     [712942,6181690]   [ 8:00,12:00]       -
 29      8:00:00    10:44:19     11.94     [712706,6181768]   [ 8:00,12:00]       -
 47      8:00:00    10:59:32      2.47     [710925,6181252]   [ 8:00,12:00]       -
 44      8:00:00    11:03:30      3.30     [710087,6181378]   [ 8:00,12:00]       -
 34      8:00:00    11:07:25      2.57     [709762,6181500]   [ 8:00,12:00]       -
 32      8:00:00    11:14:40      4.03     [707114,6181280]   [ 8:00,12:00]       -
 45      8:00:00    11:18:49      1.47     [707043,6181284]   [ 8:00,12:00]       -
 41      8:00:00    11:28:52      2.44     [711737,6182530]   [ 8:00,12:00]       -
 37      8:00:00    11:32:57      1.19     [712662,6182645]   [ 8:00,12:00]       -
 40      8:00:00    11:35:24      0.75     [713373,6182666]   [ 8:00,12:00]       -
 39      8:00:00    11:37:11      1.87     [712834,6182424]   [ 8:00,12:00]       -
 38      8:00:00    11:39:03      5.84     [712834,6182424]   [ 8:00,12:00]       -
 54      8:00:00    11:44:54      1.67     [712834,6182424]   [ 8:00,12:00]       -
 53      8:00:00    11:46:34      2.74     [712834,6182424]   [ 8:00,12:00]       -
 64      8:00:00    11:50:37      1.48     [712942,6181690]   [ 8:00,18:00]       -
 63      8:00:00    11:53:14      7.39     [712302,6181604]   [ 8:00,18:00]       -
 57      8:00:00    12:01:23      7.94     [712103,6181220]   [ 8:00,18:00]       -
 66      9:36:00    12:09:43      3.59     [711993,6181031]   [ 9:36,16:00]       -
 59      8:00:00    12:15:24      3.31     [710808,6181090]   [ 8:00,18:00]       -
 60      8:00:00    12:19:52      3.28     [710162,6181227]   [ 8:00,18:00]       -
 62      8:00:00    12:32:39      0.65     [712581,6176420]   [ 8:00,18:00]       -
 56      8:00:00    12:34:45      2.06     [713152,6175834]   [ 8:00,18:00]       -
 67     10:06:00    12:45:33      3.83     [712079,6180678]   [10:06,16:30]       -
IP  5               12:49:23               [710091,6181650]
 61      8:00:00    13:41:51      1.05     [712662,6182645]   [ 8:00,18:00]       -
 69     13:37:00    13:44:51      4.63     [713044,6181614]   [13:37,16:00]       -
 68     13:06:00    13:52:07      6.86     [711838,6180722]   [13:06,16:00]       -
IP  1               13:59:25               [711993,6181031]
 70     14:16:00    14:19:00      3.08     [712900,6182471]   [14:16,16:15]       -
IP  1               14:25:33               [711993,6181031]
 65      8:00:00    15:15:00      0.82     [710048,6181199]   [15:15,15:45]       -
 58      8:00:00    15:34:27      0.88     [700226,6185074]   [ 8:00,18:00]       -
 49      8:00:00    15:36:48      1.57     [700766,6185708]   [ 8:00,18:00]       -
 48      8:00:00    15:39:17      3.54     [700491,6186150]   [ 8:00,18:00]       -
 71     15:45:00    16:09:18      1.65     [713230,6180898]   [15:45,17:00]       -
```

```
Depot              16:20:01              [712844,6175782]

Total route length (km):          104.59
Total TW penalty time (min):        0.00
Reposition to IP:                      3
```

Appendix E

Ph. D. theses from IMM

1. **Larsen, Rasmus.** (1994). *Estimation of visual motion in image sequences. xiv* + 143 pp.
2. **Rygard, Jens Moberg.** (1994). *Design and optimization of flexible manufacturing systems. xiii* + 232 pp.
3. **Lassen, Niels Christian Krieger.** (1994). *Automated determination of crystal orientations from electron backscattering patterns. xv* + 136 pp.
4. **Melgaard, Henrik.** (1994). *Identification of physical models. xvii* + 246 pp.
5. **Wang, Chunyan.** (1994). *Stochastic differential equations and a biological system. xxii* + 153 pp.
6. **Nielsen, Allan Aasbjerg.** (1994). *Analysis of regularly and irregularly sampled spatial, multivariate, and multi-temporal data. xxiv* + 213 pp.
7. **Ersbøll, Annette Kjær.** (1994). *On the spatial and temporal correlations in experimentation with agricultural applications. xviii* + 345 pp.
8. **Møller, Dorte.** (1994). *Methods for analysis and design of heterogeneous telecommunication networks.* Volume 1-2, *xxxviii* + 282 pp., 283-569 pp.

9. **Jensen, Jens Christian.** (1995). *Teoretiske og eksperimentelle dynamiske undersøgelser af jernbanekøretøjer. viii* + 174 pp.

10. **Kuhlmann, Lionel.** (1995). *On automatic visual inspection of reflective surfaces.* Volume 1, *xviii* + 220 pp., (Volume 2, *vi* + 54 pp., fortrolig).

11. **Lazarides, Nikolaos.** (1995). *Nonlinearity in superconductivity and Josephson Junctions. iv* + 154 pp.

12. **Rostgaard, Morten.** (1995). *Modelling, estimation and control of fast sampled dynamical systems. xiv* + 348 pp.

13. **Schultz, Nette.** (1995). *Segmentation and classification of biological objects. xiv* + 194 pp.

14. **Jørgensen, Michael Finn.** (1995). *Nonlinear Hamiltonian systems. xiv* + 120 pp.

15. **Balle, Susanne M.** (1995). *Distributed-memory matrix computations. iii* + 101 pp.

16. **Kohl, Niklas.** (1995). *Exact methods for time constrained routing and related scheduling problems. xviii* + 234 pp.

17. **Rogon, Thomas.** (1995). *Porous media: Analysis, reconstruction and percolation. xiv* + 165 pp.

18. **Andersen, Allan Theodor.** (1995). *Modelling of packet traffic with matrix analytic methods. xvi* + 242 pp.

19. **Hesthaven, Jan.** (1995). *Numerical studies of unsteady coherent structures and transport in two-dimensional flows.* Risø-R-835(EN) 203 pp.

20. **Slivsgaard, Eva Charlotte.** (l995). *On the interaction between wheels and rails in railway dynamics. viii* + 196 pp.

21. **Hartelius, Karsten.** (1996). *Analysis of irregularly distributed points. xvi* + 260 pp.

22. **Hansen, Anca Daniela.** (1996). *Predictive control and identification - Applications to steering dynamics. xviii* + 307 pp.

23. **Sadegh, Payman.** (1996). *Experiment design and optimization in complex systems. xiv* + 162 pp.

24. **Skands, Ulrik.** (1996). *Quantitative methods for the analysis of electron microscope images. xvi* + 198 pp.

25. **Bro-Nielsen, Morten.** (1996). *Medical image registration and surgery simulation. xxvii* + 274 pp.

26. **Bendtsen, Claus.** (1996). *Parallel numerical algorithms for the solution of systems of ordinary differential equations. viii* + 79 pp.

27. **Lauritsen, Morten Bach.** (1997). *Delta-domain predictive control and identification for control. xxii* + 292 pp.

28. **Bischoff, Svend.** (1997). *Modelling colliding-pulse mode-locked semiconductor lasers. xxii* + 217 pp.

29. **Arnbjerg-Nielsen, Karsten.** (1997). *Statistical analysis of urban hydrology with special emphasis on rainfall modelling.* Institut for Miljøteknik, DTU. *xiv* + 161 pp.

30. **Jacobsen, Judith L.** (1997). *Dynamic modelling of processes in rivers affected by precipitation runoff. xix* + 213 pp.

31. **Sommer, Helle Mølgaard.** (1997). *Variability in microbiological degradation experiments - Analysis and case study. xiv* + 211 pp.

32. **Ma, Xin.** (1997). *Adaptive extremum control and wind turbine control. xix* + 293 pp.

33. **Rasmussen, Kim Ørskov.** (1997). *Nonlinear and stochastic dynamics of coherent structures. x* + 215 pp.

34. **Hansen, Lars Henrik.** (1997). *Stochastic modelling of central heating systems. xxii* + 301 pp.

35. **Jørgensen, Claus.** (1997). *Driftsoptimering på kraftvarmesystemer.* 290 pp.

36. **Stauning, Ole.** (1997). *Automatic validation of numerical solutions. viii* + 116 pp.

37. **Pedersen, Morten With.** (1997). *Optimization of recurrent neural networks for time series modeling. x* + 322 pp.

38. **Thorsen, Rune.** (1997). *Restoration of hand function in tetraplegics using myoelectrically controlled functional electrical stimulation of the controlling muscle. x* + 154 pp. + Appendix.

39. **Rosholm, Anders.** (1997). *Statistical methods for segmentation and classification of images. xvi* + 183 pp.

40. **Petersen, Kim Tilgaard.** (1997). *Estimation of speech quality in telecommunication systems. x* + 259 pp.

41. **Jensen, Carsten Nordstrøm.** (1997). *Nonlinear systems with discrete and continuous elements.* 195 pp.

42. **Hansen, Peter S.K.** (1997). *Signal subspace methods for speech enhancement. x* + 226 pp.

43. **Nielsen, Ole Møller.** (1998). *Wavelets in scientific computing. xiv* + 232 pp.

44. **Kjems, Ulrik.** (1998). *Bayesian signal processing and interpretation of brain scans. iv* + 129 pp.

45. **Hansen, Michael Pilegaard.** (1998). *Metaheuristics for multiple objective combinatorial optimization. x* + 163 pp.

46. **Riis, Søren Kamaric.** (1998). *Hidden markov models and neural networks for speech recognition. x* + 223 pp.

47. **Mørch, Niels Jacob Sand.** (1998). *A multivariate approach to functional neuro modeling. xvi* + 147 pp.

48. **Frydendal, Ib.** (1998.) *Quality inspection of sugar beets using vision. iv* + 97 pp. + app.

49. **Lundin, Lars Kristian.** (1998). *Parallel computation of rotating flows. viii* + 106 pp.

50. **Borges, Pedro.** (1998). *Multicriteria planning and optimization. - Heuristic approaches. xiv* + 219 pp.

51. **Nielsen, Jakob Birkedal.** (1998). *New developments in the theory of wheel/rail contact mechanics. xviii* + 223 pp.

52. **Fog, Torben.** (1998). *Condition monitoring and fault diagnosis in marine diesel engines. xii* + 178 pp.

53. **Knudsen, Ole.** (1998). *Industrial vision. xii* + 129 pp.

54. **Andersen, Jens Strodl.** (1998). *Statistical analysis of biotests. - Applied to complex polluted samples. xx* + 207 pp.

55. **Philipsen, Peter Alshede.** (1998). *Reconstruction and restoration of PET images. vi* + 132 pp.

56. **Thygesen, Uffe Høgsbro.** (1998). *Robust performance and dissipation of stochastic control systems.* 185 pp.

57. **Hintz-Madsen, Mads.** (1998). *A probabilistic framework for classification of dermatoscopic images. xi* + 153 pp.

58. **Schramm-Nielsen, Karina.** (1998). *Environmental reference materials methods and case studies. xxvi* + 261 pp.

59. **Skyggebjerg, Ole.** (1999). *Acquisition and analysis of complex dynamic intra- and intercellular signaling events.* 83 pp.

60. **Jensen, Kåre Jean.** (1999). *Signal processing for distribution network monitoring. xv* + 199 pp.

61. **Folm-Hansen, Jørgen.** (1999). *On chromatic and geometrical calibration. xiv* + 238 pp.

62. **Larsen, Jesper.** (1999). *Parallelization of the vehicle routing problem with time windows.xx* + 266 pp.

63. **Clausen, Carl Balslev.** (1999). *Spatial solitons in quasi-phase matched struktures. vi* + (flere pag.)

64. **Kvist, Trine.** (1999). *Statistical modelling of fish stocks. xiv* + 173 pp.

65. **Andresen, Per Rønsholt.** (1999). *Surface-bounded growth modeling applied to human mandibles. xxii* + 125 pp.

66. **Sørensen, Per Settergren.** (1999). *Spatial distribution maps for benthic communities.*

67. **Andersen, Helle.** (1999). *Statistical models for standardized toxicity studies. viii* + (flere pag.)

68. **Andersen, Lars Nonboe.** (1999). *Signal processing in the dolphin sonar system. xii* + 214 pp.

69. **Bechmann, Henrik.** (1999). *Modelling of wastewater systems. xviii* + 161 pp.

70. **Nielsen, Henrik Aalborg.** (1999). *Parametric and non-parametric system modelling. xviii* + 209 pp.

71. **Gramkow, Claus.** (1999). *2D and 3D object measurement for control and quality assurance in the industry. xxvi* + 236 pp.

72. **Nielsen, Jan Nygaard.** (1999). *Stochastic modelling of dynamic systems. xvi* + 225 pp.

73. **Larsen, Allan.** (2000). *The dynamic vehicle routing problem. xvi* + 183 pp.

74. **Halkjær, Søren.** (2000). *Elastic wave propagation in anisotropic inhomogeneous materials. xiv* + 133 pp.

75. **Larsen, Theis Leth.** (2000). *Phosphorus diffusion in float zone silicon crystal growth. viii* + 119 pp.

76. **Dirscherl, Kai.** (2000). *Online correction of scanning probe microscopes with pixel accuracy.* 136 pp.