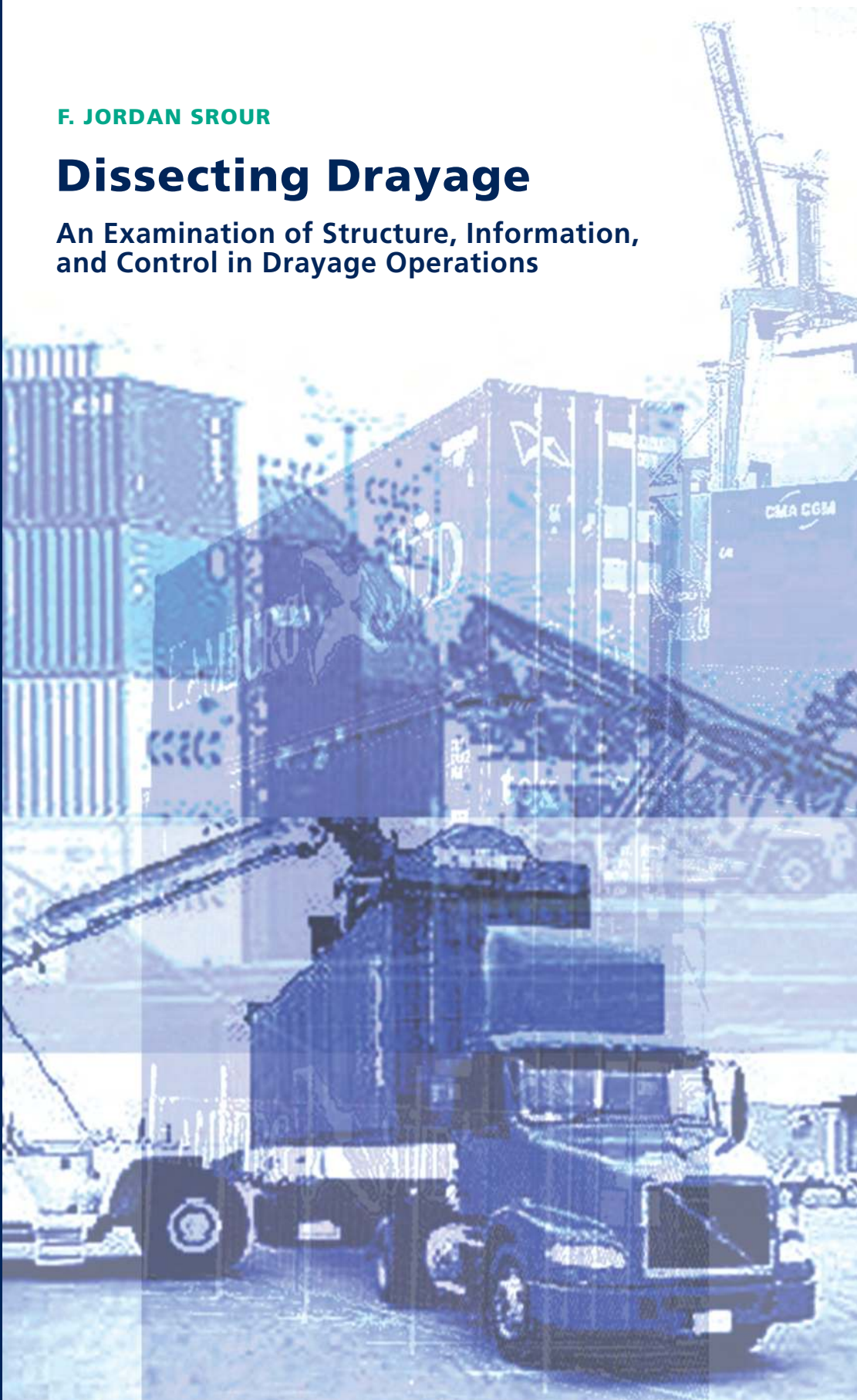


F. JORDAN SROUR

# Dissecting Drayage

An Examination of Structure, Information,  
and Control in Drayage Operations



# Dissecting Drayage

An Examination of  
Structure, Information, and Control  
in Drayage Operations



# Dissecting Drayage

An Examination of Structure, Information, and Control in Drayage Operations

Het Ontleden van Drayage

Een Onderzoek naar Structuur, Informatie en Control in Drayage Operaties

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de  
Erasmus Universiteit Rotterdam  
op gezag van de  
rector magnificus

Prof.dr. H.G. Schmidt

en volgens besluit van het College voor Promoties.

De openbare verdediging zal plaatsvinden op  
Vrijdag, 12 Februari 2010 om 11.30 uur

door

FAITH JORDAN SROUR NÉE LUDDERS  
geboren te Seattle, WA, USA



## **Promotiecommissie**

**Promotor:** Prof.dr. S. L. van de Velde  
**Overige leden:** Prof.dr. M. B. M. de Koster  
Prof.dr. C. Witteveen  
Prof.dr.ir. R. Dekker  
**Co-Promotor:** Dr. R. A. Zuidwijk

**Erasmus Research Institute of Management-ERIM**

**Rotterdam School of Management (RSM)**

**Erasmus School of Economics (ESE)**

Erasmus University Rotterdam

Internet: <http://www.irim.eur.nl>

**ERIM Electronic Series Portal:** <http://hdl.handle.net/1765/1>

**ERIM PhD Series in Management, 186**

Reference number ERIM: EPS-2010-186-LIS

ISBN 978-90-5892-226-7

© 2010, F. Jordan Srour

**Design:** B&T Ontwerp en advies [www.b-en-t.nl](http://www.b-en-t.nl) and F. Jordan Srour

**Print:** Haveka [www.haveka.nl](http://www.haveka.nl)

**Cover Art:** F. Jordan Srour and Kathleen M. Ludders

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without permission in writing from the author.

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

*To my parents,  
for teaching me the value of scientific inquiry.*



# Contents

- Acknowledgements** **v**
  
- 1 Introduction** **1**
  - 1.1 Drayage Operations . . . . . 2
  - 1.2 Dissecting Drayage . . . . . 3
    - 1.2.1 Geometric Structure . . . . . 6
    - 1.2.2 Advanced Information . . . . . 7
    - 1.2.3 Level of Control . . . . . 10
  - 1.3 Contributions . . . . . 12
  
- 2 The Stacker Crane Problem: A Literature Review** **15**
  - 2.1 Preliminaries: Traveling Salesmen and Stacker Cranes . . . . . 18
  - 2.2 Solving The Stacker Crane Problem . . . . . 26
    - 2.2.1 Variations: Swapping and Preemption Problems . . . . . 32
  - 2.3 Empirical Studies . . . . . 33
  - 2.4 Implications . . . . . 38
  
- 3 Are SCPs Easy? An Empirical Exploration** **39**
  - 3.1 ATSP Instances . . . . . 41
  - 3.2 Hard or Easy? Measuring Solvability . . . . . 43
    - 3.2.1 Concorde . . . . . 45
    - 3.2.2 Tsp\_solve version 1.3.6 . . . . . 47
    - 3.2.3 Solvability . . . . . 47



|          |                                                                                           |            |
|----------|-------------------------------------------------------------------------------------------|------------|
| 3.3      | Distance Matrix Metrics . . . . .                                                         | 50         |
| 3.3.1    | Distance Related . . . . .                                                                | 50         |
| 3.3.2    | Asymmetry Related . . . . .                                                               | 52         |
| 3.3.3    | Graph Structure Related . . . . .                                                         | 54         |
| 3.3.4    | Arc Routing Problem Related . . . . .                                                     | 58         |
| 3.3.5    | Assignment Problem Related . . . . .                                                      | 63         |
| 3.3.6    | Relationships Between Metrics . . . . .                                                   | 64         |
| 3.4      | Statistical Analysis . . . . .                                                            | 66         |
| 3.4.1    | Linear Regression . . . . .                                                               | 66         |
| 3.4.2    | Discriminant Analysis . . . . .                                                           | 72         |
| 3.4.3    | Multinomial Logistic Regression . . . . .                                                 | 76         |
| 3.5      | Verification . . . . .                                                                    | 81         |
| 3.6      | Discussion . . . . .                                                                      | 84         |
| <b>4</b> | <b>The Value of Advanced Location Information</b>                                         | <b>91</b>  |
| 4.1      | Literature Review . . . . .                                                               | 92         |
| 4.1.1    | Our Contribution . . . . .                                                                | 93         |
| 4.2      | Assumptions, Notation, and Preliminaries . . . . .                                        | 95         |
| 4.2.1    | Optimal Offline Algorithm for the TSP on $\mathbb{R}^+$ with Re-<br>lease Dates . . . . . | 96         |
| 4.2.2    | Online TSP Algorithms . . . . .                                                           | 97         |
| 4.3      | OLTSP with Two Disclosure Dates . . . . .                                                 | 99         |
| 4.4      | Fixed Amounts of Advanced Notice . . . . .                                                | 102        |
| 4.5      | Variable Amounts of Advanced Notice . . . . .                                             | 105        |
| 4.6      | Discussion . . . . .                                                                      | 112        |
| <b>5</b> | <b>Centralized versus Decentralized Control in Drayage</b>                                | <b>117</b> |
| 5.1      | Related Work . . . . .                                                                    | 119        |
| 5.1.1    | Optimization-based Approaches for Vehicle Routing . .                                     | 120        |
| 5.1.2    | Agent-based Approaches for Vehicle Routing . . . . .                                      | 124        |
| 5.2      | Experimental Design . . . . .                                                             | 128        |
| 5.2.1    | Solution Approaches . . . . .                                                             | 129        |

---

|          |                                                    |            |
|----------|----------------------------------------------------|------------|
| 5.2.2    | The Data . . . . .                                 | 146        |
| 5.2.3    | Uncertainty Scenarios . . . . .                    | 150        |
| 5.3      | Results . . . . .                                  | 155        |
| 5.3.1    | Service Time Uncertainty Results . . . . .         | 156        |
| 5.3.2    | Job Arrival Uncertainty Results . . . . .          | 161        |
| 5.3.3    | Job and Service Time Uncertainty Results . . . . . | 164        |
| 5.4      | Discussion . . . . .                               | 167        |
| <b>6</b> | <b>Conclusion</b>                                  | <b>171</b> |
| 6.1      | The Past . . . . .                                 | 171        |
| 6.2      | The Future . . . . .                               | 172        |
| <b>A</b> | <b>Generator Parameters</b>                        | <b>175</b> |
| <b>B</b> | <b>Crane2 Random Instance Generator</b>            | <b>177</b> |
|          | <b>List of Tables</b>                              | <b>181</b> |
|          | <b>List of Figures</b>                             | <b>185</b> |
|          | <b>Bibliography</b>                                | <b>187</b> |
|          | <b>Index</b>                                       | <b>203</b> |
|          | <b>Summary</b>                                     | <b>207</b> |
|          | <b>Samenvatting (Summary in Dutch)</b>             | <b>209</b> |
|          | <b>About the author</b>                            | <b>213</b> |



# Acknowledgements

The irony of this section is that it is the most difficult to write. It is far easier to write proofs or describe a few numbers than to synthesize the emotion of gratitude with only a mere 26 characters at my disposal. Nevertheless, I shall try.

For the content of this dissertation, I am grateful to my promotor Steef van de Velde. It was Steef's presence that gave me guidance, and his absence that gave me the freedom to pursue the ideas he left in my mind. For the fact that this dissertation even exists, I am indebted to the only person at the Rotterdam School of Management (RSM) with the patience to stick with me from beginning to end — Rob Zuidwijk.

In addition to the collaboration I enjoyed with my advisors, I can think of no greater collaboration than that I enjoyed with Tamás Máhr and Mathijs de Weerd of TU Delft. Every meeting with Tamás and Mathijs spawned enough ideas and enthusiasm to fill far more than one thesis. The publications we produced together are some of my favorites — if not for content, then certainly for process. I am also extremely grateful for this experience as it brought me two other tangible benefits 1) enjoyable times spent with the employees of Amende and 2) the pleasure of meeting Cees Witteveen, who worked as part of my inner committee to make this dissertation better.

In addition to Dr. Witteveen, Renè de Koster and Rommert Dekker, serving as members of my inner committee, worked to refine this dissertation into the tome you see today. I owe a special debt of gratitude to René for also being such a wonderful colleague in Department 6. René's enthusiasm for

quality work is so perfectly buttressed by his own example that one cannot but want to excel in his presence . . . whether that's in operations research or on a 7am run in the woods.

Indeed, the enthusiasm of all the members of Department 6 is truly contagious. It is invariably this *joie de vivre* that drew me to the department in the first place. In particular, I owe a world of thanks to Elfriede Krauth. We shared the same promotor and in so many ways she is like a big sister — going ahead of me past all the doctoral milestones, both good and bad, but always looking back to help me through them in my time. Other members (past and present) of Department 6, also deserving a note of thanks for making life as a PhD candidate more productive and more enjoyable, include: Mark Boons (deserves a special thanks for helping with the Dutch Summary), Irma Borst, Dirk Deichmann, René van der Eijk, Tony Hak, Murthy Halemane, José Antonio Larco-Martinelli, Mahmut Ozdemir, Hans Quak, Erik van Raaij, Kees Jan Roodbergen, Henk de Vries, Finn Wynstra, and Xiandong Zhang.

While I finished my PhD career in Department 6, I would be remiss if I did not mention those who helped me through the beginning of my PhD career in Department 1. The first words of gratitude go to Arnoud van der Maas, the person whose first words to me (“Well, why would a woman want to study math or science??”) led to a friendship characterized by many interesting discussions. Also among those coloring my first impressions and deserving a note of thanks is my first office-mate, Hans Moonen, who hooked me up with discounts on my first Dutch cell phone, my first Dutch TV subscription, my first Dutch phone subscription, and my first boat ride on the Maas, to name just a few. Other members of Department 1 whose friendship buoyed me through the tougher times include: Niels Agatz (deserves the majority of thanks for help on the Dutch Summary), Muhammad Jalil, Gabor Marotí, Ting Li, João Frota Neto Quariguasi, Bas Verheijen (also deserving a special thanks for help on the Dutch Summary), and ShengYun (Annie) Yang.

During my time at RSM, I was fortunate to meet others outside my immediate department(s). Most notably, the many wonderful staff, faculty, and students I met while serving on the Faculty Council from 2007-2008. Frank

Wijen, the chair of the faculty council at that time, deserves a special note of gratitude not only for the kindness he showed me, but for his tireless work in championing all of the issues brought before the council. Also critical to the success of the council was the secretary and legal counsel, Catheleyne Jurgens. Finally, through the council I met Bettina Wittneben who showed me that taking Bollywood dance classes is a much better way to improve in Dutch than taking Dutch classes.

While life within Erasmus University formed a large part of my PhD program, I cannot forget the debt of gratitude owed to those outside of RSM. Most notably, the entire Zerhane family. I cannot thank Naima enough for her willingness to welcome a funny American that speaks English way too fast, Arabic in broken spurts, and Dutch like a 2 year old into her family. The many meals, late nights, and laughter were the perfect remedy for any discomfort wrought by my studies or life so far from home.

I am also grateful to Jorge Castañeda and Theodora Paschoudi. How lucky to land in the Netherlands with friends already there to coach us through the myriad of forms and paperwork. A process made even better by the particularly delicious meals prepared by Theodora, with a bit of seasoning (or should that be seasoned advice?) from Jorge.

The greatest words of thanks, however, are owed to my husband, Issam Srour, who thought he was marrying an engineer, but ended up living with a mathematician. Thank you for letting me use the bathroom tiles as a whiteboard and internet-ordering as a means to prepare dinner. Fortunately, moving to Beirut alleviated my need to prepare dinners and I am particularly grateful to my parents-in-law for the many productive and pleasurable evenings spent working at Mounir's desk after a delicious meal prepared by Hassibè.

Finally, for the inspiration behind my years of academic pursuit, I thank my family, John, Kathleen, and Amity Ludders. My father's enthusiasm for asking questions and designing experiments to answer those questions inspired many of my projects — from grade-school science fairs through to this dissertation. My mother's ability to solve a myriad of *NP*-hard problems on a

daily basis is the origin of my fascination with combinatorics. Whether it is calculating the most efficient way to cut out fabric for a pattern, using the oven to prepare a multi-course meal with minimal heat-loss, or simply solving a Sudoku puzzle, my mother has always been a phenomenal optimizer. Finally, it was my older sister who started it all by teaching me the word “infrastructure”. Thank you, all, for giving me the tools I need to ask and answer so many questions of my own.

Faith Jordan Srour  
Beirut, July 2009

# Chapter 1

## Introduction

When the mathematician would solve a difficult problem, he first frees the equation of all incumbrances, and reduces it to its simplest terms. So simplify the problem of life, distinguish the necessary and the real. Probe the earth to see where your main roots run.

Henry David Thoreau, *Letter to H.G.O. Blake, 27 March 1848*

Dissection, in the medical context and throughout the ages, has served to illustrate the internal structure, while highlighting the function and relationship of each component to the whole. Not only is knowledge gained from the end-product of the dissection, but also from the process of dissection itself (Mutyalá and Cahill, 1996). In this manner, an understanding of anatomy is acquired alongside skills that can be applied outside the dissection laboratory. Just as a medical student might keep a diary of their dissections — the anatomy and the lessons learned — this thesis serves the same purpose; but the cadaver in this case is a problem originating in the drayage industry at the Port of Rotterdam, the Netherlands.



## 1.1 Drayage Operations

The term dray dates back to the 14th century when it was used commonly to describe a type of very sturdy sideless cart<sup>1</sup>. In the 1700s the word drayage came into use meaning “to transport by a sideless cart”. Today, drayage commonly refers to the transport of containerized cargo, within a limited geographic range, to and from port or rail terminals and inland locations.

With the phenomenal growth of containerized freight, since the container’s introduction in 1956, the drayage industry has also experienced significant growth. For example, the world saw total maritime container traffic grow to approximately 417 million twenty foot equivalent units (TEUs) in 2006 (BTS, 2007). Large trucks operating in the United States alone carried empty freight containers over a total of 1 billion miles in 2002 (USDOT, 2004).

Unfortunately, the drayage portion of a door-to-door container move tends to be the most costly part of the move. Morlok and Spasovic (1994) indicate that up to 40% of the cost for a 900 mile container move can be attributed to the 50 mile drayage portion of the move. There are a variety of reasons for this disproportionate assignment of costs, including a great deal of uncertainty at the interface of modes. For example, trucks moving containers to and from a port terminal are often uncertain as to how long it will take them to pick up a designated container coming from a ship, from the terminal stack, or from customs. This uncertainty leads to inefficiency in planning a profitable route for multiple containers in one day. As a result, planning processes must be designed and adopted that can rapidly exploit the underlying structure of this routing environment while incorporating real-time information. This thesis examines three properties found in drayage operations — structure, advanced information, and level of control — with this goal in mind.

Through cooperation with a Dutch logistics service provider (LSP), we received the inspiration for this research, as well as the data required to test our ideas. The LSP, participating in this study, dedicates a portion of its business to draying refrigerated (“reefer”) containers from/to the Port of Rotterdam

---

<sup>1</sup>Etymology taken from <http://www.merriam-webster.com/>

to/from various customer locations in the Netherlands. Approximately 40 trucks transport an average of 65 containers per day in this operation. In general, the containers arrive on container ships arranged by customers. They are off-loaded at sea terminals, where trucks must then pick them up. The containers are then transported to their destination at the customer, where they are emptied. The empty containers are later returned to a sea terminal. In reality, because reefers are considered high-value equipment, the same truck waits with the container until it is emptied and then returns it to a sea terminal. (In the theoretical sections [Chapters 3 and 4] of this dissertation, we, however, relax this constraint and allow the return portion of the trip to occur at a later time.) The return terminal may be the same terminal from which the container originated or it may be a different terminal. For export containers the sequence is the same, the only difference is that the containers are not emptied, but loaded at the customer's location. At each location there are time windows within which trucks can make their visits. At sea terminals the time windows correspond to the opening hours of the terminal. At customer sites, the time windows are defined by the customers. Each day the LSP must plan a set of routes capturing as much business as possible at minimum cost.

## 1.2 Dissecting Drayage

The drayage operations, outlined in the previous section, are not unique to the LSP examined in this research. Indeed, while the data provided by the Dutch LSP gives this research a touch of realism, it is the ubiquity of the drayage problem that renders this work interesting to a broader audience. For example, Caris and Janssens (2009); Cheung et al. (2008); Ileri et al. (2006); Namboothiri (2006); Smilowitz (2006); and Neuman and Smilowitz (2002) all describe similar operations. The differentiating factor amongst these studies is the method by which the authors choose to mathematically model the drayage operations they describe.

Mathematical modeling, the conversion of a problem from words to equa-

tions for the generation of a solution or solutions, is often more art than science (Hillier and Lieberman, 2001). There are usually multiple facets that can be emphasized or de-emphasized depending on the needs of the problem owner and the constraints of the modeler.

In its most general form, drayage operations may be modeled as a vehicle routing problem (VRP). VRPs are broadly defined as problems requiring the design of an optimal set of routes, serving a given set of customers with a given number of vehicles. The VRP was first introduced as the truck dispatching problem by Dantzig and Ramser (1959). Since the introduction of the VRP multiple variations have been examined including variations in vehicle capacity, fleet heterogeneity, time windows, pick-up and drop off in the same tour, multi-depot, split deliveries, and so on. Golden and Assad (1988), Ball et al. (1995), Toth and Vigo (2001), and Golden et al. (2008) provide reviews of these extensions. (The simplest form of routing problem is what is known as the transportation problem. In the transportation problem, a set of goods must flow from a given number of supply locations to a given number of demand locations, at least cost [Hillier and Lieberman, 2001].)

Given all of the VRP variants, the one most often applied to drayage operations, in the literature, is the full truckload Pick-up and Delivery Problem with Time Windows (PDPTW) (Caris and Janssens, 2009; Cheung et al., 2008; Ileri et al., 2006; Namboothiri, 2006; Neuman and Smilowitz, 2002). In the truckload PDPTW, a fleet of vehicles, capable of carrying only one job at a time, must pick up a job from one location and drop it off at another location, while arriving to each location within a specified time window. Finding the assignment of jobs to trucks that minimizes costs (in the form of total distance, empty distance, or operating costs) is the solution goal.

Defining drayage operations as a PDPTW serves to demarcate three lines along which the problem can be dissected and examined — namely, the number of vehicles, spatial characteristics of pick-up and drop-off locations, and temporal characteristics pertaining to the time windows and the revelation of job information. For example, by limiting the fleet of vehicles to a single vehicle and removing the temporal aspects of the problem, we can focus

in more detail on the role that the distances between pick-up and drop-off locations play in solvability. Alternatively, by maintaining the temporal characteristics of the problem, while simplifying the spatial characteristics, we can focus on the role that advanced job information plays in routing vehicles in real-time. Finally, by considering a fleet with multiple vehicles in addition to both temporal and spatial characteristics, we can address the role that centralized versus decentralized planning has on drayage operations. The first two subproblems delineated within drayage operations serve as the basis for the theoretical portions of this theses (Chapters 3 and 4). The final delineation most closely resembles the practical problem of drayage and is the basis of the last chapter, Chapter 5. The placement of these subproblems as they relate to different problem types within the field of VRPs, along with the chapters in which they are addressed, may be seen in Figure 1.1. The following subsections provide more detail on these subproblems.

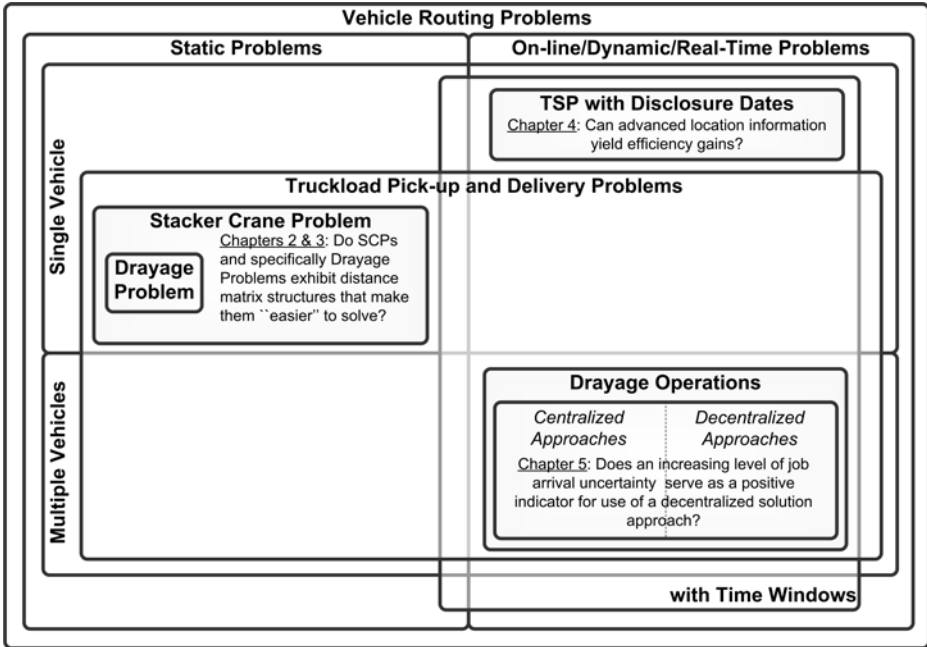


Figure 1.1: Overview of subproblems addressed in this thesis and their relationship to the PDPTW.

### 1.2.1 Geometric Structure

In order to isolate the effect that the geometric structure has on solvability, we alter two complicating features of the PDPTW. Specifically, we first assume that there is only one vehicle as opposed to a full fleet of vehicles. Second, we abandon the constraints imposed by the time windows. The resulting problem is known in the literature as the *Stacker Crane Problem (SCP)* (Frederickson et al., 1978). In the SCP, a single crane (or truck in the drayage context) must move a number of full vehicle loads (or containers in the drayage context) from specified pick-up locations to specified drop-off locations in a way that minimizes the total tour length (or equivalently, minimizes the empty distance traveled between each required move).

When considering the SCP definition in the context of drayage operations, we see that the physical location of pick-up and drop-off points yields an interesting geometric structure. Specifically, nearly all jobs originate from or are destined to one of only a few fixed freight terminals. Throughout the remainder of this thesis, we formalize this concept by designating the subclass of SCPs that have more than one coincidental pick-up and drop-off point, as *Drayage Problems*. Figure 1.2 shows, on the left, an example of a route for an arbitrary instance of the general SCP and, on the right, an example of a route for an arbitrary instance of the drayage problem. Notice in the example drayage problem, four jobs share the same pick-up and drop-off point. (In practice, these shared nodes are terminals.)

In order to appreciate this abstract representation of the drayage problem in terms of realistic drayage operations, Figure 1.3 depicts the pick-up/drop-off location structure for the Dutch LSP. In this map, the geometric structure is apparent as the vehicles must always begin at the home location, specified by the white marker, and serve jobs originating from one of the grey markers (primarily clustered near the Port of Rotterdam) destined to one of the black markers (spread throughout the Netherlands) or vice versa. In this way, jobs can often be sequenced such that the destination of one job is the origin of the next.

A review of SCP literature and related terminology is the topic of Chapter





Figure 1.3: One day of jobs in the Netherlands for the Dutch LSP. Black markers indicate customer locations; grey markers indicate terminal locations; and the white marker indicates the home terminal of the LSP.

to examine the implication of these temporal characteristics on routing, we consider the on-line arrival of jobs with release dates, but simplify the ge-

ometric structure of the PDPTW by specifying only one location for jobs. Specifically, each job is defined solely by a single point as opposed to both a pick-up and drop-off location. The resulting problem is termed the on-line Traveling Salesman Problem with two disclosure dates.

Intensely, widely, and well-studied — not to mention important — are all adjectives used to describe the well-known Traveling Salesman Problem (TSP). In short the TSP addresses the problem of finding the shortest tour through a set of jobs or cities (beginning and ending at a depot or origin city) in a given metric space. If the salesman is traveling at constant speed, finding the shortest path is equivalent to minimizing the time the salesman returns to the depot. The literature on this problem begins with the seminal papers by Dantzig et al. (1954) and Flood (1956), includes at least four books (Lawler et al., 1985; Reinelt, 1994; Gutin and Punnen, 2002; Applegate et al., 2007), multiple survey papers (Bellmore and Nemhauser, 1968; Burkard et al., 1995; Jünger et al., 1995, 1997), and a myriad of articles.

Amongst the many TSP articles are a variety of extensions to the basic problem formulation. The extension we are interested in is known as the “TSP with release dates”. In this variation, the salesman may visit each job only on or after a specified release date. If all of the job locations and their release dates are known in advance, the problem is termed static and may be solved via an offline optimization approach. As noted above, however, this is not particularly realistic. In the majority of real-world applications, jobs (or cities) and their release times are revealed over time — often after the salesman has already left the city of origin (or depot). Solution approaches designed to handle on-line problems, that is problems in which new information arrives during execution, are termed on-line algorithms.

We may add a further level of realism by assuming that the exact location of each job is also revealed over time. Specifically, the location of each job may be revealed in advance of information on the release date, which is revealed in advance of the actual release date. For example, consider a dray company, such as the one documented here and in Máhr et al. (2010), that must pick up containers from several port terminals. In the morning, the dray provider



learns the location of the terminals that will release containers for transport. Later in the day, the company learns the exact time at which those containers will be released from customs for pick-up; for some terminals this information may come early, for others this information may arrive much later in the day. While the dray company could wait until all information is known, the containers will certainly be served sooner if the company can cleverly exploit each piece of information when it arrives. For this the dray provider needs an algorithm tailored to an on-line environment.

It is this problem of finding the best ordering of the jobs for a single truck that forms a problem we term the online TSP with two disclosure dates. For the ease of analysis we restrict the metric space to the non-negative real number line with the depot located at the origin,  $\mathbb{R}_0^+$ . A graphical representation of this problem may be seen in Figure 1.4, where the y-axis represents time and the x-axis represents distance. Note, as we assume that the salesman travels at unit-speed, the salesman's trajectory in Figure 1.4 is depicted as a 45-degree line. We are thus able to indicate the value of each piece of information via a ratio of the online algorithm cost to the offline optimal algorithm cost. In the literature, this ratio is referred to as both the *competitive ratio* and the *worst-case ratio*; we will use the term worst-case ratio. The derivation of worst-case ratios for the on-line TSP with two disclosure dates is the focus of Chapter 4.

### 1.2.3 Level of Control

A key to exploiting advanced information and reacting to change prudently is the presence of an agile control structure. For example, most dray companies operate by using a central dispatcher to plan the routes of all the trucks. However, once in the field, the truck drivers may exhibit a large degree of autonomy. In this setting, a more agile control structure is one that relies on the truck drivers and customers to negotiate a solution. But will the drivers, operating without central knowledge, find the most cost effective route?

This question is at the heart of the debate between traditional (centralized) optimization techniques and (decentralized) Agent Based Modeling

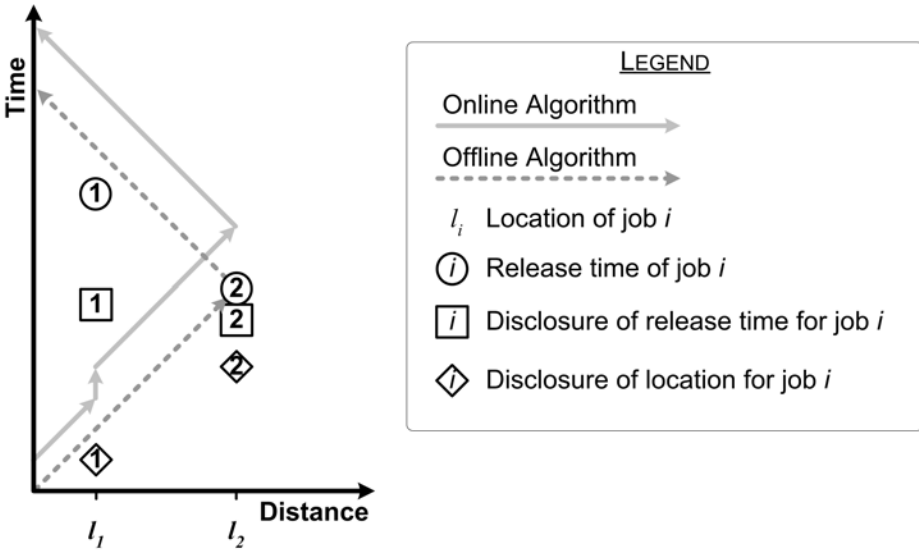


Figure 1.4: Example of the offline optimal and an online solution to the TSP on  $\mathbb{R}_0^+$  with two disclosure dates.

(ABM) (also referred to as multi-agent systems [MAS]) (Wooldridge and Jennings, 1995). ABM, with its roots in the fields of artificial intelligence, social network theory, cognitive science, has been lurking on the fringes of the operations research field for some time now (Samuelson, 2005). In the August 2006 issue of *OR/MS Today*, for example, Samuelson and Charles (2006) present agents as a serious and useful simulation technique. In general, however, the literature lacks quantitative comparisons of the strengths and weaknesses of traditional optimization to agent based techniques. To that end, we apply two structurally distinct solution approaches — a centralized solution and a decentralized solution — to the drayage problem of the Dutch LSP.

Specifically, in Chapter 5, we empirically compare a solution approach for the on-line PDPTW that focuses on a single objective and uses full system information to a solution approach based on agents that optimize their own unique objectives given the information they perceive and maintain locally (in both space and time).

### 1.3 Contributions

Table 1.1 describes the exact problem examined in each chapter while highlighting the primary contributions of those chapters.

Chapter 2 is a literature review. As such its contribution is limited to exposing clues buried in the existing body of knowledge; clues fueling a strong suspicion that SCPs, and more specifically drayage problems, are comparatively easy to solve. In turn, Chapter 3 is remarkable for both the method and outcome of investigating this suspicion. We use techniques borrowed from the field of statistics to examine the significance of 22 distance matrix metrics in indicating solvability for over 500 instances of the TSP. This method results in strong empirical evidence that drayage problems are, in general, easier than other related TSPs.

Encouraged by the “easy” distance matrix structure of drayage problems, we turn from the spatial to the temporal dimensions in Chapter 4. This chapter quantifies — via competitive analysis — the value of advanced information in a one-dimensional version of the on-line TSP with release dates. Chapter 4, demonstrates that revealing the location information alone can yield all the benefit (or all the detriment) to the cost of an on-line algorithm. The content of Chapter 4 is based on the following article:

Srour, F. Jordan and Rob A. Zuidwijk, (2008). How Much is Location Information Worth? A Competitive Analysis of the On-line Traveling Salesman Problem with Two Disclosure Dates. *ERIM Report Series*, Reference No. ERS-2008-075-LIS. Available at SSRN: <http://ssrn.com/abstract=1314164>.

Finally, in a bid to marry space and time, we devote Chapter 5 to a comparative study of centralized and decentralized control structures. Using a method dependent on MAS technology, we not only answer questions regarding the value of centralization versus decentralization, but also make a significant contribution to the agent literature by providing the first quantitative comparison of MAS to traditional operations research techniques in the field of freight transport. Chapter 5 is based on the following article:

Máhr, Tamás, F. Jordan Srour, Mathijs M. de Weerd, and Rob Zuidwijk, (2010). Can agents measure up? A comparative study of an agent-based and on-line optimization approach for a drayage problem with uncertainty. *Transportation Research, Part C: Emerging Technologies*. Volume 18, Issue 1, pp. 99-119.

The opening example of Chapter 5 is taken from:

Ketter, Wolfgang and F. Jordan Srour, (June 2009). Optimal or Agile? Tradeoffs between optimization and agent-based methods. *OR/MS Today*. Available at: <http://www.lionhrtpub.com/orms/orms-6-09/froptimalagile.html>.

Table 1.1: Overview of problems examined and our contributions.

|                                  | Chapters 2 and 3                                                                                                                                           | Chapter 4                                                                                                                                                                                                                                                      | Chapter 5                                                                                                                                                                                                                    |
|----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Problem:<br/>Description:</b> | <p><b>SCP</b></p> <p>A single vehicle must move full-loads from pick-up to drop-off locations in a way that minimizes the distance of the total route.</p> | <p><b>TSP with Disclosure Dates</b></p> <p>A single vehicle must find the shortest tour through a given set of cities, beginning and ending at the same city, in such a way that every city is visited at least once on or after a specified release time.</p> | <p><b>PDPTW</b></p> <p>A fleet of vehicles must pick-up and deliver a number of full-loads in such a way that the total route length is minimized and the time windows at the pick-up and drop-off locations are obeyed.</p> |
| <i>Pick-up and Delivery?</i>     | Yes                                                                                                                                                        | No                                                                                                                                                                                                                                                             | Yes                                                                                                                                                                                                                          |
| <i>Time Windows?</i>             | No                                                                                                                                                         | Partial                                                                                                                                                                                                                                                        | Yes                                                                                                                                                                                                                          |
| <i>No. of Trucks?</i>            | 1                                                                                                                                                          | 1                                                                                                                                                                                                                                                              | 40                                                                                                                                                                                                                           |
| <i>Static or on-line?</i>        | Static                                                                                                                                                     | On-line                                                                                                                                                                                                                                                        | On-line                                                                                                                                                                                                                      |
| <i>Dimension?</i>                | $\mathbb{R}^2$                                                                                                                                             | $\mathbb{R}^+$                                                                                                                                                                                                                                                 | $\mathbb{R}^2$                                                                                                                                                                                                               |
| <i>Advanced Information?</i>     | No                                                                                                                                                         | Yes                                                                                                                                                                                                                                                            | Varying                                                                                                                                                                                                                      |
| <i>Level of Control?</i>         | Centralized                                                                                                                                                | Centralized                                                                                                                                                                                                                                                    | Centralized and Decentralized                                                                                                                                                                                                |
| <b>Contributions:</b>            | Empirical analysis of SCPs indicates the existence of features that make SCPs, and more specifically drayage problems, “easy” to solve.                    | Advanced location information gives a measurable advantage over simultaneous disclosure dates.                                                                                                                                                                 | Decentralized heuristics can outperform centralized heuristics in certain on-line settings.                                                                                                                                  |

## Chapter 2

# The Stacker Crane Problem: A Literature Review

If you want to understand today, you have to search yesterday.

Pearl S. Buck. *American author, 1938 Nobel Prize for Literature*

Intermodal freight containers do not move themselves. An increasing number of containers move through increasingly complex supply chains, yet every move a container makes must be accompanied by another piece of equipment. This equipment can be a large ship or train, specialized yard equipment, or a standard truck and chassis (Muller, 1999; Stahlbock and Voß, 2008b,a). Whatever the conveyance may be, each move of a container represents a pick-up and delivery problem.

In its simplest form this problem may be described as follows. A vehicle (or other means of conveyance) must start from an initial location, perform a specified set of moves, and return to the initial location. These moves are defined as trips from a specified pick-up point to a delivery point. The objective is to serve all required moves with the shortest empty distance. In the context of drayage transport (the transport of containers a short distance from a port terminal by a single truck and chassis), all of the moves are considered full truckloads. We therefore focus on the single vehicle, unit-

load capacity pick-up and delivery problems. This problem is also the focus of Chapter 3. (In Chapter 4 this basic problem will be simplified further by removing pick-up and deliveries, but adding time windows and a time horizon. In Chapter 5 this basic routing problem will be expanded to include time windows and a fleet of 40 vehicles working over a time horizon.)

Recently, the literature has seen a burst of survey papers on the topic of pick-up and delivery problems. These papers collectively detail the basic problem while classifying the multiple variants (Berbeglia et al., 2007; Paragh et al., 2008a,b; Gribkovskaia and Laporte, 2008; Cordeau et al., 2008). Specifically, Berbeglia et al. (2007) introduce a classification scheme based on three categories: [*Structure* | *Visits* | *Vehicles*].

*Structure* refers to the possible pairings between pick-up and drop-off locations. More specifically, this category provides information on the level of dedication between pick-up and drop-off points. For example, if all goods may be picked-up from one location (e.g. a warehouse) and delivered to multiple locations (e.g. customers) then the problem is a one-to-many problem. The reverse structure is termed many-to-one. Alternatively, if the goods may be picked up from any number of locations and delivered to any of a set of drop-off points, then the problem is a many-to-many problem. Finally, if each item must be picked-up from only one specific location for delivery to only one specific drop-off location, then the problem is termed one-to-one (1-1, for short). *visits* refers to the activities that must (or can) be undertaken at each location (i.e. pick-up only, delivery only, or both [P-D, for short]). Finally, *vehicles* refers to the number and capacity of the vehicles. Thus, the routing problem at the heart of this and the next chapter may, in the terminology of Berbeglia et al. (2007), be described as a one-to-one (1-1), pick-up and delivery problem (P-D), with a single unit-load vehicle (1): [1-1 | P-D | 1].

One possible feature of pick-up and delivery problems, not encapsulated by this definition, is that of transshipment. Transshipment (also sometimes termed pre-emption) refers to the possibility of dropping a container before its final destination has been reached, in order to serve a more profitable load. Considering a [1-1 | P-D | 1] problem with the possibility of transshipment,

we see that three variations are possible — one in which all of the jobs must be transported directly from pick-up point to drop-off point, one in which some of the jobs must be transported directly while others may be preempted or transshipped, and one in which all jobs are available for transshipment.

In order to efficiently distinguish these three cases we introduce some notation following that of Anily and Hassin (1992). Let  $S$  represent the full set of jobs that requires service. We assume that each job is a unique object type (i.e.  $|S| = n$ , where  $n$  is the number of jobs). We then specify two subsets,  $S_m$  and  $S_d$  referring to the set of jobs that may *not* be transshipped and the set that may be “dropped” along the route, respectively. Given this notation we can see that the first problem (named the *stacker crane problem* (SCP)) is one in which  $S = S_m$  and  $S_d = \emptyset$ ; the second (named the *swapping problem* (SP)) is one in which  $S = S_m \cup S_d$ , neither subset is the empty set; and the third (named the *preemption problem*) is one in which  $S = S_d$  and  $S_m = \emptyset$ .

It is important to note, that in the literature review of Hernandez-Perez and Salazar-Gonzalez (2004), the distinction between the SCP and the SP falls along two lines — the number of origins/destinations and the permissibility of preemption. As noted above, we, however, consider the SP only under conditions in which every job is of a different object type. Therefore, in contrast to Hernandez-Perez and Salazar-Gonzalez (2004), the only distinction between the SP and the SCP is the permissibility of preemption. Specifically, preemption is forbidden in the SCP, permitted for some jobs in the SP, and permitted for all jobs in the preemption problem.

As the transfer of intermodal freight containers between vehicles generally requires specialized equipment, minimizing the possibility for preemption in the drayage context, we predominantly focus this literature review on SCPs. Furthermore, the SCP serves as the most fundamental representation of drayage operations: a single vehicle that must pick-up and deliver full loads (e.g. containers) at minimal cost. Despite this preference, we do, however, pay a brief tribute to swapping and preemption problems in Subsection 2.2.1. In the next section (Section 2.1 we introduce the terminology necessary



to appreciate the milestones in vehicle routing research, generally, and SCP research, specifically. In Section 2.2 we present the milestones of SCP theory. We follow this review with a summary of empirical studies performed on SCP instances and more generally, asymmetric traveling salesmen problems. The document concludes with a brief discussion of these results and motivation for the remainder of this thesis.

## 2.1 Preliminaries: Traveling Salesmen and Stacker Cranes

The Stacker Crane Problem is nothing more than a routing problem. At the heart of every routing problem is a distance matrix. A distance matrix is a table containing the distances between each and every city or job (also called node) in a given problem. In fact, depending on the exact nature of the routing problem, the distance matrix may be the only data underlying the problem. The famous Traveling Salesman Problem (TSP) is, for example, a problem that can be entirely described by a distance matrix. (For a comprehensive collection of results on the TSP, the reader is referred to Gutin and Punnen (2002).)

The goal of the TSP, most simply stated, is to determine a tour, passing through all of the cities exactly once, in such a way that the total length of the tour is minimized. While this problem statement may conjure up fun childhood memories of connect-the-dot puzzles or specialized maze games (see e.g. Abbott (1990)), the underlying optimization problem is 'hard' in a mathematical sense (Gutin and Punnen, 2002).

To appreciate the mathematical or theoretical meaning of the word "hard", we first describe three different versions of the TSP problem<sup>1</sup>. The first problem version is termed the *optimization problem* and reflects the definition of the TSP given in the previous paragraph. The second problem version,

---

<sup>1</sup>Note, while these three problem versions are well defined and commonly recognized, the reader is referred to (Papadimitriou and Steiglitz, 1982) for more details, as this was the source used for this exposition.

termed the *evaluation problem*, is a relaxation of the optimization version. For the TSP, the evaluation version is phrased as follows: “Given the number of cities and the inter-city distances, find the cost of the optimal solution.” Notice that the evaluation version does not require that we find the optimal tour, only that we find the cost of the optimal tour. The third problem version of the TSP, termed the *recognition problem*, can be stated as a yes/no question. For the TSP, this question is: “Given the number of cities along with all intercity distances and an integer value,  $L$ , is there a tour through all cities whose cost is less than or equal to  $L$ ?” Notice that the answer to this question is no harder to give than the solution to the evaluation problem version of the TSP. That is, once the optimal solution cost is known one must only check if that value is less than  $L$  or not. Thus, these three problem versions conveniently establish a hierarchy relating the recognition problem to the evaluation problem and the evaluation problem to the optimization problem.

We now exploit the recognition version to classify problems into two sets — recognition problems that can be solved by an algorithm whose running time is a polynomial function of the problem parameters ( $P$ ) and recognition problems that cannot be solved in polynomial time, but if an instance is known to have a “yes” answer, that can be verified in polynomial time ( $NP$ ). To explain the  $NP$  set further, in the context of the TSP, we see that if given a tour for a TSP we can readily verify if its cost is less than  $L$ , however, generating a tour with a cost less than  $L$  from scratch is far more time consuming. Noting that any recognition problem which can be solved in polynomial time can also be verified in polynomial time leads us to conclude that  $P$  is a subset of  $NP$ . Whether or not  $P$  is a proper subset of  $NP$  or alternatively whether  $P = NP$  is a famous outstanding question in mathematics and computer science<sup>2</sup>.

We can now also define another subset of  $NP$ , termed  $NP$ -complete. Specifically, a given recognition problem is  $NP$ -complete if all other problems

---

<sup>2</sup>For a formal description of this outstanding problem, the interested reader is referred to: [http://www.claymath.org/millennium/P\\_vs\\_NP/](http://www.claymath.org/millennium/P_vs_NP/)

in  $NP$  polynomially transform to the given recognition problem. (Polynomial transformation means that the parameters of one problem can be transformed into the parameters of a second problem by making a series of steps, not numbering more than a polynomial function of the original problem size.) Thus, the  $NP$ -complete problems are related in such a way that if there exists an algorithm which runs in time that is a polynomial function of problem size for one  $NP$ -complete problem, then that algorithm would work for the whole class of  $NP$ -complete problems. Now, by recalling that the recognition problems in the  $NP$ -complete class have an affiliated optimization problem version, we come to our definition of “hard”. More specifically, we use the term  $NP$ -hard. This term describes the class of problems that are not in  $NP$  (usually because they are not recognition problems), but whose recognition version is in the  $NP$ -complete class. Noting that the recognition version of the TSP is in  $NP$ -complete leads us to the conclusion that the optimization version of the TSP is in  $NP$ -hard (Gutin and Punnen, 2002; Papadimitriou and Steiglitz, 1982). The ramifications of these classifications are significant. If one can find an algorithm to solve the optimization version of the TSP problem to optimality running in time that is a polynomial function of problem size, then all other  $NP$ -hard problems could also be solved in polynomial time.

Why is finding a polynomial time algorithm so important? Well, what’s the point in trying to solve a problem, if, in the worst-case, all existing computers running the best-possible algorithms, would only be able to find a provably optimal solution well after you (and your children, your grandchildren, and so on) are dead?

Let’s clarify these statements with an example. The best known algorithm for the transportation problem (introduced in Chapter 1) runs in  $O(n^2)$  time. The  $O()$  expression is borrowed from mathematics and serves to indicate the limiting behavior of a function when a given input parameter tends towards infinity. Thus, in the context of algorithm analysis, this notation represents a worst-case bound on running time given the size of the problem expressed as a number,  $n$  (Aho et al., 1983). In the case of the transportation problem,

$n$  represents the number of supply and demand points in the problem. Thus, given a transportation problem with a total of 10 supply/demand points, it would take a computer running the best known algorithm (performing one operation per millisecond), 0.1-seconds to find the optimal solution. The best known algorithm for the traveling salesman problem, on the other hand, has a worst-case runtime bound of  $O(n^2 2^n)$  (Applegate et al., 2007). This means that, in the worst-case, a traveling salesman problem with only 10 nodes might require a computer, performing one operation per millisecond, 102.4-seconds to find the optimal solution. Maybe that doesn't sound so bad, but suppose you would like a route visiting each of the 50 States in the United States of America — using that same computer and the best known algorithm, you might, in the worst-case, have to wait over 891,959 centuries for a provably optimal route.

Despite this seemingly desperate news, there is hope. This hope comes in two forms. First, not every instance of a problem belonging to the  $NP$ -hard class will require the worst-case running time before a provably optimal solution is obtained. This concept will play a significant role in Chapter 3. Second, there exists a robust field of study focused on approximation algorithms. As stated by Hochbaum (1996), “trading-off optimality in favor of tractability is the paradigm of approximation algorithms”. Unlike heuristics, which focus on finding a feasible solution fast, without any guarantee of solution quality, approximation algorithms are designed to run in polynomial time yielding a solution guaranteed to be less than either a fixed percentage or a functional distance away from the optimal value. If a fixed constant, relative error,  $c$ , exists between the optimal value and the approximation algorithm's solution value, the algorithm is termed a  $c$ -approximation algorithm and the problem belongs to a class of problems termed APX (Papadimitriou and Steiglitz, 1982). This concept will play a significant role in Chapter 4.

To put this good news in context, we highlight two well-known heuristics and the best known  $c$ -approximation algorithm for the TSP. The first heuristic of note is a very simple constructive heuristic called Nearest-Neighbor (NN). This heuristic, introduced in the seminal TSP paper by Flood in 1956, works

by starting at one city repeatedly moving to the nearest city until all cities are visited at which point the tour is closed by returning to the starting city. The appeal of this heuristic is the speed with which it can be executed. Unfortunately, this speed comes with no guarantee of solution quality and in some very specific instances this approach can give the singular worst possible tour (Gutin et al., 2002).

To mitigate the poor (possibly abysmal) performance of NN, Lin and Kernighan introduced a tour improvement heuristic, the Lin-Kernighan (L-K) heuristic, that works by taking a given tour and swapping pairs of tour edges in such a way that the overall tour cost is reduced. This heuristic is largely based on the well-known 2-opt and 3-opt heuristics in which pairs or triplets, respectively, of edges are exchanged in a cost reducing manner. Although the L-K heuristic goes further by establishing at each iteration an appropriate number,  $k$ , for performing a  $k$ -opt exchange. The  $k$ -opt heuristics will be described in greater detail in Chapter 5 as they figure prominently in that chapter.

Finally, the best known  $c$ -approximation algorithm for the TSP is the Christofides algorithm (Christofides, 1976). This algorithm works by finding a least cost set of edges in the original problem such that every city is connected to every other city by exactly one path. The subset of cities which serve as the end point for an odd number of edges are then matched at least cost. This process yields a set of edges that permits the salesman to pass over each edge exactly once as part of a tour that begins and ends at a given city. The final step of the algorithm is to rationalize this tour by “shortcutting” edges that pass through cities which were visited more than once; recall the ultimate goal of the TSP is to visit every city exactly once. Thus, the Christofides algorithm solves the TSP, giving a solution that is within  $3/2$  of the optimal cost. Note, the Christofides algorithm serves as the basis for a subroutine in the SCP approximation algorithm highlighted in Section 2.2.

Regardless of the solution tactic taken for these “hard” problems, all TSPs and TSP solution techniques — successful and otherwise — are tied to the contents and structure of the underlying distance matrix. The best docu-

mented and influential feature of a distance matrix is symmetry (Gutin and Punnen, 2002). In words, symmetry refers to a square matrix in which the entries in the upper right are identical to those in the lower left when reflected across the diagonal. Mathematically speaking, a symmetric matrix is an  $n \times n$  matrix in which each entry in the  $i$ th row and  $j$ th column (e.g.  $d_{ij}$ , the distance from  $i$  to  $j$ ) equals the entry in the  $j$ th row and  $i$ th column (e.g.  $d_{ij} = d_{ji}$ ). When we consider this matrix feature in the context of the TSP, we can describe symmetry as the condition in which traveling from city  $i$  to city  $j$  requires the same travel time as traveling from city  $j$  to city  $i$ . If this condition holds for all cities, then the TSP actually belongs to the sub-class of *Symmetric Traveling Salesman Problems* (STSPs).

Alternatively, if the distance matrix is asymmetric, then traveling from city  $i$  to city  $j$  may take longer or shorter than traveling from city  $j$  to city  $i$ . (Mathematically, the  $n \times n$  distance matrix is such that  $d_{ij} \neq d_{ji}$ , for at least one pair of  $i$  and  $j$  with  $j \neq i$ ). In this case, the TSP belongs to the sub-class of *Asymmetric Traveling Salesman Problems* (ATSPs). While the idea, that traveling from one location to another would vary in time or distance depending on the direction of travel, may at first seem strange, there are in fact many realistic examples. The most obvious example arises in any urban environment with an over-abundance of one-way streets. In such a network, traveling from one intersection to the next may require traveling only one block or taking a trip around the block, depending on your desired direction of travel. Other examples occur when the problem in question is an *Arc Routing Problem* (ARP) as opposed to a node routing problem (Eiselt et al., 1995b,a).

It is convenient, both visually and theoretically, to describe the ARP using terminology from graph theory<sup>3</sup>. In this regard, the ARP is a routing problem on a graph ( $G$ ) with some arcs ( $A$ ) that must be traversed and other edges ( $E$ ) that may be traversed, but are not required. All arcs and edges meet at vertices ( $V$ ) which are also called nodes. (Throughout this thesis we use the

---

<sup>3</sup>For more details on graph theory the reader is directed to Trudeau (1993) and Chartrand (1977).

terms vertex and node interchangeably.) Arcs and edges, on the other hand, have a distinct meaning. In ARPs the server is required to traverse all of the arcs in a given graph at least once, at minimum cost. Edges, in ARPs, on the other hand, may be traversed as many or as few times as necessary, and thus, have the primary influence on cost. Note, if the arc is not directed, then the server must traverse that arc at least once, in any direction. If the arc is directed, then it must be traversed at least once in the given direction (e.g. left to right).

The number of arcs incident on a vertex is termed the degree of the vertex; if the arcs have a direction the term degree may be made more specific to include in-degree (arc entering the vertex or node) and out-degree (arcs leaving the vertex or node). We do not consider edges when counting degree in an ARP graph. This is because we consider edges to be optional links, while arcs are fundamental structural components in the ARP graph. Note that an arc routing problem with directed arcs can be transformed into a node routing problem (i.e. a traveling salesman problem) by exchanging each arc for a node and adding two directional edges between each node with costs equivalent to the distance from the end of one arc (in the original ARP) to the start of another.

This transformation and the accompanying distance matrix is depicted for a small problem in Figure 2.1. In the ARP of Figure 2.1 two jobs require transport from  $v_2$ , one to  $v_3$  and one to  $v_4$ , one job requires transport from  $v_1$  to  $v_4$ , and one job requires transport from  $v_3$  to  $v_2$ . Notice that in the new node routing problem, the distance matrix is asymmetric as the distance from node  $i$  (in the transformed graph) to node  $j$  (in the transformed graph) is the distance from the end of arc  $i$  (in the original graph) to the beginning of arc  $j$  (in the original graph), while the distance from node  $j$  to node  $i$  is the distance from the end of arc  $j$  to the beginning of arc  $i$ . Thus, the transformation from an ARP to a node routing problem yields an ATSP. Furthermore, the optimal solution to the new ATSP will also be the optimal solution for the original ARP, and vice-versa.

The transformation from an ARP to an ATSP formulation is well docu-

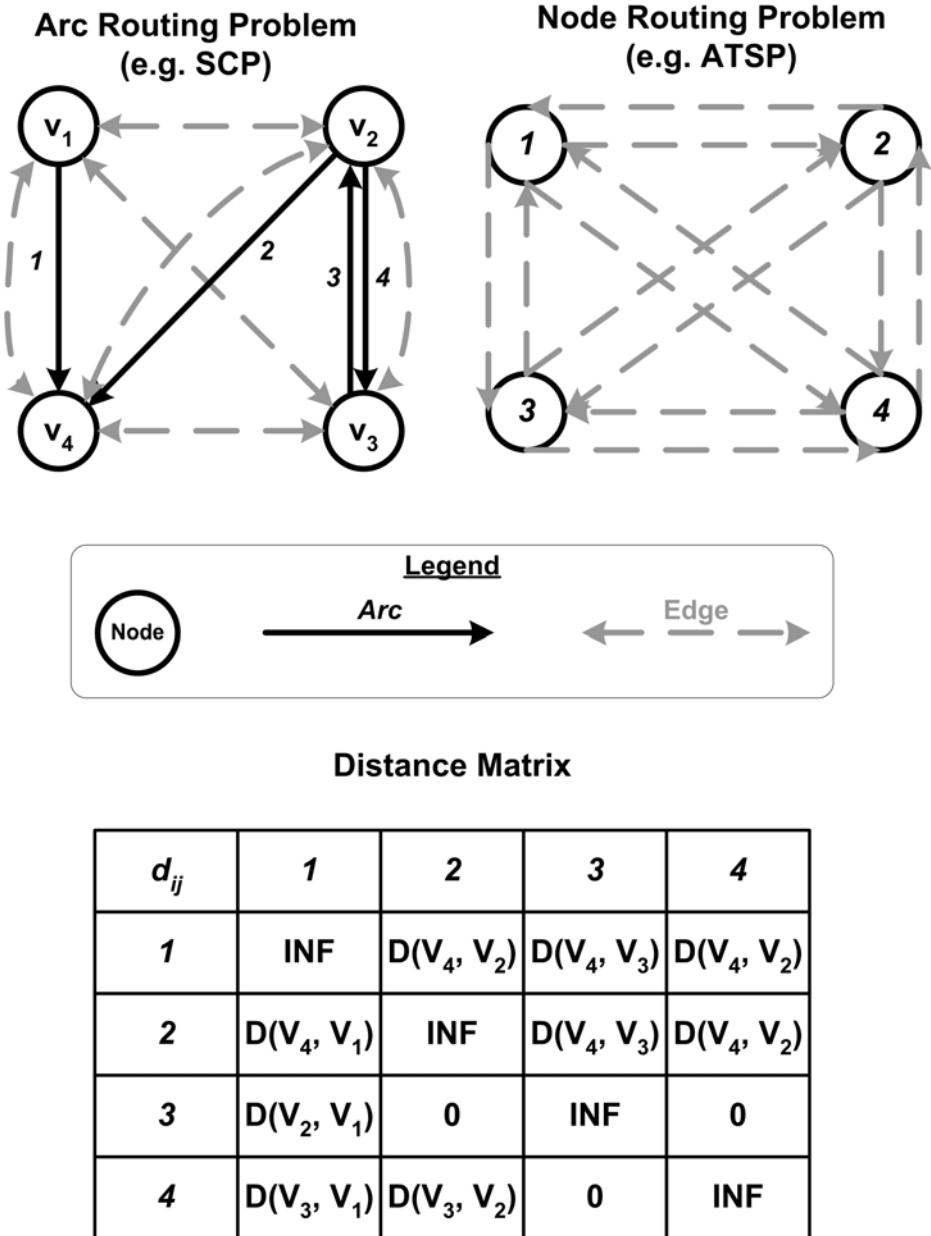


Figure 2.1: Example of the transformation from an ARP to a node routing problem.



mented as a viable and fruitful approach to solving ARPs. Specifically, Laporte (1997) demonstrated that the *Chinese Postman Problem* (CPP)<sup>4</sup> and its variants (including the Rural Postman Problem [RPP]) can be readily transformed into ATSPs and then solved using TSP techniques. Among the CPP variants, studied by Laporte (1997), was the *Stacker Crane Problem* (SCP). Using the ARP terminology, the SCP is a problem in which a tour traversing some required directed arcs out of a set of edges in a given graph must be made at minimum cost. This representation highlights the membership of the SCP in the class of *NP*-hard problems. Specifically, shrinking the distance between the pick-up and delivery points renders the SCP into the well-known, *NP*-hard, Traveling Salesman Problem (TSP).

## 2.2 Solving The Stacker Crane Problem

The first formal study of the SCP entered the literature in 1978 when Frederickson et al. described a  $c$ -approximation algorithm for the SCP as part of an exposition of algorithms for “some routing problems”. Specifically, Frederickson et al. (1978) propose an approximation algorithm guaranteeing a solution that is, in the worst-case,  $\frac{9}{5}$  times the optimal solution. The proposed approximation algorithm runs in  $O(n^3)$  time, where  $n$  is the number of jobs (or arcs if we consider the graph formulation). Interestingly, Frederickson et al. (1978) note that they did not succeed in finding an SCP instance which approaches the  $\frac{9}{5}$  worst-case bound, yet, to date, no better approximation algorithm has ever been proposed.

The SCP approximation algorithm of Frederickson et al. (1978) is comprised of two subroutines, designed to proffer a solution tailored to two possible structures of the underlying arc routing problem. The first subroutine called LARGEARC addresses settings in which the arcs (required) are longer than the edges (optional) in the underlying graph. The primary steps of LARGEARC include solving an assignment problem (i.e. a least cost match-

---

<sup>4</sup>a problem in which every arc in a given connected graph must be traversed at least once before returning to the vertex from which the tour started, see e.g. Minieka (1979)

ing between the heads and tails of all arcs) and then solving a minimum cost spanning tree<sup>5</sup> problem (i.e. the least cost set of edges that forms a connected graph with  $m - 1$  edges, when there are  $m$  vertices) in order to join any disjoint cycles. The second subroutine called LARGEEDGE addresses settings in which the edges (optional) are much longer than the arcs (required). LARGEEDGE begins by transforming the SCP into an ATSP and then applying the Christofides algorithm (Christofides, 1976).

Making a small sidestep to consider this algorithm in the context of drayage operations, we can conjecture that the solutions proffered by LARGEARC will predominate among the solutions selected in the postprocess step of the algorithm. This is because of the unique structure that drayage problems tend to exhibit. That is, most arcs (or jobs) originate from or are destined to a small number of terminals or customer locations. As such the edge lengths (or inter-arc distances) are generally much smaller (often zero) than the arc lengths.

Recognizing the influence that geometric structure, and distances, can have on algorithm design, we note that if the SCP is restricted to a line the problem becomes a Gilmore-Gomory TSP and is polynomially solvable (Kabadi, 2002). In 1964, Gilmore and Gomory examined the problem of heating and cooling a furnace to accommodate the annealing requirements of a set of jobs. While this problem has its origins in job scheduling, we can easily view it as a routing problem by considering the starting temperature of each job as its pick-up location and the ending temperature as its drop-off location, both along the real number line. The solution method proposed by Gilmore and Gomory (1964) is an  $O(n \log n)$  algorithm that first solves a least cost matching between drop-off locations and pick-up locations and then patches any sub-cycles together into a single cycle via a series of edge interchanges. More recently, Vairaktarakis (2003) revisited the original Gilmore-Gomory problem and presents a simpler algorithm (one without the edge interchange process) that also runs in  $O(n \log n)$  time.

While the underlying bipartite graph structure meticulously described and

---

<sup>5</sup>A tree, in this context, is a connected graph in which every node has degree one or two.

exploited by both Gilmore and Gomory (1964) and Vairaktarakis (2003) leads to a polynomial time formulation, it overlooks a feature of the job structure commonly found in real-world pick-up and delivery problems — especially drayage problems. That feature is the frequent coincidence of pick-up and drop-off locations. For example, as noted above, it is quite common that several jobs will originate from a specific terminal (or station) just as several jobs will be destined to the same drop-off location — the jobs originating at  $v_2$  or those destined for  $v_4$  in Figure 2.1, as an example. Recognizing this feature of the SCP in a real-world setting, Atallah and Kosaraju (1988) obtain an optimal solution to the SCP on a line with a  $O(n + k \log \beta(k, q))$  time algorithm<sup>6</sup>. In this case,  $k$  stands for the number of vertices (e.g. terminals, stations, or customer locations) in the problem;  $n$  is (as before) the number of jobs;  $q$  is the number of strongly connected components in the graph that emerges from a least cost matching step (similar to the first step of LARGEARC or Gilmore and Gomory (1964)); and  $\beta(k, q) = \min\{i \mid \log^i q \leq \frac{k}{q}\}$ . Admittedly, without introducing a significant amount of terminology related to data structures, there is no intuitive definition of  $\beta(k, q)$ . However, it is worth stating that  $\beta(k, q) \leq k$  which implies that  $k \log \beta(k, q) \leq k \log k \leq k^2$ . Thus, the complexity of this algorithm is less than the complexity of the general SCP approximation algorithm of Frederickson et al. (1978). Also of note is the fact that the number of strongly connected components that can emerge from a least cost matching is less than or equal to the minimum of the number of jobs or the number of stations (i.e.  $q \leq \min\{n, k\}$ ). Frederickson (1993) extends the work of Atallah and Kosaraju (1988) by showing that the result for a linear track also holds for a circular track.

Interestingly, the move from a linear or circular track to a tree renders the problem *NP*-hard. As such, Frederickson and Guan (1993) develop two fast approximation algorithms. One provides a  $\frac{3}{2}$ -approximation and the other a  $\frac{4}{3}$ -approximation in  $O(n + k)$  and  $O(n + k \log \beta(k, q))$  time, respec-

---

<sup>6</sup>Note, in the preparation of this review, a small error was discovered in Frederickson and Guan (1993); in their paper it was specified that  $\beta(k, q) = \min\{i \mid \log^i k \leq \frac{k}{q}\}$  when in actuality this expression should be as it appears here, as derived from the fastest minimum spanning tree algorithm available at that time (Gabow et al., 1986).

tively. Frederickson and Guan (1993) further show that a combination of the two algorithms produces a solution with cost  $\frac{5}{4}$  times the minimum cost in  $O(n + k \log \beta(k, q))$  time. Examining the  $\frac{4}{3}$ -approximation algorithm using asymptotic analysis, A.Coja-Oghlan et al. (2006) demonstrate that on almost all inputs this algorithm obtains the optimal. This result is extremely promising and serves to provide some indication as to why SCPs encountered in the “real-world” tend to be solvable to optimality (or near optimality) using these approximation algorithms.

Indeed, the real-world is more forgiving than the theoretical world, in that many applications of the SCP appearing in warehousing, manufacturing, and container yards are solvable in polynomial time. In their 1999 work on automated storage/retrieval systems (AS/RS) with dedicated storage, Van den Berg and Gademann demonstrate that a special case of the stacker crane problem with only two terminals is equivalent to the Transportation Problem. In their formulation, all storage jobs must be serviced from one terminal and all retrieval jobs are destined for a second terminal. While this case is not necessarily very realistic, it does render the problem polynomially solvable (Van den Berg and Gademann, 1999).

Interestingly other versions of the SCP, with more than one source and sink node, are also polynomially solvable under certain conditions. For example, Ball and Magazine (1988) examine a manufacturing problem arising for the insertion of chips on printed circuit boards. The problem is to find the least cost (in this case least distance) sequence of insertion operations when each chip must be taken from a specific feeder and placed in a specific location on the circuit board. Ball and Magazine (1988) reveal that when the travel metric used is the Manhattan metric (or another *additive metric*) the problem becomes polynomially solvable. In effect, this transformation reduces the problem to a Gilmore-Gomory TSP (Kabadi, 2002). In  $\mathbb{R}^2$ , with an Euclidian travel metric, however, this manufacturing problem reduces to the rural postman problem and is therefore *NP*-hard.

The reduction of SCPs to the rural postman problem (RPP) is not unusual. Theoretically, the Gilmore-Gomory TSP can be generalized into a

Minimum Cost Steiner directed pseudograph Problem with Node-Deficiency Requirements (MCNDP) (Kabadi, 2002). In turn, the RPP is a special case of the MCNDP. Vis and Roodbergen (2008) exploit this relationship between the MCNDP and RPP in studying the scheduling of container storage and retrieval at a port terminal. Specifically, they examine the scheduling of a straddle carrier as it moves containers from/to seaside to/from the stack and from/to landside to/from the stack. Given the layout of the stack and the constraints associated with straddle carrier motion, Vis and Roodbergen (2008) are able to reformulate this problem as a rural postman problem which they subsequently reformulate as an asymmetric Steiner Traveling Salesman Problem — solvable to optimality in polynomial time.

We now pause to reflect on two interesting features of the SCP algorithms cited here — the time complexity of these algorithms and the ideology behind the algorithms. All of the algorithms presented in this section, whether approximate or exact, are dominated by the time required for the minimum spanning tree (MST) algorithm. At the time that Atallah and Kosaraju (1988) and Frederickson and Guan (1993) were studying the SCP this was  $O(k \log \beta(k, q))$  as provided by Gabow et al. (1986). Since that time, Chazelle (2000) has put forth a faster MST algorithm that runs in  $O(k\alpha(k, q))$  where  $\alpha(k, q)$  is an inverse Ackermann function (Ackermann, 1928). By design the Ackermann function grows extremely fast, as such the inverse Ackermann function grows extremely slowly — rendering the time complexity of the MST algorithm nearly linear.

The fact that all of the SCP algorithms, both exact and approximate, exhibit a time complexity dependent on the MST algorithm is due to the ideology underlying these algorithms. These algorithms may be broadly described in two steps: 1) balance the network (i.e. create a network in which every node has equal in- and out- degree) and 2) connect any emerging components into a proper tour.

The first step serves to create a set of tours that include all of the jobs. These tours may be disjoint, leading to a set of strongly connected components or disjoint cycles; or if only one component emerges from this step, then

the optimal solution has been obtained at this stage. Different algorithms perform this first step in different ways, some judiciously add “augmenting” edges to create a balanced network while others solve an assignment problem version of the original SCP (or the version as transformed into the ATSP). The assignment problem version of the SCP represents a relaxation in which all jobs must precede or follow another (not necessarily distinct) job in such a way that the total inter-job costs are minimized. For example, given three jobs, the assignment problem relaxation may yield a solution such that job 1 precedes and follows job 2 just as job 2 precedes and follows job 1; meanwhile job 3 precedes and follows job 3. The assignment problem therefore yielded two disjoint cycles — one with two jobs and one with one job. This is in contrast to the SCP (or ATSP) that requires all jobs to be ordered within a single tour. Regardless of which edge augmenting or AP-based approach is used, the result is the same — a tour comprised of one or more strongly connected components.

The second step in this broad description of SCP algorithms is to connect the components in a manner that incurs the least additional cost. It is in this step that the MST algorithms are invoked on a graph whose nodes are the strongly connected components of the balanced network and whose edges are the least cost edges connecting the components. Once the MST has been identified the method of exploiting it to create a full tour including all jobs varies across the algorithms. Some algorithms use the edges of the MST to make a single tour through all components while others use the MST results to perform an intricate series of edge exchanges. Nevertheless, the outcome is a single tour including all jobs.

The SCP is a very specific case of a traveling salesman problem with pick-up and deliveries that fits into the broader range of problems known as traveling salesman problems with precedence constraints (TSPPC) (Ascheuer et al., 2000). In the TSPPC, the pick-up location must be visited before the delivery location. If no intermediate points can be visited between each pick-up and delivery, then the problem is a SCP. If, however, it is possible for the salesman to visit intermediate pick-up and delivery points for some jobs then

the problem becomes a swapping problem. The next subsection presents a brief review of the relevant literature on both the Swapping Problem and the Preemption Problem.

### 2.2.1 Variations: Swapping and Preemption Problems

The primary purpose of this review is to provide a fundamental examination of routing problems that arise in container transport. Due to the specialized equipment required to load and unload containers, preemption is not allowed in many real-world drayage problems. There are, however, some cases where a container may be dropped at an intermediate terminal for the addition or removal of cargo before continuing to its destination. Within a container terminal, containers may also be diverted to intermediate locations for customs inspections or load consolidation activities. We, therefore, make a small side-step to examine the literature on single vehicle, unit load, pick-up and delivery problems with preemption.

If only a subset of the jobs in any given problem instance are allowed to be preempted, then the problem is known as the Swapping Problem. The Swapping Problem was originally studied by Anily and Hassin (1992). They examine the problem on a general graph, demonstrate that it is *NP*-hard, and then propose a polynomial time  $\frac{5}{2}$ -approximation algorithm based on the patching algorithm of Gilmore and Gomory. Chalasani and Motwani (1999) subsequently exploited a matroid intersection technique to obtain a 2-approximation algorithm for this problem. To date, this stands as the best polynomial approximation algorithm for the swapping problem on general graphs.

Given the relationship between this problem and the Gilmore-Gomory TSP, it is not surprising that when the underlying graph is a line, then the problem may be solved to optimality in polynomial time. Anily et al. (2000) demonstrate this result by presenting an  $O(n^2)$  algorithm to compute the optimal solution for the swapping problem on a line. If this problem, on a line, is relaxed further to permit preemption for *all* jobs, then Atallah and Kosaraju (1988) show that there exists an  $O(n+k)$  polynomial time algorithm

to compute the optimal solution, where  $n$  and  $k$  are as before the number of jobs and the number of terminals, respectively. This extreme case of the Swapping Problem is what we term the Preemption Problem.

Frederickson and Guan (1992) study the preemption problem in the case that the underlying graph is a tree, demonstrating that in this case the preemption problem is also polynomially solvable. They show this by presenting two algorithms, one that solves the problem in  $O(n + kq)$  time and one that solves it in  $O(n + k \log k)$  time, where  $n$ ,  $k$ , and  $q$  are as before the number of jobs, the number of terminals, and the number of strongly connected components. If, however, job preemption is only permitted for a subset of jobs and the number of times each “preemptable” job may be preempted is restricted, then the problem is again *NP*-hard. This very specific case of the swapping problem was studied by Krumke et al. (2008). They present a  $(\frac{4}{3} + \epsilon)$ -approximation algorithm when the underlying graph is a tree. (Notice that this result is in keeping with the  $\frac{4}{3}$ -approximation algorithm of Frederickson and Guan (1993) for the stacker crane problem on trees.) It was shown by Coja-Oghlan et al. (2006), via probabilistic analysis, that when the underlying SCP graph is a tree, the  $\frac{4}{3}$ -approximation algorithm (originally proposed by Frederickson and Guan (1993)) yields a minimum cost tour with a certificate of optimality for almost all SCP inputs. The question then arises: how do these worst case results for the stacker crane, swapping, and preemption problems stand up in the average case? in cases from the “real-world”?

## 2.3 Empirical Studies

The literature highlighted in this review, thus far, has been theoretical in nature. The documents described have provided only worst case bounds on the polynomial time algorithms presented. We now turn our attention to a review of a few empirical studies performed on stacker crane problem datasets.

Johnson et al. (2002) provide a comprehensive summary of experimental results for twelve heuristics on twelve classes of problems. Eleven of these twelve classes are based on randomly generated instances designed to reflect



theoretically interesting problems as well as real applications while the twelfth class is comprised of data originating from problems in the real-world. One of the eleven randomly generated classes portrays the stacker crane problem.

The random SCP instances used in Johnson et al. (2002) have their origin in the work of Cirasella et al. (2001). Cirasella et al. (2001) develop ten instances with 100 jobs, ten with 316 jobs, three with 1,000 jobs, and one with 3,162 jobs. These instances were created by randomly selecting a pick-up location for each job out of a square that is  $10^6 \times 10^6$  units large; the delivery location for each job is then selected based on a random pick of two integers from an interval of size  $[\frac{-10^6}{u}, \frac{10^6}{u}]$ , where  $u$  is an integer ranging from 10 to 56 depending on the number of jobs in the instances. These two randomly selected integers, serving as  $x$  and  $y$  coordinates, are then added to the pick-up coordinates to obtain the delivery location. In the class of problems originating from the real-world there are four SCP instances containing 323, 358, 403, and 443 storage and retrieval jobs derived from operations in a Siemens factory in Augsburg (Carpaneto et al., 1995).

Using these datasets, Johnson et al. (2002) report the results of twelve different heuristics. The solution quality of these heuristics is reported in terms of the percentage offset from both the assignment problem relaxation (that is, the value of the least cost solution allowing subtours) and the Held-Karp (H-K) lower bound (a lower bound derived from a linear relaxation of the problem). Note, while the assignment problem is a relaxation on the tour structure (as explained in Section 2.2), the H-K bound is based on a relaxation of the constraint that every job must be preceded/followed by exactly one job. In the H-K formulation, each job may be preceded or followed by a number of “fractional” jobs so long as the sum of all fractional jobs totals one. The results on the random instances reveal that no heuristic (even those following the two broad steps outlined in Section 2.2) provided a solution better than 1.21% from the H-K bound. While these solutions may actually be the optimal solution there is no way of deducing this from the results presented in Johnson et al. (2002). The results from the real-world instances are strikingly different — all instances were solved to optimality by at least three of the heuristics.

Of note is that these three heuristics all follow the general structure as that laid out in Section 2.2.

This difference between random and real-world SCP instances is not necessarily surprising, but it does beg the question: why? We first note that the random instances were constructed such that there is roughly a constant number of other pick-up points closer to a given pick-up point than its associated delivery point. This yields some clustering, but not to the extent normally seen in real-world SCPs — and particularly, real-world drayage problems. For example, in the warehousing problem described by Van den Berg and Gademann (1999) all jobs have either a pick-up or delivery point at one of two locations; the problem studied by Ball and Magazine (1988) had pick-up points at a fixed number of locations along a line; and in the marine terminal problem of Vis and Roodbergen (2008), the pick-up and delivery points were restricted to the rows of a container stack. As another example, an average day of data for the drayage problem in this thesis has only six terminals and 19 customer locations as the end points for 65 jobs.

A second reason for the discrepancy in results may come from the metric employed to describe the distance between two jobs. In the random instances this distance was straight line distance whereas in the real-world storage and retrieval instances, travel between jobs is restricted to a manhattan like grid of racks and shelves. As Ball and Magazine (1988) showed, the travel metric can have a significant impact on a problem’s solvability.

We are not the first to wonder what separates the theoretical world from the real-world, or generally unsolvable from generally solvable problems, in the realm of asymmetric traveling salesman problems. Miller and Pekny first raised the question in their 1991 *Science* article when they stated: “The results show that the algorithm performs remarkably well for some classes of problems, determining an optimal solution even for problems with large numbers of cities, yet for other classes, even small problems thwart determination of a provably optimal solution.”

Frieze et al. (1995) began to answer this question by raising another question — “When is the assignment bound tight for the asymmetric traveling-

salesman problem?” They proceed to answer this question by showing that if the expected number of zeros in a row of the distance matrix tends to infinity, as the number of jobs in the problem tends to infinity, then the probability that the assignment problem solution is the ATSP solution tends toward one. This result is similar to an observation made by the pioneering graph theorist Frank Harary when he noted “it is not so much a matter of how many zeros a matrix has but rather their strategic location” (Harary, 1962b).

Considering realistic distributions in the distance matrix, they conjecture that if the distribution is uniform over the integers in the interval from zero to some constant times the number of jobs in the problem, then the probability that the solution to the assignment problem is the solution to the ATSP is some value less than one, but dependent on the constant. This result is similar to a result obtained by Zhang and Korf (1996). They determined that if the distances between jobs in the asymmetric TSP are drawn uniformly from the set of integers ranging from zero to  $r$ , where  $r$  is an arbitrary positive constant, then a branch and bound solution algorithm experiences an easy to hard complexity transition as  $r$  increases. Complexity transitions, similar to phase transitions in chemistry (e.g. the transition from ice to water as the temperature rises), indicate that the time required for an algorithm to run will steeply increase given a change in a specific instance parameter. Usually such transitions are found empirically, for specific algorithms, by varying one parameter across a wide range of input problem instances. In this way, by varying  $r$ , Zhang and Korf (1996) conclude that the number of distinct intercity (or interjob) distances in an asymmetric traveling salesman problem is the control parameter with the most influence on problem complexity. While the evidence supporting this conclusion is particularly compelling another interpretation is possible. That is, as  $r$  increases, not only will the number of unique distances increase, but so will the average distance. Thus, it seems that Zhang and Korf (1996) could also have concluded that the average intercity distance is the control parameter with the most influence. While this conclusion is less compelling, as any instance can be appropriately scaled before solving, it does warrant consideration in the context of container transport

within a terminal (shorter distances) versus container transport outside of a terminal (longer distances). Both the number of unique distances and the average distance in a given ATSP instance will be studied further in Chapter 3.

In a similar vein, others have examined the correlation between asymmetric traveling salesman problem parameters and the solvability of the problem. One popular parameter is *symmetry* or the related, *asymmetry*. Cirasella et al. (2001) define symmetry as the ratio of the standard deviation of intercity distances in a “symmetrized” version of the problem to the standard deviation of intercity distances in the original problem; a value of one implies that the original problem was symmetric. By this measure, the 1,000 job instance of the randomly generated stacker crane problem has a symmetry of .9998 (Cirasella et al., 2001). Johnson et al. (2002) define asymmetry as the ratio of the average value of  $|d_{ij} - d_{ji}|$ , where  $d_{ij}$  represents the distance from city (or job)  $i$  to city  $j$ , to the average value of  $|d_{ij} + d_{ij}|$ . By this metric, the randomly generated 1,000-job instances, of the SCP, have an average asymmetry of .020; the real-world problems with 443, 403, 358, and 323 jobs have an asymmetry of .229, .231, .226, and .206, respectively (Johnson et al., 2002). In a study of the online ATSP, Ausiello et al. (2008) also found it helpful to introduce a measure named *maximum asymmetry*. This measure is defined as the supremum over all jobs of the ratio of the distance between the jobs in one direction to the distance in the other direction (i.e.  $\sup_{i,j} \frac{d_{ij}}{d_{ji}}$ ).

The studies reviewed in this section are encouraging as they provide clues to features that may indicate whether an existing algorithm will yield (in reasonable time) an optimal solution to a given instance of the ATSP. While the ATSP is known to be *NP*-hard this alone does not speak to the solvability of any one instance of this problem — the instance’s parameters may serve as the telling feature. Thus, as we continue to work with drayage problems arising in the real-world, a careful examination of characteristic features is required. Namely the distance matrix should be studied for the prevalence of zeros, the presence of an underlying distribution of distances, the number of distinct distances, and measures of symmetry or asymmetry.

## 2.4 Implications

This review serves to highlight three interesting features of the Stacker Crane Problem — 1) the best approximation algorithm for a general SCP holds at  $\frac{9}{5}$ , 2) the real-world includes a myriad of SCPs that are consistently solvable to optimality in reasonable time, and 3) the difference between readily-solved and slow-solving problem instances appears to lay in discernible structures within the underlying distance matrix. These last two observations yield insights for the real-world of drayage.

Specifically, drayage providers may be working in environments that force a beneficial distance matrix structure. For example, dray providers operating in a port environment may only serve terminals that are aligned along the shoreline — thus, their problem would be analogous to one on a line, solvable in polynomial time. Alternatively, some dray providers serve only a limited number of customer or terminal locations, resulting in a distance matrix with a significant number of zeros coupled with a small number of unique distances. Furthermore, container operations in a limited geographic area (e.g. within a single terminal) will on average have shorter inter-job distances with only a limited number of unique distances. These distance matrix features may yield benefits to the routing algorithm.

In order to fully investigate these suspicions, a proper analysis of drayage problem metrics should be undertaken. More specifically, the distance matrix structure of a variety of ATSPs should be examined and if possible, correlated to the ease with which an optimal solution can be found. From this approach, we may be able to identify SCPs and more specifically drayage problems as special “easy” instances of the ATSP. Alternatively, we may be able to identify a probabilistic distribution, centered on SCP or drayage instance features relative to general ATSP features, correlating to algorithmic performance. This is the pursuit of the next chapter.

## Chapter 3

# Are SCPs Easy?

## An Empirical Exploration

We shall not cease from exploration  
And the end of all our exploring  
Will be to arrive where we started  
And know the place for the first time.

T. S. Eliot,  
*No. 4 of Four Quartets, Little Gidding.*

Is searching for a needle in a haystack hard? Yes and no. Yes, because the process can be frustrating, tedious, and time consuming. No, because the process itself is not so difficult: pick up something from the haystack, examine it, if it is a needle, stop, if not, continue.

Notice, however, that the success of this process, or algorithm, is vitally dependent on two assumptions. First, that there is at least one needle in the haystack. Second, that the haystack-searcher knows a needle when s/he sees one. If these two assumptions are fulfilled, then the time required in searching for the needle is dependent on two (possibly related) considerations. One, how big the haystack is — a big haystack will likely take longer than a small haystack. Two, where in the haystack the needle is hiding — if it is near the

top (or visible on an edge of the stack), the haystack-searcher will most likely find the needle faster than if it is at the bottom of the stack.

If we can infer that the size or structure of a haystack has an influence on the probable location of the needle in the haystack, then perhaps we can design a better search algorithm. For example, if we know that in large haystacks the needle is more likely to be on the bottom, then the haystack searcher can expedite their search by starting at the bottom of the stack. Unfortunately, haystack research is not so advanced at this stage, and the haystack searcher must toil with the most rudimentary of algorithms: one straw at a time. . .

Leaving our haystack-searcher to his/her Sisyphean task, we turn our attention to another character (previously-introduced in Chapter 2) with a similarly “hard” problem, the Stacker Crane. In the SCP, we are faced with the task of finding the least cost way to move a given number of containers, from various pick-up locations to various drop-off locations, from among a (possibly large) set of feasible ways to move the same containers. Thus, like our haystack-searcher, the SCP-solver has a task that is not necessarily difficult, but is excessively time consuming.

In fact, the SCP has an underlying graph structure that places it alongside the TSP in the class of problems termed *NP*-Hard (as defined in Chapter 2). A name which tends to influence one’s perception of the problem’s difficulty. Therefore, when we ask, “are SCPs easy?”, we are not asking about the proven worst-case performance of existing algorithms, but rather the likelihood that any one instance (or type) of SCP can be solved quickly by existing algorithms (i.e. in much less time than the forecasted worst-case time of the existing algorithms).

Indeed, as foreshadowed in the previous chapter, the literature contains several references to the SCP as an “easy” problem amidst the arc routing problems and/or the ATSPs. For example, Miller and Pekny (1991); Carpaneto et al. (1995); Frieze et al. (1995); Zhang and Korf (1996); Laporte (1997) have all noted that while ATSPs are *NP*-hard, some instances are readily solved to optimality in only a short amount of time. Laporte (1997) even goes so far as to state specifically that when an SCP is transformed to

an ATSP, it becomes easy to solve to optimality using existing TSP solution techniques. Admittedly the only proof Laporte offers up on this matter is the statement: “We have solved without difficulty instances of the directed RPP and of the SCP involving up to 220 vertices and 660 arcs by a direct application of the Carpaneto and Toth algorithm.” (Laporte, 1997). The reader is left to wonder, from where does the relative ease of solving an SCP stem? Is this due to some feature of the original arc routing graph? And does that feature manifest itself in the distance matrix?

This chapter focuses on answering these questions via an empirical analysis of SCP and ATSP distance matrices. Figure 3.1 depicts how this empirical analysis was conducted and how the sections of this chapter align with that analysis. Specifically, this approach is dependent on three tools — a set of distance matrices for a variety of ATSPs including SCPs (Section 3.1), a method by which to declare an instance “solvable” (Section 3.2), and a set of appropriate distance matrix metrics that can serve to distinguish problems in terms of solvability (Section 3.3). Following these three sections on methodology the resulting models are revealed (Section 3.4) and verified (Section 3.5).

## 3.1 ATSP Instances

The dataset used in this study contains 379 ATSP distance matrices. Those matrices came from four different sources. First, the TSPLIB instances (27 instances), as posted at <http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/atsp/>. Second, the problem instances with fewer than 1000 jobs from sources other than the TSPLIB, as described in Johnson et al. (2002) and Cirasella et al. (2001) and specified as “Benchmark Instances” at <http://www.research.att.com/~dsj/chtsp/atsp.html> (19 instances). Third, we included a set of instances that were derived from a drayage problem at the Port of Rotterdam (66 instances). Fourth, we used the twelve random instance generators described in Johnson et al. (2002) and Cirasella et al. (2001) and posted at <http://www.research.att.com/~dsj/chtsp/atsp.html>. With these generators we constructed all of the instances



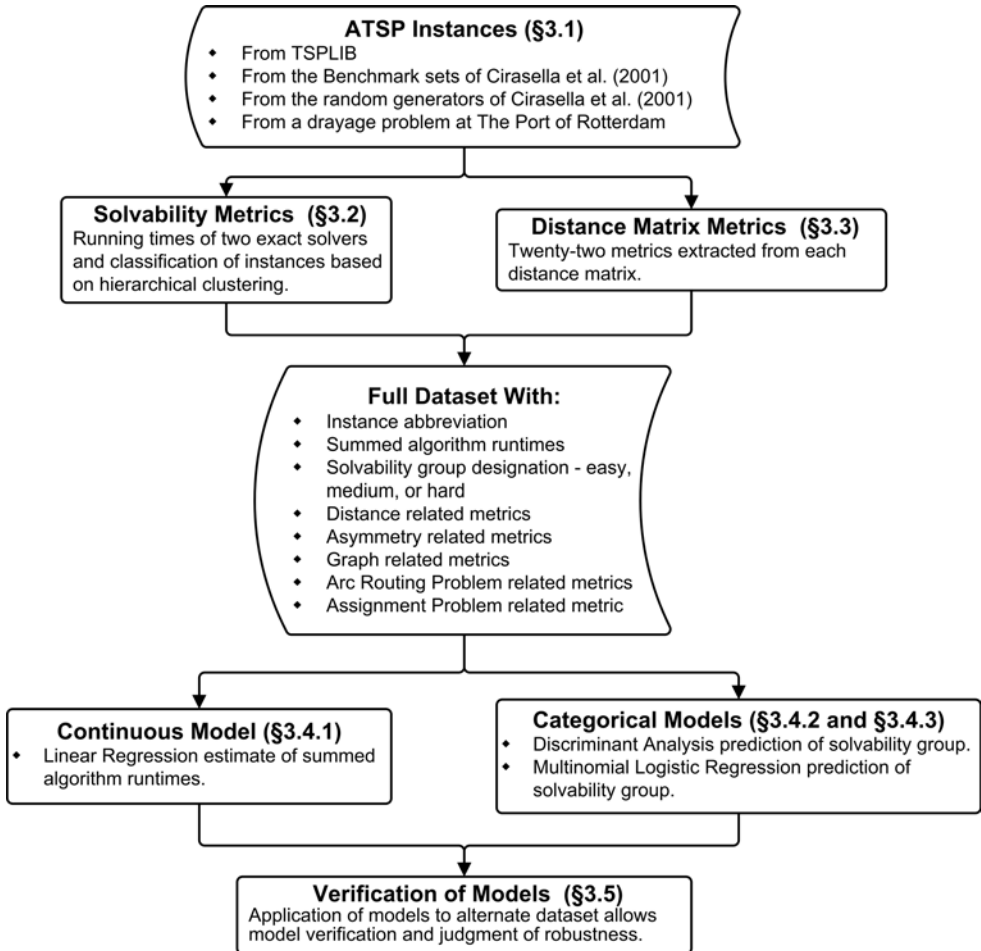


Figure 3.1: Depiction of data analysis approach along with the structure of Chapter 3.

with 100 and 316 jobs as described in Cirasella et al. (2001) (240 instances). We also used nine of generators (amat, coin, crane, disk, rect, rtilt, shop, super, tmat) to create three 66-job instances each (27 instances). We therefore had a data set including 379 distance matrices.

These 379 instances represent roughly 18 different problem types ranging in size from 16 to 932 jobs or nodes. The problem types, number of instances of that type, along with their sources are listed in Table 3.1. These instances

are all well documented in the references cited in the table. The drayage problems included as part of the real-world SCPs, stem from the Dutch LSP introduced in Chapter 1.

Specifically, the Dutch LSP provided us with a set of operational data tables. In all, these tables spanned operations from January 2002 to October 2005 as well as from January 2006 through March 2006. The tables represented jobs that were planned to be served on a given day. After a preliminary review of the data, we concluded that on average 65 jobs were served per day, at customer and terminal locations associated with less than 25 distinct zipcodes. Using these parameters, we extracted a random sample of appropriately defined jobs from the original data-set in order to generate a set of 33 days with 65 jobs per day. Recall, from Chapter 1, that each job our LSP served might actually be considered a combination of two jobs — the trip from/to a terminal to/from a customer location and the trip from/to a customer location to/from a terminal. Using this observation, we created a second set of 33 days with 130 jobs. In each set, we added one job with both the pick-up location and drop-off location at the home terminal (i.e. this job had a loaded distance of zero). The optimal route was then parsed such that this job represents the first (and last) job on the route. In this way we could ensure that we obtain a tour beginning and ending at the home terminal of the LSP. Finally, in order to match the formatting of the TSPLIB instances, we rounded all distances to the nearest integer. Hence, we added a total of 66 drayage instances, 33 with 66 jobs and 33 with 131 jobs, to our set of ATSP instances. (Note, more detail on the data from the Dutch LSP may be seen in Chapter 5, Section 5.2.2.)

## 3.2 Hard or Easy? Measuring Solvability

The trick to answering the question at the center of this research — *from where does the relative ease of solving an SCP stem?* — rests on having a good definition for “ease of solving”. Common sense dictates that a problem is easy to solve if it can be solved to optimality quickly. This definition,

Table 3.1: Summary of instances in primary dataset.

| Problem                                                                | Abbrv.    | #   | Jobs                                                                       | Source                          | Reference                                          |
|------------------------------------------------------------------------|-----------|-----|----------------------------------------------------------------------------|---------------------------------|----------------------------------------------------|
| Real-world SCP                                                         | PK66*     | 33  | 66                                                                         | Port of Rotterdam               | This Dissertation                                  |
|                                                                        | PK131*    | 33  | 66                                                                         |                                 |                                                    |
|                                                                        | rbg*      | 4   | 323, 358, 403, 443                                                         | TSPLIB                          | (Carpaneto et al., 1995), (Ascheuer, 1995)         |
| Generated SCP                                                          | crane66*  | 3   | 66                                                                         | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | crane100* | 10  | 100                                                                        |                                 |                                                    |
|                                                                        | crane316* | 10  | 316                                                                        |                                 |                                                    |
| Real-world Scheduling Problems                                         | ft*       | 2   | 53, 70                                                                     | TSPLIB                          | (Fischetti and Toth, 1992)                         |
|                                                                        | p43       | 1   | 43                                                                         |                                 |                                                    |
|                                                                        | td        | 2   | 100, 316                                                                   | Cirasella et al. (2001)         | (Cirasella et al., 2001)                           |
| Generated Scheduling Problems                                          | shop66*   | 3   | 66                                                                         | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | shop100*  | 10  | 100                                                                        |                                 |                                                    |
|                                                                        | shop316*  | 10  | 316                                                                        |                                 |                                                    |
| Real-world Routing Problems                                            | ftv*      | 17  | 33, 35, 38, 44, 47, 55, 64, 70, 90, 100, 110, 120, 130, 140, 150, 160, 170 | TSPLIB; Cirasella et al. (2001) | (Fischetti et al., 1994); (Cirasella et al., 2001) |
|                                                                        | big702    | 1   | 702                                                                        |                                 |                                                    |
| Generated Routing Problems                                             | coin66*   | 3   | 66                                                                         | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | coin100*  | 10  | 100                                                                        |                                 |                                                    |
|                                                                        | coin316*  | 10  | 316                                                                        |                                 |                                                    |
|                                                                        | disk66*   | 3   | 66                                                                         |                                 |                                                    |
|                                                                        | disk100*  | 10  | 100                                                                        |                                 |                                                    |
| disk316*                                                               | 10        | 316 |                                                                            |                                 |                                                    |
| Real-world Robotic Motion Problems                                     | atex*     | 5   | 16, 32, 48, 72, 600                                                        | Cirasella et al. (2001)         | (Cirasella et al., 2001)                           |
| Generated Robotic Motion Problems                                      | rtilt66*  | 3   | 66                                                                         | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | rtilt100* | 10  | 100                                                                        |                                 |                                                    |
|                                                                        | rtilt316* | 10  | 316                                                                        |                                 |                                                    |
|                                                                        | stilt100* | 10  | 100                                                                        |                                 |                                                    |
|                                                                        | stilt316* | 10  | 316                                                                        |                                 |                                                    |
| Real-world Data Compression Problems                                   | dc*       | 9   | 112, 126, 134, 176, 188, 563, 849, 895, 932                                | Cirasella et al. (2001)         | (Cirasella et al., 2001)                           |
| Real-world Code Optimization Problems                                  | code*     | 2   | 198, 253                                                                   | Cirasella et al. (2001)         | (Young et al., 1997)                               |
| Generated Approximate Shortest Common Superstring Problems             | super66*  | 3   | 66                                                                         | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | super100* | 10  | 100                                                                        |                                 |                                                    |
|                                                                        | super316* | 10  | 316                                                                        |                                 |                                                    |
| Randomly generated asymmetric matrices obeying the triangle inequality | tmat66*   | 3   | 66                                                                         | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | tmat100*  | 10  | 100                                                                        |                                 |                                                    |
|                                                                        | tmat316*  | 10  | 316                                                                        |                                 |                                                    |
| Randomly generated symmetric matrices                                  | smat100*  | 10  | 100                                                                        | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | smat316*  | 10  | 316                                                                        |                                 |                                                    |
| Randomly generated symmetric matrices obeying the triangle inequality  | tsmat100* | 10  | 100                                                                        | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | tsmat316* | 10  | 316                                                                        |                                 |                                                    |
| Randomly generated symmetric matrices using rectilinear distances      | rect66*   | 3   | 66                                                                         | Cirasella generators            | (Cirasella et al., 2001)                           |
|                                                                        | rect100*  | 10  | 100                                                                        |                                 |                                                    |
|                                                                        | rect316*  | 10  | 316                                                                        |                                 |                                                    |
| Symmetric matrices perturbed to be asymmetric                          | ry48p     | 1   | 48                                                                         | TSPLIB                          | (Fischetti and Toth, 1992)                         |
|                                                                        | kro124p   | 1   | 100                                                                        |                                 |                                                    |
| Unknown Origin                                                         | br17      | 1   | 17                                                                         | Cirasella generators            | (Cirasella et al., 2001)                           |

while a good start, does raise a further question: by what means should we solve the problem? To answer this question, the following two subsections describe two exact solution algorithms employed to measure solvability; the final subsection describes how the results from applying the two algorithms were used to describe instances as solvable.

### 3.2.1 Concorde

The first method used to determine ease of solvability is that encoded in the *concorde* solver<sup>1</sup>. Nearly all the details of the algorithms employed in the *concorde* code are beautifully documented in the book, *The Traveling Salesman Problem: A Computational Study* by Applegate et al. (2007). We therefore limit ourselves to simply stating that *concorde* uses a branch-and-cut approach (that is a cutting plane method embedded within a branch-and-bound search) to find the optimal solution of symmetric TSPs. Thus, in order to use *concorde* on our ATSPs we had to transform them from ATSPs into STSPs.

For this we employed the 2-node transformation of Jonker and Volgenant (1983). This transformation works by adding a copy of each node in the original problem. The new inter-node costs,  $c_{ij}$ , are then set based on the original set of inter job distances,  $d_{ij}$ , as follows: zero for the cost between a node and the copy of that node;  $d_{ij} + M$  for the cost between the copy of node  $i$  and the original node  $j$ ;  $d_{ji} + M$  for the cost between the original node  $i$  and the copy of node  $j$ ; and  $+\infty$  for all remaining edges. In order to manage the memory consumption of these instances, we tried to make a conservative selection for  $M$ , setting it to two times the maximum distance in the problem instance. This was done for all instances except *stilt316\**, *rtilt316\**, *disk66\**, *disk100\**, *disk316\**, and *code\** for which  $M$  was set to 1,000,000. Similarly, we used 99,999,999 as a proxy for  $+\infty$ . Figure 3.2 depicts this transformation on a small example with 4 nodes.

After “symmetrizing” all 379 instances, *concorde* was used with 123 specified as the random seed and 16 as the chunk size parameter (in conjunction

---

<sup>1</sup>available at <http://www.tsp.gatech.edu/concorde/>

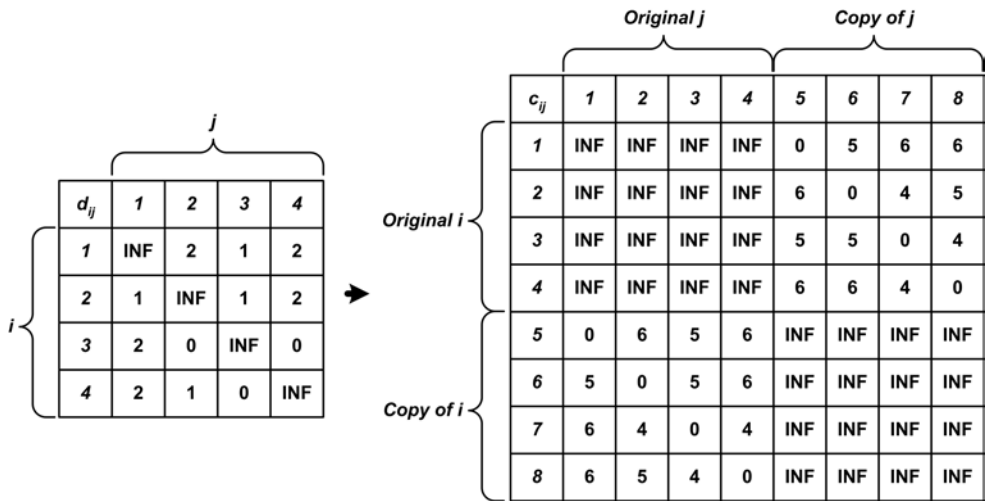


Figure 3.2: Depiction of transformation from ATSP (with 4 nodes) to STSP (with 8 nodes); INF stands for  $+\infty$ ;  $M = 4$ .

with the qsopt LP package<sup>2</sup>) to solve these instances with a 10,000-second limit on running time. Note, these specifications, both the 2-node transformation and the 10,000-second runtime, follow those found in Fischetti et al. (2002). In addition to being consistent with the literature, the selection of a 10,000-second runtime has little impact on the results. In general, if a problem is not solved in 5,000-seconds, then it will also not be solved in 10,000-seconds. From the results, we recorded, for each instance, the running time in seconds (10,000 if the time limit was reached) along with the number of branch-and-bound nodes required by the algorithm. Branch-and-bound nodes serve as a measure of the number of subproblem sets examined as part of the solution algorithm. As such the number of branch-and-bound nodes also serves as a good indicator regarding the ease of solvability.

<sup>2</sup>available at <http://www2.isye.gatech.edu/~wcook/qsopt/>

### 3.2.2 Tsp\_solve version 1.3.6

The second method used to determine ease of solvability is that encoded in `tsp_solve v. 1.3.6`. `Tsp_solve v. 1.3.6`, primarily coded by Chad Hurwitz, represents a package<sup>3</sup> of both STSP and ATSP solution algorithms and heuristics (both tour finding and tour improving). For our study, we used only one exact solution algorithm, from the suite, for the ATSP — that designated in the software as “assign”. Assign is an implementation of the branch-and-bound algorithm originally proposed by Carpeneto and Toth (1980) in which a modified assignment problem is the means by which the branching is conducted. A modified assignment problem is an assignment problem with additional constraints describing arcs that must be included and arcs that must be excluded in the solution.

Just as with the `concorde` solver, we set a 10,000-second time limit on the running time of `tsp_solve` for any one instance. As `tsp_solve` was really designed for problems with fewer than 175 nodes, there were many instances that were either not attempted by `tsp_solve` or that reached the time limit without finding the optimal solution. For these instances, the running time was recorded as 10,000-seconds.

### 3.2.3 Solvability

Given the results of both `concorde` and `tsp_solve`'s `assign` algorithm, we created two measures of solvability — one continuous and one a classification. The first measure is simply the sum of the two solvers' runtimes. The second is a classification of the instances into easy, medium, and hard categories based on the summed runtimes and the branch-and-bound nodes in `concorde`. The first measure is straight forward and requires no additional explanation. The second measure was derived using hierarchical clustering.

Hierarchical clustering initially places each instance in its own cluster then iteratively combines clusters based on the Euclidean distances (standardized to range from zero to one) between the instances, with regards to the specified

---

<sup>3</sup>freely available at <http://www.cs.sunysb.edu/~algorithm/implement/tsp/implementation.html>

solvability metrics. In this case the metrics used were the summed runtimes and the number of branch-and-bound nodes needed in the *concorde* solver. Note, if *concorde* could not solve the problem in 10,000-seconds, the runtime was specified as 10,000-seconds and the number of branch-and-bound nodes was specified as 1,000.

The success of the hierarchical clustering approach depends on a dataset without obvious outliers. To mitigate this problem, we ran the hierarchical clustering procedure, in SPSS, on the z-scores of the two measures. That is we subtracted from each runtime value the mean of all the runtimes (7976.26-seconds) and then divided by the standard deviation of runtimes (5577.01-seconds). Similarly, the z-scores were calculated for the branch-and-bound node metric, which had a mean of 87.26 and a standard deviation of 244.89. From this procedure we obtained five clusters; we then manually combined the three “hardest” groups into one cluster for a total of three clusters.

These three clusters largely correspond to three groups observable in the runtime data. Given the cutoff time of 10,000-seconds, the runtime data reveals three distinct categories with distinct means and standard deviations. First, there were 108 problem instances that could be solved by both solvers within the time limit — with a mean of 35.86-seconds and a standard deviation of 166.35-seconds. Second, there were 248 instances that could only be solved on *concorde* within the time limit — these instances had a mean of 10,319.08-seconds and a standard deviation of 1022.53-seconds. Finally, there were 23 instances that neither solver could crack — these (clearly) had a mean of 20,000-seconds with no deviation.

Returning to the clusters derived via hierarchical analysis, Table 3.2 summarizes the solvability metrics across the three hierarchical clusters corresponding to easy, medium, and hard instances. Figure 3.3 shows, by means of a scatterplot, how the three solvability clusters fall with regards to branch-and-bound nodes on the x-axis and total solver running time on the y-axis.

Table 3.2: Summary of solvability statistics by cluster.

|              |            | Sum of runtimes<br>(Concorde and tsp_solve) |                | Branch-and-bound nodes<br>(Concorde) |               |
|--------------|------------|---------------------------------------------|----------------|--------------------------------------|---------------|
| Cluster      | N          | Mean                                        | Std. dev.      | Mean                                 | Std. dev.     |
| Easy         | 108        | 35.86                                       | 166.35         | 5.07                                 | 7.09          |
| Medium       | 235        | 10122.12                                    | 345.56         | 19.10                                | 34.26         |
| Hard         | 36         | 17789.83                                    | 3240.49        | 778.75                               | 309.59        |
| <b>Total</b> | <b>379</b> | <b>7976.27</b>                              | <b>5577.01</b> | <b>87.26</b>                         | <b>244.89</b> |

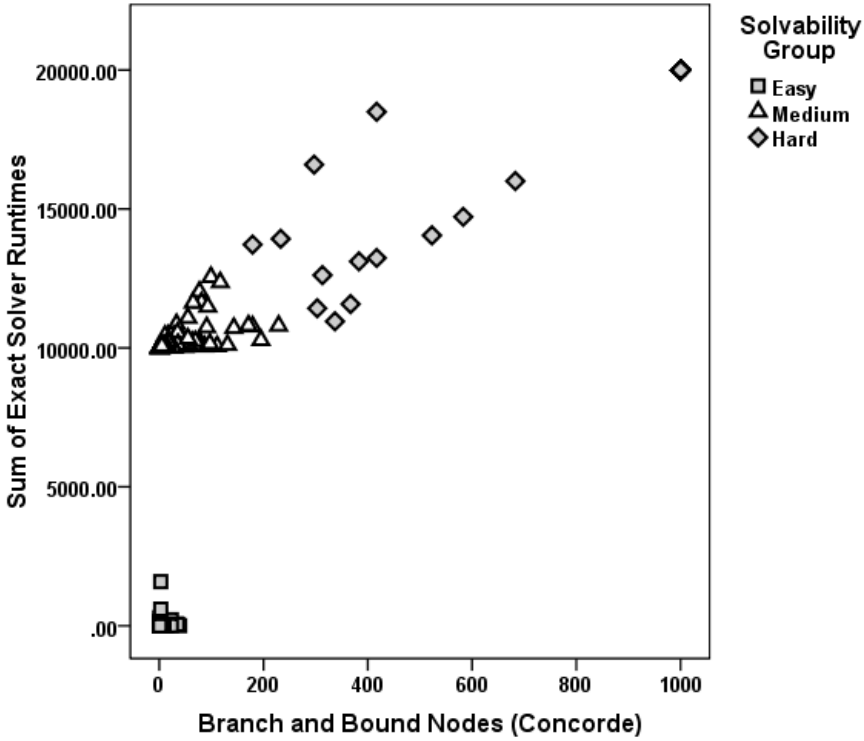


Figure 3.3: Depiction of instances with regards to solvability metrics.



### 3.3 Distance Matrix Metrics

Predicting solvability requires a method by which to describe each instance in quantifiable terms. With nothing other than a distance matrix to describe each instance, we rely on the fields of both linear algebra and graph theory to determine meaningful metrics for the matrices. In particular, the graph and matrix work of Frank Harary was quite inspirational in developing these metrics (see e.g. Harary (1959a,b,c, 1962a)). Note, we used the asymmetric matrices to derive these metrics; as opposed to the symmetrized problem version described in Section 3.2.

The most basic of the metrics studied is the number of jobs in the problem instance. This metric is abbreviated as *numJobs*. The remaining metrics fall into five primary categories: distance related (Subsection 3.3.1), asymmetry related (Subsection 3.3.2), graph structure related (Subsection 3.3.3), arc routing problem related (Subsection 3.3.4), and assignment problem related (Subsection 3.3.5).

#### 3.3.1 Distance Related

This set of distance matrix metrics describes the range and grouping of individual values appearing in the distance matrix. These metrics are primarily of a descriptive nature doing little to interpret any underlying structural properties in the matrix or graph. Admittedly, these metrics represent only a fraction of the full number of distance related metrics possible. These particular metrics were selected based on motivation found in the literature. This subsection describes these references along with introducing each distance related metric.

***zeroDist*** This is a count of the number of times that zero appears in the distance matrix. This metric may be normalized by taking the ratio of the number of times zero appears in the distance matrix to the total number of matrix entries (excluding the diagonal entries), i.e.  $\frac{|\{d_{ij}|d_{ij}=0\}|}{n(n-1)}$ ,  $i \neq j$ . This metric is of interest due to the observations of Frieze et al. (1995) regarding the role of zeros in a distance matrix. Specifically, they note,

if the expected number of zeros in a row of a given distance matrix tends toward infinity, as the given number of jobs tends toward infinity, then the probability that the solution to the problem is the solution to the related assignment problem tends toward one.

***uniqueDistCount*** This is a count of the number of unique distances in the distance matrix. This metric can be normalized by dividing the number of unique distances by the total number of matrix entries, i.e. given a sorted list of all the  $d_{ij}$ , relabeled as  $d_k$ ,  $k = 1, \dots, n(n-1)$ , this is  $\frac{|\{d_k | d_k \neq d_{k+1}\}|}{n(n-1)}$ . This feature (the number of unique distances in the distance matrix) is at the heart of a conjecture put forth by Zhang and Korf (1996). They hypothesize that the number of distinct intercity (or interjob) distances in an asymmetric traveling salesman problem is the control parameter with the most influence on problem complexity. Zhang and Korf (1996) test this hypothesis by demonstrating that a branch and bound algorithm experiences an “easy-to-hard” transition as the interval from which distances can be randomly drawn increases.

***minDist*** As the name indicates this measure reflects the minimum value present in the distance matrix. Note, if *zeroDist* is greater than zero, then this measure is zero. This measure is also related to the work of Zhang and Korf (1996) in that it represents the lower bound on the interval from which distances can be drawn.

***avgDist/maxDist*** This measure is the ratio of the average of all distances in the distance matrix (excluding the diagonal entries) to the maximum distance in the distance matrix. Again, this metric is related to the hypothesis of Zhang and Korf (1996) as it captures the spread of distances in the interval from which distances can be drawn.

***minDist/maxDist*** This measure is the minimum distance in the distance matrix divided by the maximum distance (excluding diagonal entries). Note, if *zeroDist* is greater than zero, then this measure is zero.

***maxBinSize*** This measure captures the size of the largest set of entries with

the same distance value. This measure can be normalized by the total number of entries (excluding the diagonal entries) in the distance matrix. Note that if *uniqueDistCount* is one, then the normalized version of this metric assumes a value of one; alternatively, if *uniqueDistCount* is  $n(n - 1)$  then the normalized version of this metric assumes a value of zero.

The mean and standard deviation values of these metrics across the three groups of solvability are presented in Table 3.3.

Table 3.3: Summary of distance related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses).

| Metric                      | Cluster              |                        |                        | Total                  |
|-----------------------------|----------------------|------------------------|------------------------|------------------------|
|                             | Easy                 | Medium                 | Hard                   |                        |
| <b>N</b>                    | 108                  | 235                    | 36                     | 379                    |
| <b>numJobs</b>              | 79.34<br>(21.33)     | 197.84<br>(119.14)     | 357.94<br>(154.89)     | 179.28<br>(131.32)     |
| <b>zeroDist</b>             | 509.71<br>(773.98)   | 808.62<br>(3284.09)    | 4.00<br>(24.00)        | 647.02<br>(2628.16)    |
| <b>zeroDist_norm</b>        | .12 (.18)            | .027 (.08)             | .00 (.00)              | .05 (.12)              |
| <b>uniqueDistCount</b>      | 3209.19<br>(3989.51) | 21090.20<br>(31516.20) | 49504.70<br>(41137.7)  | 18693.80<br>(30636.10) |
| <b>uniqueDistCount_norm</b> | .39 (.43)            | .45 (.38)              | .52 (.41)              | .44 (.40)              |
| <b>minDist</b>              | 92.91<br>(283.22)    | 1459.97<br>(3279.91)   | 1,738.17<br>(2010.26)  | 1096.84<br>(2732.33)   |
| <b>avgDist/maxDist</b>      | .42 (.14)            | .36 (.12)              | .32 (.07)              | .37 (.13)              |
| <b>minDist/maxDist</b>      | .02 (.05)            | .01 (.02)              | .00 (.01)              | .01 (.03)              |
| <b>maxBinSize</b>           | 741.34<br>(859.95)   | 2962.27<br>(16672.10)  | 14067.20<br>(63827.20) | 3384.22<br>(23716.40)  |
| <b>maxBinSize_norm</b>      | .15 (.17)            | .04 (.09)              | .02 (.08)              | .07 (.13)              |

### 3.3.2 Asymmetry Related

As noted in Chapter 2, symmetry is one of the most studied features of a distance matrix (Gutin and Punnen, 2002). Symmetry in this context refers to the difference between an entry and its diagonally symmetric counterpart (i.e.

$d_{ij}$  and  $d_{ji}$ ). Interestingly, while the property of asymmetry is well defined, a measure of asymmetry for asymmetric matrices is not well defined in the literature. Ausiello et al. (2008) consider a measure of asymmetry termed *maximum asymmetry*. Maximum Asymmetry is defined as the supremum over all city pairs,  $i$  and  $j$ , of the ratio  $d_{ij}$  to  $d_{ji}$ . This measure, while capturing the maximum differential in distances, does not however capture the extent of asymmetry expressed in the matrix as a whole. For example, it may be the case that there is only one pair of cities for which  $d_{ij} \neq d_{ji}$ , yet the maximum asymmetry may appear quite extreme. Meanwhile, another distance matrix may have many entries for which  $d_{ij} \neq d_{ji}$ , but the discrepancy between these distances may only be small, thus maximum asymmetry in this case would appear smaller than in the first case. Furthermore, this metric has a significant downside as it is undefined in grossly asymmetric matrices — that is in matrices where  $d_{ij} > 0$  and  $d_{ji} = 0$ , for all  $i$  and  $j$ ; as such we examine five alternate measures of symmetry, or asymmetry, as the case may be.

**asymmetry** A possibly more encompassing measure of asymmetry is that used by Johnson et al. (2002). Their measure of asymmetry is the ratio of  $|d_{ij} - d_{ji}|$ , averaged over all  $i$  and  $j$ ,  $i \neq j$ , to  $d_{ij} + d_{ji}$ , averaged over all  $i$  and  $j$ ,  $i \neq j$ . This measure encapsulates both the amount and extent by which  $d_{ij}$  may differ from  $d_{ji}$  in a given distance matrix. Note, that if  $d_{ij} = d_{ji}$ , for all  $i$  and  $j$ , the ratio will be zero; if  $d_{ji} = 0$ , for all  $i > j$ , the ratio will be one. Mathematically speaking this metric is defined as:  $\frac{|d_{ij} - d_{ji}|}{d_{ij} + d_{ji}}$ , where  $\overline{|d_{ij} - d_{ji}|} = \frac{1}{n(n-1)} \sum_{i,i \neq j} \sum_j |d_{ij} - d_{ji}|$  and  $\overline{d_{ij} + d_{ji}} = \frac{1}{n(n-1)} \sum_{i,i \neq j} \sum_j d_{ij} + d_{ji}$ .

**maxAsym** A variation of Johnson et al.’s measure of asymmetry, inspired by the maximal nature of Ausiello et al.’s maximum asymmetry, is the ratio of the maximum over all  $i$  and  $j$  of  $|d_{ij} - d_{ji}|$  to the maximum over all  $i$  and  $j$  of  $d_{ij} + d_{ji}$ . Mathematically speaking this is  $\frac{\max_{i,j}\{|d_{ij} - d_{ji}|\}}{\max_{i,j}\{d_{ij} + d_{ji}\}}$ .

**oneRatioCount** Another way to measure asymmetry is to examine the number of times that  $d_{ij} > 0$  while  $d_{ji} = 0$ . This measure may also

be normalized when taken as a fraction of the total number of non-diagonal distance matrix entries. In a sense this hints at Ausiello et al.'s maximum asymmetry, but contextualizes it in terms of the number of occurrences, rather than the scope of occurrences, within the distance matrix. Note, this normalized distance matrix metric will equal one if the matrix is an upper (or lower) triangular matrix in which all entries above (or below) the main diagonal are greater than zero; a matrix with few zeros will yield a normalized metric closer to zero. In this regard, the larger this metric, then the more "asymmetric" the matrix. Mathematically speaking, the normalized version of this metric may be written as:  $\frac{|\{d_{ij}|d_{ij}>0, d_{ji}=0\}|}{n(n-1)}$

**zeroRatioCount** It may, however, be the case that the most useful measure of asymmetry is simply the number of times  $d_{ij} = d_{ji}$  occurs. A normalized version of this metric is the ratio of the number of times  $d_{ij} = d_{ji}$  to the total number of non-diagonal distance matrix entries. Mathematically, the normalized version of this metric is:  $\frac{|\{d_{ij}|d_{ij}=d_{ji}\}|}{n(n-1)}$ . Note, if the distance matrix is symmetric the normalized version of this measure will equal one; if the matrix is fully asymmetric, then the normalized version of this measure will be zero.

**maxRatio** Finally, we turn our attention to another variation of Johnson et al.'s measure of asymmetry by considering the maximum over all  $i$  and  $j$  of the fraction,  $\frac{|d_{ij}-d_{ji}|}{d_{ij}+d_{ji}}$ . This fraction will vary between zero and one; assuming a value of one if there are any entries in the matrix such that  $d_{ij} > 0$  while  $d_{ji} = 0$ .

The mean and standard deviation values of these asymmetry related metrics across the three groups of solvability are presented in Table 3.4.

### 3.3.3 Graph Structure Related

The distance matrix for an asymmetric traveling salesman problem does, after all, describe the length of directional edges connecting nodes in a graph.

Table 3.4: Summary of asymmetry related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses).

| Metric                     | Cluster             |                        |                         | Total                  |
|----------------------------|---------------------|------------------------|-------------------------|------------------------|
|                            | Easy                | Medium                 | Hard                    |                        |
| <b>N</b>                   | 108                 | 235                    | 36                      | 379                    |
| <b>asymmetry</b>           | .27 (.14)           | .15 (.19)              | .14 (.17)               | .18 (.19)              |
| <b>maxAsym</b>             | .48 (.22)           | .24 (.27)              | .21 (.24)               | .30 (.27)              |
| <b>oneRatioCount</b>       | 407.39<br>(627.26)  | 753.12<br>(3510.931)   | .00 (.00)               | 583.07<br>(2793.085)   |
| <b>oneRatioCount_norm</b>  | .09 (.15)           | .02 (.05)              | .00 (.00)               | .04 (.09)              |
| <b>zeroRatioCount</b>      | 578.20<br>(671.425) | 16283.27<br>(34536.01) | 52052.22<br>(135381.49) | 15205.53<br>(51251.29) |
| <b>zeroRatioCount_norm</b> | .12 (.14)           | .33 (.42)              | .21 (.27)               | .26 (.36)              |
| <b>maxRatio</b>            | .93 (.15)           | .65 (.43)              | .66 (.30)               | .73 (.38)              |

We therefore turn our attention to metrics that relate to the number and placement of zero-length directional edges described by a given distance matrix. In its simplest form, this measure can be expressed as the previously introduced, *zeroDist*. We may, however, wish to study the location of these zeros as a means to infer the underlying graph structure. For example, if the distance between a node is zero in both directions then the jobs that these nodes represent are actually *co-located*.

Recalling that underlying any node routing problem there may be an arc routing problem<sup>4</sup>, we may state that two jobs (in the related ARP) are co-located if the origin of one job is the destination of the second, while the destination of the first is also the origin of the second (i.e. jobs three and four in Figure 2.1). Notice that if two jobs are co-located, the distance between the jobs, in both directions, will be zero. Jobs may also be partially co-located if the origin of one is the destination of the second, but the destination of the first is not the origin of the second (i.e. jobs two and three in Figure 2.1). If jobs are partially co-located, then one zero will appear in the distance matrix.

<sup>4</sup>The transformation between an arc routing and node routing problem was shown in Figure 2.1 of Chapter 2. As a side note, considering Figure 2.1 in the context of drayage problems, one could envision nodes  $v_1$  and  $v_2$  as terminals and  $v_3$  and  $v_4$  as customers.

Note, that if the SCP in Figure 2.1 had more than four arcs spanning just these four nodes, then proportionally more zeros would appear in the distance matrix. This may indicate why Laporte (1997) found that his SCPs, with 660 arcs on only 220 nodes, were so easily solved using TSP algorithms. These observations may also be behind Frieze et al.'s conjecture regarding the role of zeros in a distance matrix as related to the likelihood that the solution corresponds to the solution of the related assignment problem.

Thus, this set of graph structure related metrics measure the placement of zeros in the distance matrix in the context of co-location. The simplest way to measure these connections is by constructing an adjacency matrix. An adjacency matrix is similar to a distance matrix, with the exception that all non-zero distances are replaced by the number one. Figure 3.4 shows the adjacency matrix derived from the four node distance matrix presented in Figure 3.2 and associated with the graph in Figure 2.1. Using the adjacency matrix, we develop three graph related metrics.

**Adjacency Matrix**

| $a_{ij}$ | 1 | 2 | 3 | 4 |
|----------|---|---|---|---|
| 1        | 1 | 1 | 1 | 1 |
| 2        | 1 | 1 | 1 | 1 |
| 3        | 1 | 0 | 1 | 0 |
| 4        | 1 | 1 | 0 | 1 |

Figure 3.4: Example of an adjacency matrix as derived from the distance matrix in Figure 2.1 and the graph in Figure 2.1.

**noCol** This metric indicates the number of times that two nodes are connected in both directions. It is derived by calculating the number of times that  $a_{ij} + a_{ji} = 2$  for all entries such that  $i < j$ ; this value can be normalized based on half the total number of entries (for which  $i \neq j$ ) in the matrix (i.e.  $\frac{n(n-1)}{2}$ ).

**partCol** This metric indicates the number of times that two nodes are connected in only one direction. It is derived by calculating the number of times that  $a_{ij} + a_{ji} = 1$  for all entries such that  $i < j$ ; this value can be normalized based on half the total number of entries (for which  $i \neq j$ ) in the matrix (i.e.  $\frac{n(n-1)}{2}$ ). Notice that this metric is equal to half of *oneRatioCount*; and *oneRatioCount\_norm* = *partCol\_norm*.

**fullCol** This metric indicates the number of times that two nodes are not directly connected at all. It is derived by calculating the number of times that  $a_{ij} + a_{ji} = 0$  for all entries such that  $i < j$ ; this value can be normalized based on half the total number of entries (for which  $i \neq j$ ) in the matrix (i.e.  $\frac{n(n-1)}{2}$ ).

Notice that the sum of these three metrics, when normalized, will equal one. The mean and standard deviation values of these graph structure related metrics across the three groups of solvability are presented in Table 3.5.

Table 3.5: Summary of graph structure related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses).

| Metric              | Cluster              |                        |                        | Total                  |
|---------------------|----------------------|------------------------|------------------------|------------------------|
|                     | Easy                 | Medium                 | Hard                   |                        |
| <b>N</b>            | 108                  | 235                    | 36                     | 379                    |
| <b>noCol</b>        | 2976.56<br>(1907.46) | 25946.21<br>(34187.96) | 75543.39<br>(88952.15) | 24111.85<br>(42881.99) |
| <b>noCol_norm</b>   | .83 (.25)            | .96 (.10)              | 1.00 (.00)             | .93 (.17)              |
| <b>partCol</b>      | 203.69<br>(313.63)   | 376.56<br>(1755.465)   | .00 (.00)              | 291.53<br>(1396.54)    |
| <b>partCol_norm</b> | .09 (.15)            | .02 (.05)              | .00 (.00)              | .04 (.09)              |
| <b>fullCol</b>      | 153.01<br>(236.47)   | 216.03<br>(1035.40)    | 2.00 (12.00)           | 177.74<br>(826.76)     |
| <b>fullCol_norm</b> | .07 (.11)            | .02 (.06)              | .00 (.00)              | .03 (.08)              |



### 3.3.4 Arc Routing Problem Related

Continuing with the observations made in the previous subsection, we now specify a set of metrics based on the premise that in some instances the underlying graph structure can be transformed from a node routing problem into an arc routing problem (ARP). Using the following method for reconstructing the ARP graph from the structures found in the distance matrix, we can specify a set of distance matrix metrics that are based on the ARP graph.

The transformation from a distance matrix into an ARP graph description is made by first listing, in pairs, all of the jobs that are fully, partially, and not co-located (see subsection 3.3.3). We then construct a list of origin sets and destination sets. Origin sets are sets containing all of the jobs originating from the same ARP graph node; destination sets are sets containing all of the jobs destined to the same ARP graph node. Each origin set is determined by counting the number of columns with fully equal entries (save for the diagonal entries). Each destination set, on the other hand, is determined by counting the number of rows in distance matrix that contain all of the same entries (with the exception of the diagonal entries). More specifically, when comparing two rows  $i$  and  $k$ , all entries should be equal with the exception of  $d_{ii}$  and  $d_{ki}$  along with  $d_{kk}$  and  $d_{ik}$ .

Using the co-location pairs and the lists of origin and destination sets, we can now reconstruct any underlying ARP graph as follows:

1. Begin with the fully co-located pairs. Note that for these pairs the origin of one job is the destination of the other and vice versa. Thus, we specify one node in the ARP graph such that all of the jobs in the origin set containing the first job and all of the jobs in the destination set containing the second job originate from or are destined to that one node. We then specify a second node in the ARP graph such that all of the jobs in the destination set containing the first job and all of the jobs in the origin set containing the second job are destined to or originate from this second node. This process continues until all fully located job

pairs are associated with nodes in the ARP graph.

2. Next, continue with the partially co-located job pairs. Note, for these pairs the origin of one job is the destination of the other, but not the reverse. Thus, we specify only one node in the ARP graph such that all of the jobs in the origin set containing the first job and all of the jobs in the destination set containing the second job originate from or are destined to that one node. We then check if the destination of the one job and origin of the other job have already been associated with a node; if so, we continue; if not, we associate the destination set of the one job with one node and the origin set of the other job with a second node. Note, in many cases the partial co-locations have already been associated with ARP graph nodes as a result of processing the fully co-located pairs.
3. Finally, iterating over the remaining not co-located jobs we check if each job has already been associated with an ARP graph node. If so, then we continue checking the list; if not, we specify two nodes, one origin node and one destination node for each job.

For example, based on the distance matrix of Figure 2.1, we can list the co-location pairs as follows:

**Fully co-located:**  $\{3, 4\}$ ; note, in the associated adjacency matrix (Figure 3.4),  $a_{43} + a_{34} = 0$

**Partially co-located:**  $\{2, 3\}$ ; note, in the associated adjacency matrix (Figure 3.4),  $a_{23} + a_{32} = 1$

**Not co-located:**  $\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 4\}$ ; note, in the associated adjacency matrix (Figure 3.4),  $a_{ij} + a_{ji} = 2$  for all the not co-located pairs.

We can now specify the origin and destination sets as follows:

**Origin Set 1:**  $\{1\}$ ; note, no other distance matrix column is equivalent to column 1.

**Origin Set 2:**  $\{2, 4\}$ ; note, distance matrix columns 2 and 4 are equivalent.

**Origin Set 3:**  $\{3\}$ ; note, no other distance matrix column is equivalent to column 3.

**Destination Set 1:**  $\{1, 2\}$ ; note, distance matrix rows 1 and 2 are equivalent.

**Destination Set 2:**  $\{3\}$ ; note, no other distance matrix row is equivalent to row 3.

**Destination Set 3:**  $\{4\}$ ; note, no other distance matrix row is equivalent to row 4.

Recognizing that the origin and destination of a fully co-located job pair will be at the same node, we specify nodes one and two of the ARP graph, using the origin and destination sets, as follows:

**ARP Node 1:** [Origin Set containing job 3, Destination set containing job 4] =  $[\{3\}, \{4\}]$

**ARP Node 2:** [Origin Set containing job 4, Destination set containing job 3] =  $[\{2, 4\}, \{3\}]$

Now, considering the partially co-located pair,  $\{2, 3\}$ , we note that a node with the pre-requisite structure, [Origin Set containing job 2, Destination set containing job 3] =  $[\{2, 4\}, \{3\}]$ , was already placed as ARP Node 2. We thus, continue with the not co-located pairs. This leads to the following additional nodes:

**ARP Node 3:** [Origin Set containing job 1, No Destination Set] =  $[\{1\}, \{-}]$

**ARP Node 4:** [No Origin Set, Destination set containing job 2] =  $[\{-}, \{1, 2\}]$

Note, the destination sets for jobs 2, 3, and 4, were already placed. Thus, the process of associating jobs with nodes in the underlying ARP graph is complete. The resulting ARP graph has four nodes: one with out-degree one

(i.e. the origin of job one), one with out-degree two and in-degree one (i.e. the origin of jobs two and four and the destination of job three), one with out-degree one and in-degree one (i.e. the origin of job three and the destination of job four), and one with in-degree two (i.e. the destination of jobs one and two).

We must pause now to highlight one caveat of this transformation process. This process will only work well if there actually is an ARP graph underlying the distance matrix. If there is an underlying ARP graph, the following relationship should hold: if one node is partially co-located with a set of nodes, that set of nodes should appear in the same destination set. If there is no underlying ARP graph, then there is no guarantee that this structural relationship will be obeyed. In fact, within the set of 379 ATSP instances, 10 had no discernible ARP graph structure due to a violation of the structural relationship noted above. These violations were counted and are termed *degeneracies*. Nevertheless, we were able to define the following ARP graph related metrics.

***originalNodes*** This metric captures the number of nodes in the underlying ARP. At a maximum this number will be twice the number of nodes in the original node routing formulation — that is one node serving as an origin and one as a destination. As such, this metric may be normalized by dividing by  $2n$ . As an example, in Figure 2.1 this metric is four; and in normalized format, .5. In general, we have the following expression for this metric:  $originalNodes = \frac{1}{2} * (numDests + numOrigins + source + sink - degeneracies)$ , where *degeneracies* refers to the number of times that a structural relationship is violated (as described in the preceding paragraph; on average over the 379 instances this value is 1.44). *numDests*, *numOrigins*, *source*, and *sink* are defined below.

***numDests*** The number of nodes in the underlying ARP that serve as destinations for the jobs; as measured by counting the number of rows with fully equal entries. This measure can be normalized by the total number of nodes in the ARP (i.e. *originalNodes*. In Figure 2.1 this metric is 3;

and in normalized format, .75.

***numOrigins*** The number of nodes in the underlying ARP that serve as origins for the jobs; as measured by counting the number of columns with fully equal entries. This measure can be normalized by the total number of nodes in the ARP (i.e. *originalNodes*). In Figure 2.1 this metric is 3; and in normalized format, .75.

***maxOrig*** The maximum number of jobs originating from any one origin node in the ARP. This measure can be normalized by the total number of jobs in the ATSP (i.e. *numJobs*). In Figure 2.1 this metric is 2; and in normalized format, .5.

***maxDest*** The maximum number of jobs destined to any one destination node in the ARP. This measure can be normalized by the total number of jobs in the ATSP. In Figure 2.1 this metric is 2; and in normalized format, .5.

***source*** The number of origin nodes that have only jobs originating from them (i.e. no jobs simultaneously destined for those nodes). This measure may be normalized by the total number of origin nodes (i.e. *numOrigins*). In Figure 2.1 this metric is 1; and in normalized format, .33. Note that if there are no co-locations in the underlying ARP the normalized value will be 1.

***sink*** The number of destination nodes that have only jobs destined to them (i.e. no jobs simultaneously originating from those nodes). This measure may be normalized by the total number of destination nodes (i.e. *numDests*). In Figure 2.1 this metric is 1; and in normalized format, .33. Note that if there are no co-locations in the underlying ARP the normalized value will be 1.

The mean and standard deviation values of these ARP related metrics across the three groups of solvability are presented in Table 3.6.

Table 3.6: Summary of ARP related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses).

| Metric                    | Cluster           |                    |                    | Total              |
|---------------------------|-------------------|--------------------|--------------------|--------------------|
|                           | Easy              | Medium             | Hard               |                    |
| <b>N</b>                  | 108               | 235                | 36                 | 379                |
| <b>originalNodes</b>      | 120.41<br>(84.94) | 344.10<br>(258.46) | 641.61<br>(115.41) | 308.62<br>(257.08) |
| <b>originalNodes_norm</b> | .70 (.44)         | .84 (.33)          | .95 (.15)          | .81 (.36)          |
| <b>numDests</b>           | 60.85<br>(41.65)  | 174.81<br>(127.89) | 321.83<br>(60.99)  | 156.30<br>(128.01) |
| <b>numDests_norm</b>      | .58 (.14)         | .53 (.08)          | .50 (.01)          | .54 (.10)          |
| <b>numOrigins</b>         | 60.82<br>(41.68)  | 175.04<br>(128.41) | 321.78<br>(60.96)  | 156.43<br>(128.35) |
| <b>numOrigins_norm</b>    | .58 (.13)         | .53 (.07)          | .50 (.01)          | .54 (.09)          |
| <b>maxOrig</b>            | 11.76<br>(16.33)  | 10.23<br>(24.90)   | 31.64<br>(115.27)  | 12.70<br>(41.57)   |
| <b>maxOrig_norm</b>       | .18 (.25)         | .06 (.12)          | .04 (.13)          | .09 (.18)          |
| <b>maxDest</b>            | 10.50<br>(14.40)  | 8.80<br>(19.96)    | 31.64<br>(115.27)  | 11.45<br>(39.74)   |
| <b>maxDest_norm</b>       | .16 (.22)         | .06 (.11)          | .04 (.13)          | .08 (.16)          |
| <b>source</b>             | 59.57<br>(43.32)  | 170.48<br>(131.38) | 319.78<br>(54.88)  | 153.06<br>(129.72) |
| <b>source_norm</b>        | .79 (.32)         | .90 (.23)          | 1.00 (.23)         | .88 (.25)          |
| <b>sink</b>               | 59.58<br>(43.26)  | 170.18<br>(130.62) | 319.83<br>(54.91)  | 152.88<br>(129.23) |
| <b>sink_norm</b>          | .79 (.32)         | .90 (.22)          | 1.00 (.02)         | .88 (.25)          |

### 3.3.5 Assignment Problem Related

This class of distance matrix metrics contains only one metric and in some ways is not purely derived from the distance matrix. To obtain this metric, we must first solve an assignment problem version of each instance. As noted in Chapter 2, Section 2.2, the assignment problem version of the ATSP represents a relaxation in which all jobs must precede or follow another (not necessarily distinct) job in such a way that the total inter-job costs are minimized. This is not difficult as the Hungarian or Kuhn-Munkres Algorithm provides a polynomial solution approach. The solution to the assignment problem then

serves as a lowest bound solution cost for the ATSP problem. In turn this solution value can be compared to an estimate of the actual solution value (the number of jobs times the average distance) in order to ascertain a (very) rough estimate of an initial gap. If this metric is a small value (e.g.  $< .25$ ) then we can speculate that the gap between the assignment problem and the optimal solution is large. If this is the case, then it is reasonable to assume that a solution algorithm will need multiple iterations to patch the subtours and close the gap; thereby taking longer to find the optimal solution.

**AP Soln/(numJobs\*avgDist)** The value of the assignment problem solution divided by the number of jobs times the value of average distance for the entire distance matrix.

The mean and standard deviation values of these AP related metrics across the three groups of solvability are presented in Table 3.7.

Table 3.7: Summary of AP related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses).

| Metric                    | Cluster   |           |           | Total     |
|---------------------------|-----------|-----------|-----------|-----------|
|                           | Easy      | Medium    | Hard      |           |
| N                         | 108       | 235       | 36        | 379       |
| AP Soln/(numJobs*avgDist) | .30 (.19) | .17 (.17) | .13 (.20) | .20 (.20) |

### 3.3.6 Relationships Between Metrics

In this subsection, we do not introduce any new metrics, but rather devote some space to describing the known direct relationships between the metrics. We undertake this discussion as many of the statistical techniques we will introduce in the following section are dependent on using data that does not have any linear relationships. Thus, knowing which relationships exist in our data allows us to appropriately select the metrics for inclusion in the statistical analysis.

We begin by noting that there is a strong relationship between the metrics measuring the presence, number, and placement of zeros in the distance matrix. Specifically, we can identify the following five relationships:

1. If  $zeroDist \geq 1$  then  $minDist = 0$ .
2. If  $zeroDist \geq 1$  then  $minDist/maxDist = 0$ .
3.  $zeroDist = 2fullCol + partCol$ .
4.  $oneRatioCount = 2partCol$ .
5. If  $oneRatioCount \geq 1$  then  $maxRatio = 1$ .
6.  $noCol + partCol + fullCol = \frac{n(n-1)}{2}$ .

Turning our attention to those metrics that measure the presence of equal distances in symmetric locations within the distance matrix, we see the following relationship:

$$zeroRatioCount = 2fullCol + 2noCol - |\{d_{ij} \mid d_{ij} \neq d_{ji} \wedge d_{ij}, d_{ji} > 0, \forall i, j\}|.$$

Finally, we can find the following relationships between the presence and placement of zeros in the distance matrix and the ARP related metrics:

1. If  $numOrigins = source$  and  $numDests = sink$  then  $noCol = \frac{n(n-1)}{2}$ ,  $zeroDist = 0$ , and  $2partCol = 2fullCol = oneRatioCount = 0$ .
2.  $originalNodes = \frac{1}{2} * (numDests + numOrigins + source + sink - degeneracies)$ , where *degeneracies* refers to the number of times that a structural relationship is violated (as described in the definition of *originalNodes*).

Thus, when performing statistical tests that are dependent on variables without linear relationships, we must be careful not to include all of the graph related metrics together, nor should we include all of the ARP related metrics together, and we should be careful when selecting the asymmetry related metrics in conjunction with the *zeroDist* metric.



## 3.4 Statistical Analysis

We are fortunate to have two good measures of solvability — one that is continuous (sum of exact algorithm runtimes) and one that is discrete (solvability groups). This allows us to use three different statistical modeling techniques to explore the relationship between distance matrix metrics and solvability. The benefit of using three, as opposed to one, modeling techniques stems from the fact that each of the three methods works in a very different way. In this regard we can corroborate the results to strengthen our understanding of which metrics play a significant role in determining solvability. These three methods, linear regression, discriminant analysis, and multinomial logistic regression, are the topic of the following three subsections.

### 3.4.1 Linear Regression

A linear equation, as derived in linear regression, is one in which the value of the left-hand-side term (i.e. the dependent variable) is dependent on a linear combination of parameters on the right-hand-side. Parameters are constant terms modifying a set of independent variables. For example, in our study we designate the sum of exact algorithm running times as the dependent variable, a set of distance matrix metrics as the dependent variables, and then derive via regression the set of parameters whose linear combination best predicts the running time. Such a model looks something like: *sum of exact running times* =  $\beta_0 + \beta_1 * (\text{DistanceMatrixMetric}_1) + \beta_2 * (\text{DistanceMatrixMetric}_2) + \dots + \beta_n * (\text{DistanceMatrixMetric}_n) + \varepsilon$ , where the  $\beta$  terms are the parameters and  $\varepsilon$  represents an error term.

The oldest and most common method of linear regression is the method of ordinary least squares (OLS) as pioneered by Legendre and Gauss<sup>5</sup>. This method is, in effect, premised on an optimization problem. In two dimensions this problem is stated as follows, given a set of points in the plane, find the equation of a line such that the Euclidean distance from each given point to

---

<sup>5</sup>Detailed descriptions of linear regression and OLS can be found in nearly any introductory statistics book, see e.g. (StatSoft Inc., 2007); we present here only the briefest of overviews for the convenience of the reader

the line is minimized. Several tests exist for describing how well the emergent equation fits the data. One of the most popular is a goodness-of-fit test as captured by a variable called R-squared ( $R^2$ ) and the related adjusted R-squared ( $\bar{R}^2$ ). Briefly,  $R^2$  indicates the proportion of the sample variation in the dependent variable that is explained by the model;  $\bar{R}^2$  is a version of  $R^2$  adjusted to account for the number of regressors included in the model.

In order to ascribe a meaningful interpretation to the linear expression, emerging as a solution to this problem, several assumptions must be met. These assumptions, commonly known as the Gauss-Markov assumptions, focus on the error terms embedded in the regression equation. Error terms are those terms that capture the difference between a given observation and the (unobservable) mean of the population from which that observation was drawn. The Gauss-Markov assumptions state that for OLS to provide the best linear unbiased estimator of the linear parameters, the error terms must be uncorrelated drawings (assumption one) from a distribution with mean zero (assumption two) and constant variance (assumption three). Furthermore, the independent variables and the error terms must be independent (assumption four).

While we cannot be fully certain that these assumptions are fulfilled before hand, it is possible to measure the likelihood that these were met once the model has been estimated. This is done by performing a series of tests on the residuals. Residuals capture the difference between an observation and the (observable) mean of the sample to which that observation belongs. As we shall see, these assumptions do not necessarily describe our data.

Despite the obvious breaks in our data (i.e. between those problems solvable with both algorithms within 10,000-seconds, those problems solvable with only concorde within 10,000-seconds, and those solvable by neither algorithm within the timelimit), we begin our exploration of the full data set via a general to specific approach to linear regression. Specifically, using the backward stepwise entry process for regressors, as included in SPSS<sup>6</sup>, we began with

---

<sup>6</sup>SPSS is a commercially available statistics package, specifically we used SPSS for Windows, Rel. 17.0.0. 2008 as provided by SPSS Inc.

a model including 18 distance matrix metrics (non-normalized form). The metrics excluded are those with linear relations (see Subsection 3.3.6) to the metrics included in the model. Specifically, we did *not* include: *zeroDist*, *minDist/maxDist*, *maxRatio*, *partCol* and *originalNodes*. This model, with 18 metrics, was then whittled away until only the most significant regressors were left. This refined model, with an  $\bar{R}^2$  of .702, contained (in addition to a constant term) parameters for: *numJobs*, *uniqueDistCount*, *minDist*, *avgDist/maxDist*, *maxBinSize*, *zeroRatioCount*, *oneRatioCount*, *asymmetry*, *maxAsym*, *noCol*, *fullCol*, *source*, *sink*, and *AP Soln/(numJobs\*avgDist)* — all significant at  $< .05$ .

While this model might appear to be a good fit for our data, we must be sure that the Gauss-Markov assumptions are met in order to make a meaningful interpretation of the equation. We do this by first examining assumption three — constant variance of error terms. One common test of this assumption is the Breusch-Pagan test of heteroskedasticity. This test is based on the  $R^2$  of an auxiliary model in which the squared residual terms of the primary model are the dependent variables, while the regressors stay the same. The resulting test statistic is then  $N * R^2$ , which follows a Chi-squared distribution with degrees of freedom equal to the number of regressors (Verbeek, 2004). In our case the test statistic equals 49.65, which is highly significant for a Chi-squared variable with 14 degrees of freedom. We therefore conclude that we do not meet the Gauss-Markov assumption of constant variance in the error terms.

This result is not so surprising given the vastly different scale of instances included in the sample. For example, there are instances with 17 nodes and instances with 932 nodes. It is not unreasonable to expect that the variables (including error terms) of the large instances will, in general, have larger absolute values. To mitigate this problem we again used the backward stepwise entry process to derive a second linear regression model including only normalized metrics and metrics scaled to take values in the range  $[0, 1]$ . As such, our new model, with an  $\bar{R}^2$  of .442, contained (in addition to a constant term) parameters for: *uniqueDistCount\_norm*, *minDist/maxDist*,

*avgDist/maxDist*, *maxBinSize\_norm*, *oneRatioCount\_norm*, *asymmetry*, *maxAsym*, *fullCol\_norm*, *originalNodes\_norm* and *AP Soln/(numJobs\*avgDist)* — all significant at  $< .05$ . Again using the Breusch-Pagan test of heteroskedasticity, we obtain a test statistic of 31.84, which is significant for a Chi-squared variable with 9 degrees of freedom. We therefore conclude that we do not meet the Gauss-Markov assumption of constant variance in the error terms, even for this normalized version of the model.

In one last attempt to salvage linear regression as a means of analysis, we study instead the log value of all terms (both dependent and independent) in the first model — with the exception of *minDist/maxDist*, *avgDist/maxDist*, *zeroRatioCount*, *oneRatioCount*, *asymmetry*, *maxAsym*, *fullCol*, *source*, *sink*, and *AP Soln/(numJobs\*avgDist)*, as these variables do not always take non-zero positive values. The revised model, with *Log(numJobs)*, *Log(uniqueDistCount)*, *minDist/maxDist*, *avgDist/maxDist*, *Log(maxBinSize)*, *oneRatioCount\_norm*, *zeroRatioCount\_norm*, *asymmetry*, *maxAsym*, *fullCol\_norm*, *source\_norm*, *sink\_norm* and *AP Soln/(numJobs\*avgDist)* as regressors, has an  $\bar{R}^2$  of .696, but a Breusch-Pagan statistic that is even higher than before: 73.147. As such, we must carefully reconsider first, the way in which we are trying to model our dataset and second, the way we can interpret these results despite the failure to meet the Gauss-Markov assumptions.

First, we test the functional form of our model. Specifically, it may be the case that our metrics predict the dependent variable better when in a non-linear relationship. That is, we may do better by estimating a model such as:

*sum of exact running times*

$$\begin{aligned}
 = & \beta_0 + \beta_1 * (Metric_1) + \beta_2 * (Metric_1)^2 \\
 & + \beta_3 * (Metric_2) + \beta_4 * (Metric_2)^2 \\
 & + \dots + \beta_{2n-1} * (Metric_n) + \beta_{2n} * (Metric_n)^2 + \varepsilon.
 \end{aligned}$$

One way to verify this proposed model form is to estimate an auxiliary

model in which the dependent variable is the same, but the regressors are the predicted value of the primary linear model along with powers of the predicted value of the primary linear model. If the coefficients of the powers of the predicted value are non-zero, then there is cause to suspect the need to include the squared (or cubed, etc) term in the original model (see, e.g. Verbeek (2004)).

We performed this test on a model with the following regressors:  $\text{Log}(\text{numJobs})$ ,  $\text{Log}(\text{uniqueDistCount})$ ,  $\text{minDist}/\text{maxDist}$ ,  $\text{avgDist}/\text{maxDist}$ ,  $\text{Log}(\text{maxBinSize})$ ,  $\text{maxRatio}$ ,  $\text{oneRatioCount}$ ,  $\text{zeroRatioCount}$ ,  $\text{asymmetry}$ ,  $\text{maxAsym}$ ,  $\text{fullCol}$ ,  $\text{source}$ ,  $\text{sink}$  and  $\text{AP Soln}/(\text{numJobs}*\text{avgDist})$ . The outcome of the test indicated that the squared terms did have a nonzero impact on estimating the *sum of exact running times*. Thus, using the backwards step-wise entry process on all the terms and their squares, we found that a model including  $\text{Log}(\text{numJobs})$ ,  $\text{Log}(\text{uniqueDistCount})$ ,  $\text{Log}^2(\text{uniqueDistCount})$ ,  $\text{avgDist}/\text{maxDist}$ ,  $\text{maxRatio}$ ,  $\text{maxRatio}^2$ ,  $\text{zeroRatioCount}_{\text{norm}}$ ,  $\text{asymmetry}$ ,  $\text{maxAsym}$ ,  $\text{fullCol}$ , and  $\text{AP Soln}/(\text{numJobs}*\text{avgDist})$  has an  $\bar{R}^2$  of .786. Unfortunately, this model also has a significant Breusch-Pagan statistic of 31.078.

We, therefore, turn our attention to the second point of consideration regarding linear regression — the way we can interpret these results despite the failure to meet the Gauss-Markov assumptions. Basically, because we do not meet the Gauss-Markov assumptions, we cannot be certain that the coefficients of the regressors are statistically significant. While this result appears detrimental at first, we can argue that the rejection of assumption three is most likely due to the breaks in our data. Specifically, our data is really comprised of three distinct subgroups delineated by which algorithms can solve each instance within the time-limit. For example, the group of 23 instances that could not be solved by either algorithm represents a distinct subgroup — a subgroup with a mean running time value of 20,000 (and a standard deviation of zero). Given this observation coupled with the work of Hellevik (2009) (which states that using linear regression on a set of data with clear sub-groups and violations of the Gauss-Markov assumption three may not be completely inappropriate), we can conclude that while the uncertainty

estimate associated with a coefficient may not be accurate, the coefficient itself has been correctly calculated. Thus, studying the the coefficients on a given set of regressors is still instructive. Table 3.8 presents the results of the last model, for the purpose of identifying trends between the metrics and the dependent variable, *sum of exact solver times*.

Table 3.8: OLS Results for Model of Sum of Exact Runtimes.  $\bar{R}^2 = .786$ .

| Metric                                              | Coefficient | Std. Error | t      | Sig. |
|-----------------------------------------------------|-------------|------------|--------|------|
| (Constant)                                          | -28415.006  | 4038.435   | -7.036 | .000 |
| $\text{Log}(\text{numJobs})$                        | 6465.252    | 274.768    | 23.530 | .000 |
| $\text{Log}(\text{uniqueDistCount})$                | 2505.391    | 435.029    | 5.759  | .000 |
| $\text{Log}^2(\text{uniqueDistCount})$              | -192.545    | 28.472     | -6.763 | .000 |
| $\text{avgDist}/\text{maxDist}$                     | -11262.681  | 1576.894   | -7.142 | .000 |
| <i>asymmetry</i>                                    | 11711.343   | 2193.653   | 5.339  | .000 |
| <i>maxAsym</i>                                      | -10298.508  | 1557.951   | -6.610 | .000 |
| <i>zeroRatioCount_norm</i>                          | 4827.603    | 1848.069   | 2.612  | .009 |
| <i>maxRatio</i>                                     | 29043.296   | 4427.984   | 6.559  | .000 |
| <i>maxRatio</i> <sup>2</sup>                        | -26260.226  | 2721.507   | -9.649 | .000 |
| <i>fullCol</i>                                      | -.785       | .210       | -3.743 | .000 |
| $AP \text{ Soln}/(\text{numJobs} * \text{avgDist})$ | -6880.639   | 1155.260   | -5.956 | .000 |

From Table 3.8 we can conclude that the more jobs in an ATSP instance, then the longer the exact solvers will take to run; this relationship is however a log relationship. Thus, the effect of adding another job to an instance becomes less the larger the instance already is. The number of unique distances in a matrix has a similar effect — the more unique distances there are, the higher the predicted runtime, but the less each additional unique distance contributes. This result is consistent with the findings of Zhang and Korf (1996). The closer the average distance is to the maximum distance, then the more the summed runtime decreases. Examining the asymmetry related metrics as a group, we can see that for symmetric matrices, *ceteris paribus*, the summed runtime will increase by 4827.6 seconds. A fully asymmetric matrix (i.e. an upper or lower triangular matrix) will, on the other hand, have a runtime that is, *ceteris paribus*, 4195.91 seconds longer. Finally, the

more zeros that appear in the matrix in symmetric locations and the closer the AP solution is to the average distance times the number of jobs, the less the summed runtime will be, *ceteris paribus*.

Given the violations of the Gauss-Markov assumptions, the real value of this analysis does not necessarily lay in the exact value of the coefficients, but rather the direction, magnitude, and functional form of the relationship between the regressors and the dependent variable. To examine these features deeper, we examine alternate approaches to modeling our data. Recall, our goal, after all, is to have a method by which we can predict if an instance of the ATSP is hard or easy; predicting the exact runtime on a specific set of algorithms is much less important. For this reason, we turn to an examination of categorical models.

### 3.4.2 Discriminant Analysis

Discriminant analysis is a technique used to classify observations in a dataset into distinct populations given a set of characteristics (Morrison, 1990). This technique works by using a set of data for which the appropriate population classifications are known. With this information and the “fingerprint” of known characteristics, linear functions are derived to distinguish the populations. For example, if there are three populations of interest, then one discriminant function serves to discriminate between populations 1 and 2 combined with 3; a second discriminant function, in turn, discriminates between populations 2 and 3. In our case the populations of interest are ATSPs that are hard, medium, or easy to solve. For this we use the groupings constructed via hierarchical clustering in Section 3.2.

The statistical mechanism underlying discriminant analysis is an inverse of the multivariate analysis of variance (MANOVA). As such, all of the statistical assumptions required for the success of a MANOVA are also required in discriminant analysis. Chief among these assumptions is that the population variances and covariances for all independent variables are equal across the dependent variable groups. Despite the general importance of this assumption, it has been noted by several statisticians (e.g. Morrison (1990),

Spicer (2004)) that this requirement may be relaxed when all sample size requirements are met. Specifically, the samples within each category should be reasonably large and equal across categories. According to Spicer (2004), reasonably large means more than 20 instances for each independent variable included in the analysis; and no dependent grouping should have fewer than 20 instances. Therefore, in our case, to ensure that the requirements of discriminant analysis are met (or are at least, not fatal), we should not really include more than 15 independent variables, in our analysis.

Given these restrictions along with the observations made regarding influential regressors while building the linear regression, we entered all of the metrics of the last linear regression plus  $\text{Log}(\text{noCol})$ ,  $\text{Log}(\text{originalNodes})$ , and  $\text{source} + \text{sink}$  into the stepwise Discriminant Analysis model building function in SPSS. This process yielded  $\text{Log}(\text{uniqueDistCount})$ ,  $\text{Log}^2(\text{uniqueDistCount})$ ,  $\text{avgDist}/\text{maxDist}$ ,  $\text{asymmetry}$ ,  $\text{maxAsym}$ ,  $\text{maxRatio}$ ,  $\text{maxRatio}^2$ ,  $\text{Log}(\text{noCol})$ ,  $\text{Log}(\text{originalNodes})$ ,  $\text{source} + \text{sink}$ , and  $\text{AP Soln}/(\text{numJobs} * \text{avgDist})$  as the most influential independent variables for distinguishing among the easy, medium, and hard solvability groups. Table 3.9 presents the results of the univariate ANOVAs carried out for each independent variable. These results indicate that the means differ significantly for all variables in the test.

Table 3.9: Tests of equality of group means.

| Variable                                           | Wilks' Lambda | F       | Sig. |
|----------------------------------------------------|---------------|---------|------|
| $\text{Log}(\text{uniqueDistCount})$               | .556          | 150.223 | .000 |
| $\text{Log}^2(\text{uniqueDistCount})$             | .793          | 49.097  | .000 |
| $\text{avgDist}/\text{maxDist}$                    | .938          | 12.324  | .000 |
| $\text{asymmetry}$                                 | .915          | 17.426  | .000 |
| $\text{maxAsym}$                                   | .837          | 36.732  | .000 |
| $\text{maxRatio}$                                  | .891          | 23.057  | .000 |
| $\text{maxRatio}^2$                                | .888          | 23.682  | .000 |
| $\text{Log}(\text{noCol})$                         | .556          | 150.223 | .000 |
| $\text{Log}(\text{originalNodes})$                 | .719          | 73.569  | .000 |
| $\text{source} + \text{sink}$                      | .682          | 87.687  | .000 |
| $\text{AP Soln}/(\text{numJobs} * \text{avgDist})$ | .884          | 24.587  | .000 |

Running the discriminant analysis with the three ease-of-solvability groups



and the eleven specified variables, we obtain two discriminant functions. The standardized coefficients for these functions are listed in Table 3.10. Interpreting these coefficients is not quite as easy as interpreting the coefficients in a linear regression. In general, we can say that the larger the coefficient, the greater the contribution of the respective variable is to differentiating between groups. Aside from this simple interpretation these coefficients do not tell us the significance of the functions. For this we look at their associated Eigenvalues and Wilks' Lambda significance, which can be viewed in Table 3.11.

The results in Table 3.11 indicate that both of these functions are significant. Furthermore, the first function explains 94.2% of the variance and function two explains over 5%. Unfortunately these results do not tell us which groups the two functions are differentiating between, nor do the results in Tables 3.10 and 3.11 indicate exactly which variables are more strongly correlated with which function. For this we examine a structure matrix of the correlations between the variables and each function. This matrix can be seen in Table 3.12.

From Table 3.12 we can describe the discriminating functions in terms of their most significant variables. Specifically, function one is most strongly correlated to the distance, graph, ARP related, and AP metrics, while function two is most strongly correlated with the asymmetry metrics.

Figure 3.5 is a scatterplot with each instance plotted according to the discriminant scores of each function. The centroids for each group of the ease of solvability classifications are also depicted. From this figure it appears that function one is discriminating the easy group from the hard and medium groups combined. This is apparent from the way that the hard and medium groups appear to be closer together while the easy group is separated farther along the x-axis. Function two on the other hand seems to be separating the hard group from the medium group; apparent from the relative distance between the hard and medium centroids along the y-axis.

While deriving the statistical significance and relative importance of each function is interesting, the predictive capability of the two functions working

Table 3.10: Unstandardized coefficients of the two discriminant functions for the ease of solvability bins.

| Variable                                         | Function |         |
|--------------------------------------------------|----------|---------|
|                                                  | 1        | 2       |
| $\text{Log}(\text{uniqueDistCount})$             | -.603    | -.019   |
| $\text{Log}^2(\text{uniqueDistCount})$           | .046     | .018    |
| $\text{avgDist}/\text{maxDist}$                  | 4.244    | -.667   |
| $\text{asymmetry}$                               | -4.336   | -1.809  |
| $\text{maxAsym}$                                 | 3.819    | 1.076   |
| $\text{maxRatio}$                                | -7.502   | 11.105  |
| $\text{maxRatio}^2$                              | 8.375    | -10.010 |
| $\text{Log}(\text{noCol})$                       | -1.426   | -.409   |
| $\text{Log}(\text{originalNodes})$               | .349     | -.973   |
| $\text{source} + \text{sink}$                    | .001     | .006    |
| $\text{AP Soln}/(\text{numJobs}*\text{avgDist})$ | 2.771    | 1.655   |
| <i>Constant</i>                                  | 9.909    | 4.416   |

Table 3.11: Significance of discriminant functions.

| Function | Eigenvalues |               |                       | Wilks' Lambda |            |    |      |
|----------|-------------|---------------|-----------------------|---------------|------------|----|------|
|          | Eigenvalue  | % of Variance | Canonical Correlation | Wilks' Lambda | Chi-Square | df | Sig. |
| 1        | 3.058       | 93.0          | .868                  | .200          | 596.324    | 22 | .000 |
| 2        | .230        | 7.0           | .432                  | .813          | 76.664     | 10 | .000 |

together is the primary reason for undertaking discriminant analysis. For this, discriminant analysis yields a set of classification functions for each category: easy, medium, and hard. The functions, viewable in Table 3.13, serve to predict category membership based on which of the three functions returns the highest value for each instance. Using these functions, Table 3.14 shows the number of instances correctly classified based on the discriminant functions. These results indicate that the discriminant functions correctly classify 92.9% of the cases. Looking more closely at the classification results, however, reveals that in the hard bin only 61% of the instances were classified correctly using the discriminant functions. Recalling that the hard bin was actually

Table 3.12: Pooled within-groups correlations between discriminating variables and standardized canonical discriminant functions.

| Variable                                         | Function |       |
|--------------------------------------------------|----------|-------|
|                                                  | 1        | 2     |
| $\text{Log}(\text{uniqueDistCount})$             | -.300*   | -.076 |
| $\text{Log}^2(\text{uniqueDistCount})$           | -.292*   | -.001 |
| $\text{avgDist}/\text{maxDist}$                  | .146*    | -.002 |
| $\text{asymmetry}$                               | .162     | .292* |
| $\text{maxAsym}$                                 | .237     | .323* |
| $\text{maxRatio}$                                | .180     | .317* |
| $\text{maxRatio}^2$                              | .197*    | .181  |
| $\text{Log}(\text{noCol})$                       | -.511*   | .040  |
| $\text{Log}(\text{originalNodes})$               | -.358*   | -.001 |
| $\text{source} + \text{sink}$                    | -.382*   | .290  |
| $\text{AP Soln}/(\text{numJobs}*\text{avgDist})$ | .201*    | .174  |

a combination of 23 truly hard instances (i.e. no algorithm could succeed) and 13 extremely difficult medium instances (i.e. only concorde succeeded, but only after a significant amount of time), this result is in a sense more encouraging than detrimental. Furthermore, if we consider the success rate obtained by simply predicting all instances as medium (62%) versus the success rate obtained by using the discriminant functions (92.9%), then these functions perform quite well. Nevertheless, it is worthwhile to examine other classification models to gain further insight into the most influential distance matrix metrics.

### 3.4.3 Multinomial Logistic Regression

Multinomial logistic regression is similar to discriminant analysis in that it yields a model of group membership based on a set of independent variables. The method logistic regression employs is, however, very different from that of discriminant analysis or OLS. For a more detailed presentation of logistic regression, the reader is referred to Pampel (2000) or Spicer

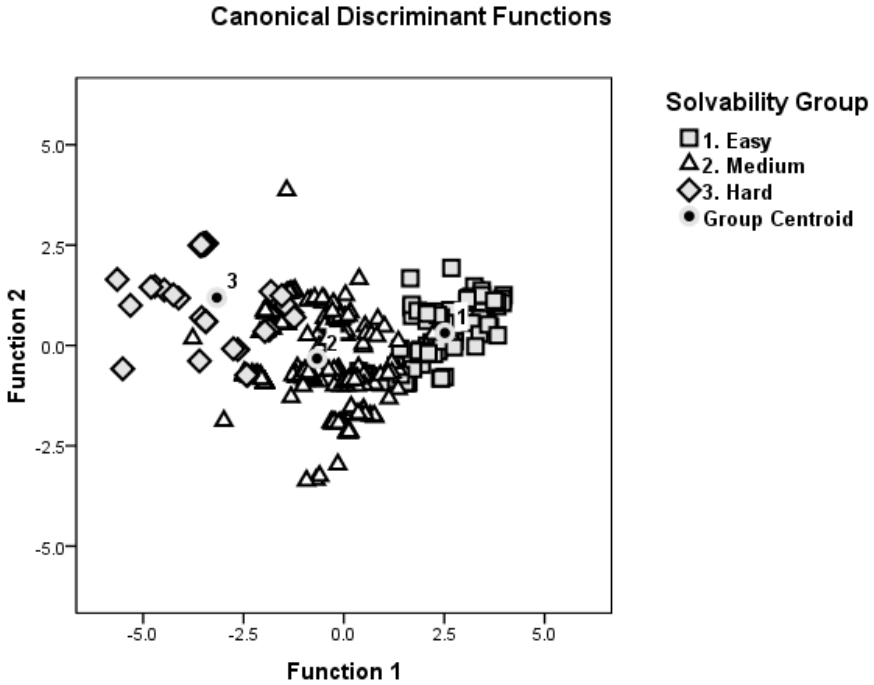


Figure 3.5: Scatterplot of discriminant function values for each instance.

(2004). Specifically, multinomial logistic regression yields a linear function, for each categorical group relative to a reference category. This linear function relies on a set of independent variables and their associated parameters (iteratively) derived (based on the maximum likelihood criterion) to produce the log odds leading to the minimum difference between the predicted probability of group membership and actual (observed) group membership. In our case, the model for the easy solvability group, relative to the medium solvability group, can be mathematically summarized as: *predicted log odds of “easy” group membership* =  $\beta_0 + \log(\beta_1) * (DistanceMatrixMetric_1) + \log(\beta_2) * (DistanceMatrixMetric_2) + \dots + \log(\beta_n) * (DistanceMatrixMetric_n)$ , where  $\beta_0$  represents a constant term. A similar model will be derived for the hard

Table 3.13: Classification functions for the groups easy, medium, and hard.

| Variable                                         | Function |          |          |
|--------------------------------------------------|----------|----------|----------|
|                                                  | Easy     | Medium   | Hard     |
| $\text{Log}(\text{uniqueDistCount})$             | 51.36    | 53.392   | 54.868   |
| $\text{Log}^2(\text{uniqueDistCount})$           | -3.107   | -3.264   | -3.35    |
| $\text{avgDist}/\text{maxDist}$                  | 163.588  | 150.495  | 138.888  |
| $\text{asymmetry}$                               | 145.631  | 160.593  | 168.677  |
| $\text{maxAsym}$                                 | -64.057  | -76.908  | -84.814  |
| $\text{maxRatio}$                                | 120.728  | 137.560  | 173.121  |
| $\text{maxRatio}^2$                              | -120.929 | -141.236 | -177.317 |
| $\text{Log}(\text{noCol})$                       | 33.179   | 37.982   | 40.923   |
| $\text{Log}(\text{originalNodes})$               | -20.926  | -21.418  | -23.763  |
| $\text{source} + \text{sink}$                    | -0.060   | -0.066   | -0.058   |
| $\text{AP Soln}/(\text{numJobs}*\text{avgDist})$ | 24.506   | 14.627   | 10.218   |
| <i>Constant</i>                                  | -213.857 | -244.521 | -269.893 |

Table 3.14: Classification of instances using classification functions.

| Ease of Solvability bins |        | Predicted Group Membership |        |      | Total |
|--------------------------|--------|----------------------------|--------|------|-------|
|                          |        | easy                       | medium | hard |       |
| Count                    | easy   | 100                        | 8      | 0    | 108   |
|                          | medium | 3                          | 230    | 2    | 235   |
|                          | hard   | 0                          | 14     | 22   | 36    |
| %                        | easy   | 92.6                       | 7.4    | 0.0  | 100.0 |
|                          | medium | 1.3                        | 97.9   | 0.8  | 100.0 |
|                          | hard   | 0.0                        | 38.9   | 61.1 | 100.0 |

solvability group, with the medium solvability group serving as the reference group.

Fortunately, the dataset assumptions required for the OLS procedure are not as necessary for multinomial logistic regression (Spicer, 2004). On the other hand, multinomial logistic regression does require a larger dataset than OLS — requiring well over 100 cases total with 50 cases for each independent variable to yield trustworthy results (Spicer, 2004). Additionally, multicollinearity (or the lack of independence between independent variables) can

cause more problems in logistic regression than in OLS regression. In our analysis, this restriction has significant implications as ultimately all independent variables are tied to the distance matrix and hence tend to exhibit significant correlations. To mitigate this problem and alleviate potential problems with dataset size, we include in our analysis far fewer metrics than were included in the discriminant analysis. We also take care to avoid including all of the metrics related by any one of the relationships identified in Subsection 3.3.6. Nevertheless, we ensure that each of the five categories of metrics have at least one representative.

Given these considerations, the metrics selected include  $minDist/maxDist$ ,  $avgDist/maxDist$ ,  $maxRatio$ ,  $maxRatio^2$ ,  $maxAsym$ ,  $noCol$ ,  $sink/numDest$ , and  $AP\ Soln/(numJobs*avgDist)$ . While there is no universally accepted goodness-of-fit criterion, SPSS does provide three pseudo  $R^2$  measures. For this model, SPSS reports that the Cox and Snell measure is .747, the Nagelkerke measure is .904, and the McFadden measure is .784. We can roughly interpret these values to indicate that this model is explaining somewhere between 75% and 90% of the variance in our dataset. These measures are extremely high, but can be corroborated by examining a cross-classification table of actual and predicted categories. Table 3.15 shows that the model derived via multinomial regression has nearly the same predictive capability as the model derived via discriminant analysis. Specifically, 91.7% of the easy instances, 94.9% of the medium instances, and 75% of the hard instances were correctly predicted. This gives an overall success rate of 92.1%.

In addition to the success of this model in predicting solvability group, it is interesting to examine which variables are most influential in differentiating easy from medium instances and hard from medium instances. For this we study the significance of the coefficients as measured by the Wald statistic. Table 3.16, presents the coefficients and affiliated statistics for each estimated model — 1) easy relative to medium and 2) hard relative to medium. Perhaps more instructive than the coefficients in this table, are the odds. In the easy to medium model, if the odds are greater than one, then the probability that the easy group will be selected over the medium group is greater given an

Table 3.15: Cross-classification of instances using multinomial logistic regression.

| Ease of Solvability bins |        | Predicted Group Membership |        |      | Total |
|--------------------------|--------|----------------------------|--------|------|-------|
|                          |        | easy                       | medium | hard |       |
| Count                    | easy   | 99                         | 9      | 0    | 108   |
|                          | medium | 9                          | 223    | 3    | 235   |
|                          | hard   | 0                          | 9      | 27   | 36    |
| %                        | easy   | 91.7                       | 8.3    | 0.0  | 100.0 |
|                          | medium | 3.8                        | 94.9   | 1.3  | 100.0 |
|                          | hard   | 0.0                        | 25.0   | 75.0 | 100.0 |

increase in that variable. Alternatively, if the odds are less than one then the probability that the easy group will be selected over the medium group is less given an increase in that variable.

Thus, from Table 3.16 we can conclude that, *ceteris paribus*, an increase in  $minDist/maxDist$ ,  $noCol$ , or  $sink/numDest$  will reduce the likelihood that an instance is easy as opposed to medium. An increase in  $avgDist/maxDist$ ,  $maxAsym$ , or  $AP\ Soln/(numJobs*avgDist)$  will, on the other hand, increase the likelihood that an instance is easy as opposed to medium. Similarly, *ceteris paribus*, an increase in  $avgDist/maxDist$ ,  $minDist/maxDist$ , or  $AP\ Soln/(numJobs*avgDist)$  will reduce the likelihood that an instance is hard as opposed to medium. Meanwhile, *ceteris paribus*, an increase in  $sink/numDest$  will yield a large increase in the likelihood that an instance is hard as opposed to medium. The impact of  $maxRatio$  in this model is a bit more intricate given the presence of  $maxRatio$  and  $maxRatio^2$ . If this metric, *ceteris paribus*, increases beyond .9 then there is a decrease in the likelihood that an instance is hard as opposed to medium; meanwhile for values of  $maxRatio$  between .7 and 1, the likelihood that an instance is easy as opposed to medium increases. On the other hand, if the metric decreases from .6 downward then the likelihood that an instance is easy as opposed to medium also decreases while the likelihood that an instance is hard as opposed to medium increases.

Table 3.16: Multinomial logistic regression statistics for solvability groups based on seven distance matrix metrics. The medium group is the reference group.

| Variable                         | Coeff.  | Wald  | Sig. | odds      |
|----------------------------------|---------|-------|------|-----------|
| Easy relative to Medium          |         |       |      |           |
| <i>intercept</i>                 | -10.55  | 16.76 | .000 | —         |
| <i>minDist/maxDist</i>           | -3.53   | .038  | .845 | .029      |
| <i>avgDist/maxDist</i>           | 29.78   | 19.42 | .000 | 8.54E12   |
| <i>maxRatio</i>                  | -15.72  | 4.62  | .032 | 1.48E-7   |
| <i>maxRatio</i> <sup>2</sup>     | 21.52   | 8.56  | .003 | 2.21E9    |
| <i>maxAsym</i>                   | 4.26    | 6.82  | .009 | 71.021    |
| <i>noCol</i>                     | -.001   | 20.90 | .000 | .99       |
| <i>sink/numDest</i>              | -3.27   | 3.09  | .079 | .038      |
| <i>AP Soln/(numJobs*avgDist)</i> | 7.41    | 10.54 | .001 | 1656.52   |
| Hard relative to Medium          |         |       |      |           |
| <i>intercept</i>                 | -411.74 | 8.38  | .004 | —         |
| <i>minDist/maxDist</i>           | -604.77 | 8.69  | .003 | 2.25E-263 |
| <i>avgDist/maxDist</i>           | -15.33  | 4.26  | .039 | 2.21E-7   |
| <i>maxRatio</i>                  | 70.23   | 10.95 | .001 | 3.17E30   |
| <i>maxRatio</i> <sup>2</sup>     | -70.87  | 10.32 | .001 | 1.67E-31  |
| <i>maxAsym</i>                   | -2.92   | 1.46  | .227 | .054      |
| <i>noCol</i>                     | .0003   | 9.378 | .002 | 1.00      |
| <i>sink/numDest</i>              | 399.11  | 8.354 | .004 | 2.15E173  |
| <i>AP Soln/(numJobs*avgDist)</i> | -8.20   | 1.84  | .175 | .0003     |

### 3.5 Verification

The results of this statistical analysis are exciting for the promise they hold in permitting one to determine, *a priori*, if an ATSP will be easy, medium, or hard to solve. We must, however, exercise extreme caution in presenting these results as such given that they were calibrated on a single, specific sample of 379 instances. It may be the case that these models do a good job of capturing noise in this specific sample, without really exposing any significant trends in the larger population. In order, to verify the predictive capabilities of these



models it is prudent to test their behavior on a second sample.

Unfortunately, the number of ATSP instances available for research is severely lacking (Gutin and Punnen, 2002). We therefore rely on the random instance generators of Cirasella et al. (2001) plus one random instance generator of our own to test our models. We must be careful when using these generators to ensure that we do not generate instances that are too similar to the original dataset. Our goal, after all, is to test the robustness of the categorical models. As such we have setup a careful regimen of 172 instances as documented in Appendix A.

As noted previously, the instance generators of Cirasella et al. (2001) and Johnson et al. (2002) are well documented. We therefore refrain from providing further detail here. New to the battery of generators, however, is one we created, termed “crane2”. This generator, using the random number generator/random location generator used in the crane generator of Cirasella et al. (2001) and Johnson et al. (2002), selects a set of  $n$  points from an  $x \times x$  square. These points then serve as the origins and destinations of  $n$  jobs. Specifically, the generator selects the first  $k$  points as origins, where  $k$  is specified by the user, and the last (moving from the back)  $m$  points as destinations, where  $m$  is specified by the user. The generator then matches origins to destinations in a round-robin fashion until all  $n$  jobs have been designated. The distance between jobs,  $d_{ij}$  is then calculated as the distance from the destination of job  $i$  to the origin of job  $j$ . In this way the generated instances should look similar to drayage problems in which the jobs originate from and are destined to a limited set of terminals or customer locations. The complete code for this generator is in Appendix B.

After generating this second sample of ATSPs, we derived the full set of distance matrix metrics for each instance. Based on these metrics we then used the linear regression model to predict the summed exact solver run-time and both the discriminant analysis model and the multinomial logistic regression model to predict the solvability group. The linear regression predicted that the 172 instances would require a mean of 10248.67 seconds with a standard deviation of 4850.14. The discriminant analysis derived classi-

fication functions predicted that 20 instances were easy, 141 instances were medium, and 11 instances were hard. The multinomial logistic regression model predicted that 34 instances were easy, 127 instances were medium, and 11 instances were hard. So, how accurate were these predictions?

After running the 172 ATSPs through both `concorde` and `tsp.solve v. 1.3.6`, the mean summed exact solver runtime was 9107.73 seconds with a standard deviation of 5468.10. Using a paired, two-tailed, t-test to compare the predicted runtimes to the actual runtimes of the 172 instances, we obtain a mean difference of -1140.94 seconds with a standard deviation of 3132.15, giving us a t-statistic of -4.78, with a significance of .000. We therefore reject the hypothesis that the predicted summed runtimes and the actual summed runtimes are equal. In fact, we can see that our linear regression consistently over-estimates the runtime. This is most likely due to the inclusion of regressors that are not actually significant; as discussed in Subsection 3.4.1.

Using the summed runtimes and the branch-and-bound nodes from `concorde` we also classified the 172 verification instances into solvability groups using the same hierarchical clustering procedure recorded in Section 3.2. This yielded 36 easy, 119 medium, and 17 hard instances. Thus, in comparison, as noted in tables 3.17 and 3.18, the discriminant analysis model predicted solvability with an 88.9% success rate and the multinomial logistic regression model with an 93.6% success rate. The results of the multinomial logistic regression are phenomenal given that the model predicted the verification set with higher success than the training data. Furthermore, given that a myopic predictive model which places all instances in the medium bin would yield a success rate of 69%, we can conclude that both models improve our predictive capabilities by over 20%. This improvement is large enough to allow us to conclude that the distance matrix metrics we selected do hold significant predictive capabilities.

Table 3.17: Cross-classification of verification instances using discriminant classification functions.

|       |        | Predicted Group Membership |        |      | Total |
|-------|--------|----------------------------|--------|------|-------|
|       |        | easy                       | medium | hard |       |
| Count | easy   | 22                         | 14     | 0    | 36    |
|       | medium | 0                          | 119    | 0    | 119   |
|       | hard   | 0                          | 5      | 12   | 17    |
| %     | easy   | 61.1                       | 38.9   | 0.0  | 100.0 |
|       | medium | 0.0                        | 100.0  | 0.0  | 100.0 |
|       | hard   | 0.0                        | 29.4   | 70.6 | 100.0 |

Table 3.18: Cross-classification of verification instances using multinomial logistic regression functions.

|       |        | Predicted Group Membership |        |      | Total |
|-------|--------|----------------------------|--------|------|-------|
|       |        | easy                       | medium | hard |       |
| Count | easy   | 30                         | 6      | 0    | 36    |
|       | medium | 0                          | 119    | 0    | 119   |
|       | hard   | 0                          | 6      | 5    | 12    |
| %     | easy   | 83.3                       | 16.7   | 0.0  | 100.0 |
|       | medium | 0.0                        | 100.0  | 0.0  | 100.0 |
|       | hard   | 0.0                        | 29.4   | 70.6 | 100.0 |

## 3.6 Discussion

We close this chapter by opening the discussion with an answer to the question asked in the title — are SCPs easy?

Table 3.19 presents the hierarchical analysis derived group memberships (as first introduced in Section 3.2) for each problem type across both the primary and verification datasets. This table shows that SCPs are not particularly easier (or harder) than other types of ATSPs. In fact, as one might expect, SCPs are comparable to the generated and real-world routing problem types in terms of solvability. This, however, is not the end of the story.

Table 3.19: Count of instances in each group based on problem type.

| <b>Problem</b>                                                         | <b>Easy</b> | <b>Medium</b> | <b>Hard</b> | <b>Total</b> |
|------------------------------------------------------------------------|-------------|---------------|-------------|--------------|
| Real-world SCP                                                         | 33          | 37            | 0           | 70           |
| Generated SCP                                                          | 7           | 37            | 7           | 51           |
| Real-world Scheduling Problems                                         | 2           | 3             | 0           | 5            |
| Generated Scheduling Problems                                          | 20          | 17            | 0           | 37           |
| Real-world Routing Problems                                            | 11          | 7             | 0           | 18           |
| Generated Routing Problems                                             | 13          | 44            | 17          | 74           |
| Real-world Robotic Motion Problems                                     | 2           | 2             | 1           | 5            |
| Generated Robotic Motion Problems                                      | 0           | 50            | 21          | 71           |
| Real-world Data Compression Problems                                   | 0           | 6             | 3           | 9            |
| Real-world Code Optimization Problems                                  | 0           | 2             | 0           | 2            |
| Generated Approximate Shortest Common Superstring Problems             | 15          | 12            | 0           | 27           |
| Randomly generated asymmetric matrices                                 | 20          | 17            | 0           | 37           |
| Randomly generated asymmetric matrices obeying the triangle inequality | 20          | 17            | 0           | 37           |
| Randomly generated symmetric matrices                                  | 0           | 34            | 0           | 34           |
| Randomly generated symmetric matrices obeying the triangle inequality  | 0           | 33            | 1           | 34           |
| Randomly generated symmetric matrices using rectilinear distances      | 0           | 34            | 3           | 37           |
| Symmetric matrices perturbed to be asymmetric                          | 0           | 2             | 0           | 2            |
| Unknown Origin                                                         | 1           | 0             | 0           | 1            |
| <b>Total</b>                                                           | <b>144</b>  | <b>354</b>    | <b>53</b>   | <b>551</b>   |

Table 3.20: Mean and (std. deviation) of the sum of exact algorithm runtimes for instances based on problem type.

| Problem Type                                                           | Easy               | Medium               | Hard                  | Total                 |
|------------------------------------------------------------------------|--------------------|----------------------|-----------------------|-----------------------|
| Real-world SCP                                                         | 1.63 (1.6)         | 10017.17<br>(41.05)  | —                     | 5295.55<br>(5035.77)  |
| Generated SCP                                                          | 55.44<br>(76.23)   | 10230.85<br>(366.04) | 13146.47<br>(1564.56) | 9234.41<br>(3881.2)   |
| Real-world Scheduling Problems                                         | 9.03 (4.28)        | 10073.45<br>(118.46) | —                     | 6047.68<br>(5513.15)  |
| Generated Scheduling Problems                                          | 20.59<br>(23.54)   | 10069.89<br>(63.12)  | —                     | 4637.84<br>(5077.39)  |
| Real-world Routing Problems                                            | 218.07<br>(487.56) | 10070.53<br>(164.92) | —                     | 4049.58<br>(4957.4)   |
| Generated Routing Problems                                             | 99.88<br>(86.27)   | 10053.71<br>(85.62)  | 20000 (0)             | 10590.02<br>(6355.81) |
| Real-world Robotic Motion Problems                                     | 2.91 (1.47)        | 10195.24<br>(119.8)  | 20000 (0)             | 8079.26<br>(8389.4)   |
| Generated Robotic Motion Problems                                      | —                  | 10542.8<br>(1005.93) | 19250.49<br>(2086.14) | 13118.31<br>(4239.21) |
| Real-world Data Compression Problems                                   | —                  | 10189.79<br>(262.62) | 18238.9<br>(3050.32)  | 12872.82<br>(4308.86) |
| Real-world Code Optimization Problems                                  | —                  | 10038.64<br>(48.42)  | —                     | 10038.64<br>(48.42)   |
| Generated Approximate Shortest Common Superstring Problems             | 2.58 (1.99)        | 10032.94<br>(67.91)  | —                     | 4460.52<br>(5079.26)  |
| Randomly generated asymmetric matrices                                 | 5.79 (4.11)        | 10031.46<br>(49.93)  | —                     | 4612.18<br>(5065.36)  |
| Randomly generated asymmetric matrices obeying the triangle inequality | 2.94 (4.34)        | 10011.41<br>(16.4)   | —                     | 4601.43<br>(5056.57)  |
| Randomly generated symmetric matrices                                  | —                  | 10040.86<br>(72.28)  | —                     | 10040.86<br>(72.28)   |
| Randomly generated symmetric matrices obeying the triangle inequality  | —                  | 10340.08<br>(706.88) | 19204.13<br>(0)       | 10600.79<br>(1671.96) |
| Randomly generated symmetric matrices using rectilinear distances      | —                  | 10471.99<br>(864.36) | 14165.66<br>(602.93)  | 10771.48<br>(1322.8)  |
| Symmetric matrices perturbed to be asymmetric                          | —                  | 10003.93<br>(3.55)   | —                     | 10003.93<br>(3.55)    |
| Unknown Origin                                                         | 63.85 (0)          | —                    | —                     | 63.85 (0)             |
| Total                                                                  | 33.69<br>(145.74)  | 10203.33<br>(559.3)  | 18352.89<br>(2875.38) | 8329.46<br>(5563.08)  |

The counts in Table 3.19 belie a range of exact algorithm runtimes. A closer inspection of mean runtimes per each group and problem type, shown in Table 3.20, reveals more. Specifically, we see that the 33 “easy” real-world SCP instances were solved in an average of 1.63-seconds. This is nearly half the time of the next most “easily” solve set of problems (generated approximate shortest common superstring problems). Furthermore, the 37 “medium” instances were solved in an average of 10,017.17-seconds, which is larger than only two other problem types (symmetric matrices perturbed to be asymmetric and randomly generated asymmetric matrices obeying the triangle inequality).

Table 3.21: Count of instances in each group based on problem type.

| <b>Problem</b>  | <b>Easy</b> | <b>Medium</b> | <b>Hard</b> | <b>Total</b> |
|-----------------|-------------|---------------|-------------|--------------|
| ATSP            | 104         | 280           | 46          | 430          |
| SCP             | 0           | 34            | 7           | 41           |
| Drayage Problem | 40          | 40            | 0           | 80           |
| Total           | 144         | 354           | 53          | 551          |

Digging even deeper into these results, we find that all 33 “easy” real-world SCPs had under 100 jobs, while the 37 “medium” instances all had more than 131 jobs. As `tsp_solve` is designed to abort when a large problem is entered, it is not surprising that this trend would appear in the data. What is, however, surprising is the fact that this trend is not consistent across all problem types — including the generated SCPs. For example, the seven generated SCPs classified as “easy” all had 100 jobs, while three of the instances classified as “medium” had only 66 jobs. Why is this? What is different about these three instances? The answer it seems can be found by refining our SCP problem type partitioning.

We proceed by grouping all of the real-world and generated ATSPs together while simultaneously partitioning the SCP problem types into SCPs (encompassing the `crane*` and `rbg*` instances) and Drayage Problems (encompassing the `PK*` and `crane2*` instances). Table 3.21 shows the count of instances falling into each solvability group across these three new problem type partitions; Table 3.22 shows the mean and (standard deviation) of the sum of exact algorithm runtimes across the solvability groups by problem type partition. From these two tables we can see that the drayage problems are consistently “easy” or “medium” while the SCPs are consistently “medium” or “hard”. Furthermore, the mean algorithm running times of the drayage problems are much lower than those of either the SCPs or ATSPs. Looking behind these data we find that the 40 dray problems designated as “easy” had 66 to 100 jobs; alternately, the 20 SCP instances with 66 to 100 jobs were classified as “medium”.

These results are stunning for they show that among ATSPs, drayage

Table 3.22: Mean and standard deviation of exact algorithm runtimes based on problem type partitions.

| Problem         | Easy              | Medium               | Hard                  | Total                 |
|-----------------|-------------------|----------------------|-----------------------|-----------------------|
| ATSP            | 42.40<br>(169.44) | 10224.29<br>(610.86) | 19145.17<br>(2090.27) | 8716.03<br>(5665.10)  |
| SCP             | —                 | 10208.18<br>(354.71) | 13146.48<br>(1564.56) | 10709.84<br>(1312.97) |
| Drayage Problem | 11.04 (36.40)     | 10052.46<br>(170.31) | —                     | 5031.75<br>(5053.87)  |
| Total           | 33.69<br>(145.74) | 10203.33<br>(559.30) | 18352.89<br>(2875.38) | 8329.46<br>(5563.08)  |

problems (of the type studied in this thesis) form a subset of easily solved problem instances. We now exploit the discriminant analysis and multinomial logistic regression models of Section 3.4 to examine the reasons behind this result. Table 3.23 presents the means and standard deviations for the distance matrix metrics of interest as per the predictive models.

The most striking feature of this table is how the drayage problems have a profile of metrics that is truly distinct from the other SCPs. Most notably, the drayage problems have a mean  $minDist/maxDist$  of zero with a standard deviation of zero. This implies (and confirms) that all drayage problems contain at least one zero element; that is, at least one partially co-located pick-up and drop-off point. Furthermore, the drayage problems have a  $maxRatio$  of one with a standard deviation of zero. This implies that in all drayage problems there is at least one pair of entries, such that  $d_{ij} > 0$  while  $d_{ji} = 0$ . The drayage problems also have significantly fewer jobs that are not co-located as measured by  $noCol$  and  $Log(noCol)$ . This implies that these drayage problems contain multiple distance matrix entries such that  $d_{ij} > 0$  while  $d_{ji} = 0$  or  $d_{ij} = d_{ji} = 0$ . Such a matrix structure would imply a high level of asymmetry, which is indeed verified in the *asymmetry* measure that is, on average, higher for drayage problems than other SCPs. These metric trends also seem, as per the multinomial logistic regression model, to indicate an increase in the likelihood that an instance is easy as opposed to medium. Furthermore, these

Table 3.23: Mean and standard deviation of distance matrix metrics by problem type.

| Metric                   | ATSP                   | SCP                   | Dray                  | Total                  |
|--------------------------|------------------------|-----------------------|-----------------------|------------------------|
| $N$                      | 430                    | 41                    | 80                    | 551                    |
| $minDist/maxDist$        | 0.01 (.03)             | .003 (.003)           | 0 (0)                 | .009 (.003)            |
| $avgDist/maxDist$        | 0.37 (0.12)            | 0.41 (0.06)           | 0.35 (0.11)           | 0.37 (0.12)            |
| $Log(uniqueDistCount)$   | 7.79 (2.55)            | 8.36 (2.73)           | 5.06 (1.70)           | 7.44 (2.64)            |
| $Log^2(uniqueDistCount)$ | 67.19<br>(36.98)       | 77.23<br>(42.75)      | 28.47<br>(19.75)      | 62.31<br>(38.18)       |
| $asymmetry$              | 0.19 (0.2)             | 0.06 (0.06)           | 0.21 (0.12)           | 0.18 (0.19)            |
| $maxAsym$                | 0.31 (0.3)             | 0.1 (0.14)            | 0.36 (0.2)            | 0.3 (0.28)             |
| $maxRatio$               | 0.65 (0.42)            | 0.69 (0.4)            | 1 (0)                 | 0.7 (0.4)              |
| $maxRatio^2$             | 0.59 (0.43)            | 0.63 (0.38)           | 1 (0)                 | 0.65 (0.42)            |
| $noCol$                  | 27069.34<br>(40461.77) | 28283.49<br>(24408.1) | 7613.91<br>(11884.57) | 24334.94<br>(37259.53) |
| $Log(noCol)$             | 9.43 (1.34)            | 9.63 (1.26)           | 8.17 (1.21)           | 9.26 (1.39)            |
| $Log(originalNodes)$     | 5.67 (0.73)            | 5.71 (0.62)           | 3.19 (1.14)           | 5.31 (1.18)            |
| $source + sink$          | 322.16<br>(252.55)     | 354.88<br>(228.34)    | 34.03 (60)            | 282.76<br>(254.28)     |
| $sink/numDest$           | .85 (.34)              | .92 (.23)             | .43 (.19)             | .80 (.35)              |
| $AP/(numjob*avgdist)$    | 0.2 (0.2)              | 0.12 (0.06)           | 0.2 (0.1)             | 0.19 (0.18)            |



trends, harken back to the observations of Frieze et al. (1995) and Harary (1962b) who note that the number and placement of zeros in a distance matrix appear to be critical indicators of solvability. We also note that the drayage problems, on average, have fewer unique distances in their distance matrices as compared to the other SCPs. This is another feature previously expressed by Zhang and Korf (1996) as having implications for solvability.

In addition to those metrics previously noted in the literature, we identify two metrics that appear to hold implications for solvability — the number of nodes in any underlying ARP (*originalNodes*) and the number of source only and sink only nodes in that same underlying ARP (*source + sink*). On average, the drayage problems have lower values for the  $\text{Log}(\textit{originalNodes})$  and *source + sink* metrics than other SCPs (or even other ATSPs). This indicates that the ARPs underlying drayage problems tend to have comparatively more vertices that serve as hubs; that is vertices that serve as an origin and destination for multiple jobs. The reason these features influence solvability may stem from the two SCP structures originally noted by Frederickson et al. (1978) which led to the development of the subroutines LARGEARC and LARGEEDGE in the  $\frac{9}{5}$ -approximation algorithm for the SCP.

Our method of exploration, the use of statistical analysis to identify key ATSP instance features and relate them to solvability, highlights the importance of using statistics as a tool in operations research. While others (e.g. Zhang and Korf (1996)) have examined complexity transitions focused on altering one variable, we have studied a large set of instances that vary across multiple variables. Nevertheless, we were able to discern metrics that can significantly predict whether an instance will take a long time to solve or whether it will take a shorter time to solve. Thus, the power of this type of analysis in studying algorithms should not be overlooked. Furthermore, the set of instances we used relate to a variety of problem types that occur in the real-world. As such, we have also been able to name a subclass of “easy” problems — drayage problems.

## Chapter 4

# The Value of Advanced Location Information

Time, time, time, see what's become of me, while I look around for my possibilities.

Simon & Garfunkel, *A Hazy Shade of Winter*

We cannot know the future. This is a fact and frustration of life in this reality. Despite this disability, we do possess a robust skill set in planning for and adjusting to the ever unfolding by-and-by. This chapter, formalizes, and in a sense quantifies, our planning and adjusting skills in the context of a routing problem with release dates. Given that the ability to reason about the future and make adaptable plans is a distinctly human capability, it is only appropriate that this chapter, amongst all the other chapters, relies most heavily on a second distinctly human capability — mathematical thinking.

In this chapter<sup>1</sup> we derive the worst-case ratio of an algorithm for the online Traveling Salesman Problem (TSP) with two disclosure dates. This problem, a variant of the online TSP with release dates, is characterized by the disclosure of a job's location at one point in time followed by the disclosure of that job's release date at a later point in time. We present an algorithm

---

<sup>1</sup>This chapter is based on Srour and Zuidwijk (2008).

for this online problem restricted to the positive real number line. We then derive the worst-case ratio of our algorithm and show that it is best-possible in two contexts — the first, in which the amount of time between the disclosure events and release time are fixed and equal for all jobs; and a second in which the time between disclosure events varies for each job. We conclude that the value of advanced information can be attributed to the location information alone — yielding an optimal solution in favorable instances.

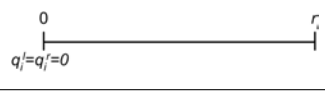
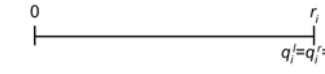
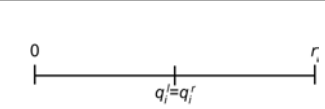
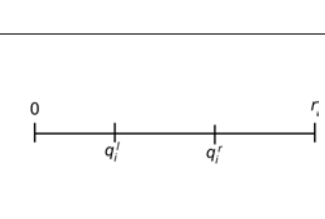
## 4.1 Literature Review

The offline TSP with release dates on  $\mathbb{R}_0^+$  is not new. Psaraftis et al. (1990) introduced this problem as one of routing and scheduling along a shoreline. They examine both path and tour versions of the problem and demonstrate that in the tour version on such a restricted metric space these problems are trivially solved in polynomial time.

Blom et al. (2001) provide an algorithm with a worst-case ratio of  $\frac{3}{2}$  for the online variant of this problem; they term this version of the problem the *online TSP* (OLTSP). They prove that their algorithm, Move-Right-If-Necessary (MRIN), is best-possible for the OLTSP with release dates. MRIN is a zealous algorithm that sends the salesman immediately to any job on the right and back to the origin (left) if there are no other jobs to the right. Jaillet and Wagner (2006) and Wagner (2006), however, note that the result of Blom et al. (2001) is dependent on the assumption that the disclosure time of a job's location and release time occurs at the moment of release. In this way, Jaillet and Wagner (2006) formulate a TSP scenario with advanced information and demonstrate the benefit of that advanced information.

Specifically, Jaillet and Wagner (2006) introduce a disclosure time, at which both the location and the release time are announced. If this disclosure time is equal to the release time then we are in the case where MRIN yields a solution with worst-case ratio of  $\frac{3}{2}$ . If, however, the disclosure time occurs a fixed amount of time in advance of the release date then the worst-case ratio for an arbitrary homing (or tour) online algorithm improves to at least

Table 4.1: Overview of work to date and our contribution.

| Problem                              | Depiction of Info. Arrival (Time)                                                 | Main Result on $\mathbb{R}_0^+$                                                                                                                                                                                                                           | Reference                  |
|--------------------------------------|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| Offline TSP with Release Dates       |  | Optimal algorithm in $O(n)$ time.                                                                                                                                                                                                                         | (Psaraftis et al., 1990)   |
| Online TSP with Release Dates        |  | Best-possible algorithm with worst-case ratio of $\frac{3}{2}$ .                                                                                                                                                                                          | (Blom et al., 2001)        |
| Online TSP with Disclosure Dates     |  | Disclosure dates give advantage over release dates; worst-case ratio for both fixed and variable advanced notice is dependent on time between disclosure and release, but bounded by $\frac{3}{2}$ .                                                      | (Jaillet and Wagner, 2006) |
| Online TSP with Two Disclosure Dates |  | Advanced location information gives an advantage over simultaneous disclosure dates; worst-case ratio for both fixed and variable advanced notice is dependent on time between both disclosure dates and the release time, but bounded by $\frac{3}{2}$ . | This chapter               |

$(\frac{3}{2} - \frac{a}{2l_{\max}}) \in [1, \frac{3}{2}]$  where  $a$  is the fixed amount of advanced notice time and  $l_{\max}$  is the location of the job farthest from the origin on  $\mathbb{R}_0^+$ . Note, we use the expression homing in a manner similar to Ausiello et al. (2001) in order to indicate that the algorithm must return to the depot or origin at the point in time when all known jobs have been served.

### 4.1.1 Our Contribution

We position our work as depicted in Table 4.1. In this table the name of the problem examined appears in the far left column. The second column provides a graphical depiction of information arrival over time that characterizes the associated problem; note,  $q_i^l$  represents the time the location of a job  $i \in N = \{1, \dots, n\}$  is disclosed,  $q_i^r$  represents the time the release time of job  $i$  is disclosed, and  $r_i$  represents the release time of the job. The third and fourth columns indicate the main result and reference for the associated problem, respectively. This table emphasizes the focus of our work on the impact of early location disclosure in the context of the TSP on  $\mathbb{R}_0^+$ .

In our case, we have two disclosure dates — the disclosure of the job

location and the disclosure of the release time. This split information arrival serves to give our online algorithm a greater advantage over other online algorithms in comparison to the optimal offline strategy. This case is also more realistic as there are many real-world instances in which the job locations are known early in execution, but the release times come later. We begin by introducing an online algorithm for  $\mathbb{R}_0^+$  designed to exploit both pieces of information as they are made available. Our online algorithm is a homing algorithm as the salesman must return to the origin following the completion of all known jobs.

We prove that our online algorithm is best-possible with a worst-case ratio of  $\max \left\{ 1, \frac{3}{2} - \frac{(a+b)}{2l_{\max}} \right\}$ , where  $a$  and  $b$  represent fixed amounts of time between the disclosure events and release of the job.

We also address the case of variable amounts of advanced notice (i.e. the case where  $a$  and  $b$  vary by job taking any positive real value). In this case we obtain a ratio of

$$1 \leq 1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \\ \leq \frac{3}{2}$$

We show that this ratio is the best possible in this setting.

The remainder of this chapter is organized as follows: in Section 4.2 we state the problem of interest in mathematical terms and define the necessary notation; we also present in greater detail the optimal offline algorithm and online algorithms for the TSP with release dates and TSP with disclosure dates. In Section 4.3 we present our algorithm, Move-Right-Early-Left-Late (MRELL), for the OLTSP on  $\mathbb{R}_0^+$  with two disclosure dates. In Section 4.4 we derive the worst-case ratio for the case in which the amount of time between disclosure events is fixed; we also demonstrate that MRELL is best possible in this case. In Section 4.5 we study the case in which the amount of time between disclosure events varies across jobs; we demonstrate that MRELL is best possible for that case as well. Finally, we conclude with a discussion of

these results and statement of future research in Section 4.6.

## 4.2 Assumptions, Notation, and Preliminaries

To facilitate an understanding of the exact nature of the problem under consideration, we begin by stating some assumptions and describing the notation we will use throughout this chapter.

1. All job locations are along the positive real number line,  $\mathbb{R}^+$ .
2. The origin is at the point, 0, on  $\mathbb{R}^+$  which is where the salesman begins at the start of each problem at time 0 and must return to after visiting all jobs.
3. The location of a job,  $i$ , is only revealed to the salesman at a time in advance of its release time (and the disclosure of that time); this location disclosure time is denoted  $q_i^l$ .
4. A job's release time is only revealed to the salesman at a time after the disclosure of its location, but before the time of release; this release disclosure time is denoted  $q_i^r$ .
5. The salesman always travels at unit speed along  $\mathbb{R}^+$ ; otherwise he is idle.
6. The objective of this online TSP is to minimize the time required to serve all jobs and return to the origin.
7. In the online problem, the salesman does not know in advance how many jobs are in a single problem instance. In the offline problem, all jobs and their release times are known *a priori*.
8. A problem instance,  $N$ , is a collection of  $n$  jobs, numbered  $1, \dots, n$ .

Note, we can completely describe a job  $i \in N$  by the following vector:  $(q_i^l, q_i^r, r_i, l_i)$  where  $l_i$  represents the location of job  $i$  on  $\mathbb{R}_0^+$ ;  $q_i^l$  is the point in time at which  $l_i$  is revealed; and  $q_i^r$  is the point in time at which  $r_i$  is

revealed, where  $r_i$  represents the release time of job  $i$ . In our variation of the online TSP, the information arrives such that  $0 \leq q_i^l \leq q_i^r \leq r_i$ . We further specify  $l_{\max}$  to represent the job that is farthest from the origin; that is,  $l_{\max} = \max_{i \in N} \{l_i\}$ . Similarly,  $r_{\max} = \max_{i \in N} \{r_i\}$  represents the job that is released the latest. The job at  $l_{\max}$  is not necessarily the same job with release time  $r_{\max}$ . The notation  $(x)^+$  is used as a short hand for  $\max\{x, 0\}$ .

As the remainder of this document focuses on competitive analysis, we use the notation  $C_A(N)$  to represent the cost of an algorithm,  $A$ , on an instance,  $N$ , of  $n$  jobs. Furthermore, we define the performance ratio of an algorithm  $A$  on an instance  $N$  as  $\frac{C_A(N)}{C_{OPT}(N)}$ . The value  $\rho_A$ , the worst-case ratio (as introduced in Chapter 1), is thus defined as the infimum over all performance ratios, which implies that  $C_A(N) \leq \rho_A C_{OPT}(N)$  for any instance  $N$ . A best possible algorithm is thus defined as an algorithm guaranteed to achieve a performance ratio less than or equal to the infimum over all algorithms of  $\rho_A$ . Finally, throughout this chapter we use the language of Jaillet and Wagner (2006) when writing out the relevant algorithms and affiliated costs.

The remainder of this section is divided into two subsections — the first in which we describe the optimal offline algorithm of Psaraftis et al. (1990) and the second in which we describe the online algorithms of Blom et al. (2001) and Jaillet and Wagner (2006).

#### 4.2.1 Optimal Offline Algorithm for the TSP on $\mathbb{R}^+$ with Release Dates

The offline version of the TSP with release dates on  $\mathbb{R}_0^+$  was first introduced in the context of routing and scheduling on a shoreline by Psaraftis et al. (1990). They propose an optimal offline algorithm entitled TRAVERSE and prove that it solves the problem exactly in  $O(n)$  time. The formal steps of the algorithm are repeated here, for convenience. TRAVERSE works by going to the farthest job from the origin, waiting at that job until the point in time where a smooth (i.e. no waiting) return to the origin can be made.

It is clear that the cost of this algorithm (that is the earliest point in time the salesman will return to the origin) is the time required to travel to

---

**Algorithm 1** TRAVERSE or OPT

---

1. Go directly to job  $l_{\max}$ .
  2. Wait at  $l_{\max}$  for  $\max_{i \in N} \{\max\{0, r_i - (2l_{\max} - l_i)\}\}$  units of time.
  3. Proceed directly back to the origin.
- 

the farthest location and back to the origin plus any waiting time incurred at that farthest location. Thus, the following is a closed form expression for  $C_{\text{TRAVERSE}}(N)$  (termed  $C_{\text{OPT}}(N)$  for future reference):

$$C_{\text{OPT}}(N) = \max_{i \in N} \{\max\{2l_i, r_i + l_i\}\} \quad (4.1)$$

### 4.2.2 Online TSP Algorithms

In this subsection, we review two different cases of advanced information arrival; for each we present the best-possible online algorithms. The first case is one in which a job's location and release time are disclosed at the moment of release, that is  $q_i^l = q_i^r = r_i$ . This first case is identical to that of the "OLTSP with release dates" originally proposed and studied by Blom et al. (2001). The second case is one in which the location and release time are disclosed simultaneously at a time in advance of the release time, that is  $q_i^l = q_i^r < r_i$ . This second case is identical to the "OLTSP with disclosure dates" originally proposed by Jaillet and Wagner (2006).

#### OLTSP with Release Dates

In their study of zealous algorithms and fair adversaries for the OLTSP with release dates, Blom et al. (2001) specify the Move-Right-if-Necessary (MRIN) algorithm as a strategy in the  $\mathbb{R}_0^+$  metric space. MRIN is a zealous algorithm in which the salesman moves to jobs on his right as soon as they are released and returns to the origin if there are no more jobs on the right.

Blom et al. (2001) show that MRIN is a best-possible online algorithm for the OLTSP with release dates on  $\mathbb{R}_0^+$  with a worst-case ratio of  $\frac{3}{2}$ . Therefore, in the case where  $q_i^l = q_i^r = r_i$ , MRIN is the best possible strategy.



---

**Algorithm 2** MRIN
 

---

1. If there is an unserved job to the right of the salesman, he moves toward it at unit speed.
  2. If there are no unserved jobs to the right of the salesman, he moves back toward the origin at unit speed.
  3. Upon reaching the origin, the salesman becomes idle.
- 

**OLTSP with Disclosure Dates**

In their study of online routing problems, Jaillet and Wagner (2006) introduce the OLTSP with disclosure dates and specify the Move-Left-If-Beneficial (MLIB) algorithm as a strategy in the  $\mathbb{R}_0^+$  metric space. MLIB is based on the idea that with prior knowledge of jobs to the left of the salesman it is better to wait as far right for as long as possible. In this way, MLIB represents a compromise strategy between the optimal offline, TRAVERSE algorithm and the online MRIN strategy.

---

**Algorithm 3** MLIB
 

---

1. If there is an unserved job to the right of the salesman, he moves toward it at unit speed.
  2. If there are no unserved jobs to the right of the salesman, he moves back toward the origin if and only if the return trajectory reaches all unserved jobs on or after their release date; otherwise the salesman remains idle at his current location.
  3. Upon reaching the origin, the salesman becomes idle.
- 

Jaillet and Wagner (2006) show that MLIB is a best-possible online algorithm for the OLTSP with disclosure dates on  $\mathbb{R}_0^+$  when the amount of advanced notice (i.e. the time between disclosure and release) is fixed. We extend their results slightly to show that MLIB is also best-possible when the amount of advanced notice is variable (see Section 4.5). In both settings (fixed and variable advanced notice) the worst-case ratio of MLIB is not constant

and instead varies based on the amount of advanced notice; nevertheless the worst-case ratio never exceeds  $\frac{3}{2}$ . Therefore, in the case where  $q_i^l = q_i^r \leq r_i$ , MLIB is the best possible strategy.

### 4.3 OLTSP with Two Disclosure Dates

The primary focus of this chapter is one in which the location is disclosed earlier than the release time which is disclosed earlier than the release itself, that is  $q_i^l \leq q_i^r \leq r_i$ . In this instance, we can construct an algorithm that not only exploits the advanced release information but also the earlier disclosed location information. The Move-Right-Early-Left-Late (MRELL) algorithm is based on the idea that it is better to wait as far in the field as long as possible than hastily return to the origin.

---

#### Algorithm 4 MRELL

---

1. If there is a job for which the location has been revealed to the right of the salesman he moves towards it at unit speed.
  2. If there are no jobs to the right of the salesman, he moves back to the origin according to the following rules:
    - (a) If the salesman knows the release time of all the jobs to his left, whose locations have been disclosed, then the salesman returns to the origin at the point in time that allows him to pass all jobs on or after their release time.
    - (b) If the salesman knows the release time of only some of all jobs to his left, whose locations have been disclosed, then the salesman remains idle until the time that allows him to pass all release time disclosed jobs on or after their release time, but the salesman must stop along this trajectory and wait at any job for which only the location is known.
    - (c) If the salesman knows none of the release times for all the location-disclosed jobs to his left, then he moves toward the nearest job waiting there until its release.
  3. Upon reaching the origin, the salesman remains idle.
-

Note that if this algorithm is applied to a case where  $q_i^l = q_i^r = r_i$  then MRELL is equivalent to MRIN (Blom et al., 2001). Furthermore, if this algorithm is applied to a case where  $q_i^l = q_i^r < r_i$  then MRELL is equivalent to MLIB (Jaillet and Wagner, 2006). Additionally, note that if  $0 = q_i^l = q_i^r < r_i, \forall i \in N$  then this algorithm is indistinguishable from the optimal offline algorithm (see Psaraftis et al., 1990).

**Lemma 1.** *The cost of MRELL is bounded as follows:*

$$C_{MRELL}(N) \leq \max_{i \in N} \left\{ \max \left\{ q_i^l + 2l_i, r_i + l_i \right\} \right\} \quad (4.2)$$

**Proof** Using logic similar to Jaillet and Wagner (2006), we derive the cost of MRELL by analyzing the final segment of the salesman's journey. That is, the segment of the salesman's journey in which he leaves a job to return directly to the origin without stopping to wait at any other job along the way. We say that this final segment will begin at a time,  $t_0$  with the salesman arriving at the origin at time  $z = C_{MRELL}(N)$ . According to the algorithm, MRELL, the salesman may begin his final segment to the origin from any job (a job we will term the *final departure job*) to the right of the origin, on the condition that all jobs in between the final departure job and the origin will be passed on or after their release time. We proceed by analyzing two cases.

1. Salesman leaves final departure job as soon as he arrives.

- This represents the case where the salesman arrives to the final departure job after the release time of that job and at a point in time at which all jobs between that final departure job and the origin can be passed on or after their release times. Note that in this case, the salesman was traveling away from the origin just before turning back for the final segment at the final departure job. Thus, the salesman begins his return segment immediately after arriving to the final departure job,  $k$ . This gives us that  $t_0 = \text{arrival to } k \leq q_k^l + l_k$ . Note that  $q_k^l + l_k$  represents departure from the origin and hence the worst case. Thus,  $z \leq q_k^l + l_k + l_k = q_k^l + 2l_k$ .

2. Salesman leaves final departure job after waiting.

- This case represents a situation where the salesman must wait for the release of some job between the final departure job and the origin (possibly the final departure job itself). In this second case the salesman will already have spent some time at the final departure job before returning to the origin - thus he may have come to that final departure job from either the left or the right. In this case the final segment is timed to pass through some job,  $m$ , at a time  $t > t_0$  such that  $t = r_m$  and  $r_m$  is the latest release time remaining. Thus, the salesman will finish the final segment at  $z = r_m + l_m$ .

Because the last segment of the salesman's trajectory can only be of one case type, we may say that  $z \leq \max \{q_k^l + 2l_k, r_m + l_m\}$ . Furthermore, because these cases represent the latest event in the trajectory of the salesman we can write,  $C_{MRELL}(N) = z \leq \max_{i \in N} \{\max \{q_i^l + 2l_i, r_i + l_i\}\}$ .  $\square$

The following corollary further illustrates the relationship between  $C_{MRELL}(N)$  and  $C_{OPT}(N)$ . Corollary 1 will also be used in proving Theorem 2.

**Corollary 1.** *If  $q_i^l = 0$  and  $q_i^r \leq r_i, \forall i \in N$ , then  $C_{MRELL}(N) = C_{OPT}(N)$ .*

Related to Corollary 1 we have Lemma 2 that will be used in the proof of both Theorem 2 and Theorem 4.

**Lemma 2.** *For any instance,  $N$ , of the online TSP with release dates on  $\mathbb{R}^+$ , we can construct a related instance,  $\tilde{N}$ , in which all jobs in the set  $Q = \{i \in N \mid q_i^l = 0\}$  are excluded. The performance ratio for instance  $\tilde{N}$  will not be less than the performance ratio for instance  $N$ .*

**Proof** Let  $C_{MRELL}(\tilde{N})$  be the cost of MRELL on the instance  $\tilde{N} = N \setminus Q$ ; similarly let  $C_{OPT}(\tilde{N})$  be the cost of OPT on the instance  $\tilde{N} = N \setminus Q$ .

If  $\max_{i \in Q} \{r_i + l_i, 2l_i\} \geq C_{MRELL}(\tilde{N})$ , then  $C_{MRELL}(N) = \max_{i \in Q} \{r_i + l_i, 2l_i\} = C_{OPT}(N)$ ; which gives us a performance ratio of 1. As the performance ratio for instance  $\tilde{N}$  must be greater than or equal to 1

we have that the performance ratio for instance  $\tilde{N}$  will not be less than the performance ratio for instance  $N$ .

If instead,  $\max_{i \in Q} \{r_i + l_i, 2l_i\} < C_{MRELL}(\tilde{N})$  then  $C_{MRELL}(\tilde{N}) = C_{MRELL}(N)$ . Since,  $C_{OPT}(\tilde{N}) \leq C_{OPT}(N)$ , the performance ratio for instance  $N$  is less than or equal to the performance ratio of  $\tilde{N}$ .

□

## 4.4 Fixed Amounts of Advanced Notice

In this case, we imagine that the salesman is told the location of each job at a point in time  $(a + b)$  units of time before the release of the job. Similarly, the release time of each job is announced  $a$  units of time before the release of the job. We may also write this as follows. For each job in a problem instance, there exist constants  $a$  and  $b$  such that  $(a + b) \in [0, r_{\max}]$ , yielding  $q_i^r = (r_i - a)^+$ ,  $\forall i \in N$  and  $q_i^l = (r_i - a - b)^+$ ,  $\forall i \in N$ . Given this notation and noting that  $2l_{\max}$  is a lower bound on the length of the optimal TSP tour through all jobs, we have the following theorem.

**Theorem 1.** *Let  $A$  be an arbitrary homing online algorithm with cost  $C_A(N)$  on an instance of  $n$  jobs. Then for all  $n \geq 2$  there exists an instance  $N$ , of size  $n$ , where the performance ratio is at least  $\left[\frac{3}{2} - \left(\frac{a+b}{2l_{\max}}\right)\right] \in [1, \frac{3}{2}]$ .*

**Proof** Using logic similar to Jaillet and Wagner (2006), we begin by establishing an arbitrary instance  $N'$  of  $n - 1$  jobs. Given this instance, the time at which the salesman finishes serving all  $n - 1$  jobs and returns to the origin is given by our arbitrary algorithm,  $A$ , as  $C_A(N')$ . We now designate an  $n^{\text{th}}$  job which is further out on  $\mathbb{R}^+$  than any of the previous  $n - 1$  jobs. Thus,  $l_n = l_{\max}$ . To specify the exact location of  $l_{\max}$ , we note that  $C_A(N') \geq C_{OPT}(N') \geq 2l_i, \forall i \in N'$ . Thus, by setting  $l_n$  equal to  $C_A(N')$  plus some constant term, we are assured that  $l_n$  is  $l_{\max}$  for this instance of  $n$  jobs. We therefore select  $l_n = (a + b) + C_A(N')$ . Note, if  $(a + b) = 0$  then  $q_n^l = r_n$  and the analysis of Blom et al. (2001) applies thus completing our

proof. However, if  $(a + b) > 0$ , we obtain the following description of job  $n$ :  
 $(q_n^l, q_n^r, r_n, l_n) = (C_A(N'), a + C_A(N'), (a + b) + C_A(N'), (a + b) + C_A(N'))$ .

Given this job and the knowledge that the salesman is at the origin at  $q_n^l = C_A(N')$ , we obtain the following:  $C_A(N) \geq q_n^l + 2l_n = 3C_A(N') + 2(a + b)$ .

Turning our attention to the optimal offline algorithm, we have:

$$C_{OPT}(N) = \max\{\max\{2l_i, r_i + l_i\}\} = 2C_A(N') + 2(a + b)$$

As  $(a + b) > 0$  then  $C_{OPT}(N) > 0$ . We now obtain the desired result:

$$\begin{aligned} \frac{C_A(N)}{C_{OPT}(N)} &\geq \frac{3C_A(N') + 2(a + b)}{2C_A(N') + 2(a + b)} \\ &= 1 + \frac{C_A(N')}{2l_{\max}} \\ &= 1 + \frac{l_{\max} - (a + b)}{2l_{\max}} \\ &= \frac{3}{2} - \left(\frac{a + b}{2l_{\max}}\right) \end{aligned}$$

Given that  $(a + b) \leq l_{\max}$ , we conclude that  $\frac{3}{2} - \left(\frac{a+b}{2l_{\max}}\right) \in [1, \frac{3}{2}]$ .  $\square$

**Theorem 2.** *When the amount of advanced notice is fixed such that,  $q_i^r = (r_i - a)^+$  and  $q_i^l = (r_i - a - b)^+$ ,  $\forall i \in N$ , then MRELL is a best-possible algorithm.*

**Proof** Define  $\mathfrak{L} = \{i \in N | q_i^l > 0\}$ . Note that if  $\mathfrak{L} = \emptyset$  then the location of all jobs are known at the start of the day. Thus, by Lemma 2, we obtain  $C_{MRELL}(N) = C_{OPT}(N)$ . However, if  $\mathfrak{L}$  is not empty, then we rewrite inequality 4.2 as:

$$C_{MRELL}(N) \leq \max \left\{ \max_{i \in \mathfrak{L}} \left\{ \max \left\{ q_i^l + 2l_i, r_i + l_i \right\} \right\}, \max_{i \in N \setminus \mathfrak{L}} \left\{ \max \{2l_i, r_i + l_i\}\} \right\}$$

Now, by Lemma 2 we can ignore all the jobs not in  $\mathfrak{L}$  without risk of

reducing the competitive ratio. Thus we obtain:

$$C_{MRELL}(N) \leq \max_{i \in \mathcal{L}} \left\{ \max \left\{ q_i^l + 2l_i, r_i + l_i \right\} \right\}$$

Using the definition of  $q_i^l$  we do the following algebra:

$$\begin{aligned} C_{MRELL}(N) &\leq \max_{i \in \mathcal{L}} \left\{ \max \left\{ q_i^l + 2l_i, r_i + l_i \right\} \right\} \\ &= \max_{i \in \mathcal{L}} \left\{ \max \left\{ r_i - a - b + 2l_i, r_i + l_i \right\} \right\} \\ &= \max_{i \in \mathcal{L}} \left\{ r_i + l_i + \max \left\{ l_i - a - b, 0 \right\} \right\} \\ &\leq \max_{i \in \mathcal{L}} \left\{ r_i + l_i + \max \left\{ l_{\max} - a - b, 0 \right\} \right\} \end{aligned} \quad (4.3)$$

We now analyze two cases:

**Case 1:**  $l_{\max} - a - b < 0 \Leftrightarrow l_{\max} < a + b$ . This case implies that  $C_{MRELL}(N) \leq \max_{i \in \mathcal{L}} \{r_i + l_i\} \leq C_{OPT}(N)$  which implies that  $C_{MRELL}(N) = C_{OPT}(N)$ .

**Case 2:**  $l_{\max} - a - b \geq 0 \Leftrightarrow l_{\max} \geq a + b$ . In this case we may rewrite inequality 4.3 in the following way.

$$\begin{aligned} C_{MRELL}(N) &\leq \max_{i \in \mathcal{L}} \{r_i + l_i + l_{\max} - a - b\} \\ &= \max_{i \in \mathcal{L}} \{r_i + l_i\} + l_{\max} - a - b \\ &\leq C_{OPT}(N) + l_{\max} - a - b \end{aligned}$$

Rewriting  $l_{\max} - a - b$  as  $\frac{l_{\max} - a - b}{l_{\max}} l_{\max}$ , we obtain the desired result.

$$\begin{aligned} C_{MRELL}(N) &\leq C_{OPT}(N) + \frac{l_{\max} - a - b}{2l_{\max}} 2l_{\max} \\ &\leq C_{OPT}(N) + \frac{l_{\max} - a - b}{2l_{\max}} C_{OPT}(N) \\ &= \left[ \frac{3}{2} - \frac{(a + b)}{2l_{\max}} \right] C_{OPT}(N) \end{aligned}$$

Recognizing that these cases are disjoint, we state,

$$C_{MRELL}(N) \leq \max \left\{ 1, \frac{3}{2} - \left( \frac{a+b}{2l_{\max}} \right) \right\} C_{OPT}(N).$$

As Theorem 1 gives us that  $\max \left\{ 1, \frac{3}{2} - \left( \frac{a+b}{2l_{\max}} \right) \right\}$  is the lowest possible performance ratio for any algorithm we may conclude that MRELL is a best-possible algorithm. □

## 4.5 Variable Amounts of Advanced Notice

In this subsection, we explore the worst-case ratio of MRELL in the context of variable amounts of advanced notice time for both the location and release time disclosures. In examining Lemma, 1 we note that the job driving the cost of MRELL (we will call this job  $d$ ) can be one of two types: (1) the job may be such such that  $\max_{i \in N} \{ \max \{ q_i^l + 2l_i, r_i + l_i \} \} = r_d + l_d$  or (2) the job may be such that  $\max_{i \in N} \{ \max \{ q_i^l + 2l_i, r_i + l_i \} \} = q_d^l + 2l_d$ . If job  $d$  is of type one, then the cost of MRELL will be equal to the cost of the optimal offline algorithm. Given this phenomenon, the worst-case ratio is primarily determined by the value of  $\max_{i \in N} \{ q_i^l + 2l_i \}$ .

**Theorem 3.** *Let  $A$  be an arbitrary homing online algorithm with cost  $C_A(N)$  on an instance,  $N$ , of  $n$  jobs. Then for all  $n \geq 2$  there exists an instance of size  $n$  where the performance ratio is at least*

$$1 + \min \left\{ \left( \frac{\max_{i \in N} \{ q_i^l + 2l_i \}}{\max_{i \in N} \{ r_i + l_i \}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{ q_i^l + 2l_i \}}{\max_{i \in N} \{ 2l_i \}} - 1 \right) \right\} \in \left[ 1, \frac{3}{2} \right].$$

**Proof** Applying the same logic as in Theorem 1, we specify an arbitrary instance  $N'$  of  $n - 1$  jobs that the salesman serves and then returns to the origin. Thus, the salesman is at the origin at time  $C_A(N')$ . We now specify the  $n^{\text{th}}$  job at a location on  $\mathbb{R}^+$  that is further from the origin than any other of the  $n - 1$  jobs with a release time later than all others. We may therefore



describe the  $n^{\text{th}}$  job fully as follows.

$$\begin{aligned} & \left( q_n^l, q_n^r, r_n, l_n \right) \\ &= \left( q_{\max}^l, q_{\max}^r, r_{\max}, l_{\max} \right) \\ &= \left( C_A(N'), C_A(N') + \max_{i \in N \setminus n} \{r_i - q_i^r\}, C_A(N') + \delta, C_A(N') + \delta \right) \end{aligned}$$

Note that  $\delta = \max_{i \in N \setminus n} \{r_i - q_i^l\}$ .

This plus the knowledge that the salesman is at the origin at time  $C_A(N')$  yields:

$$C_A(N) \geq q_n^l + 2l_n = 3C_A(N') + 2\delta$$

Turning our attention to the cost of the optimal offline algorithm we have:

$$C_{OPT}(N) = \max_{i \in N} \{ \max \{2l_i, r_i + l_i\} \} = 2C_A(N') + 2\delta$$

This gives us the following:

$$\begin{aligned} & \frac{C_A(N)}{C_{OPT}(N)} \\ & \geq \frac{3C_A(N') + 2\delta}{2C_A(N') + 2\delta} \\ & = 1 + \frac{C_A(N')}{2C_A(N') + 2\delta} \\ & = 1 + \frac{C_A(N') + 2(C_A(N') + \delta) - 2(C_A(N') + \delta)}{2(C_A(N') + \delta)} \\ & \geq 1 + \min \left\{ \frac{q_n^l + 2l_n - (r_n + l_n)}{r_n + l_n}, \frac{q_n^l + 2l_n - 2l_n}{2l_n} \right\} \\ & = 1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \end{aligned} \tag{4.4}$$

We now note that if we let  $\delta$  decrease to 0 in equation (4.4), then this fraction increases to  $\frac{3}{2}$ ; alternately if we take the limit of  $\delta$  approaching  $\infty$ ,

then this fraction decreases to 1. Therefore,

$$1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \in \left[ 1, \frac{3}{2} \right].$$

□

The following theorem establishes that MRELL is also a best-possible algorithm in the context of variable notice.

**Theorem 4.** *When the amount of advanced notice varies for each job,  $i \in N$ , then  $\rho_{MRELL} \leq 1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \leq \frac{3}{2}$  and MRELL is a best-possible algorithm.*

**Proof** Let  $q_m^l + 2l_m = \max_{i \in N} \{q_i^l + 2l_i\}$ ,  $r_p + l_p = \max_{i \in N} \{r_i + l_i\}$ , and  $2l_{\max} = \max_{i \in N} \{2l_i\}$ . Note that jobs  $m$ ,  $p$ , and  $\max$  in the case of  $l_{\max}$  may actually represent the same job depending on the instance. We further define two sets:

$$\begin{aligned} \mathcal{L}(N) &= \left\{ j \in N \mid q_j^l + 2l_j = \max \left\{ \max_{i \in N} \{q_i^l + 2l_i, r_i + l_i\} \right\} \right\} \\ \mathcal{R}(N) &= \left\{ k \in N \mid r_k + l_k = \max \left\{ \max_{i \in N} \{q_i^l + 2l_i, r_i + l_i\} \right\} \right\} \end{aligned}$$

Given these two sets we proceed with the proof by examining three cases: (1)  $\mathcal{L}(N) = \emptyset, \mathcal{R}(N) \neq \emptyset$ , (2)  $\mathcal{L}(N) \neq \emptyset, \mathcal{R}(N) \neq \emptyset$ , and (3)  $\mathcal{L}(N) \neq \emptyset, \mathcal{R}(N) = \emptyset$ . Note if  $\mathcal{L}(N) = \mathcal{R}(N) = \emptyset$  then there are no jobs in the problem instance.

**Case 1:**  $\mathcal{L}(N) = \emptyset, \mathcal{R}(N) \neq \emptyset$  In this case  $p \in \mathcal{R}(N)$ . Thus,

$$r_p + l_p = \max \left\{ \max_{i \in N} \{q_i^l + 2l_i, r_i + l_i\} \right\} \geq 2l_{\max}$$

which implies that  $C_{MRELL}(N) \leq r_p + l_p \leq C_{OPT}(N)$ . Thus,  $\frac{C_{MRELL}(N)}{C_{OPT}(N)} \leq 1 \leq \frac{3}{2}$ . We further note, that in this case  $r_p + l_p > q_m^l + 2l_m$ . Hence

$\frac{q_m^l + 2l_m}{r_p + l_p} < 1$  which yields

$$\left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+ = 0.$$

Therefore, in this case,

$$\begin{aligned} & \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \\ & = 0. \end{aligned}$$

As a result we can conclude that in this case,

$$\begin{aligned} & \frac{C_{MRELL}(N)}{C_{OPT}(N)} \\ & \leq 1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \\ & \leq \frac{3}{2}. \end{aligned}$$

**Case 2:**  $\mathcal{L}(N) \neq \emptyset, \mathcal{R}(N) \neq \emptyset$  In this case  $m \in \mathcal{L}(N)$  and  $p \in \mathcal{R}(N)$  which gives us that  $q_m^l + 2l_m = q_j^l + 2l_j = \max \{ \max_{i \in N} \{q_i^l + 2l_i, r_i + l_i\} \} = r_k + l_k = r_p + l_p$ . Thus,  $C_{MRELL}(N) \leq r_p + l_p \leq C_{OPT}(N)$  yielding  $\frac{C_{MRELL}(N)}{C_{OPT}(N)} \leq 1 \leq \frac{3}{2}$ . In this case we also note that  $r_p + l_p = q_m^l + 2l_m$  which gives that  $\left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+ = 0$ . Therefore,

$$\min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} = 0.$$

As a result we conclude that in this case,

$$\begin{aligned} \frac{C_{MRELL}(N)}{C_{OPT}(N)} &\leq \\ 1 + \min &\left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \\ &\leq \frac{3}{2}. \end{aligned}$$

**Case 3:**  $\mathcal{L}(N) \neq \emptyset, \mathcal{R}(N) = \emptyset$  In this case,  $m \in \mathcal{L}(N)$ . Thus,

$$\begin{aligned} &q_m^l + 2l_m \\ &= \max \left\{ \max_{i \in N} \{q_i^l + 2l_i, r_i + l_i\} \right\}. \end{aligned}$$

So we may conclude that  $q_m^l + 2l_m > r_p + l_p$ . We may also conclude that  $q_m^l + 2l_m > 2l_{\max}$ , because Lemma 2 allows us to ignore all jobs for which  $q_i^l = 0$ . We thus examine two cases, (1)  $q_m^l + 2l_m > r_p + l_p > 2l_{\max}$  and (2)  $q_m^l + 2l_m > 2l_{\max} > r_p + l_p$ .

**Case 3.1:**  $q_m^l + 2l_m > r_p + l_p > 2l_{\max}$

$$\begin{aligned} C_{MRELL}(N) &\leq q_m^l + 2l_m \\ &= \frac{q_m^l + 2l_m}{r_p + l_p} (r_p + l_p) \\ &\leq \left[ \left( \frac{q_m^l + 2l_m}{2l_{\max}} - 1 \right) + 1 \right] C_{OPT}(N) \end{aligned}$$

$$\text{Thus, } \frac{C_{MRELL}(N)}{C_{OPT}(N)} \leq 1 + \left( \frac{q_m^l + 2l_m}{r_p + l_p} - 1 \right)^+ \leq 1 + \left( \frac{q_m^l + 2l_m}{2l_{\max}} - 1 \right).$$

Which gives us the result that

$$\frac{C_{MRELL}(N)}{C_{OPT}(N)} \leq 1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\}$$

**Case 3.2:**  $q_m^l + 2l_m > 2l_{\max} > r_p + l_p$

$$\begin{aligned} C_{MRELL}(N) &\leq q_m^l + 2l_m \\ &= \frac{q_m^l + 2l_m}{2l_{\max}} (2l_{\max}) \\ &\leq \left[ \left( \frac{q_m^l + 2l_m}{r_p + l_p} - 1 \right) + 1 \right] C_{OPT}(N) \end{aligned}$$

Thus,  $\frac{C_{MRELL}(N)}{C_{OPT}(N)} \leq 1 + \left( \frac{q_m^l + 2l_m}{2l_{\max}} - 1 \right) \leq 1 + \left( \frac{q_m^l + 2l_m}{r_p + l_p} - 1 \right)^+$ , which gives us the result that

$$\frac{C_{MRELL}(N)}{C_{OPT}(N)} \leq 1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\}$$

We conclude Case 3 by proving that

$$\begin{aligned} &1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \\ &\leq \frac{3}{2}. \end{aligned}$$

Proving this statement is done via contradiction. Assume that

$$1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} > \frac{3}{2}.$$

This implies:

$$\begin{aligned} & \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+ > \frac{1}{2} \\ & \wedge \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) > \frac{1}{2} \\ \Rightarrow & \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} > \frac{3}{2} \\ \Rightarrow & 2(q_m^l + 2l_m) > 3(r_p + l_p) > 3(r_m + l_m) > 3(q_m^l + l_m) \\ \Rightarrow & l_m > q_m^l \\ \Rightarrow & 2(3l_m) > 2(q_m^l + 2l_m) > 3(2l_{\max}) \\ \Rightarrow & 2l_m > 2l_{\max} = \max_{i \in N} \{2l_i\} \end{aligned}$$

Which is a contradiction. Thus,

$$1 + \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \leq \frac{3}{2}.$$

As these three cases cover all possible situations, we obtain the desired result:

$$\begin{aligned} \frac{C_{MRELL}(N)}{C_{OPT}(N)} &\leq \\ 1 + \min &\left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \\ &\leq \frac{3}{2} \end{aligned}$$

□

This analysis also serves to further the results of Jaillet and Wagner (2006). In their paper, Jaillet and Wagner (2006) give a rather complex worst-case ratio of MLIB under conditions of variable advanced notice. However, by noting that when  $q_i^l = q_i^r$  the two algorithms, MRELL and MLIB, are equivalent, we may give the following expression as the competitive ratio of MLIB under conditions of variable advanced notice.

$$\begin{aligned} \frac{C_{MLIB}(N)}{C_{OPT}(N)} &\leq \\ 1 + \min &\left\{ \left( \frac{\max_{i \in N} \{q_i^r + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^r + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\} \\ &\leq \frac{3}{2}. \end{aligned}$$

Furthermore, by following a similar set of arguments as outlined in Theorems 3 and 4, it is possible to prove that MLIB is a best-possible algorithm when  $q_i^l = q_i^r \quad \forall i \in N$  and conditions of variable amounts of advanced notice prevail.

## 4.6 Discussion

Given these elaborate worst-case ratios for MRELL under conditions of fixed and variable advanced notice, what can be said about the value of location information? We begin by noting that MRIN is an algorithm that uses no

advanced information; all actions are take at  $r_i$ . MRELL on the other hand uses advanced location and release time information; actions are taken at both  $q_i^l$  and  $q_i^r$ . Therefore by comparing these two extreme algorithms we may specify a value for the advanced location information.

In previous papers (see e.g. Jaillet and Wagner (2006)) the comparison between different algorithms was undertaken by subtracting the worst-case ratios of the two algorithms. We too will begin our comparison between MRIN and MRELL using this methodology. We then show that this method may yield a deceptive value for the location information. As a final result we specify a realistic range of values and describe policies that give MRELL a consistent improvement over MRIN.

We begin our comparison by studying the difference  $\rho_{MRIN} - \rho_{MRELL}$ . If we calculate this value directly we obtain:

$$\rho_{MRIN} - \rho_{MRELL} = \frac{1}{2} - \min \left\{ \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{r_i + l_i\}} - 1 \right)^+, \left( \frac{\max_{i \in N} \{q_i^l + 2l_i\}}{\max_{i \in N} \{2l_i\}} - 1 \right) \right\}$$

As this expression is strictly positive, we may be inclined to conclude that advanced location information is similarly strictly beneficial. However, if we recall that  $1 \leq \frac{C_{MRIN}(N)}{C_{OPT}(N)} \leq \rho_{MRIN} \leq \frac{3}{2}$  and  $1 \leq \frac{C_{MRELL}(N)}{C_{OPT}(N)} \leq \rho_{MRELL} \leq \frac{3}{2}$ . Then, we may say:

$$-\frac{1}{2} \leq 1 - \rho_{MRELL} \leq \frac{C_{MRIN}(N) - C_{MRELL}(N)}{C_{OPT}(N)} \leq \rho_{MRIN} - 1 \leq \frac{1}{2} \quad (4.5)$$

Equation (4.5) implies that in some instances advanced location information can be detrimental. Given these conflicting observations, stemming from the broad range in which  $\frac{C_{MRIN}(N) - C_{MRELL}(N)}{C_{OPT}(N)}$  can fall, we cannot immediately specify a value for advanced location information. We therefore explore



the full implications of this range in more detail.

We begin our more complete comparison of MRIN and MRELL by examining the extreme left of the range. It appears from the analysis in equation (4.5) that

$$\frac{C_{MRIN}(N) - C_{MRELL}(N)}{C_{OPT}(N)}$$

can be as low as  $-\frac{1}{2}$ . This is, however, not true as there are no instances such that

$$\frac{C_{MRIN}(N)}{C_{OPT}(N)} = 1$$

at the same time that

$$\frac{C_{MRELL}(N)}{C_{OPT}(N)} = \frac{3}{2}.$$

Instead, we put forth the following conjecture.

**Conjecture 1.**  $\frac{C_{MRIN}(N) - C_{MRELL}(N)}{C_{OPT}(N)} \geq -\frac{1}{3}$  for all instances  $N$ .

An example of one such instance that drives the difference in the algorithms' costs to its lowest value of  $-\frac{1}{3}$  is:  $l_1 = l_2 = 2$ ,  $r_1 = 2$ ,  $q_1^l = 0$ ,  $q_1^r = 1$ ,  $r_2 = q_2^l = q_2^r = 4$ .

We now explore the extreme positive end of the range for the difference in  $C_{MRIN}(N)$  and  $C_{MRELL}(N)$  as compared to  $C_{OPT}(N)$ . We can immediately see that there exist instances such that  $\frac{C_{MRIN}(N) - C_{MRELL}(N)}{C_{OPT}(N)} = \frac{1}{2}$ . Take for example the instance where  $q_1^l = 0$  and  $q_1^r = r_1 = 1$ . We, therefore, conclude that:

$$-\frac{1}{3} \leq \frac{C_{MRIN}(N) - C_{MRELL}(N)}{C_{OPT}(N)} \leq \frac{1}{2}. \quad (4.6)$$

If we assume a uniform distribution of instances across this range then we can say that on average using MRELL to exploit advanced location information will yield a cost improvement of  $\frac{1}{12}$ . Of course, if the instances are distributed differently the benefit of advanced location information may be drastically reduced. We therefore turn our attention toward policies that can improve the value of advanced location information.

We first note that the instances rendering advanced location information detrimental are those for which an earlier job drives the cost of MRIN while

a later job with no advanced notice drives the cost of MRELL. Thus, the best policy strategy is one that requires all job locations to be announced at some point in advance of their release date. In fact this is the reasoning behind the analysis of fixed information in Section 4.4. It is important to note that in instances of fixed advanced notice, where  $a > 0$  and  $b > 0$ ,  $C_{MRIN}(N) \geq C_{MRELL}(N)$ . This is because given the point in time that the location is revealed, the release time can be computed. As both  $a$  and  $b$  are positive this information can be computed in advance of the actual release time thereby avoiding the types of detrimental instances examined above.

A second strategy is to introduce a job pricing scheme that charges a premium for those jobs not willing or able to announce the location until a time close to the job's release date. This premium can be set dynamically to cover any costs originating from acting too soon for a previous job. For example, by specifying a price per job equal to the time the location is revealed plus the round trip distance of the job (i.e.  $q_i^l + 2l_i$ ), then customers will have an incentive to provide the job location information early. If a job location is revealed late then such a fee would cover the cost of service regardless of the situation created by a previous job. Admittedly, while this scheme is theoretically sufficient to cover the cost of jobs revealed too late it may be confusing to customers who are likely to prefer fixed rates based solely on distance. Nevertheless this still provides some benefit to the customer as they do not need to reveal the release time any earlier — only the location of the job.

This observation yields the following question, does providing information on the release time early yield any benefit? We answer this question by noting that the earliest that the release time may be disclosed is  $q_i^r = q_i^l$ . If this is done for all jobs  $i \in I$ , then  $C_{MLIB}(N) = C_{MRELL}(N)$ ; thus,  $\frac{C_{MLIB}(N) - C_{MRELL}(N)}{C_{OPT}(N)} = 0$ . From this analysis, we may conclude that the value of location information is immense. The revelation of location information alone brings all the benefit or detriment. This value ranges, dependent on the problem instance, from  $-\frac{1}{3}$  to  $\frac{1}{2}$  in terms of the difference in the cost of these algorithms as compared to the optimal solution.

These results represent only a first step towards a meaningful analysis of the drayage problem at the center of this thesis. A clear first extension to this work is an analysis of the same problem in more realistic metric spaces, such as a general metric space or  $\mathbb{R}^2$ . A second extension of interest is the design of an online job selection algorithm. For example, by rejecting jobs based on a comparison of their disclosed locations to already accepted job locations might yield significant performance gains. Finally, we recommend studying other versions of the TSP — such as the TSP with pick-up and delivery or the ATSP. As a side note, to date and to the best of our knowledge, only one paper attacks an online analysis of an ATSP. Ausiello et al. (2008) study the online asymmetric traveling salesman problem demonstrating that the competitive ratio for any online asymmetric TSP, that must return to a specified location, is at least one plus the *golden ratio*.

Finally, we note that, in the simplified context of the TSP there is but one server — the salesman, alone. This obfuscates the need for a higher level of control. In reality, however, many drayage companies operate more than one vehicle. Therefore, the key to exploiting advanced information rests largely on the agility of the control mechanism. That is, the mechanism by which jobs are assigned to individual vehicles. The quantifiable merits of centralized versus decentralized control, in realistic drayage problems, is the topic of the next chapter.

## Chapter 5

# Centralized versus Decentralized Control in Drayage

Only the governed exist to govern themselves.

Kahlil Gibran, *The Wanderer: His Parables and Sayings*

Imagine working as a dispatcher for a medium sized freight logistics company. Your day begins by matching a set of orders to a group of drivers. Maybe a computer helps you in this task, but ultimately the outcome is the same - a schedule for the day. This schedule has been carefully constructed to serve all orders at least cost while taking a variety of constraints (e.g. equipment type, time windows, hours of service regulations, etc) into account. Immediately after enacting this plan, changes occur. A truck breaks down, a customer cancels, a load is bigger than expected. The phone starts ringing, and your growing headache reminds you that you should ask your boss for a raise.

The next day you try an experiment. After giving all your drivers cell phones, PDAs, and GPS navigation systems, you tell them to communicate

with each other and the customers to create their own schedule. You also make the drivers responsible for negotiating solutions to any troubles encountered en route. In effect, you have rendered your job as a central dispatcher obsolete, leaving more time for other office management tasks. But will the drivers, operating without central knowledge, find the most cost effective route? Which solution will fulfill (or exceed) company goals and objectives?

Experiments studying the behavior of agent based methods in comparison to traditional optimization methods are generally absent from the literature. Independent of comparative benchmarks, the literature holds several claims that agent-based solutions perform well in uncertain domains (Fischer et al., 1995), that is, in domains where the problem (and its associated solution space) is continually evolving. The key objective of this chapter is to test these claims by studying the performance of an agent-based solution and an on-line optimization approach with respect to handling uncertainty in the context of the previously introduced drayage company at the Port of Rotterdam, the Netherlands. Recall, in our case, a container transport company with a fleet of 40 vehicles must pick up containers from terminals at the Port of Rotterdam, transport the containers to a customer location in the hinterland arriving within a given time window, wait with the container until it is loaded or unloaded, and then return the full or empty container to another port terminal.

Given this method of operations, the problem may be described, in a static manner, as a pick-up and delivery problem with time windows (PDPTW). Reality, however, reminds us that this problem is anything but static and as such we study this problem in an on-line context taking into account two types of uncertainty — service time uncertainty and job arrival uncertainty. Job arrival uncertainty is the most basic type of uncertainty, as a job cannot be planned for or served until it is made known to the planning system. Furthermore, while the presence of a job may be known in advance, the amount of time required to pick it up from the terminal, service it at the customer location, and process it at the return terminal can be highly variable.

The remainder of this chapter<sup>1</sup> describes the foundational literature on the PDPTW under conditions of uncertainty as solved by both optimization-based and agent-based solution approaches (Section 5.1). In Section 5.2, we provide a detailed description of our experimental design including a description of the solution approaches, the data used, and the two types of uncertainty examined. Results, on the performance of both systems across several scenarios of varying service time and job arrival uncertainty, appear in Section 5.3. A discussion of these results and suggestions for future research conclude this chapter.

## 5.1 Related Work

As noted in the introduction, at the heart of this research is a case study in drayage. This case can be described in operations research terms as a truckload pick-up and delivery problem with time-windows (PDPTW). In the PDPTW, a fleet of vehicles, capable of carrying only one job at a time, must pick up a job from one location and drop it off at another location within a specified time period. Finding the assignment of jobs to trucks that minimizes costs (in the form of total distance, empty distance, or operating costs) is the solution goal.

While it is easiest to describe and classify these problems in a static manner, these problems may in reality be studied from either a static or dynamic perspective (Ghiani et al., 2003). Static vehicle routing problems (VRPs) assume that all relevant problem instance information or input data is known ahead of time and may be exploited in the solution process. On the other hand, the input data for dynamic (also known as on-line (Jaillet and Wagner, 2006) or real-time (Yang et al., 1999)) VRPs is revealed over time. In response, solution mechanisms are designed to give an answer on what decision to make, based on available information, in the face of an uncertain future. When the time between the problem-changing events is short, there may not be a lot of time to plan for or optimize decisions — there may only be time

---

<sup>1</sup>This chapter is based on Máhr et al. (2008, 2010).

to obtain a good feasible solution. Given the possibly sub-optimal nature of on-line solution mechanisms, one could argue that all on-line algorithms are in effect heuristics. As defined in Chapter 2, Section 2.1, heuristics are solution mechanisms designed to yield a feasible solution rapidly, without any guarantee on quality.

Dynamism, also referred to as uncertainty, can have different sources. The type of uncertainty classically studied in vehicle routing is new job arrival. Other types of uncertainty, often studied in the related field of scheduling, include variable activity durations or variable resource failures. Reviews of dynamism in the field of scheduling include Herroelen and Leus (2005) and Sgall (1998). For our study, we focus on two types of uncertainty - variable service times (similar to variable activity durations) and job arrivals revealed over time.

To summarize the position of our work in the literature, we examine two structurally distinct solution approaches — an optimization-based solution approach and an agent-based solution approach — for the dynamic truckload pick-up and delivery problem with time windows under two types of uncertainty. The structurally differentiating feature of the two solution approaches is the level of control — centralized versus decentralized. Specifically, an optimization-based solution approach, focusing on a single objective and using full system information, exemplifies centralized control. In comparison, agents, optimizing their own unique objectives using information they perceive and maintain locally, exemplify decentralized control. The following two subsections describe in greater detail both optimization-based and agent-based approaches to the Dynamic Vehicle Routing Problem (DVRP).

### **5.1.1 Optimization-based Approaches for Vehicle Routing**

When studying centrally controlled optimization-based approaches for vehicle routing, the role of a dispatcher serves as a natural metaphor. A dispatcher is responsible for using all known information (such as job pick-up and drop-off locations, job size, job time windows, fleet size, vehicle capacity, and vehicle locations, etc.) to develop, what they perceive to be, a very good (possibly

optimal) feasible routing of vehicles to service all jobs at the least cost. In practice, the one characteristic the dispatcher does not possess is clairvoyance. The dispatcher is unaware of most future demands or service times until they occur. Thus, previously optimized plans must be updated to accommodate new demands as they arrive to the system or longer/shorter service times as they are realized; possibly moving the previous plan to a point far from optimal.

Solving dynamic vehicle routing problems from a centralized perspective has been an active area in the literature for over 20 years. Psaraftis' 1988 survey of results in this problem domain is valuable as it clearly lists the primary differences between the static and dynamic versions of the VRP. Chief among these differences is the essential nature of time coupled with the need for information update mechanisms and fast computation times for solutions.

In general, solution approaches to the dynamic vehicle routing problem tend to be premised on innovative manipulations of the static version of the vehicle routing problem. These problems may depend on either stochastic or deterministic input data (Powell et al., 1995). Stochastic approaches are designed to exploit statistical information gleaned from data on past problem instances. Alternately, deterministic approaches use only known information to solve the routing problem at specific decision instances. In this study, neither the agent-based, nor the optimization-based approach has access to stochastic or forecasted data. This design decision was largely driven by the lack of data commonly found at drayage providers; without proper data, these drayage companies have little capability to appropriately calibrate a stochastic model. We therefore focus our literature review on deterministic approaches.

The decision instances used in deterministic approaches may be designated as the moment of a single job arrival or when a pre-set number of jobs arrive or a specified amount of time has passed. In this way no unknown information is assumed at each decision epoch. The method by which the problem is solved at each decision epoch is the main differentiator between the methods



described in the past ten years.

Regan et al. (1995, 1996), for example, propose a set of rule-based heuristics for load acceptance and assignment decisions. These heuristics are sometimes termed *incremental approaches* as they incrementally change the problem at each decision instance without fully re-solving the problem. The use of routing heuristics in an on-line setting stems from their history and success in an offline context. Both Cordeau et al. (2002) and Laporte et al. (2000) provide comprehensive surveys on routing heuristics. Two primary branches of classical heuristic approaches, as previously introduced in Chapter 2, Section 2.1, are the *constructive methods* and the *improvement methods*. From the first group, a classical example is the *insertion heuristic*, a polynomial-time heuristic without performance guarantees (Solomon, 1987). This algorithm is widely used to create an initial solution that is further optimized by improvement methods.

In the realm of improvement methods, Thompson and Psaraftis (1993) introduce cyclic transfers of jobs among vehicles as one way to improve an initial solution. In their method, a *b-cyclic k-transfer* shifts  $k$  jobs from each vehicle to the next one in a circular permutation of  $b$  vehicles. This is a very general framework that can express many different and complicated exchanges of jobs between vehicles. As a result, the space of cyclic transfers is too big to consider a full search. Subsequently, Breedam (1994) and Kindervater and Savelsbergh (1997) introduce simpler moves such as: load reallocation (moving loads from one vehicle to another), load exchange (exchanging loads between vehicles), and crossover (mixing the routes of two vehicles).

In contrast to these heuristic approaches, Yang et al. (2004) demonstrate the superiority of an exact mixed integer programming formulation of the PDPTW, solved in a rolling horizon framework at each decision instance. They compare their re-optimization approaches to three heuristic approaches (a simple round robin assignment, an insertion heuristic, and a reordering approach). This comparison reveals that the re-optimization approaches systematically outperform the heuristic approaches by about 10%. This superior re-optimization approach, has its origins in the paper by Yang et al. (1999).

Mahmassani et al. (2000) further this line of work by examining a hybrid rule-based heuristic and optimization approach. They find an improvement in performance occurs for this hybrid approach only when the frequency of job arrivals is low and the number of trucks is not too large. Most recently, Chen and Xu (2006) investigate a dynamic column generation technique as a means to handle a set-partitioning-type formulation at each decision epoch. They show, similar to Yang et al. (2004), that dynamic column generation on average outperforms the insertion heuristic by about 10%.

While re-optimization or plan/re-plan approaches are appealing as they tend to closely mirror manual operations (i.e. the situation where a dispatcher plans and re-plans routes as new jobs arrive), there are also drawbacks. Powell et al. (2000) highlight the myopic nature of these approaches. For example, optimal solutions implemented early in the day may no longer be optimal in light of additional information that arrives later in the day. Furthermore, the term “re-optimization” may be misleading as the solution to the exact mathematical programming formulation may only be feasible (rather than optimal) at each decision instance. This phenomenon can be exacerbated when the problem size is large as plan/re-plan approaches tend to suffer from the burden of rapidly responding to new information, especially in cases where an optimization-based algorithm is used.

In order to address these issues of large problem size, the idea of breaking the problem into component parts is appealing. Ghiani et al. (2003) provide a review of real-time vehicle routing strategies with a particular emphasis on parallel computing strategies. Their review focuses on different control and communication structures (e.g. master-slave, etc.) for efficiently searching the solution space arising from a centralized problem formulation. While they present a useful method for classifying parallel computing strategies for the VRP, they overlook computing possibilities that begin with a decentralized problem formulation. For example, it may be beneficial to reframe the vehicle routing problem based on roles within the problem, i.e. jobs and vehicles. Dividing the problem in this manner renders the solution approach into a decentralized approach. The following section describes decentralized agent

based approaches in the context of vehicle routing.

### 5.1.2 Agent-based Approaches for Vehicle Routing

Modeling a vehicle routing problem as a decentralized system is primarily motivated by the decentralized nature of the environment in which these problems occur. When a company needs to organize transportation for a set of jobs, it has to deal with the customers the jobs came from, the drivers who will execute the actual transportation, the legal and social environment of the company, and last but not least the company's need to make a profit. If one does not want to abstract away from this setting then a decentralized model considering all the players forms a natural metaphor.

Such a model is provided by multi-agent systems (MAS) (Wooldridge, 2002). Multi-agent systems consist of a group of autonomous decision makers (artificial agents) that are capable of interacting with each other, while pursuing a goal. If the agents share a common goal, they can cooperate to achieve it. When the goals are contradicting, the agents are competitive, they may hide sensitive information from each other, and try to achieve their goals individually. Applications of agent systems span from vehicle routing and logistics in general (Dorer and Calisti, 2005), through business process management (Hutzschenreuter et al., 2008), procurement and contracting (Jakob et al., 2008), aerospace applications (Scerri et al., 2008), energy management (Das et al., 2008), to security applications (Rehak et al., 2008).

In vehicle routing, a simple agent model may contain job agents and vehicle agents pursuing an assignment by minimizing some given objective. More complex models may include agents for the company, the planners, or for customers having more than one job. Interactions between the agents constitute a major part of the solution mechanism.

In their frequently cited paper, Fischer et al. (1995) argue that such multi-agent models fit the transportation domain particularly well. Their main reasons are (similar to those mentioned above): (i) the domain is inherently decentralized (trucks, customers, companies etc.); (ii) a decentralized MAS architecture can cope with multiple dynamic events; (iii) commercial companies

may be reluctant to provide proprietary data needed for global optimization while agents can use local information; and (iv) inter-company cooperation can be more easily facilitated by agents.

To illustrate their idea, the authors provide a detailed MAS architecture for transportation problems that evolve over time thereby exhibiting job arrival uncertainty. This architecture makes a distinction between a higher and a lower architectural level. At the higher level, company agents negotiate transportation requests to eliminate ill-fitting jobs. On the lower level, truck agents (clustered per company) participate in simulated market places, where they bid on offered transportation jobs. Truck agents use simple insertion heuristics to calculate their costs and use those costs to bid in auctions. Although the heuristics used by the truck agents to calculate bids are rather crude, Fischer et al.'s research (1995) suggest that in dynamic problems (problems with high uncertainty), such methods survive better than sophisticated optimization methods.

Their bi-level approach recognizes that one shortcoming of a fully decentralized system is that agents only have access to local information. The need to find a balance between the omniscience of a centralized model and the agility of a decentralized model, was similarly recognized by Mes et al. (2007). They also introduce a higher level of agents, but with a different role than the high-level agents of Fischer et al. (1995). Mes et al.'s two high-level agents (the planner and the customer agent) gather information from and provide information to agents assigned beneath them. The role of the higher level agents is to centralize information essential for the lower level agents to make the right decisions.

Some researchers have gone even further in proposing centralization in agent-based models. These researchers concentrate problem information in some agents for the purpose of making better decentralized decisions. In one of the few models that has been applied in a commercial company, Dorer and Calisti (2005) cluster trucks geographically, using one agent per cluster. This way, one agent plans for multiple trucks. They use insertion heuristics to initially assign jobs to trucks, and then use cyclic transfers (Thompson and

Psarafitis, 1993) to enhance the solution. In a similar but slightly different approach, Leong and Liu (2006) introduce a planner agent that has complete information about the other agents. In their method, planner agents coordinate the improvement procedures performed by the job and vehicle agents considering global objectives, such as the minimization of the number of vehicles used, and the total traveled distance. The authors analyze the performance of their model on a selection of Solomon benchmark sets, and show that it performs competitively on those sets.

As noted previously, however, the move towards centralization can hinder the ability of the agents to react quickly on local information. Given the uncertain environment of our problem, we are interested in the competitiveness of a system with fully decentralized agents. One example of a fully decentralized agent approach in the transportation domain is that of Buerckert et al. (2000). They propose a more detailed (holonic) agent model. In this model, they distinguish between truck, driver, chassis, and container agents that have to form groups (called holons) to serve jobs. Already formed holons use the same techniques to allocate tasks as Fischer et al., but the higher agent level is omitted, since they model only a single company case. The main focus of their research is computer-human cooperative planning, which makes their contribution interesting in spite of the fact that a thorough comparison of their approach to other ways of dealing with this problem is missing.

In most of the approaches mentioned above, agents use simple heuristic techniques to make decisions. In the related domain of production planning, Persson et al. (2005) embed optimization in the agents to improve local decisions. They show that optimizing agents outperform the heuristic agents, but they also show that central optimization still outperforms the optimizing, yet decentralized, agents. A similar result was demonstrated by Davidsson et al. (2007) on a problem of resource allocation in which decisions about both production and how much to send to each customer must be made simultaneously in the inherently uncertain domain of food production.

While Persson et al. (2005) and Davidsson et al. (2007) concentrated on making optimal decisions within agents, there is still a need to combine these

individual solutions in an optimal way. For example, in the transport problem context, when jobs are assigned to trucks sequentially, at every assignment the truck with the cheapest insertion gets the job. Later, however, it might turn out that it would have been cheaper to assign the same job, together with newly arrived jobs, to another truck. From the truck perspective this means that trucks that bid early and win assignments might not be able to bid later on more beneficial (better fitting) jobs. This problem is called *the eager bidder problem* (Schillo et al., 2002) and several researchers propose alternative techniques to deal with this problem. Kohout and Erol (1999) introduce an enhancement process between pairs of agents. The process mimics the well-known 'swapping' or two-exchange improvement techniques (Cordeau et al., 2001). Kohout and Erol implement this swapping process in a fully decentralized way, and show that it yields significant improvement.

Perugini et al. (2003) extend Fischer et al.'s contract-net protocol to allow trucks to place multiple possibly-conflicting bids for partial routes. These bids are not binding. Trucks are requested to commit to them only when one of the bids is accepted by a job agent. Since auctions are not necessarily cleared before other auctions are started, agents have a chance to "change their mind" if the situation changes. This extension helps to overcome the eager-bidder problem and thereby produces better results.

Another possible way to tackle the same problem is to use leveled commitment contracts, as introduced by Sandholm and Lesser (2001). Leveled commitment contracts represent agreements between agents that can be withdrawn. If a truck agent finds a new job that fits better, it can decommit an already committed job and take the new one. Hoen and La Poutré (2003) employ truck agents that bid for new jobs considering decommitting already assigned ones. Note, while decommitment sounds very negative, it does not necessarily imply that the job is rejected. In fact, the "decommitted" job may be able to negotiate an even more profitable contract with another truck. Thus, Hoen and La Poutré (2003) show that decommitment yields better plans in a single-company cooperative case.

Returning to Fischer's reasoning, however, the primary reason for using

decentralized agent solutions is that they are usually expected to outperform central optimization methods in problem instances with high levels of uncertainty. Researchers seemingly take this for granted and allocate their research efforts to demonstrating the value of their decentralized algorithm against other decentralized algorithms. Experiments studying the behavior of agent based methods over varying levels of uncertainty in comparison to optimization methods are generally absent from the literature. This is especially true in the logistics and transportation domain where the lack of appropriate comparisons between agent-based approaches and existing techniques appears to indicate a belief on the part of agent researchers that agent-based systems outperform traditional methods (Davidsson et al., 2005).

In this work, we dare to question this underlying assumption professed by agent enthusiasts. If advanced swapping and decommitment techniques are used, can fully decentralized agents really perform competitively with (or better than) centralized optimization in highly uncertain settings? Can the time gained in doing local operations compensate for the loss of not considering crucial global information? In our opinion these questions have not been fully answered. Our goal is to scrutinize these prevalent assumptions in the agent literature by studying an agent-based system in comparison to an optimization based approach for a real-world dynamic transportation problem. In the following section we describe our experimental design employed to perform this comparison.

## 5.2 Experimental Design

Through cooperation with a Dutch logistics service provider (LSP), we received the data required to test our solution approaches. The operations of the Dutch LSP were introduced in Chapter 1. Briefly, the LSP devotes approximately 40 trucks to the transport of approximately 65 containers per day between port terminals and customer locations, in such a way that time windows at both the customer and port terminals are obeyed. Given the number and geographic range of jobs, each truck can serve approximately two, and

maximally three, jobs per day. (Note, if the trucks were to serve only one job per day, then there would be no need for a planning system.)

Our objective is to compare the performance of a traditional optimization based approach to an agent based approach in determining a feasible set of routes for one day of execution within this PDPTW. In order to examine the advantages/disadvantages of these two methodologies, we used data from the LSP to generate experimental datasets exhibiting two sources of uncertainty (service time duration uncertainty and job arrival time uncertainty). The performance of each solution is compared in terms of empty distance traveled, the lost profit of rejected jobs, and the lateness of the vehicles in reaching each job pick-up, delivery, and return location. This section describes the two solution approaches, the datasets, and the sources of uncertainty.

### 5.2.1 Solution Approaches

The primary objective of the planning methods is to route a uniform fleet of forty trucks on the Netherlands' road network at lowest cost without violating time windows. Costs consist of time traveling empty plus the penalty for rejected jobs. Jobs may be rejected when they cannot be served within the time restrictions. The penalty for rejecting a job equals the loaded time of that job. The loaded time of a job is the time from the start of the pick-up action to the end of the return action — including all loading, unloading, and traveling time. This is an appropriate penalty for a rejected job as it represents the profit lost in not serving the job. Admittedly, rejecting a job may also yield a loss in customer good will or relations. Given the difficulty in quantifying this loss, we however choose to use the loaded time as a low estimate of the cost associated with job rejection. As we simulate only one planning day, in each instance, rejected jobs are simply rejected, although in practice they are reconsidered for service the next day. We now describe the mechanisms by which each solution approach solves this operational problem.



## Agent-based Solution Approach

In building our agent system, our goal was to define a fully decentralized solution approach, in which no information regarding different agents or different sub-problems is concentrated at a higher level. Additionally, we wanted to equip our agents with state-of-the-art mechanisms to successfully solve the routing problem.

Following the traditions of Fischer et al. (1995), we modeled every truck as an agent. But, instead of defining company agents on a higher hierarchical level to sell jobs to truck agents on auctions, we modeled the containers themselves as agents. Each container agent sells itself on an auction to the truck agents. Truck agents use an extended insertion heuristic to bid on the auctions, and a container-exchange heuristic to improve the current solution. Container agents also actively try to improve the current solution by a re-allocation heuristic.

**Auctioning Containers** Container agents organize auctions immediately after the container is announced to the planning system. If containers are revealed to the system at well-spaced intervals, then the auctions are fully sequential. If, however, multiple containers are announced at about the same time, those auctions are held in parallel. This parallelism is the result of our decision to model every container as an agent. This way, auctions are held by separate agents instead of a central company agent. Although this requires extra coordination to handle parallel auctions, introducing a central entity to hold sequential auctions would go against our design goal of having a totally flat MAS architecture.

The container agents collect quotes for transportation via these auctions. The bids truck agents send for auctions contain the cost they would incur should they win the auction and transport the container. This includes the *loaded time* of the container, plus the *extra costs* of driving to that container empty, and driving to the next container also empty. A container agent only accepts bids that are less than its reservation price. The reservation price is the sum of the loaded time and the rejection penalty of the container (thus

twice the loaded time). In this way, containers that have an extra cost, higher than their rejection penalty, are rejected.

Container agents implement a single-shot second-price closed-bid (a.k.a. Vickrey) auction. This auction type is popular in the literature because of its simplicity (Hoen and La Poutré, 2003). Not only is a Vickrey Auction simple to implement, but it also has a structure in which the optimal bidding strategy for each bidder is to bid their true value (Vickrey, 1961). We use this mechanism because, by setting the price to the second-best bid, the market position of the container is implicitly communicated to the winning truck. This information is used by truck agents in making decisions, as explained later. Having the second-best bid as the price also ensures that the truck agent realizes a profit. If the winner is the only truck agent bidding on the auction, or the second-best bid is higher than the reservation price, the container agent sets the price of the contract to the reservation price.

The winning truck agent can accept the contract only if its plan is unchanged since the time of the bidding. If the plan has changed in the interim, due to winning another container, for example, the truck agent must re-calculate the transportation costs for this container considering the new plan. If the new costs are less than or equal to the price in the contract, the truck agent accepts the second container. Otherwise it has to reject it, since transporting the container in this new situation costs more than the price it would receive for the job. If the winner rejects the container, the container agent will try to close a deal with the second, the third, etc. truck agents in a similar manner.

When a container agent succeeds in making a contract with a truck agent, it sends a message to the other containers to notify them about the changed state of the contracted truck agent. This information is crucial for rejected containers that can try to re-auction themselves in the hope that they will have some options now that the situation has changed. Every container agent has a latest possible auctioning time. After that time it is not possible to pick-up and transport the container within the given time windows. Container agents try to re-auction themselves only if this latest possible auction time has not

yet passed. To avoid an avalanche of auctions from rejected container agents every time a contract has been made, re-auctions take place at randomly chosen intervals with an exponential distribution and a mean value of one-tenth of the time remaining for the container to be successfully scheduled.

**Insertion with Substitution** When a truck agent bids on a container, its bid includes the additional cost it would incur should it win the auction and transport the container. To calculate this additional cost, a truck agent has to compute the difference between the cost of executing its plan with the new container included and excluded. In general, a truck agent needs to solve a TSP-like problem in order to find the optimal order of the containers in its plan. Similar to other agent based methods, our agent system uses a fast heuristic, rather than a full optimization scheme. The fast heuristic our truck agents employ has appeared in previous work (Máhr et al., 2008). For the convenience of the reader, we describe our insertion and substitution heuristic again here. Algorithm 5 summarizes the extended insertion algorithm.

In step  $i$  of our insertion and substitution heuristic, a truck agent computes the cost of inserting container  $k$  before container  $i$  and the cost of substituting container  $i$  by container  $k$ . The cost of inserting container  $k$  between container  $i$  and  $j$  is calculated as the difference of the empty-travel times with and without container  $k$ :  $ins_{ij}^k = d_{ik} + d_{kj} - d_{ij}$ , where  $d_{xy}$  is the time the truck travels empty from the drop off location of container  $x$  to the pick-up location of container  $y$ . Note,  $d_{xx}$  represents the loaded distance of container  $x$ ; that is, the distance from the pick-up location  $x$  to the drop-off location of  $x$ . The cost of substituting container  $i$ , which is between container  $h$  and  $j$  in the plan, with container  $k$  is defined as:  $subs_i^k = ins_{hj}^k + profit_{hj}^i$ . The first term, inserting container  $k$  between  $h$  and  $j$  is as defined above. The lost profit of container  $i$ ,  $profit_{hj}^i$ , is computed as the price offered for container  $i$  minus the loaded time and the insertion cost of container  $i$ . Thus,  $profit_{hj}^i = price^i - d_{ii} - ins_{hj}^i$ . The loaded time,  $d_{ii}$ , is subtracted because it is also part of the bids trucks submit. The insertion cost element of the lost profit is recalculated for every substitution decision, because the preceding or the

subsequent container of  $i$  may have changed since it was included in the plan. This algorithm is linear (considering the insertion of one new container into the plan of a single truck), but the solution found can be arbitrarily far from the optimal, which might only be found by fully reordering all containers.

In addition to calculating costs, our trucks also check time-window constraints, and do not accept any solution that violates time windows. The bid that is finally submitted to the container agent is the sum of the lowest insertion or substitution cost and the loaded time of the container.

The advantage of computing both insertion and substitution costs stems from the fact that insertion alone is very sensitive to the order of job arrivals. By allowing trucks to substitute earlier-committed containers, we reduce this ordering-sensitivity problem. Furthermore, the substitution cost expression we use supports the replacement of containers that had many similar competing bids. The price of such a container, computed from the second-best bid, is only a little bit higher than the associated costs. This means that the profit is low, therefore it becomes a good candidate for replacement. At the same time such a container can easily find another truck for a similar price, since there was at least one truck that bid very close to the winner. Along the same lines, the proposed calculation prevents the substitution of containers that could have difficulties in finding another truck for a similar price.

If a truck wins an auction where its bid corresponds to an insertion position, then it simply inserts the new container to the specified position. If the bid corresponds to a substitution position, then the truck first releases the container in that position and then inserts the new container to its place.

The released container starts a new auction just as it did following its arrival time. This new auction may, of course, result in another substitution, which invokes a new auction, and so on. The seemingly endless chain of substitutions is limited by the rule that truck agents accept containers for substitution only if they fit better (by at least  $\epsilon$ ) in their plan than the one that is to be replaced. This  $\epsilon$  can be chosen in such a way as to guarantee a bound on the length of substitution chains. For example, if  $\epsilon$  is chosen as a fraction of the maximum possible improvement, the length of the resulting

---

**Algorithm 5** Insertion and substitution of jobs

---

1. Iterate over the plan and collect the insertion and substitution costs corresponding to positions in the plan.
  2. Sort the merged list of insertion and substitution costs and positions by increasing order of costs.
  3. Iterate over the list of costs and positions, and
    - (a) if the position indicates a substitution and the insertion cost of the new container is not less than  $\epsilon$  less than the insertion cost of the container currently in the problem, then drop this alternative,
    - (b) if it is not possible to insert or substitute the new container at the given position without violating its time windows, then drop this alternative,
    - (c) if the time windows of the subsequent containers are violated by the insertion or substitution of the new container, then drop this alternative, or
    - (d) else return the position and the cost as the cheapest feasible insertion or substitution position.
-

substitution chain is bounded by a constant. With any choice of  $\epsilon$ , the end of a substitution chain is either an insertion, or rejection, which occurs if all truck agents reject the container agent in the auction due to time-window infeasibility, or if all bids are higher than the reservation price of the container.

In addition to the basic task of bid calculation, both the container and truck agents are endowed with additional techniques to improve the initial solution.

**Random Reallocation** Whenever the plan of a truck agent changes, in principle any container agent has a chance that the truck agent with the new plan can now transport it at a lower cost than their current truck agent. Rejected container agents receive notification about such events. Therefore they can try to close a deal with the truck agent with the changed state. Since it is also useful for container agents that already are in possession of a contract to try to find better options, they use a randomized algorithm to search for those options.

Every container agent has a timer that fires in exponentially distributed random intervals with a mean value of  $\mu_r$ . Whenever the timer goes off and the container agent has a contract, it tries to reallocate itself. Reallocations are randomized this way to minimize the chance of two re-auctions happening at the same time. In our simulations, with 65 container agents per day, we selected  $\mu_r$  to be one hour. This yields approximately one reallocation per minute. To commence reallocations, the container agent, whose timer has fired, sends a message to the truck agent, with which it has a contract, in order to prevent the truck from transporting it. The truck agent puts the container *on hold*, and reports the current insertion cost of the container to the container agent. Then the container agent re-auctions itself among the other truck agents to see if any of them offers a better price than its current insertion cost. The container agent only accepts new offers that are better than this limit. If the container agent finds a cheaper option, it breaks its current contract and makes a new one with the new winning truck agent. If not, it sends a message to the current contracted truck agent in order to

remove the hold, allowing the truck to transport it. The steps of this algorithm are summarized in Algorithm 6.

---

**Algorithm 6** Reallocation

---

1. The container agent sends a message to its contracted truck agent to prevent the truck to transport it.
  2. If the transportation of the container has not started yet, the truck agent puts the container on hold, and sends back its current insertion cost.
  3. The container agent re-auctions itself among the other truck agents and collects offers that are better than its current insertion cost.
  4. The container agent sorts the list of collected offers from best to worse, and iterates through the list.
    - (a) Notify the truck agent that it won the auction.
    - (b) If the truck agent accepts the new contract, the container agent notifies the previous truck agent that it leaves. The previous truck agent removes the container from its plan, and the algorithm stops.
  5. If there are no offers left (or the list was empty because no new offer was better the current one) the container agent sends a message to the currently contracted truck to remove the on-hold status.
- 

By periodically attempting a reallocation, container agents check new possibilities that arose from changes in the plans of trucks since the last auction. In our agent system, truck agents also actively try to improve the solution. The next section discusses a decentralized algorithm that exchanges containers between pairs of trucks.

**Exchange of Containers** An efficient way to improve solution quality in vehicle routing problems is to try to exchange jobs between trucks as noted in Section 5.1. Our decentralized algorithm of container exchanges searches a neighborhood of 2-cyclic  $k$ -exchanges in a randomized fashion. Like container agents, truck agents also have a timer set to fire at exponentially random

intervals with a mean value of  $\mu_e$ . For similar reasons explained at the reallocation algorithm,  $\mu_e$  is chosen to be one hour in our experiments. When a truck agent's timer goes off, it initiates a container-exchange procedure. It first checks if there are any containers in its plan that are not executed yet. These containers are available for exchange with another truck. The truck agent puts the first exchangeable container on hold, to prevent it from being executed. Then it copies the relevant part of the plan and sends it to another truck. In principle, truck agents can apply sophisticated heuristics to choose a partner truck. Geographical coordinates, personal preferences of trucks, or business considerations of the company can be taken into account. However, these heuristics rely on the availability of appropriate information regarding all (or a subset of the) trucks. Maintaining such information implies the aggregation or centralization of data. As one of our goals was to design and test an agent system in which no information is concentrated, our truck agents do not discriminate in choosing a partner truck; selection of a partner truck is made randomly.

The truck that receives the “not-yet-executed” chunk of plan, produces a similar sub-plan from its own plan. It also puts the first exchangeable container on hold, and then performs a full search on the  $k$ -exchange neighborhood of the two plan segments. First it tries to exchange single containers, then chains of two, three, etc. containers in any combination. In the case of our test instances, the plan segments are never longer than three containers. This search returns the exchange combination of containers that yields the highest saving for the *combination* of these two trucks. If a solution is found, it is reported back to the initiator truck.

If the initiator receives a certain container-exchange combination from the responder truck, it implements the exchange in its plan, and sends new contracts to newly assigned container agents. The last step of the initiator is to send the expired contracts of the exchanged containers to the responder truck. To conclude the procedure, the responder also implements the container exchanges in its plan, and sends new contracts to the newly received container agents. The container-exchange algorithm can thus be explained as a four-



step negotiation between the initiator and the responder trucks. Algorithm 7 summarizes all the steps.

---

**Algorithm 7** Container Exchange

---

**The Initiator** truck

1. puts the first exchangeable container on hold,
2. copies the segment of its plan that contains only exchangeable containers, and
3. sends this segment to a randomly chosen other truck.

**The Responder** truck

1. also puts its first exchangeable container on hold,
2. makes a copy of the exchangeable part of its plan,
3. performs a full search on the  $k$ -exchange neighborhood of the two plan segments, and
4. sends back the best exchange combination to the initiator, if one is found that leads to better plans.

**The Initiator** truck

1. implements the proposed exchange,
2. sends new contracts to the newly acquired container agents, and
3. sends back the expired contracts of the exchanged container agents to the responder.

**The Responder** truck

1. implements the proposed exchanges in its plan, and
2. sends new contracts to the newly acquired container agents.

---

The above described container-exchange algorithm implements a randomized search on the 2-cyclic  $k$ -exchange neighborhood of the container transportation problem. The search is performed in a decentralized fashion, and it requires cooperative truck agents. Truck agents are required to reveal parts of their plan to other truck agents. The responder agent searches for an ex-

change that maximally decreases the costs of the two truck agents together. Such considerations prohibit the direct application of this algorithm in a competitive agent system.

We now consider an instance with  $N$  jobs and  $K$  trucks in order to analyze the worst-case runtime performance of the agent-based heuristic. In the insertion and substitution algorithms, computing the insertion and substitution costs is linear in the number of containers ( $N$ ), ordering the costs is  $O(N \log N)$ , and checking the time constraints is  $O(N^2)$ . The container exchange algorithm consists of a full search of the exponential  $k$ -exchange neighborhood, but we limit the size of the neighborhood by three (in our experiments because this is the maximum number of containers a truck can transport per day). Limiting the size of the neighborhood also limits the runtime, thus the algorithm runs in quasi-linear time.

During the auctions, all  $K$  trucks submit a bid, the bids are computed in  $O(N^2)$  time, and clearing an auction is polynomial in the number of trucks ( $O(K \log K)$ ). The run-time of this approach thus mainly depends on the number of auctions. Auctions are held for three different reasons.

First, there are auctions that container agents organize when they arrive. In the worst case, should they all have their first auction at the same time, only  $K$  contracts can be made. Then, if the second round of  $N - K$  auctions is synchronised again, it yields only  $K$  contracts as well. The sum of all such auctions is  $N + (N - K) + (N - 2K) + \dots$ . The number of extra rounds needed after the first one is  $m = \lfloor \frac{N}{K} \rfloor$ . The total sum is  $(m + 1)N - \frac{m(m+1)}{2}K$ , which is in the order of  $(O(\frac{N^2}{K}))$ .

Second, there are auctions caused by substitutions. The bound on the length of substitution chains depends on the  $\epsilon$  parameter of Algorithm 5. In our experiments, we chose  $\epsilon$  to be small enough to allow any substitution that strictly improves the solution, but dependent on the maximum possible cost of a container. Consequently, the number of auctions in a substitution chain is limited by a constant  $C$ . The number of substitution chains is at most the number of successful initial auctions, thus in the worst case there are  $CN$  auctions caused by substitutions.

Finally, auctions may be held during a reallocation attempt. These happen a constant number of times because such attempts are made periodically regardless of the number of containers. The worst case is that each of these auctions start a substitution chain, which generates auctions in the order of  $O(N)$ . To summarize, the number of auctions is polynomial ( $O(\frac{N^2}{K})$ ) in the number of containers. Considering the complexity of the truck bidding algorithms and the number of auctions, we can conclude that the agent solution runs in polynomial time on the order of ( $O(\frac{N^4}{K} + N^2 \log K)$ ). While this runtime might look particularly bad for a heuristic, it should be noted that this worst-case is highly dependent on the timing of the auctions. Thus, in general instances, where the auctions are far more asynchronous, the agent runtime is actually quite reasonable.

The insertion, substitution, random reallocation, and the exchange of containers all contribute to the quality of the routes produced for each truck. In particular, these methods only change the plan if doing so leads to a strict decrease of the costs. Each of these algorithms has been used previously in multi-agent transportation planning systems, but to the best of our knowledge, their combination is unique. From this we conclude that the approach presented here is a very good representative of agent methods, and as such is a good candidate for comparison to a centralized optimization-based approach.

### **Optimization-based Solution Approach**

Our objective in selecting a comparative approach, to the decentralized agent based approach, was to choose a proven approach that depends on the centralization of full system information. Furthermore, we wanted this centralized approach to be recognized as state-of-the-art and perform better than other centralized approaches. As such, we selected a solution approach based on a mixed integer program (MIP) for a truck-load pick-up and delivery problem with time windows originally put forth by Yang et al. 1999. Specifically, as noted in Section 5.1, this model provides competitive results in comparison to fast running heuristics (Yang et al., 2004).

The on-line optimization approach used in our experiments works by solv-

ing the MIP, with CPLEX<sup>2</sup>, at 30-second intervals. We selected 30-second intervals for two reasons. First, this interval provides a good tradeoff between problem size and solver runtime. For example, if the interval is lengthened then CPLEX gains more time to run, but simultaneously the problem size increases as more jobs will arrive in the intervening time. Thus, 30-seconds tends to ensure that only one or two jobs have arrived since the last feasible solution was implemented. Our second reason for selecting 30-seconds is to remain competitive with the event driven agent-based solution approach. As the on-line optimization approach may only find a feasible (not necessarily optimal) solution in each 30-second interval, this approach acts more like a heuristic in the on-line context. We therefore believe that the centralized on-line optimization approach makes a good comparison to the decentralized agent approach. The complete description of our modifications to and use of Yang et al.'s MIP in our on-line optimization scheme is the focus of this section.

**The Mixed Integer Program** Before introducing the notation and mathematical formulation for this problem, we begin with a small example to illustrate exactly how Yang et al.'s MIP works to exploit the structure of this truckload pick-up and delivery problem with time windows. Imagine a scenario with three trucks and four jobs. The model of Yang et al. is constructed such that it will find a set of least cost cycles describing the order in which each truck should serve the jobs. For example, as depicted in Figure 5.1, the outcome may be a tour from truck 1 to job 1, then job 2, then truck 2, then job 3, then back to truck 1. This would indicate that truck 1 serves job 1 and 2, while truck 2 serves job 3. The cycle including only truck 3 indicates that truck 3 remains idle. Similarly, the cycle including only job 4 indicates that job 4 is rejected.

Given this problem description, we designate the following notation for the given information.

---

<sup>2</sup>CPLEX is a commercially available solver package, specifically we used ILOG CPLEX 11.0 as provided by ILOG Inc.(ILOG, Inc., 1992)

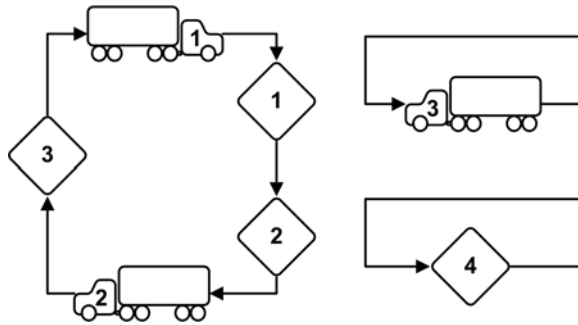


Figure 5.1: Cycles in the MIP solution structure.

- $K$  the total number of vehicles available in the fleet.
- $N$  the total number of known demands.
- $d_{ij}$  as introduced in 5.2.1, the travel time required to go from demand  $i$ 's return terminal to the pick-up terminal of demand  $j$ . Note, if  $i = j$  then the travel time  $d_{ii}$  represents the loaded distance of job  $i$ ; this distance includes the time from pick-up at the originating terminal to completion of service at the return terminal.
- $d_{0i}^k$  the travel time required to move from the location where truck  $k$  started to the pick-up terminal of demand  $i$ .
- $d_{iH}^k$  the travel time from the return terminal of demand  $i$  to the home terminal of vehicle  $k$ .
- $v^k$  the time vehicle  $k$  becomes available.
- $\tau_i^-$  earliest possible arrival at demand  $i$ 's pick-up terminal.
- $\tau_i^+$  latest possible arrival at demand  $i$ 's pick-up terminal.
- $M$  a large number set to be  $2 \cdot \max_{i,j} \{d_{ij}\}$ .

Note that  $\tau_i^-$  and  $\tau_i^+$  are calculated to ensure that all subsequent time windows (at the customer location and return terminal) are respected.

Given the problem of interest, we specify the following two variables.

- $x_{uv}$  a binary variable indicating whether arc  $(u, v)$  is used in the final routing;  $u, v = 1, \dots, K + N$ .
- $\delta_i$  a continuous variable designating the time of arrival at the pick-up terminal of demand  $i$ .

Using the notation described above, we formulate a MIP that explicitly permits job rejections, based on the loaded distance of a job.

$$\begin{aligned} \min \quad & \sum_{k=1}^K \sum_{i=1}^N d_{0i}^k x_{k,K+i} + \sum_{i=1}^N \sum_{j=1}^N d_{ij} x_{K+i,K+j} \\ & + \sum_{i=1}^N \sum_{k=1}^K d_{iH}^k x_{K+i,k} \end{aligned}$$

such that

$$\sum_{v=1}^{K+N} x_{uv} = 1 \quad \forall u = 1, \dots, K + N \quad (5.1)$$

$$\sum_{v=1}^{K+N} x_{vu} = 1 \quad \forall u = 1, \dots, K + N \quad (5.2)$$

$$\delta_i - \sum_{k=1}^K (d_{0i}^k + v^k) x_{k,K+i} \geq 0 \quad \forall i = 1, \dots, N \quad (5.3)$$

$$\begin{aligned} \delta_j - \delta_i - M x_{K+i,K+j} + \\ (d_{ii} + d_{ij}) x_{K+i,K+i} & \quad \forall i, j = 1, \dots, N \quad (5.4) \\ \geq d_{ii} + d_{ij} - M \end{aligned}$$

$$\tau_i^- \leq \delta_i \leq \tau_i^+ \quad \forall i = 1, \dots, N \quad (5.5)$$

$$\delta_i \in \mathbb{R}^+ \quad \forall i = 1, \dots, N \quad (5.6)$$

$$x_{uv} \in \{0, 1\} \quad \forall u, v = 1, \dots, K + N \quad (5.7)$$

In words, the objective of this model is to minimize the total amount of time spent traveling without a profit generating load. Specifically, we wish to minimize the penalty incurred from rejecting jobs, time spent traveling empty to pick up a container, between containers and when returning to the home depot. This objective is subject to the following seven constraints:

**(5.1)** Each demand and vehicle node must have one and only one arc entering.

**(5.2)** Each demand and vehicle node must have one and only one arc leaving.

**(5.3)** If demand  $i$  is the first demand assigned to vehicle  $k$ , then the start time of demand  $i$  ( $\delta_i$ ) must be later than the available time of vehicle  $k$

plus the time required to travel from the available location of vehicle  $k$  to the pick up location of demand  $i$ .

- (5.4) If demand  $i$  follows demand  $j$  then the start time of demand  $j$  must be later than the start time of demand  $i$  plus the time required to serve demand  $i$  plus the time required to travel between demand  $i$  and demand  $j$ ; if however, demand  $i$  is rejected, then the pick up time for job  $i$  is unconstrained.
- (5.5) The arrival time at the pick up terminal of demand  $i$  must be within the specified time windows. (Note, this constraint prevents a truck from arriving early or arriving late to a demand  $i$ .)
- (5.6)  $\delta_i$  is a positive real number.
- (5.7)  $x_{uv}$  is binary.

Mathematically this model specification serves to find the least-cost (in terms of time) set of cycles that includes all nodes given in the set  $\{1, \dots, K, K+1, \dots, K+N\}$ . We define  $x_{uv}$ , ( $u, v = 1, \dots, K+N$ ) to indicate whether arc  $(u, v)$  is selected in one of the cycles. These tours require interpretation in terms of vehicle routing. This is done by noting that node  $k$ , ( $1 \leq k \leq K$ ) represents the vehicle  $k$  and node  $K+i$ , ( $1 \leq i \leq N$ ) corresponds to demand  $i$ . Thus, each tour that is formed may be seen as a sequential assignment of demands to vehicles respecting time window constraints.

The model described above is used to provide the optimal (yet realistically unattainable) lower bound for each day of data in the experiments concerning job arrival uncertainty. We denote this approach as the static *a priori* case. In this case, we obtain the route and schedule as if all the jobs are known and we have hours to find the optimal solution. Thus, not only is this lower bound realistically unattainable due to a relaxation on the amount of information available, but also due to a relaxation on the amount of time available for CPLEX to obtain the optimal solution.

**Optimization in an On-line Context** In order to provide a fair comparison with the agent-based approach, the MIP is manipulated for use in on-line operations. In our on-line approach, this MIP is invoked at 30 second intervals. At each interval, the full and current state of the world is captured, and then encoded in the MIP. This “snapshot” of the world includes information on all jobs that are available and in need of scheduling, as well as the calculated next available location and time of all trucks. The MIP is then solved via a call to CPLEX, which is initialized with the previous feasible solution. In this way, if nothing has changed since the last decision point, the optimization can continue and the solution can improve.

The decision to use 30 second intervals was driven by the desire to be comparable to the agent-based approach while still providing CPLEX enough time to find a feasible solution for each snapshot problem. While this interval may seem extremely short for finding a feasible solution, it is not as damaging as one might think. First, in the baseline dataset, we see that even with all of the jobs arriving in one clump at the start of the day, the on-line optimization rejects no jobs. Second, in most datasets, the snapshot problem being solved at each decision instance is significantly smaller than the full problem size.

The solution given by CPLEX, at the end of the 30-second interval, is parsed and any jobs that are within two intervals (i.e. 60 seconds) of being late (i.e. missing the time specified by  $\delta_i$  in the latest plan), if travel is not commenced in the next interval, are permanently assigned. Any jobs that were designated for rejection in the solution are permanently rejected only if they are within two intervals of violating a time window; otherwise they are considered available for scheduling in a subsequent interval.

If CPLEX cannot find an initial feasible solution in any one interval then the plan from the last feasible interval is invoked and parsed to fix assignments and rejections as described. If CPLEX cannot find an initial feasible solution for three consecutive intervals (90 seconds) then one job is selected for rejection based on the following hierarchy:

1. a job that arrived since the last feasible plan was made.



2. a job with a loaded distance less than or equal to 13,000-seconds. (Note, 13,000-seconds represents a job with a limited amount of loaded distance in the context of our data.)
3. a job that is 30 minutes away from the end of its time window.
4. a random job.

The procedure continues solving problem instances and parsing solutions in this fashion until the end of the working day at which point all jobs have been served or rejected.

### 5.2.2 The Data

In this subsection, we describe how our data was inspired and fed by the operations of the Dutch LSP. Recall that the LSP is transporting comparatively high-value reefer containers. As such, the trucks always wait at customer sites for the containers to be (un)loaded, and they never exchange containers. We therefore handle each pick-up, delivery, and return sequence as one job. Note, more than one job starts and ends at the same terminal locations. (Recall the drayage structure described and analyzed in Chapter 3.) Moreover, some customers have more than one job serviced in a day. Nevertheless, we handle each job separately, as if they all belonged to different customers. Each job is specified by two data vectors — one spatial and one temporal.

The spatial vector contains the location of the pick-up terminal, the customer site, and the return terminal. This data was derived from a set of operational data tables provided by the LSP. In all, we were given data from January 2002 to October 2005 as well as from January 2006 through March 2006. The tables represented jobs that were planned to be served on a given day. Unfortunately, the exact timing of the jobs each day was nearly absent from the data. Further problems were presented by some of the addresses that referred to postal boxes instead of real customer or terminal locations; therefore these had to be pruned from the data. Nevertheless, after a preliminary review of the data, we could conclude that on average 65 jobs were served

in a day, at customer and terminal locations associated with less than 25 distinct zipcodes. The rare timing information suggested that the jobs were served uniformly throughout the day. Using these parameters, we extracted a random sample of appropriately defined jobs from the original data-set in order to generate a set of 33 days with 65 jobs per day using the locations in the sample. Note, we consider each day as a single instance. Thus, there are no jobs that persist in the planning system from on day (or instance) to the next. Figure 5.2 depicts the geography of the Netherlands and the full set of locations represented in our data.

The temporal vector is comparatively more complex — containing three data types: data on time windows, data on service times, and data on job arrival. The data on time windows includes the terminal operating time windows and the customer time window. The data on service times includes the service time required at the three job locations. Finally, the data on job arrival includes one element — the time the job is announced in the planning system. As mentioned earlier, such timing information was sparsely recorded in the data tables. Therefore this part of the job descriptions was entirely generated based on the experiences of the human planners.

To standardize the data for our experimental purposes, we specified time windows at all locations as follows: terminals are open for pick up between 6am and 6pm, and for return between 6am to 5:59am on the next day. The wide return time windows reflect the practice that trucks can bring containers to the terminals on the following day, if they were too late on the same day. These time windows are the same for all jobs. Delivery time windows are set to two hour intervals, and their start times are distributed uniformly over the working day between 8am and 5pm. Figure 5.3 displays the number of open time windows for all jobs at any time point of the working day. Since time windows open regularly and stay open for two hours, the number of open time windows gradually builds up and reaches the top between 12am and 2pm. After that, the number of open time windows, and therefore the number of jobs requiring service before the end of the day decreases.

The service time data type refers to the time trucks need to complete



Figure 5.2: All locations in the Netherlands. Black markers indicate customer locations; grey markers indicate terminal locations; and the white marker indicates the home terminal of the LSP

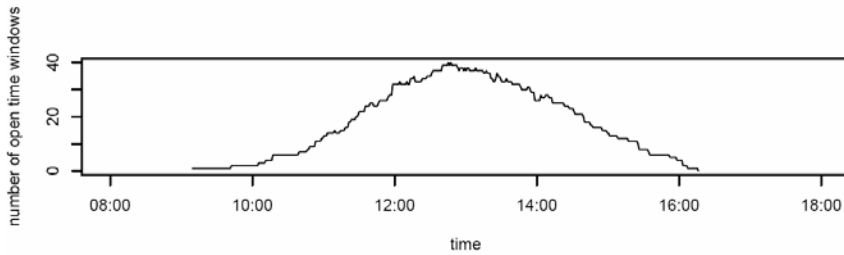


Figure 5.3: Number of open time windows for all jobs throughout the working day.

service at the different locations. When a truck arrives at a sea terminal or a customer, it spends some time to pick up, to deliver, or to return a container. The length of this time depends on various factors. Picking up a container for example, can be delayed by customs clearing, paperwork, or problems with putting the container on the truck. Emptying a container at the customer can be quick if the customer is ready to unload the goods, but it can be delayed if a warehouse is very busy. Similarly, when a container is returned, technical issues may delay the trucks. In discussions with the LSP, it seems that the human planners, by experience, allocate one hour for picking up, one hour for delivering, and half an hour for returning a container. As such, the baseline dataset (referred to as  $R_0$ ) sets all service time values to these times for all jobs.

The job arrival data type refers to the time a container is made known to the planning system. Before this time, the planning methods do not know about the job, after this time, the locations and the time windows are revealed. In the baseline dataset all job arrivals occur at the start of the day, 6am.

In all instances that we generated, we added a homogeneous fleet of 40 trucks starting at a base location close to Rotterdam. Although the number of trucks used by human planners varies each day, we chose to use 40 trucks, because this proved to be enough to solve each problem *a priori*, in the baseline dataset. Using more trucks would not yield different results. Using fewer trucks would yield a higher rejection rate, with possible differences in the kind of jobs rejected by the two methods.

Within the baseline dataset, each job requires, on average, approximately 4.2 hours of loaded distance. When the routing is optimal (in the *a priori* baseline case in which all jobs are known at the start of the day and all service times equal their expected values) the average empty time per job is approximately 25 minutes (or 27 hours total per day).

### 5.2.3 Uncertainty Scenarios

The objective of this work is to determine how the two solution approaches perform on the given pick-up and delivery problem with time windows under different types of uncertainty. The three main sources of uncertainty encountered by the drayage company are: service time uncertainty, travel time uncertainty, and job-arrival time uncertainty. The complexity of pursuing travel time uncertainty in a correct way, made us decide to use a deterministic estimate of the travel time, and focus only on uncertainty regarding service-time and job arrivals. Note, however, that when periodic changes in the travel time are included in the simulations and taken into account by both approaches, it should not influence the results. The main difficulty in simulating with uncertain travel times is the problem of identifying which trucks, traveling which routes, are subject to the variation. Nevertheless, up to a certain extent, such travel-time uncertainty is not that different from our model of service-time uncertainty in the case where service times are dependent upon each other.

#### Scenarios with Service-Time Uncertainty

In reality service times vary, forcing plans to be updated in real-time (or more generally, after every pick up, delivery, or return action). To simulate this source of uncertainty, we define different service times for every pick-up, delivery, or return action drawn from a uniform random distribution. This exact information is, however, hidden from the planning methods, which consider the mean service times as derived from human experience, and encounter the variable service times only during execution. Service time uncertainty may be viewed from two perspectives. The first perspective is that incidents affecting

Table 5.1: Summary of variation and corresponding bounds on service times for service time uncertainty scenarios.

| Experiment | Bound on Pick-up and Delivery Service Times | Bound on Return Service Times |
|------------|---------------------------------------------|-------------------------------|
| 10 minutes | [50min, 70min]                              | [20min, 40min]                |
| 20 minutes | [40min, 80min]                              | [15min, 50min]                |
| 30 minutes | [30min, 90min]                              | [15min, 60min]                |
| 4 hours    | [30min, 5hours]                             | [15min, 4.5hours]             |

service times are random and occur in a uniformly distributed way across the day and across terminal and customer locations (*independent*). The second perspective comes from recognizing that service time disruption events are most likely clustered in both time and location (*dependent*).

In scenarios with independent service times, we generated durations for all actions as random variables drawn independently from a uniform distribution. The boundaries of the uniform distributions for each scenario were defined with the intention to generate four different scenarios with service-time variations of 10, 20, 30 minutes, and 4 hours. The 10, 20, and 30 minutes are common delays according to the LSP; 4 hours is included to view any effect in exaggeration. Table 5.1 summarizes the bounds on the service times for each service action type as well as the nominal values used to denote the scenarios. Note that the lower bound of the return actions was not allowed to fall below 15 minutes. This was based on the assumption that returning a container can never be quicker than 15 minutes. Similarly, in the most extreme case, the lower bounds on pick up and delivery actions were minimized at 30 minutes. Furthermore, it is important to note that regardless of the bounds, the planning systems always planned using the mean values derived from the experience of the human planners at the LSP (i.e. one hour for pick up, one hour for delivery, and a half hour for return).

In addition to generating service times for all actions independently, we also considered scenarios where service times at the same locations are always the same. To achieve this, we generated service times for every location in an

instance independently according to a uniform distribution. Then, we always assigned the same service times to actions happening at the same locations. The scenarios were generated according to the same variations as in the independent case (summarized in Table 5.1). This resulted in four dependent service-time variations scenarios:  $Sd_{600}$ ,  $Sd_{1200}$ ,  $Sd_{1800}$ , and  $Sd_{14400}$ .

The motivation for the scenarios with dependent service times stems from practice. For example, if the cause of a delay at a sea terminal is a faulty crane then it will influence all trucks visiting that terminal. Note, such dependent delays have an effect similar to traffic jams, where trucks traveling toward the same location (and using the same roads) suffer from the same traffic conditions.

Considering this type of uncertainty in the context of the solution approaches, we note that when a truck agent experiences a change in its plan due to the actual service times, it tries to adjust the plan to the new situation. If simply shifting the remaining containers in time solves the problem, then that is done. If any subsequent containers become infeasible due to time-window violations, the truck agent removes the infeasible containers from its plan. The removed containers experience a removal similar to a substitution, and they start new auctions to find another truck agent. In this way, service-time variation is handled and the agents continue their improvement efforts as before.

As a comparison, the on-line approach has a more basic way of handling service-time variation. Whenever an actual service time is revealed, the planner adjusts the timings in the plan of the truck in question. Jobs that are already permanently assigned in a plan are not put back into the pool of unplanned jobs — even if they will now be late. In this way job rejections are minimized at the expense of being late. This is distinctly different than the *a priori* optimal which, with access to the service time data and in accordance with the constraints, will instead reject the jobs that will be late. As a result, a comparison of the on-line and *a priori* optimal solutions, in the case of service time uncertainty, would be inappropriate or unfair. Given these differences we only compare the agent-based and on-line optimization

solution approaches for the service time uncertainty experiments ignoring the *a priori* lower bound.

Considering, the meaning of these two methods of handling service time uncertainty in the real-world, we are immediately confronted with policies regarding hours-of-service. Hours-of-service laws specify the number of consecutive hours that any one truck driver can work. Regarding such policies, the agents behave in a way that would avoid violating these laws. However, after a point (i.e. in the most uncertain case, 4hours) each truck will serve only one job per day. This, in turn, renders the need for planning obsolete. The on-line optimization, on the other hand, supports the need for planning, but in the extremes cases will lead to a violation of hours-of-service. Admittedly, we can counter this violation by noting that it is the truck and not necessarily the driver that is in service.

### **Scenarios with Arrival-Time Uncertainty**

In addition to service time uncertainty, human dispatchers are often faced with a great deal of uncertainty regarding the time when a container may enter the planning system. Although the load may be known to the transportation company beforehand (e.g. from ship arrival data), the exact moment that the container will actually be offloaded is often unknown. Some containers are handled on a previous day, or during the night, while others may become available only in the afternoon.

In every problem instance, we defined a job-arrival time for each container. The earliest job arrival was at 6am; containers with such an arrival time we call static jobs. We call them static because they are actually known to the planning systems when planning starts for the day. Containers with an arrival time later than 6am we call dynamic jobs, referring to the fact that the planning systems need to incorporate them into the plans during execution. When a container is randomly selected to be dynamic in a certain problem instance, we set its arrival time to exactly two hours before the start of its customer time window (i.e. four hours before the end of the customer location time window, leaving slightly less than two hours on average before the latest



departure time from the pick-up location). This choice was made to ensure that the planning methods were confronted with time pressure. On average, the distance to the time window cannot be shorter than two hours, because it would render the instances infeasible. It could be longer, but then the resulting instances would be easier to solve, since there would be ample time for planning.

We generated five different scenarios with varying levels of job-arrival uncertainty. The varying levels of uncertainty were expressed in the percentage of dynamic jobs present in the instances. The baseline dataset represents the zero percent scenario ( $R_0$ ), where all jobs are known at the start of the working day, 6am. In the 25% scenario ( $R_{25}$ ), one quarter of the jobs (selected randomly from the 65 containers) arrive two hours before their customer time window, and the rest are static. In the 50% scenario ( $R_{50}$ ) half of the jobs are static and half of them are dynamic. In the 75% scenario ( $R_{75}$ ), one quarter of the jobs are static. Finally, in the 100% cases ( $R_{100}$ ) all containers arrive in a dynamic fashion.

The agents implicitly handle job-arrival uncertainty; they do not start their first auction before a job's designated arrival time. In the on-line optimization approach new arrivals are taken into account in the next epoch. For the job arrival uncertainty scenarios, we compare all three solution approaches (*a priori* optimal, on-line optimization, and agent-based).

### Scenarios with Arrival-Time and Service-Time Uncertainty

Having new jobs arriving during the day, and experiencing variations in service times are both quite common in practice. In instances with service-time uncertainty (denoted  $S_*$  and  $Sd_*$ ), all containers are static (they all arrive at the beginning of the day). In those instances the only source of uncertainty is service-time variations. Similarly, in instances with dynamic job arrivals (in  $R_*$  instances), all service times experienced during execution correspond to the average service times. To study the combined effect of both sources of uncertainty, we defined scenarios where dynamic job arrivals and variable service times are both present.

Table 5.2: Scenarios with different sources of uncertainty

| Scenario Title                                                   | Scenario Description                                                                                      |
|------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| $S_{600}, S_{1200}, S_{1800}, S_{14400}$                         | Scenarios with 10, 20, 30 minutes, and 4 hours independent service time variations.                       |
| $Sd_{600}, Sd_{1200}, Sd_{1800}, Sd_{14400}$                     | Scenarios with 10, 20, 30 minutes, and 4 hours dependent service time variations.                         |
| $R_0, R_{25}, R_{50}, R_{75}, R_{100}$                           | Scenarios with zero, 25%, 50%, 75%, and 100% dynamic jobs.                                                |
| $R_{50}S_{600}, R_{50}S_{1200}, R_{50}S_{1800}, R_{50}S_{14400}$ | Scenarios with 50% dynamic jobs, and 10, 20, 30 minutes, and 4 hours independent service time variations. |

In conversations with the planners at the LSP, it was revealed that typically 50% of the containers served in one day are dynamic. Therefore, we based our combined scenarios on the  $R_{50}$  instances in which 50% of the containers are dynamic.

We generated four variations of the  $R_{50}$  instances with 10, 20, 30 minutes and 4 hour service time variations. The resulting scenarios are denoted as  $R_{50}S_{600}$ ,  $R_{50}S_{1200}$ ,  $R_{50}S_{1800}$ , and  $R_{50}S_{14400}$  respectively. Consequently in these scenarios, in addition to 50% of the containers being dynamic, durations of all service actions are varied independently according to the same uniform distributions as the independent service-time variation case explained in Section 5.2.3. Table 5.2 summarizes all the scenarios and notation as used in this chapter.

### 5.3 Results

The basis on which both methods are compared across all datasets is total routing cost. Specifically, total routing cost, as defined here, is the sum of the amount of time spent traveling empty, the penalty affiliated with rejecting loads, and the amount of time incurred from delivering jobs outside their appointed time windows. Note, in the design of both systems only time spent

traveling empty and job rejection penalties are considered explicitly. Late penalties are a by-product of service time variability in an on-line setting. To demonstrate the effect of this design consideration, the results are presented in two formats. The tables presented in the following subsections include the mean and standard errors of the total routing costs, while the graphs illustrate the split of these mean total costs across their three component costs.

### 5.3.1 Service Time Uncertainty Results

The results highlighted here illustrate the capabilities of each system—agents and on-line optimization - to handle independent and dependent service time disruptions. Recall, we do not include the *a priori* optimal solution in this set of results as it would be unfair to compare a system that can never yield a solution permitting time window violations with solutions that allow such violations.

Figure 5.4 shows, by means of a bar chart, the mean routing costs of the agent and on-line optimization solution approaches for the case of independent service time variability. Of particular interest is the fact that the empty time (hours) attributable to both systems does not vary significantly across the five scenarios; for the on-line optimization the empty time remains consistently at 27.9 hours whereas for the agents it ranges slightly from 32 to 36 hours. The primary difference, both across scenarios and between systems, can be seen in terms of the mean over the total amount of time late for the jobs' time windows. For the on-line optimization approach, the amount of lateness increases sharply across the five scenarios — starting at zero ranging through 3.5, 8.25, 11.7 and ending at 203.79 hours. The agents, however, have a comparatively small amount of lateness across the five scenarios — starting at zero ranging through .02, .05, .14, and ending at 47.75 hours in the most extreme scenario. The penalty affiliated with job rejection is, on the other hand, relatively small for both systems, until the most extreme case at which point the job rejection penalty of the agents (126.02 hrs) far exceeds that of the on-line optimization (1.62 hrs). We should note here that all of the jobs rejected (27 in total) in the on-line optimization were rejected due to the

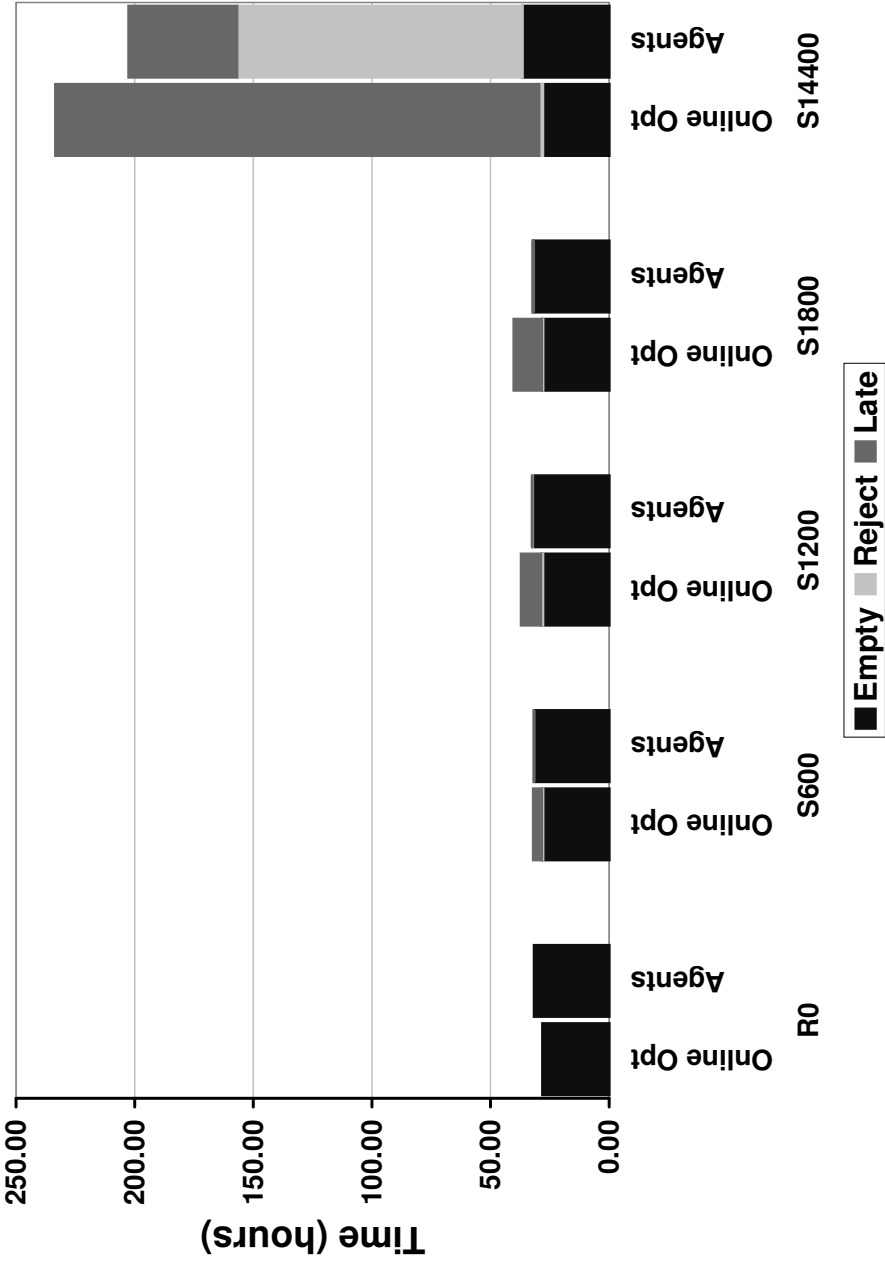


Figure 5.4: Mean over 33 days of the routing cost components (empty time, rejection penalty, late time) for the agent and on-line optimization solution approaches across five independent service time uncertainty scenarios.

inability of the system to find a feasible solution within 90 seconds. While this might appear to be a major shortcoming of the on-line optimization, this represents only .0025% of all jobs examined across all  $S_*$  instances.

Figure 5.5 shows the mean routing costs of the agent and on-line optimization solution approaches for the case of dependent service time variability. Similar to the independent service time scenarios, the empty time (hours) attributable to both systems does not vary significantly across the five scenarios. In fact the values for both solution approaches in the dependent scenarios are also nearly identical to those obtained in the independent scenarios; for the on-line optimization the empty time remains consistently at 28 hours whereas for the agents it ranges slightly from 32 to 36 hours. Again, the primary difference, both across scenarios and between systems, can be seen in terms of the mean over the total amount of time late for the jobs' time windows. For the on-line optimization approach the amount of lateness increases sharply across the five scenarios — starting at zero ranging through 8.72, 15.25, 27.02 and ending at 203.53 hours. The agents, however, have a comparatively small amount of lateness across the five scenarios — starting at zero ranging through .08, .32, .79, and ending at 49.05 in the most extreme scenario. The penalty affiliated with job rejection is, on the other hand, relatively small for both systems until the most extreme case at which point the job rejection penalty of the agents (129.52 hrs) far exceeds that of the on-line optimization (1.36 hrs). Again, we note that all of the jobs rejected (24 in total) in the on-line optimization were rejected due to the inability of the system to find a feasible solution within 90seconds; this represents only .0022% of all jobs examined across all  $Sd_*$  instances.

Table 5.3 summarizes the mean plus/minus the standard error for the total routing costs averaged over the 33 days of data for both the independent and dependent service time uncertainty cases. Also displayed in the table are the results of a paired t-test comparing the mean total cost of the on-line optimization and the agent-based method for each dataset.

These results indicate that the agent approach can deal slightly better with uncertainty regarding the service times than the on-line optimization

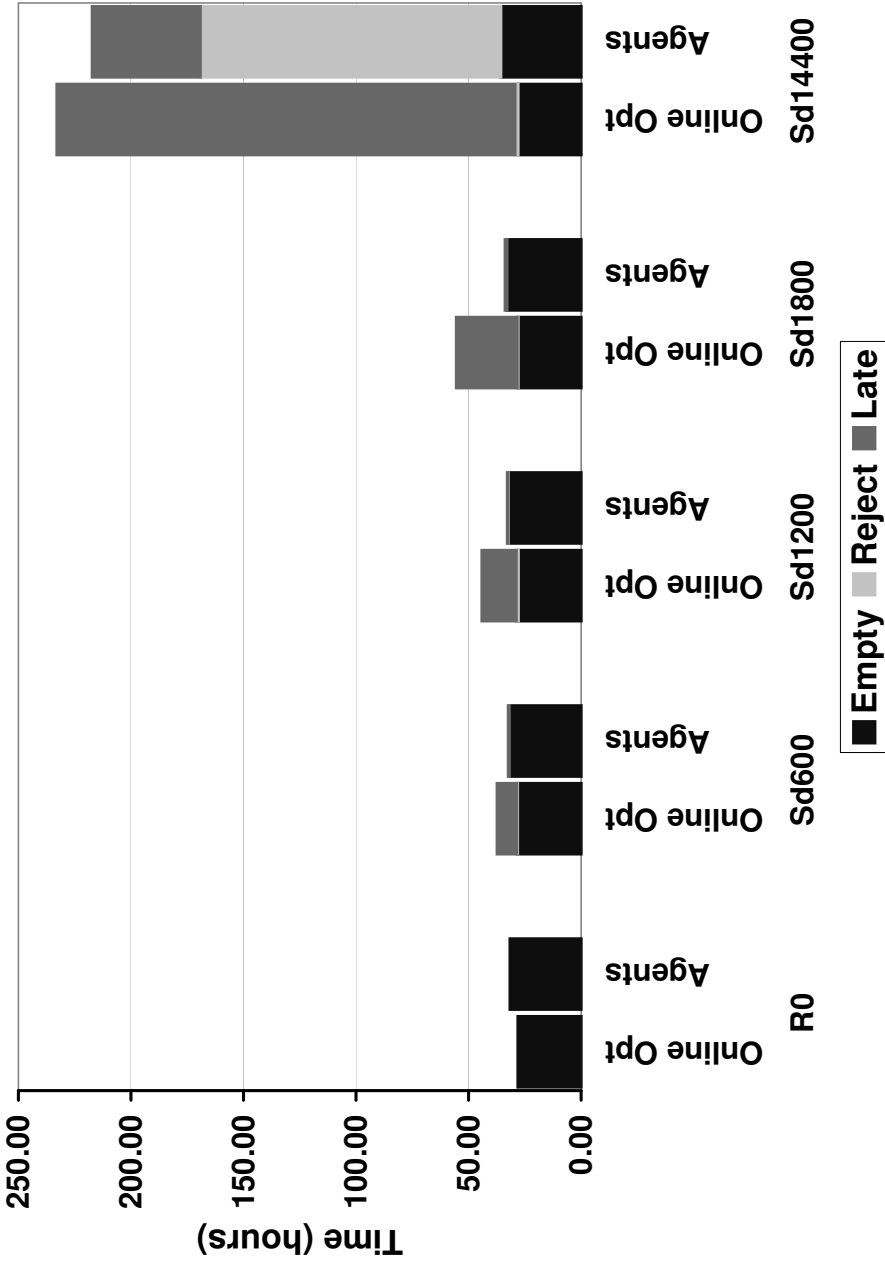


Figure 5.5: Mean over 33 days of the routing cost components (empty time, rejection penalty, late time) for the agent and on-line optimization solution approaches across five dependent service time uncertainty scenarios.

Table 5.3: mean  $\pm$  std. error of total routing costs (lateness plus rejected penalties plus time spent empty) in hours for all service time uncertainty experiments;  $n = 33$ .

| Dataset      | On-line Optimization | Agents            | Paired t-test: Difference in means equal to 0? |
|--------------|----------------------|-------------------|------------------------------------------------|
| $S_{600}$    | $31.95 \pm .68$      | $31.62 \pm .33$   | Fail to Reject, $p = .60$                      |
| $S_{1200}$   | $37.05 \pm .86$      | $32.44 \pm .29$   | Reject, $p < .0001$                            |
| $S_{1800}$   | $40.17 \pm 1.00$     | $32.09 \pm .36$   | Reject, $p < .0001$                            |
| $S_{14400}$  | $233.36 \pm 3.19$    | $202.51 \pm 5.17$ | Reject, $p < .0001$                            |
| $Sd_{600}$   | $37.33 \pm .94$      | $32.33 \pm .38$   | Reject, $p < .0001$                            |
| $Sd_{1200}$  | $44.06 \pm 1.81$     | $32.79 \pm .39$   | Reject, $p < .0001$                            |
| $Sd_{1800}$  | $55.43 \pm 2.65$     | $33.77 \pm .39$   | Reject, $p < .0001$                            |
| $Sd_{14400}$ | $232.89 \pm 4.08$    | $217.20 \pm 9.56$ | Fail to Reject, $p = .05$                      |

approach. (This is, of course, at the expense of significant job rejection.) This result is completely caused by the many late jobs in the on-line optimization approach. The tardiness manifested in the on-line optimization results comes from the assignment process of the approach when used in a rolling-horizon framework. The MIP is calibrated to specify an arrival time at each job in each snapshot problem. If a truck is close to violating the time specified in the previous solution, then the job is permanently fixed and the truck begins execution. In this way many jobs are assigned that later encounter delay during execution as a result of the variable service times. This also explains the many late jobs in the extreme case of 4 hours, where the agent approach incurs a high penalty for rejection, because the agent approach can find no way to squeeze that many jobs into an already delayed schedule.

Furthermore, it is clear that the amount of late time incurred by the on-line optimization solution in the dependent case is higher than that in the independent case. The difference in costs for the agents is however not so different between the independent and dependent service time uncertainty scenarios. In fact, a paired t-test indicates that in all, but the  $S_{14400}$  and  $Sd_{14400}$  scenarios, there is a statistically significant difference ( $p < .0001$ ) in the means of the independent and dependent service time scenarios for the

on-line optimization approach. On the other hand, the agents do not exhibit a statistically significant difference in means (at  $p > .01$ ) for all but the  $S_{1800}$  and  $Sd_{1800}$  cases.

These results indicate that the on-line optimization tends to suffer in terms of late time when the service time uncertainty has underlying dependencies. This is most likely because jobs that are close together location-wise often end up in the same route. When jobs are on the same route, if one job demonstrates delay in execution, subsequent jobs are immediately assigned as they appear more urgent in the system. From this initial solution many job assignments are then fixed and service is undertaken; only to suffer from even more lateness occurring in execution. From this argumentation, we may expect better performance of the on-line optimization when service times are static. We now turn our attention to the results of the job arrival uncertainty experiments to see whether we can support this experimentally.

### 5.3.2 Job Arrival Uncertainty Results

Job arrival uncertainty is the most common type of uncertainty classically studied in the dynamic vehicle routing literature. As the instances used to test job arrival uncertainty do not contain jobs that would cause lateness, regardless of their arrival time, it is valid in these instances to compare all three routing systems—the *a priori* optimal (a realistically unattainable lower bound), the on-line optimization approach, and the agent approach.

Figure 5.6 shows the mean routing costs of the *a priori* optimal, on-line optimization, and agent solution approaches for the case of job arrival uncertainty. Of note is the fact that the empty time (hours) attributable to each system does not vary significantly across the five scenarios, but does show a marked difference when studied between systems. The *a priori* optimal remains the lowest for all three systems with a value between 27 and 28 hours for all scenarios. The slight increase is due to the tightening of the time constraints stemming from the later availability of some jobs. It should also be noted that in scenarios  $R_0$ ,  $R_{25}$ ,  $R_{50}$ , and  $R_{75}$ , CPLEX was unable to find a confirmed optimal solution to the *a priori* problem in three instances before



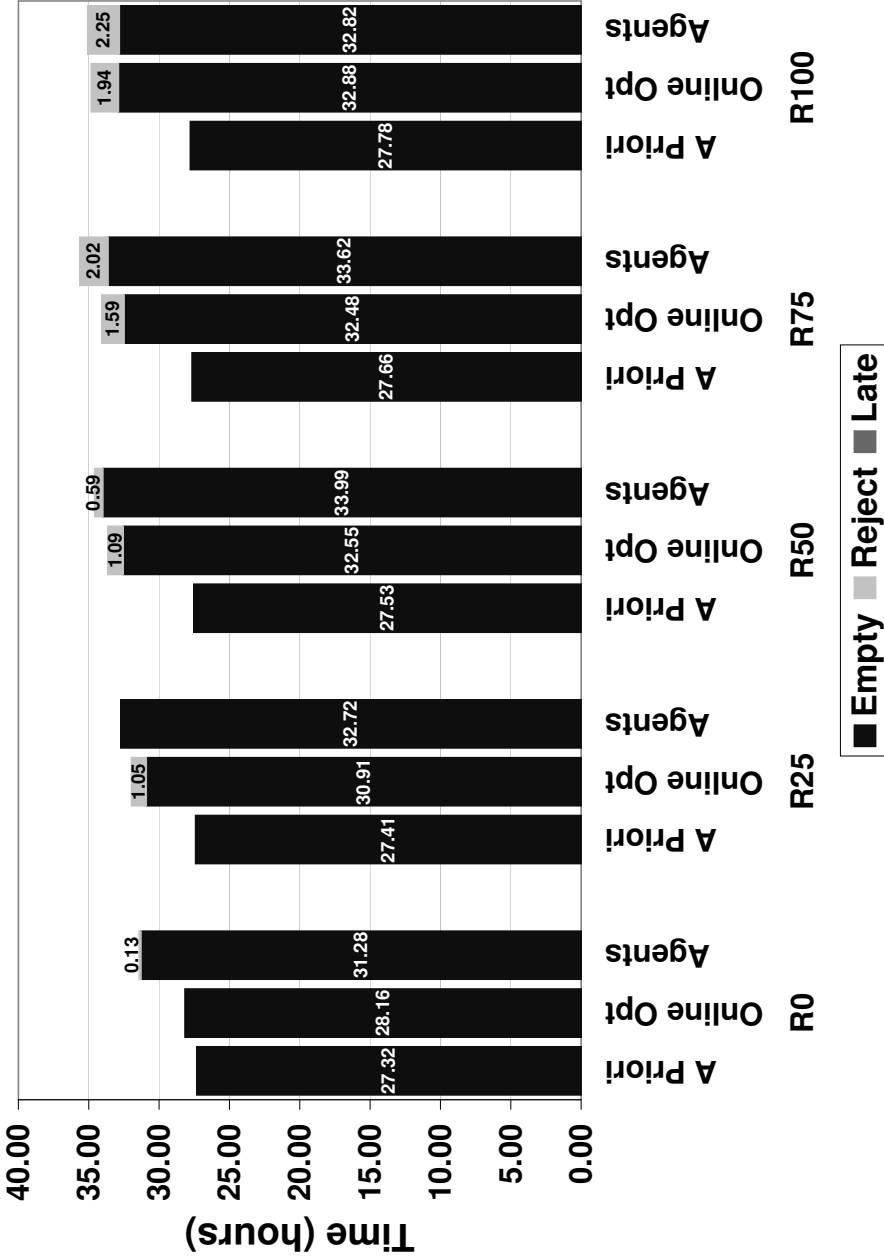


Figure 5.6: Mean over 33 days of the routing cost components (empty time, rejection penalty, late time) for the agent and on-line optimization solution approaches across five job arrival uncertainty scenarios.

running out of memory. In scenario  $R_{100}$  there were four instances for which the optimal could not be found. In these cases the relaxed lower bound was taken as the objective value. On average, the lowest bound was 1.05% from the best known integer solution at the time the model quit running. Thus, the data presented here represents a lowest bound on the routing costs.

The on-line optimization, on the other hand, performs consistently in the range between the *a priori* optimal and the agent system but does experience a steady increase across scenarios — ranging from 28.16 hours when all jobs are known at time zero to 32.88 hours in the case when all jobs are revealed over time. The agents system consistently yields a higher (and in four cases the highest) level of empty time across all five scenarios; ranging from 31.28 hours in the case with the least uncertainty to 32.82 in the case with the most uncertainty. With the exception of the low uncertainty case, both the on-line optimization and the agent system incur a penalty for rejecting jobs. The rejection penalty grows at an increasing rate for the agent system (ranging from 0.13 to 2.25) while it appears to fluctuate in a smaller range for the on-line optimization approach (ranging from 1.05 to 1.94). Interestingly, in the on-line optimization only one job was rejected due to the inability of the system to find a feasible solution within 90seconds.

Overall we can say that the on-line optimization approach consistently outperforms the agent approach. However, the more jobs are uncertain, the smaller the difference becomes — to the point that both systems can be considered competitive. This is supported statistically, with a paired t-test indicating that the difference in the means is not significant in the  $R_{100}$  case at  $p = .61$ . A summary of the means plus/minus standard errors, along with the results of the paired t-tests, can be seen in Table 5.4.

Now that we have examined both service time and job arrival uncertainty in isolation, we turn our attention to the scenarios that combine both types of uncertainty.

Table 5.4: mean  $\pm$  std. error of total routing costs (lateness plus rejected penalties plus time spent empty) in hours for all job arrival uncertainty experiments;  $n = 33$ .

| Dataset   | <i>A Priori</i><br>Optimal | On-line<br>Optimization | Agents          | Paired t-test: Difference in on-line opt and agent means equal to 0? |
|-----------|----------------------------|-------------------------|-----------------|----------------------------------------------------------------------|
| $R_0$     | $27.32 \pm .35$            | $28.16 \pm .34$         | $31.41 \pm .38$ | Reject, $p < .0001$                                                  |
| $R_{25}$  | $27.41 \pm .36$            | $31.96 \pm .46$         | $32.72 \pm .37$ | Fail to Reject, $p = .08$                                            |
| $R_{50}$  | $27.53 \pm .36$            | $33.64 \pm .62$         | $34.58 \pm .47$ | Fail to Reject, $p = .13$                                            |
| $R_{75}$  | $27.66 \pm .37$            | $34.07 \pm .71$         | $35.64 \pm .69$ | Reject, $p = .01$                                                    |
| $R_{100}$ | $27.78 \pm .37$            | $34.82 \pm .73$         | $35.07 \pm .71$ | Fail to Reject, $p = .61$                                            |

### 5.3.3 Job and Service Time Uncertainty Results

This final set of experiments focuses on the effect of combining job arrival uncertainty at the 50% level with independently distributed service time uncertainty. For this scenario, we are again forced to leave out the results for the *a priori* optimal, because only the on-line optimization and the agents can incur lateness.

Figure 5.7 shows the mean routing costs of the agent and on-line optimization solution approaches for the case of independent service time variability combined with a 50% job arrival uncertainty level. As in the service time uncertainty cases, the empty time remains relatively constant across all cases for both solution approaches. Again, as in the service time uncertainty scenarios, the primary difference, both across scenarios and between systems, can be seen in terms of the mean over the total amount of time late for the jobs' time windows. For the on-line optimization approach the amount of lateness increases sharply across the five scenarios—starting at 0 ranging through 4.39, 9.07, 13.72 and ending at 144.98 hours. The agents, however, have a comparatively small amount of lateness across the five scenarios—starting at 0 ranging through .12, .42, 1.07, and ending at 57.51 in the most extreme scenario. The penalty affiliated with job rejection is, on the other hand, relatively small for both systems until the most extreme case at which point the job rejection

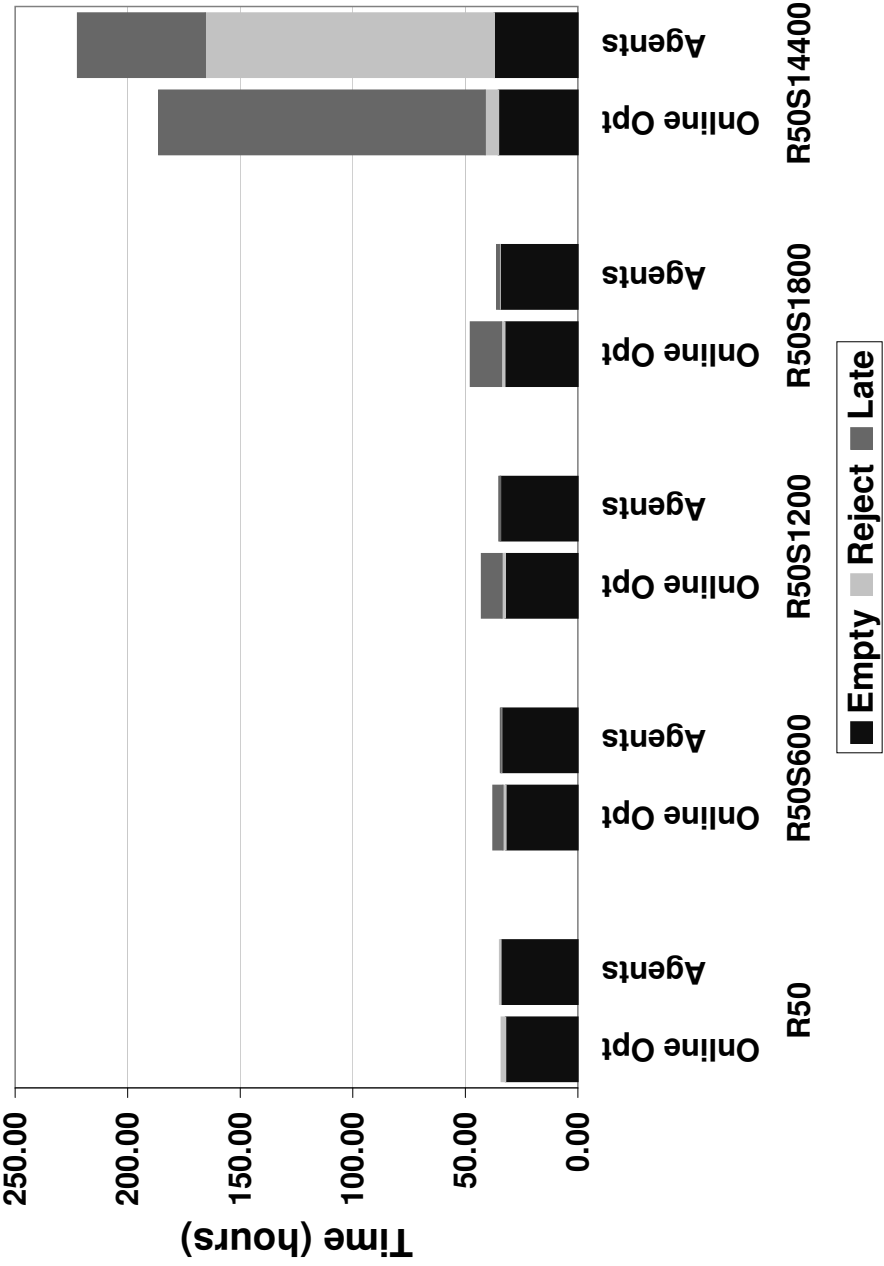


Figure 5.7: Mean over 33 days of the routing cost components (empty time, rejection penalty, late time) for the agent and on-line optimization solution approaches across five combined job arrival and service time uncertainty scenarios.

Table 5.5: mean  $\pm$  std. error of total routing costs (lateness plus rejected penalties plus time spent empty) in hours for all service time combined with job arrival uncertainty experiments;  $n = 33$ .

| Dataset           | On-line Optimiza-<br>tion | Agents            | Paired t-test: Dif-<br>ference in means<br>equal to 0? |
|-------------------|---------------------------|-------------------|--------------------------------------------------------|
| $R_{50}S_{600}$   | $37.71 \pm .75$           | $34.36 \pm .44$   | Reject, $p < .0001$                                    |
| $R_{50}S_{1200}$  | $42.79 \pm 1.01$          | $35.06 \pm .44$   | Reject, $p < .0001$                                    |
| $R_{50}S_{1800}$  | $47.71 \pm 1.21$          | $36.04 \pm .47$   | Reject, $p < .0001$                                    |
| $R_{50}S_{14400}$ | $186.17 \pm 2.94$         | $222.25 \pm 4.41$ | Reject, $p < .0001$                                    |

penalty of the agents (132.64 hrs) far exceeds that of the on-line optimization (5.94 hrs). We should note here that only nine of the jobs rejected, in all  $R_{50}S_*$  instances of the on-line optimization, were rejected due to the inability of the system to find a feasible solution within 90 seconds.

Overall we can see that agents do better here, and perform similarly to the setting without job arrival uncertainty. However, for the extreme case of a four hour service time uncertainty, the on-line optimization approach has much lower costs. This difference between the agents and on-line optimization is not, however, statistically significant (most likely due to the difference in variance exhibited by both datasets). The means plus/minus standard error as well as the paired t-test results can be seen in Table 5.5.

What is interesting, however, is that the difference in means between the on-line optimization  $S_{14400}$  and  $R_{50}S_{14400}$  instances is statistically significant ( $p < 0.001$ ). The moderating effect that job arrival uncertainty seems to have on service time uncertainty is most likely due to the fact that when jobs arrive over time, each problem instance is smaller and can therefore solve to optimality within the 30 second horizon allotted to it. Furthermore, the on-line optimization is less likely to fix long routes early as in the case with only service time uncertainty, since it does not have access to information regarding co-located jobs. Thus, each truck may only be assigned one (or even zero) jobs in the early iterations of the optimization, this helps to keep the trucks from permanently being assigned longer routes of co-located jobs.

## 5.4 Discussion

We now return to the fundamental question in this research — can agents perform as well as on-line optimization? From the results presented here, we are safe to say: yes, but that is not the end of the story. Not only do the agents perform competitively in comparison to the on-line optimization approach on the spatially realistic drayage case, but they also demonstrate a certain number of strengths in handling uncertainty.

These strengths are most obvious in the experiments focused on service time uncertainty. In both independent ( $S_*$ ) and dependent ( $Sd_*$ ) cases the agents consistently yield lower routing costs when the variation in service times is greater than 10 minutes. We attribute this advantage to the limitations of the heuristics agents use to compute a solution. These heuristics provide sub-optimal results when faced with a static problem. In a series of problems however, being suboptimal in the beginning means having more trucks on the move, yielding more options to accommodate changes.

There are, on the other hand, settings in which the on-line optimization approach gives significantly better results. In particular, in settings where job arrival uncertainty is a dominant feature. Admittedly, this is not surprising, as job arrival uncertainty is the type of uncertainty most commonly studied in the literature and the type of uncertainty the on-line optimization approach was truly calibrated for.

Given these findings, one might ask the question: “how much are these differences attributable to the centralized versus decentralized nature of the systems as opposed to the optimization versus heuristic characteristics of the approaches?” We begin to answer this question by noting that the comparison between the two approaches is a comparison of heuristics. While the on-line optimization has the opportunity to find an optimal solution, it more often finds only a good feasible solution within each 30-second interval. As such, neither system can claim to produce an optimal solution. Nevertheless, we cannot, so easily, disentangle the centralization/decentralization and heuristic/optimization facets of each approach. The near optimal solutions

put forth by the on-line optimization are highly dependent on the centralization of all problem data; the agility exhibited by the decentralized agents is dependent on fast running heuristics.

As the on-line optimization operates by batching all system information at 30-second intervals, it is also forced to batch routing decisions in the same intervals. This process of centralizing information, and using all this information in making job assignment as well as rejection decisions yields assignments that may later turn out to be late, because service times may vary. On the other hand, when service times are static, this approach yields assignments that very effectively balance empty distance against job rejection.

In contrast, the agents, simulated in an individualized manner, are not dependent on batched decisions made every 30-seconds. They can react at any time, but they have a limited view. The individualized outlook implies that the truck agents do not recognize the cost of a rejected container. The container agents, in turn, are the victims of the trucks' inefficiency (or service time variability). This yields a high level of rejected jobs, but a minimal level of lateness when service time uncertainty is high. However, when job arrival uncertainty is the only source of uncertainty, the agents lack the ability to globally weigh routing costs against rejection penalties.

The advantage of a global perspective, regarding the ability to balance empty distance against rejection costs, is supported by our results. Paired t-tests, comparing only the combined costs of empty distance and rejection penalties (leaving out lateness), reveal a significant difference in these mean costs for all service time uncertainty scenarios; and for all, but the  $R_{100}$ , job arrival uncertainty scenarios. Alternatively, the combined service time and job arrival uncertainty scenarios show a significant difference only in the most uncertain case ( $R_{50}S_{14400}$ ). These results lead us to the following conclusion, when the on-line optimization approach performed better, it was by capitalizing on the optimal (or near optimal) balance between routing and rejection costs. In the cases where the agents performed better, their flexibility provided by their distributed nature was the competitive advantage.

Aside from the observable and quantifiable results presented in this com-

parative study, we would be remiss if we did not highlight a few qualitative differences between agent-based and on-line optimization approaches for vehicle routing. Agents permit modeling a complex system with a high likelihood of achieving cognitive fit (Krauth, 2008). Cognitive fit refers to how closely the theoretical representation of the problem being solved is correlated with the actual problem being solved. This feature of agent-based systems can make them especially appealing to the lay-person. Thus, when we consider the real-world environment of a drayage company, it is easy to imagine a dispatcher embracing a solution approach that directly maps onto their existing operations. Additionally, in settings where more than one company (or business unit within a single company) must compete for limited resources or when the direct sharing of sensitive information is an issue, then the distributed agent model may be advantageous. Alternatively, agents may provide a good solution method in environments where there is no central authority, such as the routing of automated guided vehicles.

On the other hand, agent technology is new. New technology often represents a risky choice. In contrast, many existing decision support systems depend on optimization techniques. Furthermore, there is a perception that optimization and operations research represents a technique with years of scientific research underpinning it. Additionally, the management level of logistics companies often prefers to specify their goals for the company as encapsulated in an objective function. The notion of emergent behavior or autonomous fulfillment of local goals may appear threatening to those that must select and invest in a new decision support system.

Given both the quantitative and qualitative advantages and disadvantages of both approaches, the future of this research lays in the investigation of how ideas from an agent-based approach can be used in an on-line approach, and vice versa. At the most basic level, the results from this agent-based study can be used to calibrate the trade-off between rejection and lateness in the objective function of the MIP. At a deeper level, a hybrid approach with the ability to exhibit the right behavior depending on the current situation may ultimately be the best way of dealing with uncertainty. Furthermore,



the agent-based approach can itself be augmented to include learning. In this chapter we describe agents that do not make any effort to adjust their behavior based on what they have seen before. One may conjecture that some of the global state could be inferred from past experiences, and that agents with limited information could benefit from learning. An interesting challenge, for future research, would be to find the conditions in which agents can learn and adapt to uncertainties in comparison to a centralized system based on stochastic programming.

In order to find the exact conditions under which one approach outperforms the other, and to understand the fundamental reasons for that behavior, further experiments are required. Not only should we study more general vehicle routing problems (using standard datasets such as the Solomon benchmark sets (Solomon, 1987)), we should also study other sources of uncertainty, such as travel time uncertainty (to see if this indeed shows similar behavior to the dependent service times considered in this chapter) and truck break downs. Furthermore, as instances grow larger, the effect of limited time and/or memory may influence performance drastically, and should thus be carefully taken into consideration. Finally, we would like to see a similar study focused on a comparison of agents with stochastic approaches.

## Chapter 6

# Conclusion

...to the traveler, a mountain outline varies with every step, and it has an infinite number of profiles, though absolutely but one form. Even when cleft or bored through it is not comprehended in its entirety.

Henry David Thoreau, *Walden*, 1854

There is but one problem at the heart of this thesis. I have cleaved it, bored through it, and examined its many profiles, yet it remains largely enigmatic. Why is this so? While the answer to this question may elude us yet, we attempt closure by summarizing the route this thesis traveled and speculating what further exploration could reveal.

### 6.1 The Past

In Chapter 2, we found clues in the literature indicating that the pick-up and delivery structure of drayage problems may make them easier than general ATSPs. This hypothesis was empirically tested in Chapter 3. The result of this testing was the identification of a set of distance matrix metrics with a significant influence on the ease of solvability for ATSPs. These metrics, in turn, imply that drayage problems — a special type of ATSP — are generally easier to solve. In particular we reveal that drayage problems, with

a limited number of terminals and customer locations, tend to have a large number of well-placed zeros in their distance matrices. Specifically, the placement of zeros is such that drayage problem distance matrices have more zeros in symmetric locations indicating the presence of only a few vertices in an underlying ARP; as compared to their general ATSP counterparts. These properties seem to make the problem easier.

Abandoning the spatial dimension for the temporal, in Chapter 4, we learned that advanced job information can be beneficial to the cost of an on-line algorithm — bringing it closer to optimal. The real contribution of this realization, however, is that the advanced information can be as minimal as revealing the location of the job, with no revelation of the job's release time. This is an important finding as drayage operations are such that job locations are generally static while job release times are not always revealed in a timely manner.

While these findings alone gave us hope, the movement away from a single-vehicle setting to full fleet operations, in Chapter 5, raised and partially answered questions regarding control. Specifically, we asked what the quantifiable benefits of using a centralized control structure over a decentralized control structure might be. The answer indicated a dependence on uncertainty. The more uncertainty, in terms of job arrival and service time duration, the better the agents performed. This result has significant implications for the drayage industry as most (possibly all) drayage firms follow a traditional control structure dependent on a central dispatcher. These results indicate that driver empowerment may yield higher profits — or, at least, less costs.

## 6.2 The Future

We describe our vision for the future by first looking back to the opening lines of this book. This book documents the dissection of a large problem into smaller, more easily examined, parts. In some cases, the tool of dissection, has isolated structures to a point where the function and relationship to the

whole is barely discernible. This may frustrate some nevertheless the isolated parts have themselves yielded fascinating questions — inspiration for future work. While many of these questions were raised at the end of their respective chapters we present some additional considerations here.

When considering the results of Chapter 2 and Chapter 3, the questions begging for answers are: how can we exploit the models predicting solvability? Can we use them to design a method by which a “hard” ATSP can be transformed into an “easy” ATSP? Given the clues in the data of Chapter 3, can we prove that drayage problems are, like Gilmore-Gomory TSPs, a sub-class of ATSPs that are polynomially solvable?

Turning to Chapter 4, we begin with the question: What happens when we extend our simple TSP on  $\mathbb{R}^+$  to include pick-up and delivery? at first blush, we can answer that pick-up and delivery on  $\mathbb{R}^+$  represents a Gilmore-Gomory TSP. As such, this problem is solvable in polynomial time. Recognizing this fact we can ask a myriad of questions revolving around the worst-case ratios for on-line Gilmore-Gomory TSPs and the advantage of advanced information in this context. Furthermore, the vector of advanced information in pick-up and delivery problems is more complex — e.g. both the pick-up (location and time) and the drop-off (location and time) can be revealed at different disclosure dates. Additionally, we can ask the question: what other fields can benefit from these results? While a line ( $\mathbb{R}^+$ ) does not seem the most likely space for real-world applications, these models do find a place in the scheduling of elevators or the movement of robots in manufacturing. Thus, we wish to know how must these TSP related results be modified to become useful to a broader audience?

Finally, Chapter 5 raises questions at the boundary of (at least) three fields — operations research, computer science, and artificial intelligence. If anything, Chapter 5 highlights the need for a more natural and smoother integration of both agent based and optimization based approaches. How can the handoff from an optimal solution to the MAS implementation be orchestrated? How will the MAS execution affect the optimality of the optimization based solution? How can the emergent behavior of the MAS be monitored

and fed back into the optimization? How can the emergent behavior of the MAS be monitored and controlled to trust their use in uncontrolled environments such as elevator scheduling? These are the questions that await a new generation of interdisciplinary researchers.

Ultimately, while the process of dissection may force an examination of isolated parts, the results of each isolated examination indicate a common vision for the future. The first three content chapters indicate that benefits are to be gained when the drayage problem is pared down to a single vehicle case. Similarly, the result of the last chapter highlights that cleaving the problem along the lines of its vehicular parts generates benefits in highly dynamic settings. Thus, our entreaty for future work is a proposal for an agent system in which the agents exploit the results and algorithms of Chapters 3 and 4.

As this work moves into the future, we bring one last thought to bear, a thought that has been almost entirely absent from this thesis: any and all new drayage planning software must consider the human dimension. The fact that, in the end, all of the routes are made or at least executed by human beings means that new planning systems must include humans as part of the decision process. As remarked in Chapter 4 the ability to plan and adjust is singularly human. Therefore a system that can incorporate hints from the user will ultimately find more favor amongst dispatchers — dispatchers that are known for remarking: “you can plan all you want, but in a few seconds that plan is shot all to pieces.”

# Appendix A

## Generator Parameters

Table A.1: Description of instances generated for verification of models.

| Generator | No. Instances | Parameters                                                                                       |
|-----------|---------------|--------------------------------------------------------------------------------------------------|
| smat      | 5             | $n = 300, d_{ij} \in [0, 500]$ , seed numbers 0 to 4                                             |
| smat      | 2             | $n = 300, d_{ij} \in [0, 10^6]$ , seed numbers 5 and 6                                           |
| smat      | 2             | $n = 100, d_{ij} \in [0, 10^6]$ , seed numbers 7 and 8                                           |
| smat      | 5             | $n = 100, d_{ij} \in [0, 100]$ , seed numbers 9 and 13                                           |
| tsmat     | 5             | $n = 300, d_{ij} \in [0, 500]$ , seed numbers 0 to 4                                             |
| tsmat     | 2             | $n = 300, d_{ij} \in [0, 10^6]$ , seed numbers 5 and 6                                           |
| tsmat     | 2             | $n = 100, d_{ij} \in [0, 10^6]$ , seed numbers 7 and 8                                           |
| tsmat     | 5             | $n = 100, d_{ij} \in [0, 100]$ , seed numbers 9 and 13                                           |
| rect      | 5             | $n = 300$ , square 500 by 500, seed numbers 0 to 4                                               |
| rect      | 2             | $n = 300$ , square $10^6$ by $10^6$ , seed numbers 5 and 6                                       |
| rect      | 2             | $n = 100$ , square $10^6$ by $10^6$ , seed numbers 7 and 8                                       |
| rect      | 5             | $n = 100$ , square 100 by 100, seed numbers 9 and 13                                             |
| amat      | 5             | $n = 300, d_{ij} \in [0, 500]$ , seed numbers 0 to 4                                             |
| amat      | 2             | $n = 300, d_{ij} \in [0, 10^6]$ , seed numbers 5 and 6                                           |
| amat      | 2             | $n = 100, d_{ij} \in [0, 10^6]$ , seed numbers 7 and 8                                           |
| amat      | 5             | $n = 100, d_{ij} \in [0, 100]$ , seed numbers 9 and 13                                           |
| tmat      | 5             | $n = 300, d_{ij} \in [0, 500]$ , seed numbers 0 to 4                                             |
| tmat      | 2             | $n = 300, d_{ij} \in [0, 10^6]$ , seed numbers 5 and 6                                           |
| tmat      | 2             | $n = 100, d_{ij} \in [0, 10^6]$ , seed numbers 7 and 8                                           |
| tmat      | 5             | $n = 100, d_{ij} \in [0, 100]$ , seed numbers 9 and 13                                           |
| rtilt     | 5             | $n = 300$ , square 500 by 500, $u_x = 1, u_y^+ = 2, u_y^- = 0$ , seed numbers 0 to 4             |
| rtilt     | 2             | $n = 300$ , square $10^6$ by $10^6$ , $u_x = 1, u_y^+ = 2, u_y^- = 0$ , seed numbers 5 and 6     |
| rtilt     | 2             | $n = 100$ , square $10^6$ by $10^6$ , $u_x = 1, u_y^+ = 2, u_y^- = 0$ , seed numbers 7 and 8     |
| rtilt     | 5             | $n = 100$ , square 100 by 100, $u_x = 1, u_y^+ = 2, u_y^- = 0$ , seed numbers 9 and 13           |
| stilt     | 5             | $n = 300$ , square 500 by 500, $u_x = 2, u_y^+ = 4, u_y^- = 1$ , seed numbers 0 to 4             |
| stilt     | 2             | $n = 300$ , square $10^6$ by $10^6$ , $u_x = 2, u_y^+ = 4, u_y^- = 1$ , seed numbers 5 and 6     |
| stilt     | 2             | $n = 100$ , square $10^6$ by $10^6$ , $u_x = 2, u_y^+ = 4, u_y^- = 1$ , seed numbers 7 and 8     |
| stilt     | 5             | $n = 100$ , square 100 by 100, $u_x = 2, u_y^+ = 4, u_y^- = 1$ , seed numbers 9 and 13           |
| crane     | 5             | $n = 300$ , square 500 by 500, $u = 4,000$ , seed numbers 0 to 4                                 |
| crane     | 2             | $n = 300$ , square $10^6$ by $10^6$ , $u = 10$ , seed numbers 5 and 6                            |
| crane     | 2             | $n = 100$ , square $10^6$ by $10^6$ , $u = 10$ , seed numbers 7 and 8                            |
| crane     | 5             | $n = 100$ , square 100 by 100, $u = 20,000$ , seed numbers 9 and 13                              |
| disk      | 5             | $n = 300, x \in [0, 500]$ , $u = 2,000$ , $speed = 10.0$ , seed numbers 0 to 4                   |
| disk      | 2             | $n = 300, x \in [0, 10^6]$ , $u = 10$ , $speed = 10.0$ , seed numbers 5 and 6                    |
| disk      | 2             | $n = 100, x \in [0, 10^6]$ , $u = 10$ , $speed = 10.0$ , seed numbers 7 and 8                    |
| disk      | 5             | $n = 100, x \in [0, 100]$ , $u = 10,000$ , $speed = 10.0$ , seed numbers 9 and 13                |
| coin      | 5             | $n = 300$ , 100 blocks, $maxcoord = 10$ , seed numbers 0 to 4                                    |
| coin      | 2             | $n = 300$ , 173 blocks, $maxcoord = 10$ , seed numbers 5 and 6                                   |
| coin      | 2             | $n = 100$ , 100 blocks, $maxcoord = 10$ , seed numbers 7 and 8                                   |
| coin      | 5             | $n = 100$ , 25 blocks, $maxcoord = 10$ , seed numbers 9 and 13                                   |
| shop      | 5             | $n = 300, k = 20, task\ length \in [0, 250]$ , seed numbers 0 to 4                               |
| shop      | 2             | $n = 300, k = 50, task\ length \in [0, 1000]$ , seed numbers 5 and 6                             |
| shop      | 2             | $n = 100, k = 50, task\ length \in [0, 1000]$ , seed numbers 7 and 8                             |
| shop      | 5             | $n = 100, k = 10, task\ length \in [0, 100]$ , seed numbers 9 and 13                             |
| super     | 2             | $n = 300, k = 2, string\ length = 20$ , seed numbers 5 and 6                                     |
| super     | 2             | $n = 100, k = 2, string\ length = 20$ , seed numbers 7 and 8                                     |
| crane2    | 5             | $n = 300$ , square 500 by 500, $origins = 150, destinations = 200$ , seed numbers 0 to 4         |
| crane2    | 2             | $n = 300$ , square $10^6$ by $10^6$ , $origins = 300, destinations = 200$ , seed numbers 5 and 6 |
| crane2    | 2             | $n = 100$ , square $10^6$ by $10^6$ , $origins = 100, destinations = 65$ , seed numbers 7 and 8  |
| crane2    | 5             | $n = 100$ , square 100 by 100, $origins = 50, destinations = 75$ , seed numbers 9 and 13         |

# Appendix B

## Crane2 Random Instance Generator

```
/* Crane2 random instance generator, designed by F. Jordan Srour,
   is intended to mimic drayge problems; it is based on the
   random instance generators of Cirasella et al, 2001
   and Johnson et al, 2002 - available at:
   http://www.research.att.com/~dsj/chtsp/atsp.html*/

#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include "genrand.h" //available at http://www.research.att.com/~dsj/chtsp/atsp.html
#define MAXN 3200
#define MAXCOORD 1000000
#define BINSIZE (1 << 30)
#define FACTOR 1

int x[MAXN],y[MAXN],z[MAXN],w[MAXN];

int dist(int a, int b);

main(argc,argv)
int argc;
char *argv[];
{
int maxcoord = MAXCOORD;
int N;
int i,j, node1, node2;
int z1, w1;
int seed;
int origins;
int destinations;
int arg = 1;
int tsplib = 0;

if (argc < 4) {
```



```

printf("Usage: cranegen2 [-tsplib] N seed origins destinations [maxcoord] > filename\n");
exit(1);
}
if (strcmp(argv[arg], "-tsplib") == 0) {
    tsplib = 1;
    ++arg;
}
N = atoi(argv[arg++]);
seed = atoi(argv[arg++]);
origins = atoi(argv[arg++]);
destinations = atoi(argv[arg++]);
if (argc > arg) maxcoord = atoi(argv[arg++]);

if (origins > N) origins = N;
if (destinations < (N - origins)) destinations = N - origins + 1;

/* initialize random number generator */

sprand(seed);

for (i=1;i<=N;i++) {
    x[i] = rangerand(maxcoord);
    y[i] = rangerand(maxcoord);
}

if (tsplib) {
    printf("NAME: cranegen_%d_%d_%d_%d\n", N, seed, origins, destinations, maxcoord);
    printf("TYPE: ATSP\n");
    printf("COMMENT: Asymmetric TSP (generated with 'cranegen2 %d %d %d %d')\n",N, seed, origins, destinations, maxcoord);
    printf("DIMENSION: %d\n", N);
    printf("EDGE_WEIGHT_TYPE: EXPLICIT\n");
    printf("EDGE_WEIGHT_FORMAT: FULL_MATRIX\n");
    printf("EDGE_WEIGHT_SECTION\n");
}
else {
    printf("%d A\n",N);
}

for (i=1;i<=N;i++)
{
    for (j=1;j<=N;j++)
    {
        if ((i != j) && (i <= destinations) && (j <= origins)){node1 = N+1 - i; node2 = j;}
        if ((i != j) && (i <= destinations) && (j > origins)) {node1 = N+1 - i; node2 = j - origins;}
        if ((i != j) && (i > destinations) && (j <= origins)) {node1 = N+1 - (i-destinations); node2 = j;}
        if ((i != j) && (i > destinations) && (j > origins)) {node1 = N+1 - (i-destinations); node2 = j-origins;}
        if (i == j) {node1 = 0; node2 = 0;}

        printf("%d\n",dist(node1,node2));
    }
}

if (tsplib) {
    printf("EOF\n");
}
else {
    printf("cranegen2 %d %d %d %d\n",N,seed, origins, destinations,maxcoord);
};
}

dist (a,b)
int a,b;

```

---

```
{
    double max, sum, t;
    int rd;

    if (a == 0 && b == 0) return (1<<29);
    t = (double) (x[a]-x[b]);
    if (t < 0.0) t = -t;
    max = t; sum = t*t;

    t = (double) (y[a]-y[b]);
    if (t < 0.0) t = -t;
    if (t > max) max = t;
    sum += t*t;

    if (sum == 0.0) return 0;

    max *= 2.0;
    max = 0.5 * (max + sum/max);
    max = 0.5 * (max + sum/max);
    max = 0.5 * (max + sum/max);
    max = 0.5 * (max + sum/max);
    max = 0.5 * (max + sum/max);
    rd = (int) max;
#ifdef ROUNDUP
    if (max-rd > .00000001) rd++;
#else
    if (max-rd > .5) rd++;
#endif
    return(rd);
}
```



# List of Tables

|      |                                                                                                                                                     |    |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 1.1  | Overview of problems examined and our contributions. . . . .                                                                                        | 14 |
| 3.1  | Summary of instances in primary dataset. . . . .                                                                                                    | 44 |
| 3.2  | Summary of solvability statistics by cluster. . . . .                                                                                               | 49 |
| 3.3  | Summary of distance related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses). . . . .        | 52 |
| 3.4  | Summary of asymmetry related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses). . . . .       | 55 |
| 3.5  | Summary of graph structure related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses). . . . . | 57 |
| 3.6  | Summary of ARP related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses). . . . .             | 63 |
| 3.7  | Summary of AP related distance matrix metrics by cluster; each cell contains the mean and standard deviation (in parentheses). . . . .              | 64 |
| 3.8  | OLS Results for Model of Sum of Exact Runtimes. $\bar{R}^2 = .786$ . .                                                                              | 71 |
| 3.9  | Tests of equality of group means. . . . .                                                                                                           | 73 |
| 3.10 | Unstandardized coefficients of the two discriminant functions for the ease of solvability bins. . . . .                                             | 75 |
| 3.11 | Significance of discriminant functions. . . . .                                                                                                     | 75 |

|      |                                                                                                                                                                                 |     |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 3.12 | Pooled within-groups correlations between discriminating variables and standardized canonical discriminant functions. . . . .                                                   | 76  |
| 3.13 | Classification functions for the groups easy, medium, and hard.                                                                                                                 | 78  |
| 3.14 | Classification of instances using classification functions. . . . .                                                                                                             | 78  |
| 3.15 | Cross-classification of instances using multinomial logistic regression. . . . .                                                                                                | 80  |
| 3.16 | Multinomial logistic regression statistics for solvability groups based on seven distance matrix metrics. The medium group is the reference group. . . . .                      | 81  |
| 3.17 | Cross-classification of verification instances using discriminant classification functions. . . . .                                                                             | 84  |
| 3.18 | Cross-classification of verification instances using multinomial logistic regression functions. . . . .                                                                         | 84  |
| 3.19 | Count of instances in each group based on problem type. . . . .                                                                                                                 | 85  |
| 3.20 | Mean and (std. deviation) of the sum of exact algorithm runtimes for instances based on problem type. . . . .                                                                   | 86  |
| 3.21 | Count of instances in each group based on problem type. . . . .                                                                                                                 | 87  |
| 3.22 | Mean and standard deviation of exact algorithm runtimes based on problem type partitions. . . . .                                                                               | 88  |
| 3.23 | Mean and standard deviation of distance matrix metrics by problem type. . . . .                                                                                                 | 89  |
| 4.1  | Overview of work to date and our contribution. . . . .                                                                                                                          | 93  |
| 5.1  | Summary of variation and corresponding bounds on service times for service time uncertainty scenarios. . . . .                                                                  | 151 |
| 5.2  | Scenarios with different sources of uncertainty . . . . .                                                                                                                       | 155 |
| 5.3  | mean $\pm$ std. error of total routing costs (lateness plus rejected penalties plus time spent empty) in hours for all service time uncertainty experiments; $n = 33$ . . . . . | 160 |
| 5.4  | mean $\pm$ std. error of total routing costs (lateness plus rejected penalties plus time spent empty) in hours for all job arrival uncertainty experiments; $n = 33$ . . . . .  | 164 |

- 
- 5.5 mean  $\pm$  std. error of total routing costs (lateness plus rejected penalties plus time spent empty) in hours for all service time combined with job arrival uncertainty experiments;  $n = 33$ . . . 166
- A.1 Description of instances generated for verification of models. . . 176



# List of Figures

|     |                                                                                                                                                                                                                 |     |
|-----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 1.1 | Overview of subproblems addressed in this thesis and their relationship to the PDPTW. . . . .                                                                                                                   | 5   |
| 1.2 | Example of a route in a SCP instance (left) and a route in a drayage problem instance (right). . . . .                                                                                                          | 7   |
| 1.3 | One day of jobs in the Netherlands for the Dutch LSP. Black markers indicate customer locations; grey markers indicate terminal locations; and the white marker indicates the home terminal of the LSP. . . . . | 8   |
| 1.4 | Example of the offline optimal and an online solution to the TSP on $\mathbb{R}_0^+$ with two disclosure dates. . . . .                                                                                         | 11  |
| 2.1 | Example of the transformation from an ARP to a node routing problem. . . . .                                                                                                                                    | 25  |
| 3.1 | Depiction of data analysis approach along with the structure of Chapter 3. . . . .                                                                                                                              | 42  |
| 3.2 | Depiction of transformation from ATSP (with 4 nodes) to STSP (with 8 nodes); INF stands for $+\infty$ ; $M = 4$ . . . . .                                                                                       | 46  |
| 3.3 | Depiction of instances with regards to solvability metrics. . . . .                                                                                                                                             | 49  |
| 3.4 | Example of an adjacency matrix as derived from the distance matrix in Figure 2.1 and the graph in Figure 2.1. . . . .                                                                                           | 56  |
| 3.5 | Scatterplot of discriminant function values for each instance. . . . .                                                                                                                                          | 77  |
| 5.1 | Cycles in the MIP solution structure. . . . .                                                                                                                                                                   | 142 |



- 
- 5.2 All locations in the Netherlands. Black markers indicate customer locations; grey markers indicate terminal locations; and the white marker indicates the home terminal of the LSP . . . 148
- 5.3 Number of open time windows for all jobs throughout the working day. . . . . 149
- 5.4 Mean over 33 days of the routing cost components (empty time, rejection penalty, late time) for the agent and on-line optimization solution approaches across five independent service time uncertainty scenarios. . . . . 157
- 5.5 Mean over 33 days of the routing cost components (empty time, rejection penalty, late time) for the agent and on-line optimization solution approaches across five dependent service time uncertainty scenarios. . . . . 159
- 5.6 Mean over 33 days of the routing cost components (empty time, rejection penalty, late time) for the agent and on-line optimization solution approaches across five job arrival uncertainty scenarios. . . . . 162
- 5.7 Mean over 33 days of the routing cost components (empty time, rejection penalty, late time) for the agent and on-line optimization solution approaches across five combined job arrival and service time uncertainty scenarios. . . . . 165

# Bibliography

- Abbott, R. 1990. *Mad Mazes: Intriguing Mind Twisters for Puzzle Buffs, Game Nuts and Other Smart People*. Bob Adams Inc., Publishers.
- Ackermann, W. 1928. Zum hilbertschen aufbau der reellen zahlen. *Mathematische Annalen* **99** 118–133. URL <http://www.springerlink.com/content/px2g58k5j3q45535>.
- A.Coja-Oghlan, S. O. Krumke, T. Nierhoff. 2006. A heuristic for the stacker crane problem on trees which is almost surely exact. *Journal of Algorithms* **61**(1) 1–19. doi:<http://dx.doi.org/10.1016/j.jalgor.2004.07.007>.
- Aho, A. V., J. E. Hopcroft, J. D. Ullman. 1983. *Data Structures and Algorithms*. Addison-Wesley.
- Anily, S., M. Gendreau, G. Laporte. 2000. The swapping problem on a line. *SIAM J. Comput.* **29**(1) 327–335.
- Anily, S., R. Hassin. 1992. The swapping problem. *Networks* **22**(4) 419–433.
- Applegate, D. L., R. E. Bixby, V. Chvátal, W. J. Cook. 2007. *The Traveling Salesman Problem: A Computational Study*. Princeton Series in Applied Mathematics, Princeton University Press.
- Ascheuer, N. 1995. Hamiltonian path problems in the on-line optimization of flexible manufacturing systems. Ph.D. thesis, Technical University of Berlin, Berlin, Germany. URL <http://www.zib.de/ZIBbib/Publications/>.

- Ascheuer, N., M. Jünger, G. Reinelt. 2000. A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints. *Computational Optimization and Applications* **17** 61–84.
- Atallah, M. J., S. R. Kosaraju. 1988. Efficient solutions to some transportation problems with applications to minimizing robot arm travel. *SIAM J. Comput.* **17**(5) 849–869. doi:<http://dx.doi.org/10.1137/0217053>.
- Ausiello, G., V. Bonifaci, L. Laura. 2008. The on-line asymmetric traveling salesman problem. *Journal of Discrete Algorithms* **6**(2) 290–298. doi:10.1016/j.jda.2007.03.002.
- Ausiello, G., E. Feuerstein, S. Leonardi, L. Stougie, M. Talamo. 2001. Algorithms for the on-line travelling salesman. *Algorithmica* **29**(4) 560–581.
- Ball, M. O., M. J. Magazine. 1988. Sequencing of Insertions in Printed Circuit Board Assembly. *Operations Research* **36**(2) 192–201. doi:10.1287/opre.36.2.192. URL <http://or.journal.informs.org/cgi/content/abstract/36/2/192>.
- Ball, M. O., T.L. Magnanti, C.L. Monna, G.L. Nemhauser, eds. 1995. *Network Routing*. Handbooks in Operations Research and Management Science, vol. 8, Elsevier Science, North-Holland, Amsterdam.
- Bellmore, M., G.L. Nemhauser. 1968. The traveling salesman problem: a survey. *Operations Research* **16**(3) 538–558.
- Berbeglia, G., J.-F. Cordeau, I. Gribkovskaia, G. Laporte. 2007. Static pickup and delivery problems: a classification scheme and survey. *TOP* **15**(1) 1–31.
- Berg, J. P. van den, A. J. R. M. Gademann. 1999. Optimal routing in an automated storage/retrieval system with dedicated storage. *IIE Transactions* **31**(5) 407–415.
- Blom, M., S. O. Krumke, W. de Paepe, L. Stougie. 2001. The online tsp against fair adversaries. *INFORMS Journal on Computing* **13**(2) 138–148.

- Breedam, A. van. 1994. An analysis of the behavior of heuristics for the vehicle routing problem for a selection of problems with vehicle-related, customer-related, and time-related constraints. Ph.D. thesis, University of Antwerp.
- BTS. 2007. Transportation statistics annual report 2007. URL [http://www.bts.gov/publications/transportation\\_statistics\\_annual\\_report/2007/html/preface.html](http://www.bts.gov/publications/transportation_statistics_annual_report/2007/html/preface.html).
- Buerckert, H.-J., K. Fischer, G. Vierke. 2000. Holonic transport scheduling with Teletruck. *Applied Artificial Intelligence* **14**(7) 697–725. doi:10.1080/08839510050119253. URL <http://www.ingentaconnect.com/content/tandf/uaai/2000/00000014/00000007/art00005>.
- Burkard, R., V. Deineko, R. van Dal, J. van der Veen, G. Woeginger. 1995. Well-solvable special cases of the tsp: A survey. Manuscript, Institute of Mathematics, University of Technology, Graz, Austria. URL <http://citeseer.ist.psu.edu/burkard95wellsolvable.html>.
- Caris, A., G.K. Janssens. 2009. A local search heuristic for the pre- and end-haulage of intermodal container terminals. *Computers and Operations Research In Press, Corrected Proof*. doi:DOI:10.1016/j.cor.2008.12.007. URL <http://www.sciencedirect.com/science/article/B6VC5-4V74XT3-2/2/c9a223a87b2ad35eea7793b592ae352e>.
- Carpaneto, G., M. Dell’Amico, P. Toth. 1995. Exact solution of large-scale, asymmetric traveling salesman problems. *ACM Trans. Math. Softw.* **21**(4) 394–409. doi:<http://doi.acm.org/10.1145/212066.212081>.
- Carpeneto, G., P. Toth. 1980. Some new branching and bounding criteria for the asymmetric travelling salesman problem. *Management Science* **26**(7) 736–743. URL <http://www.jstor.org/stable/2630706>.
- Chalasani, P., R. Motwani. 1999. Approximating capacitated routing and delivery problems. *SIAM Journal of Computing* **28**(6) 2133–2149. doi:<http://dx.doi.org/10.1137/S0097539795295468>.

- Chartrand, G. 1977. *Graphs and Mathematical Models*. Prindle, Weber & Schmidt, Inc., Boston.
- Chazelle, B. 2000. A minimum spanning tree algorithm with inverse-ackermann type complexity. *Journal of the Association for Computing Machinery* **47**(6) 1028–1047. doi:<http://doi.acm.org/10.1145/355541.355562>.
- Chen, Z.-L., H. Xu. 2006. Dynamic column generation for dynamic vehicle routing with time windows. *Transportation Science* **40**(1) 74–88.
- Cheung, R. K., N. Shi, W. B. Powell, H. P. Simao. 2008. An attributed decision model for cross-border drayage problem. *Transportation Research, Part E* **44** 217–234.
- Christofides, N. 1976. Worst-case analysis of a new heuristic for the travelling salesman problem. Report 388, Graduate School of Industrial Administration, Carnegie Mellon University.
- Cirasella, J., D. S. Johnson, L. A. McGeoch, W. Zhang. 2001. The asymmetric traveling salesman problem: Algorithms, instance generators, and tests. *ALLENEX '01: Revised Papers from the Third International Workshop on Algorithm Engineering and Experimentation*. Springer-Verlag, London, UK, 32–59.
- Cordeau, J.-F., G. Desaulniers, J. Desrosiers, M. M. Solomon, F. Soumis. 2001. VRP with time windows. *The vehicle routing problem*. Society for Industrial and Applied Mathematics, 157–193.
- Cordeau, J.-F., M. Gendreau, G. Laporte, J.-Y. Potvin, F. Semet. 2002. A guide to vehicle routing heuristics. *Journal of the Operational Research Society* **53**(5) 512–522.
- Cordeau, J.-F., G. Laporte, S. Ropke. 2008. Recent models and algorithms for one-to-one pickup and delivery problems. B. L. Golden, S. Raghavan, E. A. Wasil, eds., *Vehicle Routing: Latest Advances and Challenges*. Kluwer, Boston, 327–357.

- Dantzig, G., R. Fulkerson, S. Johnson. 1954. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America* **2**(4) 393–410.
- Dantzig, G. B., R. H. Ramser. 1959. The truck dispatching problem. *Management Science* **6** 80–91.
- Das, R., J. O. Kephart, C. Lefurgy, G. Tesauro, D. W. Levine, H. Chan. 2008. Autonomic multi-agent management of power and performance in data centers. *Proceedings of the 7th international joint conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*. IFAAMAS, Richland, SC, 107–114.
- Davidsson, P., L. Henesey, L. Ramstedt, J. Törnquist, F. Wernstedt. 2005. An analysis of agent-based approaches to transport logistics. *Transportation Research Part C* **13**(4) 255–271.
- Davidsson, Paul, Jan A. Persson, Johan Holmgren. 2007. On the integration of agent-based and mathematical optimization techniques. *Agents and Multi-Agent Systems: Technologies and Applications, LNAI series*, vol. 4496. Springer, 1–10.
- Dorer, K., M. Calisti. 2005. An adaptive solution to dynamic transport optimization. *Proceedings of the 4th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005)*. ACM Press, 45–51. doi:<http://doi.acm.org/10.1145/1082473.1082803>.
- Eiselt, H. A., M. Gendreau, G. Laporte. 1995a. Arc routing problems, part i: The chinese postman problem. *Operations Research* **43**(2) 231–242. URL <http://www.jstor.org/stable/171832>.
- Eiselt, H. A., M. Gendreau, G. Laporte. 1995b. Arc routing problems, part ii: The rural postman problem. *Operations Research* **43**(3) 399–414. URL <http://www.jstor.org/stable/171865>.
- Fischer, K., J. P. Muller, M. Pischel, D. Schier. 1995. A model for cooperative transportation scheduling. *Proceedings of the First International*

- Conference on Multiagent Systems..* AAAI Press / MIT Press, Menlo park, California, 109–116. URL [citeseer.ist.psu.edu/fischer95model.html](http://citeseer.ist.psu.edu/fischer95model.html).
- Fischetti, M., A. Lodi, P. Toth. 2002. Exact methods for the asymmetric traveling salesman problem. G. Gutin, A. P. Punnen, eds., *The Traveling Salesman Problem and Its Variations, Combinatorial Optimization*, vol. 12. Springer, 167–205.
- Fischetti, M., P. Toth. 1992. An additive bounding procedure for the asymmetric travelling salesman problem. *Mathematical Programming* **53**(1) 173–197.
- Fischetti, M., P. Toth, D. Vigo. 1994. A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs. *Operations Research* **42**(5) 846–859. URL <http://www.jstor.org/stable/171544>.
- Flood, M. M. 1956. The traveling-salesman problem. *Operations Research* **4**(1) 61–75.
- Frederickson, G. N. 1993. A note on the complexity of a simple transportation problem. *SIAM Journal on Computing* **22**(1) 57–61. doi:10.1137/0222005. URL <http://link.aip.org/link/?SMJ/22/57/1>.
- Frederickson, G. N., D. J. Guan. 1992. Preemptive ensemble motion planning on a tree. *SIAM Journal on Computing* **21**(6) 1130–1152. doi:10.1137/0221066. URL <http://link.aip.org/link/?SMJ/21/1130/1>.
- Frederickson, G. N., D. J. Guan. 1993. Nonpreemptive ensemble motion planning on a tree. *Journal of Algorithms* **15**(1) 29–60.
- Frederickson, G. N., M. S. Hecht, C. E. Kim. 1978. Approximation algorithms for some routing problems. *SIAM Journal on Computing* **7**(2) 178–193.
- Frieze, A., R. M. Karp, B. Reed. 1995. When is the assignment bound tight for the asymmetric traveling-salesman problem? *SIAM Journal on Computing* **24**(3) 484–493. doi:10.1137/S0097539792235384. URL <http://link.aip.org/link/?SMJ/24/484/1>.

- Gabow, H. N., Z. Galil, T. Spencer, R. E. Tarjan. 1986. Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorica* 109–122 URL <http://www.springerlink.com/content/r238n1x159371v7j>.
- Ghiani, G., F. Guerriero, G. Laporte, R. Musmanno. 2003. Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies. *European Journal of Operational Research* **151**(1) 1–11.
- Gilmore, P. C., R. E. Gomory. 1964. Sequencing a One State-Variable Machine: A Solvable Case of the Traveling Salesman Problem. *Operations Research* **12**(5) 655–679. doi:10.1287/opre.12.5.655. URL <http://or.journal.informs.org/cgi/content/abstract/12/5/655>.
- Golden, B. L., A. A. Assad, eds. 1988. *Vehicle Routing: Methods and Studies, Studies in Management Science and Systems*. Elsevier Science Publishers, Amsterdam.
- Golden, B.L., R. Raghavan, E. Wasil, eds. 2008. *The Vehicle Routing Problem: Latest Advances and New Challenges, Operations Research/Computer Science Interfaces Series*, vol. 43. Springer.
- Gribkovskaia, I., G. Laporte. 2008. One-to-many-to-one single vehicle pickup and delivery problems. B. L. Golden, S. Raghavan, E. A. Wasil, eds., *Vehicle Routing: Latest Advances and Challenges*. Kluwer, Boston, 359–377.
- Gutin, G., A.P. Punnen, eds. 2002. *The Traveling Salesman Problem and Its Variations*. Combinatorial Optimization , Vol. 12, Springer.
- Gutin, G., A. Yeo, A. Zverovich. 2002. Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the tsp. *Discrete Applied Mathematics* **117** 81–86.
- Harary, F. 1959a. An elementary theorem of graphs. *The American Mathematical Monthly* **66**(5) 405–407.



- Harary, F. 1959b. A graph theoretic method for the complete reduction of a matrix with a view toward finding its eigenvalues. *Journal of Mathematics and Physics* **38** 104–111.
- Harary, F. 1959c. Graph theoretic methods in the management sciences. *Management Science* **5**(4) 387–403.
- Harary, F. 1962a. The determinant of the adjacency matrix of a graph. *SIAM Review* **4**(3) 202–210.
- Harary, F. 1962b. A graph theoretic approach to matrix inversion by partitioning. *Numerische Mathematik* **4** 128–135.
- Hellevik, Ottar. 2009. Linear versus logistic regression when the dependent variable is a dichotomy. *Quality and Quantity* **43** 59–74.
- Hernandez-Perez, H., J.-J. Salazar-Gonzalez. 2004. Heuristics for the One-Commodity Pickup-and-Delivery Traveling Salesman Problem. *Transportation Science* **38**(2) 245–255. doi:10.1287/trsc.1030.0086. URL <http://transci.journal.informs.org/cgi/content/abstract/38/2/245>.
- Herroelen, W., R. Leus. 2005. Project scheduling under uncertainty: Survey and research potentials. *European Journal of Operational Research* **165**(2) 289–306.
- Hillier, F. S., G. J. Lieberman. 2001. *Introduction to Operations Research*. McGraw Hill.
- Hochbaum, D. S., ed. 1996. *Approximation Algorithms for NP-hard Problems*. Course Technology.
- Hoen, P. J. 't, J. A. La Poutré. 2003. A decommitment strategy in a competitive multi-agent transportation setting. *Proceedings of 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2003)*. ACM Press, 1010–1011. doi:<http://doi.acm.org/10.1145/860575.860770>.

- Hutzschenreuter, A. K., P. A. N. Bosman, I. Blonk-Altena, J. van Aarle, H. La Poutré. 2008. Agent-based patient admission scheduling in hospitals. *Proceedings of the 7th international joint conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*. IFAAMAS, 45–52.
- Ileri, Y., M. Bazaraa, T. Gifford, G. Nemhauser, J. Sokol, E. Wikum. 2006. An optimization approach for planning daily drayage operations. *Central European Journal of Operations Research* **14** 141–156.
- ILOG, Inc. 1992. Using the CPLEX Callable Library and CPLEX Mixed Integer Library.
- Jaillet, P., M. R. Wagner. 2006. Online routing problems: Value of advanced information as improved competitive ratios. *Transportation Science* **40**(2) 200–210.
- Jakob, M., M. Pěchouček, S. Miles, M. Luck. 2008. Case studies for contract-based systems. *Proceedings of the 7th international joint conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*. IFAAMAS, 55–62.
- Johnson, D. S., G. Gutin, L. A. McGeoch, A. Yeo, W. Zhang, A. Zverovitch. 2002. Experimental analysis of heuristics for the atsp. G. Gutin, A.P. Punnen, eds., *The Traveling Salesman Problem and Its Variations, Combinatorial Optimization*, vol. 12. Springer, 445–487.
- Jonker, R., T. Volgenant. 1983. Transforming asymmetric into symmetric traveling salesman problems. *Operations Research Letters* **2** 161–153.
- Jünger, M., G. Reinelt, G. Rinaldi. 1995. The traveling salesman problem. M. Ball, T. Magnanti, C. Monma, G. Nemhauser, eds., *Network models*. Handbooks in operations research and management science, Elsevier, Amsterdam, 225–330.
- Jünger, M., G. Reinelt, G. Rinaldi. 1997. The traveling salesman problem. M. Dell’Amico, F. Maffioli, S. Martello, eds., *Annotated bibliographies in combinatorial optimization*. Wiley, Chichester, 199–221.

- Kabadi, S. N. 2002. Polynomially solvable cases of the TSP. G. Gutin, A.P. Punnen, eds., *The Traveling Salesman Problem and Its Variations, Combinatorial Optimization*, vol. 12. Springer, 490–583.
- Kindervater, G. A. P., M. W. P. Savelsbergh. 1997. Vehicle routing: Handling edge exchanges. E.H.L. Aarts, J.K. Lenstra, eds., *Local Search in Combinatorial Optimization*. Wiley, Chichester, UK, 337–360.
- Kohout, R., K. Erol. 1999. In-time agent-based vehicle routing with a stochastic improvement heuristic. *Proceedings of the 16th national conference on Artificial Intelligence and the 11th on Innovative Applications of Artificial Intelligence (AAAI/IAAI 1999)*. AAAI press, Menlo Park, CA, USA, 864–869.
- Krauth, E. I. 2008. Real-time planning support: A task-technology fit perspective. Ph.D. thesis, Rotterdam School of Management, Erasmus University, Rotterdam, The Netherlands. URL <http://repub.eur.nl/publications/index/262578220/>.
- Krumke, S. O., D. Rübiger, R. Schrader. 2008. Semi-preemptive routing on trees. *Discrete Appl. Math.* **156**(17) 3298–3304. doi:<http://dx.doi.org/10.1016/j.dam.2008.05.016>.
- Laporte, G. 1997. Modeling and solving several classes of arc routing problems as traveling salesman problems. *Computers and Operations Research* **24**(11) 1057–1061. doi:DOI:10.1016/S0305-0548(97)00013-0. URL <http://www.sciencedirect.com/science/article/B6VC5-3SX0K5F-16/2/202a14f75ea59dbb47610c25825652d1>.
- Laporte, G., M. Gendreau, J.-Y. Potvin, F. Semet. 2000. Classical and modern heuristics for the vehicle routing problem. *International Transactions in Operations Research* **7**(4-5) 285–300.
- Lawler, E. L., J. K. Lenstra, A. H. G. Rinnooy Kan, D. B. Shmoys, eds. 1985. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. John Wiley & Sons Ltd., New York.

- Leong, H. W., M. Liu. 2006. A multi-agent algorithm for vehicle routing problem with time window. *Proceedings of the ACM Symposium on Applied Computing (SAC 2006)*. ACM Press, New York, NY, USA, 106–111. doi: <http://doi.acm.org/10.1145/1141277.1141301>.
- Lin, Shen, B. W. Kernighan. 1973. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research* **21** 498–516.
- Mahmassani, H. S., Y.-J. Kim, P. Jaillet. 2000. Local optimization approaches to solve dynamic commercial fleet management problems. *Transportation Research Record* **1733** 71–79.
- Máhr, T., F. J. Srouf, M. M. de Weerd, R. Zuidwijk. 2010. Can agents measure up? a comparative study of an agent-based and on-line optimization approach for a drayage problem with uncertainty. *Transportation Research: Part C* **18**(1) 99–119.
- Máhr, T., J. Srouf, M. de Weerd, R. Zuidwijk. 2008. Agent performance in vehicle routing when the only thing certain is uncertainty. *Proceedings of the workshop on Agents in Traffic and Transportation (ATT)*.
- Mes, M., M. van der Heijden, A. van Harten. 2007. Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems. *European Journal of Operational Research* **181**(1) 59–75. URL <http://www.sciencedirect.com/science/article/B6VCT-4KV3XT6-1/2/0a71abb449f8943fb58fa984d38a7317>.
- Miller, D. L., J. F. Pekny. 1991. Exact Solution of Large Asymmetric Traveling Salesman Problems. *Science* **251**(4995) 754–761. doi:10.1126/science.251.4995.754. URL <http://www.sciencemag.org/cgi/content/abstract/251/4995/754>.
- Minieka, E. 1979. The Chinese Postman Problem for Mixed Networks. *Management Science* **25**(7) 643–648. doi:10.1287/mnsc.25.7.643. URL <http://mansci.journal.informs.org/cgi/content/abstract/25/7/643>.

- Morlok, E. K., L. N. Spasovic. 1994. Redesigning rail-truck intermodal drayage operations for enhanced service and cost performance. *Journal of Transportation Research Forum* **34**(1) 16–31.
- Morrison, D. F. 1990. *Multivariate Statistical Methods*. 3rd ed. McGraw-Hill Publishing Company.
- Muller, G. 1999. *Intermodal Freight Transportation, 4th Edition*. Eno Transportation Foundation, Inc., Washington D.C.
- Mutyala, S., D. R. Cahill. 1996. Catching up. *Clinical Anatomy* **9** 53–56.
- Namboothiri, R. 2006. Planning container drayage operations at congested seaports. Ph.D. thesis, Georgia Institute of Technology, Atlanta, Georgia, USA. URL [http://etd.gatech.edu/theses/available/etd-05182006-143046/unrestricted/namboothiri\\_rajeev\\_k\\_200608\\_phd.pdf](http://etd.gatech.edu/theses/available/etd-05182006-143046/unrestricted/namboothiri_rajeev_k_200608_phd.pdf).
- Neuman, C., K. Smilowitz. 2002. Strategies for coordinated drayage movements. R. Hall, P. Mirchandani, eds., *NSF Real Time Logistics Workshop*. Long Beach, CA, USA, 864–869.
- Pampel, F. C. 2000. *Logistic Regression: A Primer*. Quantitative Applications in the Social Sciences, Vol. 132, Sage Publications, Inc.
- Papadimitriou, C., K. Steiglitz. 1982. *Combinatorial Optimization: Algorithms and Complexity*. Prentice Hall, Inc.
- Parragh, S., K. Doerner, R. Hartl. 2008a. A survey on pickup and delivery problems: Part i: Transportation between customers and depot. *Journal für Betriebswirtschaft* **58** 21–51.
- Parragh, S., K. Doerner, R. Hartl. 2008b. A survey on pickup and delivery problems: Part ii: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft* **58** 81–117. doi:doi:10.1007/s11301-008-0036-4. URL <http://www.ingentaconnect.com/content/klu/11301/2008/00000058/00000002/00000036>.

- Persson, J. A., P. Davidsson, S. J. Johansson, F. Wernstedt. 2005. Combining agent-based approaches and classical optimization techniques. *Proceedings of the European workshop on Multi-Agent Systems (EUMAS 2005)*. 260–269.
- Perugini, D., D. Lambert, L. Sterling, A. Pearce. 2003. A distributed agent approach to global transportation scheduling. *IEEE/WIC Int. Conf. on Intelligent Agent Technology (IAT 2003)*. 18–24.
- Powell, W. B., P. Jaillet, A. Odoni. 1995. Stochastic and dynamic networks and routing. M. O. Ball, T.L. Magnanti, C.L. Monna, G.L. Nemhauser, eds., *Network Routing, Handbooks in Operations Research and Management Science*, vol. 8. North-Holland, Amsterdam, 141–295.
- Powell, W. B., M. T. Towns, A. Marar. 2000. On the value of optimal myopic solutions for dynamic routing and scheduling problems in the presence of user noncompliance. *Transportation Science* **34**(1) 67–85. doi:<http://dx.doi.org/10.1287/trsc.34.1.67.12283>.
- Psaraftis, H. N. 1988. Dynamic vehicle routing problems. B. L. Golden, A. A. Assad, eds., *Vehicle Routing: Methods and Studies, Studies in Management Science and Systems*, vol. 16. Elsevier Science Publishers, Amsterdam, 223–248.
- Psaraftis, H. N., M. M. Solomon, T. L. Magnanti, T. U. Kim. 1990. Routing and scheduling on a shoreline with release times. *Management Science* **36**(2) 212–223.
- Regan, A. C., H. S. Mahmassani, P. Jaillet. 1995. Improving the efficiency of commercial vehicle operations using real-time information: potential uses and assignment strategies. *Transportation Research Record* **1493** 188–198.
- Regan, A. C., H. S. Mahmassani, P. Jaillet. 1996. Dynamic decision making for commercial fleet operations using real-time information. *Transportation Research Record* **1537** 91–97.

- Rehak, M., M. Pechoucek, P. Celeda, J. Novotny, P. Minarik. 2008. CAM-NEP: agent-based network intrusion detection system. *Proceedings of the 7th international joint conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*. IFAAMAS, Richland, SC, 133–136.
- Reinelt, G. 1994. *The Traveling Salesman, Computational Solutions for TSP Applications, Lecture Notes in Computer Science*, vol. 840. Springer.
- Samuelson, A. 2005. Agents of change: How agent based modeling may transform social science. *OR/MS Today* **32**(1).
- Samuelson, A., M. Charles. 2006. Agent-based simulation comes of age: Software opens up many new areas of application. *OR/MS Today* **33**(4).
- Sandholm, T. W., V. R. Lesser. 2001. Leveled commitment contracts and strategic breach. *Games and Economic Behaviour* **35**(1-2) 212–270. URL [citeseer.ist.psu.edu/article/sandholm01leveled.html](http://citeseer.ist.psu.edu/article/sandholm01leveled.html).
- Scerri, P., T. von Gonten, G. Fudge, S. Owens, K. Sycara. 2008. Transitioning multiagent technology to uav applications. *Proceedings of the 7th international joint conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*. IFAAMAS, Richland, SC, 89–96.
- Schillo, M., C. Kray, K. Fischer. 2002. The eager bidder problem: a fundamental problem of DAI and selected solutions. *Proceedings of 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*. ACM Press, New York, NY, USA, 599–606. doi: <http://doi.acm.org/10.1145/544862.544886>.
- Sgall, J. 1998. On-line scheduling. A. Fiat, G. J. Woeginger, eds., *Online Algorithms: The State of the Art*. Lecture Notes in Computer Science, 1442, Springer, 196–231.
- Smilowitz, K. 2006. Multi-resource routing with flexible tasks: An application in drayage operations. *IIE Transactions* **38**(7) 577–590.

- Solomon, M. M. 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research* **35**(2) 254–265.
- Spicer, J. 2004. *Making Sense of Multivariate Data Analysis: An Intuitive Approach*. Sage Publications, Inc.
- Srour, F. J., R. A. Zuidwijk. 2008. How much is location information worth? a competitive analysis of the online traveling salesman problem with two disclosure dates. *ERIM Report Series* (ERS-2008-075-LIS). URL <http://ssrn.com/abstract=1314164>.
- Stahlbock, R., S. Voß. 2008a. Operations research at container terminals: a literature update. *OR Spectrum* **30** 1–52.
- Stahlbock, R., S. Voß. 2008b. Vehicle routing problems and container terminal operations an update of research. B. Golden, S. Raghavan, E. Wasil, eds., *The Vehicle Routing Problem: Latest Advances and New Challenges, Operations Research/Computer Science Interfaces Series*, vol. 43. Springer, 551–589. doi:10.1007/978-0-387-77778-8\_25. URL <http://www.springerlink.com/content/p85r781013p20668>.
- StatSoft Inc. 2007. *Electronic Statistics Textbook*. StatSoft, Tulsa, OK. URL <http://www.statsoft.com/textbook/stathome.html>.
- Thompson, P. M., H. N. Psaraftis. 1993. Cyclic transfer algorithms for multivehicle routing and scheduling problems. *Operations research* **41**(5) 935–946.
- Toth, P., D. Vigo. 2001. The vehicle routing problem. *Monographs on Discrete Mathematics and Applications 9*. SIAM, Philadelphia.
- Trudeau, R. J. 1993. *Introduction to Graph Theory*. Dover Publications, Inc.
- USDOC. 2004. United states: 2002 economic census, vehicle inventory and use survey.



- Vairaktarakis, G. L. 2003. Simple algorithms for gilmore–gomory’s traveling salesman and related problems. *J. of Scheduling* **6**(6) 499–520. doi:<http://dx.doi.org/10.1023/A:1026200209386>.
- Verbeek, M. J. C. M. 2004. *A Guide to Modern Econometrics, 2nd edition*. John Wiley and Sons, Chichester.
- Vickrey, W. 1961. Counterspeculation, auctions, and competitive sealed tenders. *Journal of Finance* **16** 8–37.
- Vis, I. F. A., K. J. Roodbergen. 2008. Scheduling of container storage and retrieval. *Operations Research* .
- Wagner, M. R. 2006. Online optimization in routing and scheduling. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.
- Wooldridge, M. 2002. *An Introduction to MultiAgent Systems*. John Wiley & Sons, Chichester, England.
- Wooldridge, M., N. Jennings. 1995. Intelligent agents: theory and practice. *Knowledge Engineering Review* **10**(2) 115–152.
- Yang, J., P. Jaillet, H. Mahmassani. 1999. On-line algorithms for truck fleet assignment and scheduling under real-time information. *Transportation Research Record* **1667** 107–113.
- Yang, J., P. Jaillet, H. Mahmassani. 2004. Real-time multi-vehicle truckload pick-up and delivery problems. *Transportation Science* **38**(2) 135–148.
- Young, C., D. S. Johnson, D. R. Karger, M. D. Smith. 1997. Near-optimal intraprocedural branch alignment. *Proceedings 1997 Symposium on Programming Languages, Design, and Implementation*. 183–193.
- Zhang, W., R. E. Korf. 1996. A study of complexity transitions on the asymmetric traveling salesman problem. *Artificial Intelligence* **81** 223–239.

# Index

- NP*, 19
- NP*-complete, 19
- NP*-hard, 20
- P*, 19
- 2-node transformation, 45
  
- Ackermann function, 30
- adjacency matrix, 56
- agent based modeling, 11
- agent-based approach, 120, 124
- approximation algorithms, 21
- arc routing problem, 23, 58
- assignment problem, 26, 36, 63
- asymmetric, 23
- asymmetry, 37, 52
- automated storage/retrieval systems, 29
  
- b-cyclic, k-transfer heuristic, 122
- big “oh” notation, 20
- branch-and-bound, 47
- branch-and-cut, 45
- Breusch-Pagan test, 68
  
- centralized level of control, 120
- Chinese postman problem, 26
- Christofides algorithm, 27
  
- competitive ratio, 10
- computer-human cooperative planning, 126
- concorde, 45
- contract-net protocol, 127
- cyclic transfers, 122
  
- decentralized level of control, 120, 124
- decommitment, 127
- degree, 24
- dependent variable, 66
- directed arcs, 24
- discriminant analysis, 72–73
- dispatcher, 120
- dissection, 1
- distance matrix, 18, 22, 24, 41
  - asymmetric, 23
    - 2-node transformation, 45
  - asymmetry, 52
  - metrics, 41
  - symmetric, 23
  - symmetry, 23, 52
- dray, 2, 15
  - drayage, 2
  - operations, 27, 28

- drayage, *see* dray  
 drayage problem, 6  
 dynamic vehicle routing problems  
     centralized approach, 121  
 eager bidder problem, 127  
 easy, 40, 47  
 evaluation problem, 19  
  
 Gauss-Markov assumptions, 67  
 goodness-of-fit, 67  
 graph theory, 23  
  
 haystack, 39  
 Held-Karp lower bound, 34  
 heuristics, 21, 122  
     constructive methods, 122  
     improvement methods, 122  
 hierarchical clustering, 47  
 Hungarian algorithm, 63  
  
 in-degree, 24  
 independent variable, 66  
 insertion heuristic, 122  
 intermodal freight containers, 15  
  
 Kuhn-Munkres algorithm, 63  
  
 LARGEARC, *see* stacker crane problem  
 LARGEEDGE, *see* stacker crane problem  
  
 Lin-Kernighan Heuristic, 22  
 linear equation, 66  
 linear regression, 66–67  
  
     adjusted R-squared, 67  
     Breusch-Pagan test, 68  
     goodness-of-fit, 67  
     ordinary least squares, 66  
     parameters, 66  
     pseudo R-squared, 79  
     R-squared, 67  
 logistics service provider, 2, 128  
  
 mathematical modeling, 3  
 minimum cost spanning tree problem, 27  
 minimum cost Steiner directed pseudograph problem, 30  
     with node-deficiency, 30  
 minimum spanning tree, 30  
 mixed integer program, 141  
 Move-Left-If-Beneficial algorithm, 98  
 Move-Right-Early-Left-Late algorithm, 99  
 Move-Right-If-Necessary algorithm, 92, 97  
 multi-agent systems, 11, 124  
     for transportation, 124–128  
 multinomial logistic regression, 76–79  
 multivariate analysis of variance, 72  
  
 Nearest-Neighbor Heuristic, 21  
 node routing problem, 24, 58  
  
 offline, 10  
 online algorithm, 10  
 optimal algorithm, 10

- optimization problem, 18
- optimization-based approach, 120
- ordinary least squares, 66
- out-degree, 24
- pick-up and delivery problem
  - dynamic, 119
  - single vehicle, unit-load, 16
  - static, 119
  - with time windows, 4, 118
    - agent-based approach, 130–140
    - arrival-time and service-time uncertainty, 154–155
    - arrival-time uncertainty, 153–154, 167
    - optimization-based approach, 140–146
    - service-time uncertainty, 150–153, 167
- plan/re-plan, 123
- preemption problem, 17, 32–33
- production planning, 126
- R-squared, 67
  - adjusted, 67
  - pseudo R-squared, 79
- re-optimization, 123
- recognition problem, 19
- rural postman problem, 26, 29
- solvability, *see* solvable
- solvable, 41, 47
- stacker crane problem, 6, 17, 18, 26–32
  - approximation algorithm, 26
  - LARGEARC, 26
  - LARGEEDGE, 27
  - on a circular track, 28
  - on a line, 27
  - on a tree, 28
  - randomly generated instances, 34
- stacker crane problems
  - in the “real-world”, 29
- static, 9
- swapping, 127
- swapping problem, 17, 32–33
- symmetry, 23, 37, 52
- transportation problem, 4, 29
  - runtime, 20
- traveling salesman problem, 9, 18, 24, 40
  - asymmetric, 18, 23, 24, 35, 40
    - randomly generated instances, 33
  - asymmetric Steiner, 30
  - concorde, 45
  - Gilmore-Gomory, 27, 29
  - Held-Karp lower bound, 34
  - online, 92
  - runtime, 21
  - symmetric, 23
  - TSPLIB, 41
  - two disclosure dates, 91

- with disclosure dates, online, 93
  - MLIB algorithm, 98
- with release dates, 9, 91
- with release dates on  $\mathbb{R}^+$ , 92
- with release dates, offline, 93
  - TRAVERSE algorithm, 97
- with release dates, online, 93, 97
  - MRIN algorithm, 92, 97
- with two disclosure dates, online, 93
  - fixed amounts of advanced notice, 102
  - MRELL algorithm, 99
  - value of advanced location information, 112
  - variable amounts of advanced notice, 105
- TRAVERSE algorithm, 97
- tree, 27
- tsp\_solve v.1.3.6, 47
- TSPLIB, 41
  
- vehicle routing problem, 4
  - deterministic, 121
  - dynamic, 119, 120
  - real-time, 123
  - static, 119
  - stochastic, 121
- Vickrey auction, 131
  
- worst-case ratio, 10, 91

# Summary

The term dray dates back to the 14th century when it was used commonly to describe a type of very sturdy sideless cart<sup>1</sup>. In the 1700s the word drayage came into use meaning “to transport by a sideless cart”. Today, drayage commonly refers to the transport of containerized cargo to and from port or rail terminals and inland locations. With the phenomenal growth of containerized freight since the container’s introduction in 1956, the drayage industry has also experienced significant growth. For example, the world saw total maritime container traffic grow to approximately 417 million twenty foot equivalent units (TEUs) in 2006<sup>2</sup>.

Unfortunately, the drayage portion of a door-to-door container move tends to be the most costly part of the move. Morlok and Spasovic<sup>3</sup> indicate that up to 40% of the cost for a 900 mile container move can be attributed to the 50 mile drayage portion of the move. There are a variety of reasons for this disproportionate assignment of costs, including a great deal of uncertainty at the interface of modes. For example, trucks moving containers to and from a port terminal are often uncertain as to how long it will take them to pick up a designated container coming from a ship, from the terminal stack, or from customs. Whatever the reason, the effect is the same – without the ability to plan and use transportation capacity intelligently and efficiently a

---

<sup>1</sup>Etymology taken from <http://www.merriam-webster.com/>

<sup>2</sup>The United States Dept. of Transportation, Bureau of Transportation Statistics, <http://www.transtats.bts.gov>.

<sup>3</sup>Morlok, E. and Spasovic, L. (1994). Redesigning rail-truck intermodal drayage operations for enhanced service and cost performance. *Journal of Transportation Research Forum*, 34(1):16-31.

supply chain is only a handful of loose links. We study mechanisms to improve capacity utilization in drayage operations from three perspectives using both empirical and theoretical techniques.

In chapters 2 and 3, we study the first tactic employed in conquering the drayage problem — an empirical analysis of the problem’s geometric structure. Specifically, in the drayage problem considered here, all jobs originate from or are destined to one, of only a few, fixed freight terminals. In this way, jobs can often be sequenced such that the destination of one job is the origin of the next. When considering this structure in the context of single-vehicle routing, a review of relevant literature (Chapter 2), leads us to the understanding that indeed these underlying structural features make drayage problems easier to solve than other single-vehicle routing problems. Using a dataset of over 350 problem instances, we test this hypothesis empirically (Chapter 3).

The second level, and basis for Chapter 4, involves delving deeper into the heart of the drayage problem. It is there that we realize the key to improving efficiency may lay in exploiting all of the pieces of job related information as they arrive. For example, in the case of a dray company at the Port of Rotterdam, the transport company knows almost 80% of their jobs ahead of time, but only learns the release time of those jobs in real-time - thus, we ask the question, what value does advanced location information provide in the absence of advanced job release time information? To study this question from a fundamental perspective, we use competitive analysis to examine several on-line algorithms in the (comparatively) more basic traveling salesman problem with release dates and split information arrival.

Finally, in Chapter 5, we examine agent-based solutions as an agile mechanism for handling uncertainty in drayage operations. We compare a decentralized (agent-based) solution approach to a centralized (on-line optimization) approach. We examine the performance of both systems across four scenarios of job arrival uncertainty and four scenarios of service time duration uncertainty. We conclude that when both job arrival and service times are unknown at the start of the day then an agent-based approach performs competitively with, and sometimes better than, an on-line optimization approach.

# Samenvatting (Summary in Dutch)

Het Engelse woord “dray” dateert uit de 14e eeuw, toen het werd gebruikt om een sterk type open kar<sup>4</sup> te beschrijven. Rond 1700 kwam het woord “drayage” in gebruik voor het vervoer per open kar. Vandaag de dag verwijst “drayage” in het algemeen naar het vervoer van containervracht van en naar een haven of spoorwegeindpunt. Met de fenomenale groei van containervrachtvervoer sinds de introductie van de container in 1956 is ook de drayage-industrie sterk gegroeid. De totale maritieme containervracht is bijvoorbeeld gegroeid tot ongeveer 417 miljoen twintig voet containers (TEUs) in 2006<sup>5</sup>.

Jammer genoeg is het drayage gedeelte van “door-to-door” container bewegingen vaak het duurste deel van het transport. Morlok en Spasovic<sup>6</sup> geven aan dat tot 40% van de totale kosten van een 900 mijl containerbeweging kan worden toegeschreven aan het drayage gedeelte van 50 mijl. Een verscheidenheid aan redenen ligt ten grondslag van dit disproportioneel grote aandeel in de kosten. Bijvoorbeeld de grote onzekerheid in de interactie tussen de transportmodaliteiten; voor vrachtwagens die containers van en naar een haven transporteren is er bijvoorbeeld vaak onzekerheid met betrekking tot de

---

<sup>4</sup>Etymologie uit <http://www.merriam-webster.com/>

<sup>5</sup>The United States Dept. of Transportation, Bureau of Transportation Statistics, <http://www.transtats.bts.gov>.

<sup>6</sup>Morlok, E. and Spasovic, L. (1994). Redesigning rail-truck intermodal drayage operations for enhanced service and cost performance. *Journal of Transportation Research Forum*, 34(1):16-31.



tijd die nodig is om een bepaalde container op te halen. Deze onzekerheid leidt tot veel inefficiëntie in het plannen van meerdere containers per dag. Onafhankelijk van de reden voor de hoge kosten blijft het effect hetzelfde; zonder de mogelijkheid tot plannen en gebruik van transportcapaciteit op een intelligente en efficiënte manier is een “supply chain” slechts een handvol losse “links”. In dit proefschrift bestuderen wij mechanismen om het gebruik van drayage capaciteit te verbeteren vanuit drie verschillende perspectieven met zowel empirische als theoretische technieken.

In hoofdstuk 2 and 3 onderzoeken wij de eerste tactiek om het drayage probleem aan te pakken. Deze tactiek is een empirische analyse van de geometrische structuur van het probleem. Specifiek voor het hier beschouwde drayage probleem is dat alle ritten beginnen of eindigen in één van een kleine verzameling vrachtterminals. Hierdoor kunnen de ritten vaak op een zodanige manier worden gerangschikt dat de bestemming van de ene rit de start van de volgende is. Wanneer we deze structuur in acht nemen in de context van routeringsproblemen met één enkel voertuig, leidt een overzicht van relevante literatuur (Hoofdstuk 2) ons tot het begrip dat dit probleem structurele eigenschappen heeft die het makkelijker oplosbaar maken dan andere routeringsproblemen met één enkel voertuig. Gebruikmakend van een dataset van meer dan 350 van dit type routeringsproblemen testen wij deze hypothese empirisch (Hoofdstuk 3).

Het tweede perspectief, en de basis voor Hoofdstuk 4, richten we onze aandacht op de kern van het drayageprobleem. Daar realiseren we ons dat de sleutel tot het verbeteren van de efficiëntie ligt in het gebruiken van alle ritinformatie op het moment dat deze beschikbaar wordt. Bijvoorbeeld, een drayagebedrijf bij de Haven van Rotterdam kent bijna 80% van de vervoersbewegingen vantevoren, maar komt de tijden van vrijgave van de orders pas in “real-time” te weten. We stellen dan ook de vraag welke waarde de beschikbare ritinformatie heeft wanneer de tijd waarop de ritten worden vrijgegeven niet bekend is. Om deze vraag op een fundamentele manier te bestuderen, analyseren we de waarde van informatie door te kijken naar de prestatie van verschillende online algoritmes in het relatief eenvoudige onderliggende han-

delsreizigersprobleem.

Tot slot, in Hoofdstuk 5, onderzoeken wij agent-gebaseerde oplossingen als flexibele mechanisme om om te gaan met de onzekerheid van de ritaankomst in de context van een drayage casus bij de Haven van Rotterdam. Wij vergelijken een gedecentraliseerde (agent-gebaseerde) oplossingsbenadering met een gecentraliseerde (online optimalisering) benadering. Wij onderzoeken de prestaties van beide systemen in vier onzekerheidsscenario's van zowel de orderaankomst als de ritduur. Wij concluderen dat wanneer zowel de ritaankomst als de rittijden aan het begin van de dag onbekend zijn, een agent-gebaseerde benadering concurrerend is met, en soms beter is dan, een online optimaliseringsbenadering.



# About the author



This thesis was concluded while the author, Faith Jordan Srou (née Ludders), was residing on the campus of the American University of Beirut, in Beirut, Lebanon. What brought her to that milestone and location is an interesting combination of both planning and luck. Born on 20 July 1978 in Seattle, WA, Jordan crossed the United States as her parents moved from one university town to the next — eventually completing highschool in 1996 at Ithaca Highschool in Ithaca, NY (the hometown of Cornell University).

She then struck out on her own to pursue a Bachelor of Arts (BA) in the Classics at Carleton College in Northfield, MN. Somewhere amidst corn fields, snow, and the scent of the Malt-O-meal cereal factory, she however lost interest in Cicero and found a new passion for differential geometry. This passion led to a BA in Mathematics, granted *cum laude*, in June 2000.

Jordan then moved down the road (I-35) to the warmer climes of central Texas. There, she had the pleasure and privilege of working with Dr. Hani Mahmassani and Dr. Patrick Jaillet at The University of Texas at Austin. Just before Christmas 2001, this work culminated in a Masters of Science in the field of Civil Engineering from the department of Transportation Engineering.

After dragging her feet about having to grow up and join the working world, Jordan landed a job as a Transportation Engineer at Science Applications International Corporation (SAIC). Pretending to be an adult in the

corporate world proved to be more fun than expected and also did a great job of filling the coffers. For almost four years, Jordan evaluated an array of new technologies designed to improve the safety, security, and efficiency of transportation. Not only did she gain a deep understanding of many technologies, their strengths and weaknesses, but she also learned about the role of government contracting in America, its strengths and weaknesses. (Amidst all this hard work, Jordan even managed to take the 20th of December, 2002 off to marry Issam Mounir Srour at the Arlington Courthouse in Arlington, Virginia.)

Leaving SAIC in the spring of 2006, Jordan donned the title of “knowledge-migrant” when she followed Issam across the Atlantic to the Netherlands. Having relatively little knowledge of where she was migrating to, it was luck that led her to work with Dr. Steef van de Velde and Dr. Rob Zuidwijk at the the Rotterdam School of Management (RSM), Erasmus University. At RSM, Jordan, serving as a PhD candidate, benefitted from and contributed to both the Department of Decision and Information Sciences and the Department of Management of Technology and Innovation.

Among the benefits, garnered from Jordan’s enrollment at RSM, was the funding to participate in numerous conferences — including presentations at the 2007, 2008, and 2009 annual meetings of the Institute for Operations Research and Management Science (INFORMS), the 2008 annual meeting of the Transportation Research Board (TRB), the 2006 and 2009 editions of the International Workshop on Distribution Logistics (IWDL), the 2007 European Logistics Association (ELA) doctoral workshop, the 2006 Production and Operations Management Society (POMS) conference, and the 2007 Consortium of European Management Schools’ seminar on Supply Chain Management (CEMS SCM), among others. All of these international plane and train rides afforded Jordan ample time to think and write, leading to the (co-)authorship of articles appearing in *Transportation Research Record*, *Transportation Research, Part C: Emerging Technologies*, and *OR/MS Today*. Jordan also contributed to RSM by assisting in the teaching of the master’s level class, *Global Logistics and IT*, in both 2006 and 2007; co-reading (as the first co-reader)

for one masters student; editing, *pro bono publico* (and sometimes *pro cioccolata*), numerous dissertations, articles, applications, business letters, and questionnaires; co-organizing the 2008 Department 6 strategic days retreat; and serving as the PhD representative to the Faculty Council for the academic year 2007-2008.

In Fall of 2008, Jordan followed Issam, even farther from the shores of America, to the coast of Lebanon. There, Issam serves as an assistant professor at the American University of Beirut and Jordan, in between giving the occasional lecture, serves as a part-time chicken farmer.



**ERIM PH.D. SERIES**

**RESEARCH IN MANAGEMENT**

ERIM Electronic Series Portal: <http://hdl.handle.net/1765/1>

Agatz, N.A.H., *Demand Management in E-Fulfillment*, Promotor: Prof.dr.ir. J.A.E.E. van Nunen, EPS-2009-163-LIS, ISBN: 978-90-5892-200-7, <http://hdl.handle.net/1765/15425>

Althuizen, N.A.P., *Analogical Reasoning as a Decision Support Principle for Weakly Structured Marketing Problems*, Promotor: Prof.dr.ir. B. Wierenga, EPS-2006-095-MKT, ISBN: 90-5892-129-8, <http://hdl.handle.net/1765/8190>

Alvarez, H.L., *Distributed Collaborative Learning Communities Enabled by Information Communication Technology*, Promotor: Prof.dr. K. Kumar, EPS-2006-080-LIS, ISBN: 90-5892-112-3, <http://hdl.handle.net/1765/7830>

Appelman, J.H., *Governance of Global Interorganizational Tourism Networks: Changing Forms of Co-ordination between the Travel Agency and Aviation Sector*, Promotors: Prof.dr. F.M. Go & Prof.dr. B. Nootboom, EPS-2004-036-MKT, ISBN: 90-5892-060-7, <http://hdl.handle.net/1765/1199>

Asperen, E. van, *Essays on Port, Container, and Bulk Chemical Logistics Optimization*, Promotor: Prof.dr.ir. R. Dekker, EPS-2009-181-LIS, ISBN: 978-90-5892-222-9, <http://hdl.handle.net/1765/1>

Assem, M.J. van den, *Deal or No Deal? Decision Making under Risk in a Large-Stake TV Game Show and Related Experiments*, Promotor: Prof.dr. J. Spronk, EPS-2008-138-F&A, ISBN: 978-90-5892-173-4, <http://hdl.handle.net/1765/13566>

Baquero, G., *On Hedge Fund Performance, Capital Flows and Investor Psychology*, Promotor: Prof.dr. M.J.C.M. Verbeek, EPS-2006-094-F&A, ISBN: 90-5892-131-X, <http://hdl.handle.net/1765/8192>

Berens, G., *Corporate Branding: The Development of Corporate Associations and their Influence on Stakeholder Reactions*, Promotor: Prof.dr. C.B.M. van Riel, EPS-2004-039-ORG, ISBN: 90-5892-065-8, <http://hdl.handle.net/1765/1273>



Berghe, D.A.F. van den, *Working Across Borders: Multinational Enterprises and the Internationalization of Employment*, Promotors: Prof.dr. R.J.M. van Tulder & Prof.dr. E.J.J. Schenk, EPS-2003-029-ORG, ISBN: 90-5892-05-34, <http://hdl.handle.net/1765/1041>

Berghman, L.A., *Strategic Innovation Capacity: A Mixed Method Study on Deliberate Strategic Learning Mechanisms*, Promotor: Prof.dr. P. Mattyssens, EPS-2006-087-MKT, ISBN: 90-5892-120-4, <http://hdl.handle.net/1765/7991>

Bijman, W.J.J., *Essays on Agricultural Co-operatives: Governance Structure in Fruit and Vegetable Chains*, Promotor: Prof.dr. G.W.J. Hendrikse, EPS-2002-015-ORG, ISBN: 90-5892-024-0, <http://hdl.handle.net/1765/867>

Bispo, A., *Labour Market Segmentation: An investigation into the Dutch hospitality industry*, Promotors: Prof.dr. G.H.M. Evers & Prof.dr. A.R. Thurik, EPS-2007-108-ORG, ISBN: 90-5892-136-9, <http://hdl.handle.net/1765/10283>

Blindenbach-Driessen, F., *Innovation Management in Project-Based Firms*, Promotor: Prof.dr. S.L. van de Velde, EPS-2006-082-LIS, ISBN: 90-5892-110-7, <http://hdl.handle.net/1765/7828>

Boer, C.A., *Distributed Simulation in Industry*, Promotors: Prof.dr. A. de Bruin & Prof.dr. A. Verbraeck, EPS-2005-065-LIS, ISBN: 90-5892-093-3, <http://hdl.handle.net/1765/6925>

Boer, N.I., *Knowledge Sharing within Organizations: A situated and Relational Perspective*, Promotor: Prof.dr. K. Kumar, EPS-2005-060-LIS, ISBN: 90-5892-086-0, <http://hdl.handle.net/1765/6770>

Boer-Sorbán, K., *Agent-Based Simulation of Financial Markets: A modular, Continuous-Time Approach*, Promotor: Prof.dr. A. de Bruin, EPS-2008-119-LIS, ISBN: 90-5892-155-0, <http://hdl.handle.net/1765/10870>

Boon, C.T., *HRM and Fit: Survival of the Fittest!?*, Promotors: Prof.dr. J. Paauwe & Prof.dr. D.N. den Hartog, EPS-2008-129-ORG, ISBN: 978-90-5892-162-8, <http://hdl.handle.net/1765/12606>

Braun, E., *City Marketing: Towards an Integrated Approach*, Promotor: Prof.dr. L. van den Berg, EPS-2008-142-MKT, ISBN: 978-90-5892-180-2, <http://hdl.handle.net/1765/13694>

Brito, M.P. de, *Managing Reverse Logistics or Reversing Logistics Management?* Promotors: Prof.dr.ir. R. Dekker & Prof.dr. M. B. M. de Koster, EPS-2004-035-LIS, ISBN: 90-5892-058-5, <http://hdl.handle.net/1765/1132>

---

Brohm, R., *Polycentric Order in Organizations: A Dialogue between Michael Polanyi and IT-Consultants on Knowledge, Morality, and Organization*, Promotors: Prof.dr. G. W. J. Hendrikse & Prof.dr. H. K. Letiche, EPS-2005-063-ORG, ISBN: 90-5892-095-X, <http://hdl.handle.net/1765/6911>

Brumme, W.-H., *Manufacturing Capability Switching in the High-Tech Electronics Technology Life Cycle*, Promotors: Prof.dr.ir. J.A.E.E. van Nunen & Prof.dr.ir. L.N. Van Wassenhove, EPS-2008-126-LIS, ISBN: 978-90-5892-150-5, <http://hdl.handle.net/1765/12103>

Burgers, J.H., *Managing Corporate Venturing: Multilevel Studies on Project Autonomy, Integration, Knowledge Relatedness, and Phases in the New Business Development Process*, Promotors: Prof.dr.ir. F.A.J. Van den Bosch & Prof.dr. H.W. Volberda, EPS-2008-136-STR, ISBN: 978-90-5892-174-1, <http://hdl.handle.net/1765/13484>

Campbell, R.A.J., *Rethinking Risk in International Financial Markets*, Promotor: Prof.dr. C.G. Koedijk, EPS-2001-005-F&A, ISBN: 90-5892-008-9, <http://hdl.handle.net/1765/306>

Chen, C.-M., *Evaluation and Design of Supply Chain Operations Using DEA*, Promotor: Prof.dr. J.A.E.E. van Nunen, EPS-2009-172-LIS, ISBN: 978-90-5892-209-0, <http://hdl.handle.net/1765/1>

Chen, H., *Individual Mobile Communication Services and Tariffs*, Promotor: Prof.dr. L.F.J.M. Pau, EPS-2008-123-LIS, ISBN: 90-5892-158-1, <http://hdl.handle.net/1765/11141>

Chen, Y., *Labour Flexibility in China's Companies: An Empirical Study*, Promotors: Prof.dr. A. Buitendam & Prof.dr. B. Krug, EPS-2001-006-ORG, ISBN: 90-5892-012-7, <http://hdl.handle.net/1765/307>

Damen, F.J.A., *Taking the Lead: The Role of Affect in Leadership Effectiveness*, Promotor: Prof.dr. D.L. van Knippenberg, EPS-2007-107-ORG, <http://hdl.handle.net/1765/10282>

Daniševská, P., *Empirical Studies on Financial Intermediation and Corporate Policies*, Promotor: Prof.dr. C.G. Koedijk, EPS-2004-044-F&A, ISBN: 90-5892-070-4, <http://hdl.handle.net/1765/1518>

Delporte-Vermeiren, D.J.E., *Improving the Flexibility and Profitability of ICT-enabled Business Networks: An Assessment Method and Tool*, Promotors: Prof. mr. dr. P.H.M. Vervest & Prof.dr.ir. H.W.G.M. van Heck, EPS-2003-020-LIS, ISBN: 90-5892-040-2, <http://hdl.handle.net/1765/359>

Derwall, J.M.M., *The Economic Virtues of SRI and CSR*, Promotor: Prof.dr. C.G. Koedijk, EPS-2007-101-F&A, ISBN: 90-5892-132-8, <http://hdl.handle.net/1765/8986>

Diepen, M. van, *Dynamics and Competition in Charitable Giving*, Promotor: Prof.dr. Ph.H.B.F. Franses, EPS-2009-159-MKT, ISBN: 978-90-5892-188-8, <http://hdl.handle.net/1765/14526>

Dietz, H.M.S., *Managing (Sales)People towards Performance: HR Strategy, Leadership & Teamwork*, Promotor: Prof.dr. G.W.J. Hendrikse, EPS-2009-168-ORG, ISBN: 978-90-5892-210-6, <http://hdl.handle.net/1765/1>

Dijksterhuis, M., *Organizational Dynamics of Cognition and Action in the Changing Dutch and US Banking Industries*, Promotors: Prof.dr.ir. F.A.J. van den Bosch & Prof.dr. H.W. Volberda, EPS-2003-026-STR, ISBN: 90-5892-048-8, <http://hdl.handle.net/1765/1037>

Eijk, A.R. van der, *Behind Networks: Knowledge Transfer, Favor Exchange and Performance*, Promotors: Prof.dr. S.L. van de Velde & Prof.dr.drs. W.A. Dolfsma, EPS-2009-161-LIS, ISBN: 978-90-5892-190-1, <http://hdl.handle.net/1765/14613>

Elstak, M.N., *Flipping the Identity Coin: The Comparative Effect of Perceived, Projected and Desired Organizational Identity on Organizational Identification and Desired Behavior*, Promotor: Prof.dr. C.B.M. van Riel, EPS-2008-117-ORG, ISBN: 90-5892-148-2, <http://hdl.handle.net/1765/10723>

Erken, H.P.G., *Productivity, R&D and Entrepreneurship*, Promotor: Prof.dr. A.R. Thurik, EPS-2008-147-ORG, ISBN: 978-90-5892-179-6, <http://hdl.handle.net/1765/14004>

Fenema, P.C. van, *Coordination and Control of Globally Distributed Software Projects*, Promotor: Prof.dr. K. Kumar, EPS-2002-019-LIS, ISBN: 90-5892-030-5, <http://hdl.handle.net/1765/360>

Fleischmann, M., *Quantitative Models for Reverse Logistics*, Promotors: Prof.dr.ir. J.A.E.E. van Nunen & Prof.dr.ir. R. Dekker, EPS-2000-002-LIS, ISBN: 35-4041-711-7, <http://hdl.handle.net/1765/1044>

Flier, B., *Strategic Renewal of European Financial Incumbents: Coevolution of Environmental Selection, Institutional Effects, and Managerial Intentionality*, Promotors: Prof.dr.ir. F.A.J. van den Bosch & Prof.dr. H.W. Volberda, EPS-2003-033-STR, ISBN: 90-5892-055-0, <http://hdl.handle.net/1765/1071>

---

Fok, D., *Advanced Econometric Marketing Models*, Promotor: Prof.dr. Ph.H.B.F. Franses, EPS-2003-027-MKT, ISBN: 90-5892-049-6, <http://hdl.handle.net/1765/1035>

Ganzaroli, A., *Creating Trust between Local and Global Systems*, Promotors: Prof.dr. K. Kumar & Prof.dr. R.M. Lee, EPS-2002-018-LIS, ISBN: 90-5892-031-3, <http://hdl.handle.net/1765/361>

Gertsen, H.F.M., *Riding a Tiger without Being Eaten: How Companies and Analysts Tame Financial Restatements and Influence Corporate Reputation*, Promotor: Prof.dr. C.B.M. van Riel, EPS-2009-171-ORG, ISBN: 90-5892-214-4, <http://hdl.handle.net/1765/1>

Gilsing, V.A., *Exploration, Exploitation and Co-evolution in Innovation Networks*, Promotors: Prof.dr. B. Nooteboom & Prof.dr. J.P.M. Groenewegen, EPS-2003-032-ORG, ISBN: 90-5892-054-2, <http://hdl.handle.net/1765/1040>

Gijsbers, G.W., *Agricultural Innovation in Asia: Drivers, Paradigms and Performance*, Promotor: Prof.dr. R.J.M. van Tulder, EPS-2009-156-ORG, ISBN: 978-90-5892-191-8, <http://hdl.handle.net/1765/14524>

Gong, Y., *Stochastic Modelling and Analysis of Warehouse Operations*, Promotors: Prof.dr. M.B.M. de Koster & Prof.dr. S.L. van de Velde, EPS-2009-180-LIS, ISBN: 978-90-5892-219-9, <http://hdl.handle.net/1765/1>

Govers, R., *Virtual Tourism Destination Image: Glocal Identities Constructed, Perceived and Experienced*, Promotors: Prof.dr. F.M. Go & Prof.dr. K. Kumar, EPS-2005-069-MKT, ISBN: 90-5892-107-7, <http://hdl.handle.net/1765/6981>

Graaf, G. de, *Tractable Morality: Customer Discourses of Bankers, Veterinarians and Charity Workers*, Promotors: Prof.dr. F. Leijnse & Prof.dr. T. van Willigenburg, EPS-2003-031-ORG, ISBN: 90-5892-051-8, <http://hdl.handle.net/1765/1038>

Greeven, M.J., *Innovation in an Uncertain Institutional Environment: Private Software Entrepreneurs in Hangzhou, China*, Promotor: Prof.dr. B. Krug, EPS-2009-164-ORG, ISBN: 978-90-5892-202-1, <http://hdl.handle.net/1765/15426>

Groot, E.A. de, *Essays on Economic Cycles*, Promotors: Prof.dr. Ph.H.B.F. Franses & Prof.dr. H.R. Commandeur, EPS-2006-091-MKT, ISBN: 90-5892-123-9, <http://hdl.handle.net/1765/8216>

Guenster, N.K., *Investment Strategies Based on Social Responsibility and Bubbles*, Promotor: Prof.dr. C.G. Koedijk, EPS-2008-175-F&A, ISBN: 978-90-5892-206-9, <http://hdl.handle.net/1765/1>

Gutkowska, A.B., *Essays on the Dynamic Portfolio Choice*, Promotor: Prof.dr. A.C.F. Vorst, EPS-2006-085-F&A, ISBN: 90-5892-118-2, <http://hdl.handle.net/1765/7994>

Hagemeyer, R.E., *The Unmasking of the Other*, Promotors: Prof.dr. S.J. Magala & Prof.dr. H.K. Letiche, EPS-2005-068-ORG, ISBN: 90-5892-097-6, <http://hdl.handle.net/1765/6963>

Halderen, M.D. van, *Organizational Identity Expressiveness and Perception Management: Principles for Expressing the Organizational Identity in Order to Manage the Perceptions and Behavioral Reactions of External Stakeholders*, Promotor: Prof.dr. S.B.M. van Riel, EPS-2008-122-ORG, ISBN: 90-5892-153-6, <http://hdl.handle.net/1765/10872>

Hartigh, E. den, *Increasing Returns and Firm Performance: An Empirical Study*, Promotor: Prof.dr. H.R. Commandeur, EPS-2005-067-STR, ISBN: 90-5892-098-4, <http://hdl.handle.net/1765/6939>

Hermans, J.M., *ICT in Information Services; Use and Deployment of the Dutch Securities Trade, 1860-1970*, Promotor: Prof.dr. drs. F.H.A. Janszen, EPS-2004-046-ORG, ISBN 90-5892-072-0, <http://hdl.handle.net/1765/1793>

Hessels, S.J.A., *International Entrepreneurship: Value Creation Across National Borders*, Promotor: Prof.dr. A.R. Thurik, EPS-2008-144-ORG, ISBN: 978-90-5892-181-9, <http://hdl.handle.net/1765/13942>

Heugens, P.P.M.A.R., *Strategic Issues Management: Implications for Corporate Performance*, Promotors: Prof.dr.ir. F.A.J. van den Bosch & Prof.dr. C.B.M. van Riel, EPS-2001-007-STR, ISBN: 90-5892-009-9, <http://hdl.handle.net/1765/358>

Heuvel, W. van den, *The Economic Lot-Sizing Problem: New Results and Extensions*, Promotor: Prof.dr. A.P.L. Wagelmans, EPS-2006-093-LIS, ISBN: 90-5892-124-7, <http://hdl.handle.net/1765/1805>

Hoedemaekers, C.M.W., *Performance, Pinned down: A Lacanian Analysis of Subjectivity at Work*, Promotors: Prof.dr. S. Magala & Prof.dr. D.H. den Hartog, EPS-2008-121-ORG, ISBN: 90-5892-156-7, <http://hdl.handle.net/1765/10871>

Hooghiemstra, R., *The Construction of Reality: Cultural Differences in Self-serving Behaviour in Accounting Narratives*, Promotors: Prof.dr. L.G. van der Tas RA & Prof.dr. A.Th.H. Pruijn, EPS-2003-025-F&A, ISBN: 90-5892-047-X, <http://hdl.handle.net/1765/871>

---

Hu, Y., *Essays on the Governance of Agricultural Products: Cooperatives and Contract Farming*, Promotors: Prof.dr. G.W.J. Hendrkse & Prof.dr. B. Krug, EPS-2007-113-ORG, ISBN: 90-5892-145-1, <http://hdl.handle.net/1765/10535>

Huij, J.J., *New Insights into Mutual Funds: Performance and Family Strategies*, Promotor: Prof.dr. M.C.J.M. Verbeek, EPS-2007-099-F&A, ISBN: 90-5892-134-4, <http://hdl.handle.net/1765/9398>

Huurman, C.I., *Dealing with Electricity Prices*, Promotor: Prof.dr. C.D. Koedijk, EPS-2007-098-F&A, ISBN: 90-5892-130-1, <http://hdl.handle.net/1765/9399>

Iastrebova, K., *Manager's Information Overload: The Impact of Coping Strategies on Decision-Making Performance*, Promotor: Prof.dr. H.G. van Dissel, EPS-2006-077-LIS, ISBN: 90-5892-111-5, <http://hdl.handle.net/1765/7329>

Iwaarden, J.D. van, *Changing Quality Controls: The Effects of Increasing Product Variety and Shortening Product Life Cycles*, Promotors: Prof.dr. B.G. Dale & Prof.dr. A.R.T. Williams, EPS-2006-084-ORG, ISBN: 90-5892-117-4, <http://hdl.handle.net/1765/7992>

Jansen, J.J.P., *Ambidextrous Organizations*, Promotors: Prof.dr.ir. F.A.J. Van den Bosch & Prof.dr. H.W. Volberda, EPS-2005-055-STR, ISBN: 90-5892-081-X, <http://hdl.handle.net/1765/6774>

Jaspers, F.P.H., *Organizing Systemic Innovation*, Promotor: Prof.dr.ir. J.C.M. van den Ende, EPS-2009-160-ORG, ISBN: 978-90-5892-197-), <http://hdl.handle.net/1765/14974>

Jennen, M.G.J., *Empirical Essays on Office Market Dynamics*, Promotors: Prof.dr. C.G. Koedijk & Prof.dr. D. Brounen, EPS-2008-140-F&A, ISBN: 978-90-5892-176-5, <http://hdl.handle.net/1765/13692>

Jiang, T., *Capital Structure Determinants and Governance Structure Variety in Franchising*, Promotors: Prof.dr. G. Hendrikse & Prof.dr. A. de Jong, EPS-2009-158-F&A, ISBN: 978-90-5892-199-4, <http://hdl.handle.net/1765/14975>

Jiao, T., *Essays in Financial Accounting*, Promotor: Prof.dr. G.M.H. Mertens, EPS-2009-176-F&A, ISBN: 978-90-5892-211-3, <http://hdl.handle.net/1765/1>

Jong, C. de, *Dealing with Derivatives: Studies on the Role, Informational Content and Pricing of Financial Derivatives*, Promotor: Prof.dr. C.G. Koedijk, EPS-2003-023-F&A, ISBN: 90-5892-043-7, <http://hdl.handle.net/1765/1043>

Kaa, G. van, *Standard Battles for Complex Systems: Empirical Research on the Home Network*, Promoters: Prof.dr.ir. J. van den Ende & Prof.dr.ir. H.W.G.M. van Heck, EPS-2009-166-ORG, ISBN: 978-90-5892-205-2, <http://hdl.handle.net/1765/1>

Keizer, A.B., *The Changing Logic of Japanese Employment Practices: A Firm-Level Analysis of Four Industries*, Promoters: Prof.dr. J.A. Stam & Prof.dr. J.P.M. Groenewegen, EPS-2005-057-ORG, ISBN: 90-5892-087-9, <http://hdl.handle.net/1765/6667>

Kijkuit, R.C., *Social Networks in the Front End: The Organizational Life of an Idea*, Promotor: Prof.dr. B. Nooteboom, EPS-2007-104-ORG, ISBN: 90-5892-137-6, <http://hdl.handle.net/1765/10074>

Kippers, J., *Empirical Studies on Cash Payments*, Promotor: Prof.dr. Ph.H.B.F. Franses, EPS-2004-043-F&A, ISBN: 90-5892-069-0, <http://hdl.handle.net/1765/1520>

Klein, M.H., *Poverty Alleviation through Sustainable Strategic Business Models: Essays on Poverty Alleviation as a Business Strategy*, Promotor: Prof.dr. H.R. Commandeur, EPS-2008-135-STR, ISBN: 978-90-5892-168-0, <http://hdl.handle.net/1765/13482>

Knapp, S., *The Econometrics of Maritime Safety: Recommendations to Enhance Safety at Sea*, Promotor: Prof.dr. Ph.H.B.F. Franses, EPS-2007-096-ORG, ISBN: 90-5892-127-1, <http://hdl.handle.net/1765/7913>

Kole, E., *On Crises, Crashes and Comovements*, Promoters: Prof.dr. C.G. Koedijk & Prof.dr. M.J.C.M. Verbeek, EPS-2006-083-F&A, ISBN: 90-5892-114-X, <http://hdl.handle.net/1765/7829>

Kooij-de Bode, J.M., *Distributed Information and Group Decision-Making: Effects of Diversity and Affect*, Promotor: Prof.dr. D.L. van Knippenberg, EPS-2007-115-ORG, <http://hdl.handle.net/1765/10722>

Koppius, O.R., *Information Architecture and Electronic Market Performance*, Promoters: Prof.dr. P.H.M. Vervest & Prof.dr.ir. H.W.G.M. van Heck, EPS-2002-013-LIS, ISBN: 90-5892-023-2, <http://hdl.handle.net/1765/921>

Kotlarsky, J., *Management of Globally Distributed Component-Based Software Development Projects*, Promotor: Prof.dr. K. Kumar, EPS-2005-059-LIS, ISBN: 90-5892-088-7, <http://hdl.handle.net/1765/6772>

Krauth, E.I., *Real-Time Planning Support: A Task-Technology Fit Perspective*, Promoters: Prof.dr. S.L. van de Velde & Prof.dr. J. van Hillegersberg, EPS-2008-155-LIS, ISBN: 978-90-5892-193-2, <http://hdl.handle.net/1765/14521>

---

Kuilman, J., *The Re-Emergence of Foreign Banks in Shanghai: An Ecological Analysis*, Promotor: Prof.dr. B. Krug, EPS-2005-066-ORG, ISBN: 90-5892-096-8, <http://hdl.handle.net/1765/6926>

Kwee, Z., *Investigating Three Key Principles of Sustained Strategic Renewal: A Longitudinal Study of Long-Lived Firms*, Promotors: Prof.dr.ir. F.A.J. Van den Bosch & Prof.dr. H.W. Volberda, EPS-2009-174-STR, ISBN: 90-5892-212-0, <http://hdl.handle.net/1765/1>

Langen, P.W. de, *The Performance of Seaport Clusters: A Framework to Analyze Cluster Performance and an Application to the Seaport Clusters of Durban, Rotterdam and the Lower Mississippi*, Promotors: Prof.dr. B. Nooteboom & Prof. drs. H.W.H. Welters, EPS-2004-034-LIS, ISBN: 90-5892-056-9, <http://hdl.handle.net/1765/1133>

Le Anh, T., *Intelligent Control of Vehicle-Based Internal Transport Systems*, Promotors: Prof.dr. M.B.M. de Koster & Prof.dr.ir. R. Dekker, EPS-2005-051-LIS, ISBN: 90-5892-079-8, <http://hdl.handle.net/1765/6554>

Le-Duc, T., *Design and Control of Efficient Order Picking Processes*, Promotor: Prof.dr. M.B.M. de Koster, EPS-2005-064-LIS, ISBN: 90-5892-094-1, <http://hdl.handle.net/1765/6910>

Leeuwen, E.P. van, *Recovered-Resource Dependent Industries and the Strategic Renewal of Incumbent Firm: A Multi-Level Study of Recovered Resource Dependence Management and Strategic Renewal in the European Paper and Board Industry*, Promotors: Prof.dr.ir. F.A.J. Van den Bosch & Prof.dr. H.W. Volberda, EPS-2007-109-STR, ISBN: 90-5892-140-6, <http://hdl.handle.net/1765/10183>

Lentink, R.M., *Algorithmic Decision Support for Shunt Planning*, Promotors: Prof.dr. L.G. Kroon & Prof.dr.ir. J.A.E.E. van Nunen, EPS-2006-073-LIS, ISBN: 90-5892-104-2, <http://hdl.handle.net/1765/7328>

Li, T., *Informedness and Customer-Centric Revenue Management*, Promotors: Prof.dr. P.H.M. Vervest & Prof.dr.ir. H.W.G.M. van Heck, EPS-2009-146-LIS, ISBN: 978-90-5892-195-6, <http://hdl.handle.net/1765/14525>

Liang, G., *New Competition: Foreign Direct Investment and Industrial Development in China*, Promotor: Prof.dr. R.J.M. van Tulder, EPS-2004-047-ORG, ISBN: 90-5892-073-9, <http://hdl.handle.net/1765/1795>

Liere, D.W. van, *Network Horizon and the Dynamics of Network Positions: A Multi-Method Multi-Level Longitudinal Study of Interfirm Networks*, Promotor: Prof.dr. P.H.M. Vervest, EPS-2007-105-LIS, ISBN: 90-5892-139-0, <http://hdl.handle.net/1765/10181>



Loef, J., *Incongruity between Ads and Consumer Expectations of Advertising*, Promotors: Prof.dr. W.F. van Raaij & Prof.dr. G. Antonides, EPS-2002-017-MKT, ISBN: 90-5892-028-3, <http://hdl.handle.net/1765/869>

Maeseneire, W., de, *Essays on Firm Valuation and Value Appropriation*, Promotor: Prof.dr. J.T.J. Smit, EPS-2005-053-F&A, ISBN: 90-5892-082-8, <http://hdl.handle.net/1765/6768>

Londoño, M. del Pilar, *Institutional Arrangements that Affect Free Trade Agreements: Economic Rationality Versus Interest Groups*, Promotors: Prof.dr. H.E. Haralambides & Prof.dr. J.F. Francois, EPS-2006-078-LIS, ISBN: 90-5892-108-5, <http://hdl.handle.net/1765/7578>

Maas, A.A., van der, *Strategy Implementation in a Small Island Context: An Integrative Framework*, Promotor: Prof.dr. H.G. van Dissel, EPS-2008-127-LIS, ISBN: 978-90-5892-160-4, <http://hdl.handle.net/1765/12278>

Maas, K.E.G., *Corporate Social Performance: From Output Measurement to Impact Measurement*, Promotor: Prof.dr. H.R. Commandeur, EPS-2009-182-STR, ISBN: 978-90-5892-225-0, <http://hdl.handle.net/1765/1>

Maeseneire, W., de, *Essays on Firm Valuation and Value Appropriation*, Promotor: Prof.dr. J.T.J. Smit, EPS-2005-053-F&A, ISBN: 90-5892-082-8, <http://hdl.handle.net/1765/6768>

Mandele, L.M., van der, *Leadership and the Inflection Point: A Longitudinal Perspective*, Promotors: Prof.dr. H.W. Volberda & Prof.dr. H.R. Commandeur, EPS-2004-042-STR, ISBN: 90-5892-067-4, <http://hdl.handle.net/1765/1302>

Meer, J.R. van der, *Operational Control of Internal Transport*, Promotors: Prof.dr. M.B.M. de Koster & Prof.dr.ir. R. Dekker, EPS-2000-001-LIS, ISBN: 90-5892-004-6, <http://hdl.handle.net/1765/859>

Mentink, A., *Essays on Corporate Bonds*, Promotor: Prof.dr. A.C.F. Vorst, EPS-2005-070-F&A, ISBN: 90-5892-100-X, <http://hdl.handle.net/1765/7121>

Meyer, R.J.H., *Mapping the Mind of the Strategist: A Quantitative Methodology for Measuring the Strategic Beliefs of Executives*, Promotor: Prof.dr. R.J.M. van Tulder, EPS-2007-106-ORG, ISBN: 978-90-5892-141-3, <http://hdl.handle.net/1765/10182>

---

Miltenburg, P.R., *Effects of Modular Sourcing on Manufacturing Flexibility in the Automotive Industry: A Study among German OEMs*, Promotors: Prof.dr. J. Paauwe & Prof.dr. H.R. Commandeur, EPS-2003-030-ORG, ISBN: 90-5892-052-6, <http://hdl.handle.net/1765/1039>

Moerman, G.A., *Empirical Studies on Asset Pricing and Banking in the Euro Area*, Promotor: Prof.dr. C.G. Koedijk, EPS-2005-058-F&A, ISBN: 90-5892-090-9, <http://hdl.handle.net/1765/6666>

Moitra, D., *Globalization of R&D: Leveraging Offshoring for Innovative Capability and Organizational Flexibility*, Promotor: Prof.dr. K. Kumar, EPS-2008-150-LIS, ISBN: 978-90-5892-184-0, <http://hdl.handle.net/1765/14081>

Mol, M.M., *Outsourcing, Supplier-relations and Internationalisation: Global Source Strategy as a Chinese Puzzle*, Promotor: Prof.dr. R.J.M. van Tulder, EPS-2001-010-ORG, ISBN: 90-5892-014-3, <http://hdl.handle.net/1765/355>

Mom, T.J.M., *Managers' Exploration and Exploitation Activities: The Influence of Organizational Factors and Knowledge Inflows*, Promotors: Prof.dr.ir. F.A.J. Van den Bosch & Prof.dr. H.W. Volberda, EPS-2006-079-STR, ISBN: 90-5892-116-6, <http://hdl.handle.net/1765>

Moonen, J.M., *Multi-Agent Systems for Transportation Planning and Coordination*, Promotors: Prof.dr. J. van Hillegersberg & Prof.dr. S.L. van de Velde, EPS-2009-177-LIS, ISBN: 978-90-5892-216-8, <http://hdl.handle.net/1>

Mulder, A., *Government Dilemmas in the Private Provision of Public Goods*, Promotor: Prof.dr. R.J.M. van Tulder, EPS-2004-045-ORG, ISBN: 90-5892-071-2, <http://hdl.handle.net/1765/1790>

Muller, A.R., *The Rise of Regionalism: Core Company Strategies Under The Second Wave of Integration*, Promotor: Prof.dr. R.J.M. van Tulder, EPS-2004-038-ORG, ISBN: 90-5892-062-3, <http://hdl.handle.net/1765/1272>

Nalbantov G.I., *Essays on Some Recent Penalization Methods with Applications in Finance and Marketing*, Promotor: Prof. dr P.J.F. Groenen, EPS-2008-132-F&A, ISBN: 978-90-5892-166-6, <http://hdl.handle.net/1765/13319>

Nederveen Pieterse, A., *Goal Orientation in Teams: The Role of Diversity*, Promotor: Prof.dr. D.L. van Knippenberg, EPS-2009-162-ORG, <http://hdl.handle.net/1765/15240>

Nguyen, T.T., *Capital Structure, Strategic Competition, and Governance*, Promotor: Prof.dr. A. de Jong, EPS-2008-148-F&A, ISBN: 90-5892-178-9, <http://hdl.handle.net/1765/14005>

Niessen, E.M.M.I., *Regulation, Governance and Adaptation: Governance Transformations in the Dutch and French Liberalizing Electricity Industries*, Promotors: Prof.dr. A. Jolink & Prof.dr. J.P.M. Groenewegen, EPS-2009-170-ORG, ISBN: 978-90-5892-208-3, <http://hdl.handle.net/1765/1>

Nieuwenboer, N.A. den, *Seeing the Shadow of the Self*, Promotor: Prof.dr. S.P. Kaptein, EPS-2008-151-ORG, ISBN: 978-90-5892-182-6, <http://hdl.handle.net/1765/14223>

Ning, H., *Hierarchical Portfolio Management: Theory and Applications*, Promotor: Prof.dr. J. Spronk, EPS-2007-118-F&A, ISBN: 90-5892-152-9, <http://hdl.handle.net/1765/10868>

Noeverman, J., *Management Control Systems, Evaluative Style, and Behaviour: Exploring the Concept and Behavioural Consequences of Evaluative Style*, Promotors: Prof.dr. E.G.J. Vosselman & Prof.dr. A.R.T. Williams, EPS-2007-120-F&A, ISBN: 90-5892-151-2, <http://hdl.handle.net/1765/10869>

Nuijten, I., *Servant Leadership: Paradox or Diamond in the Rough? A Multidimensional Measure and Empirical Evidence*, Promotor: Prof.dr. D.L. van Knippenberg, EPS-2009-183-ORG, <http://hdl.handle.net/1765/1>

Oosterhout, J., van, *The Quest for Legitimacy: On Authority and Responsibility in Governance*, Promotors: Prof.dr. T. van Willigenburg & Prof.mr. H.R. van Gunsteren, EPS-2002-012-ORG, ISBN: 90-5892-022-4, <http://hdl.handle.net/1765/362>

Oostrum, J.M., van, *Applying Mathematical Models to Surgical Patient Planning*, Promotor: Prof.dr. A.P.M. Wagelmans, EPS-2009-179-LIS, ISBN: 978-90-5892-217-5, <http://hdl.handle.net/1765/1>

Paape, L., *Corporate Governance: The Impact on the Role, Position, and Scope of Services of the Internal Audit Function*, Promotors: Prof.dr. G.J. van der Pijl & Prof.dr. H. Commandeur, EPS-2007-111-MKT, ISBN: 90-5892-143-7, <http://hdl.handle.net/1765/10417>

Pak, K., *Revenue Management: New Features and Models*, Promotor: Prof.dr.ir. R. Dekker, EPS-2005-061-LIS, ISBN: 90-5892-092-5, <http://hdl.handle.net/1765/362/6771>

---

Pattikawa, L.H., *Innovation in the Pharmaceutical Industry: Evidence from Drug Introduction in the U.S.*, Promotors: Prof.dr. H.R.Commandeur, EPS-2007-102-MKT, ISBN: 90-5892-135-2, <http://hdl.handle.net/1765/9626>

Peeters, L.W.P., *Cyclic Railway Timetable Optimization*, Promotors: Prof.dr. L.G. Kroon & Prof.dr.ir. J.A.E.E. van Nunen, EPS-2003-022-LIS, ISBN: 90-5892-042-9, <http://hdl.handle.net/1765/429>

Pietersz, R., *Pricing Models for Bermudan-style Interest Rate Derivatives*, Promotors: Prof.dr. A.A.J. Pelsser & Prof.dr. A.C.F. Vorst, EPS-2005-071-F&A, ISBN: 90-5892-099-2, <http://hdl.handle.net/1765/7122>

Poel, A.M. van der, *Empirical Essays in Corporate Finance and Financial Reporting*, Promotors: Prof.dr. A. de Jong & Prof.dr. G.M.H. Mertens, EPS-2007-133-F&A, ISBN: 978-90-5892-165-9, <http://hdl.handle.net/1765/13320>

Popova, V., *Knowledge Discovery and Monotonicity*, Promotor: Prof.dr. A. de Bruin, EPS-2004-037-LIS, ISBN: 90-5892-061-5, <http://hdl.handle.net/1765/1201>

Pouchkarev, I., *Performance Evaluation of Constrained Portfolios*, Promotors: Prof.dr. J. Spronk & Dr. W.G.P.M. Hallerbach, EPS-2005-052-F&A, ISBN: 90-5892-083-6, <http://hdl.handle.net/1765/6731>

Prins, R., *Modeling Consumer Adoption and Usage of Value-Added Mobile Services*, Promotors: Prof.dr. Ph.H.B.F. Franses & Prof.dr. P.C. Verhoef, EPS-2008-128-MKT, ISBN: 978/90-5892-161-1, <http://hdl.handle.net/1765/12461>

Puvanavari Ratnasingam, P., *Interorganizational Trust in Business to Business E-Commerce*, Promotors: Prof.dr. K. Kumar & Prof.dr. H.G. van Dissel, EPS-2001-009-LIS, ISBN: 90-5892-017-8, <http://hdl.handle.net/1765/356>

Quak, H.J., *Sustainability of Urban Freight Transport: Retail Distribution and Local Regulation in Cities*, Promotor: Prof.dr. M.B.M. de Koster, EPS-2008-124-LIS, ISBN: 978-90-5892-154-3, <http://hdl.handle.net/1765/11990>

Quariguasi Frota Neto, J., *Eco-efficient Supply Chains for Electrical and Electronic Products*, Promotors: Prof.dr.ir. J.A.E.E. van Nunen & Prof.dr.ir. H.W.G.M. van Heck, EPS-2008-152-LIS, ISBN: 978-90-5892-192-5, <http://hdl.handle.net/1765/14785>

Radkevitch, U.L., *Online Reverse Auction for Procurement of Services*, Promotor: Prof.dr.ir. H.W.G.M. van Heck, EPS-2008-137-LIS, ISBN: 978-90-5892-171-0, <http://hdl.handle.net/1765/13497>

Rinsum, M. van, *Performance Measurement and Managerial Time Orientation*, Promotor: Prof.dr. F.G.H. Hartmann, EPS-2006-088-F&A, ISBN: 90-5892-121-2, <http://hdl.handle.net/1765/7993>

Romero Morales, D., *Optimization Problems in Supply Chain Management*, Promotors: Prof.dr.ir. J.A.E.E. van Nunen & Dr. H.E. Romeijn, EPS-2000-003-LIS, ISBN: 90-9014078-6, <http://hdl.handle.net/1765/865>

Roodbergen, K.J., *Layout and Routing Methods for Warehouses*, Promotors: Prof.dr. M.B.M. de Koster & Prof.dr.ir. J.A.E.E. van Nunen, EPS-2001-004-LIS, ISBN: 90-5892-005-4, <http://hdl.handle.net/1765/861>

Rook, L., *Imitation in Creative Task Performance*, Promotor: Prof.dr. D.L. van Knippenberg, EPS-2008-125-ORG, <http://hdl.handle.net/1765/11555>

Rosmalen, J. van, *Segmentation and Dimension Reduction: Exploratory and Model-Based Approaches*, Promotor: Prof.dr. P.J.F. Groenen, EPS-2009-165-MKT, ISBN: 978-90-5892-201-4, <http://hdl.handle.net/1765/15536>

Rus, D., *The Dark Side of Leadership: Exploring the Psychology of Leader Self-serving Behavior*, Promotor: Prof.dr. D.L. van Knippenberg, EPS-2009-178-ORG, <http://hdl.handle.net/1765/1>

Samii, R., *Leveraging Logistics Partnerships: Lessons from Humanitarian Organizations*, Promotors: Prof.dr.ir. J.A.E.E. van Nunen & Prof.dr.ir. L.N. Van Wassenhove, EPS-2008-153-LIS, ISBN: 978-90-5892-186-4, <http://hdl.handle.net/1765/14519>

Schaik, D. van, *M&A in Japan: An Analysis of Merger Waves and Hostile Takeovers*, Promotors: Prof.dr. J. Spronk & Prof.dr. J.P.M. Groenewegen, EPS-2008-141-F&A, ISBN: 978-90-5892-169-7, <http://hdl.handle.net/1765/13693>

Schauten, M.B.J., *Valuation, Capital Structure Decisions and the Cost of Capital*, Promotors: Prof.dr. J. Spronk & Prof.dr. D. van Dijk, EPS-2008-134-F&A, ISBN: 978-90-5892-172-7, <http://hdl.handle.net/1765/13480>

Schramade, W.L.J., *Corporate Bonds Issuers*, Promotor: Prof.dr. A. De Jong, EPS-2006-092-F&A, ISBN: 90-5892-125-5, <http://hdl.handle.net/1765/8191>

Schweizer, T.S., *An Individual Psychology of Novelty-Seeking, Creativity and Innovation*, Promotor: Prof.dr. R.J.M. van Tulder, EPS-2004-048-ORG, ISBN: 90-5892-077-1, <http://hdl.handle.net/1765/1818>

---

Six, F.E., *Trust and Trouble: Building Interpersonal Trust Within Organizations*, Promotors: Prof.dr. B. Nooteboom & Prof.dr. A.M. Sorge, EPS-2004-040-ORG, ISBN: 90-5892-064-X, <http://hdl.handle.net/1765/1271>

Slager, A.M.H., *Banking across Borders*, Promotors: Prof.dr. R.J.M. van Tulder & Prof.dr. D.M.N. van Wensveen, EPS-2004-041-ORG, ISBN: 90-5892-066-6, <http://hdl.handle.net/1765/1301>

Sloot, L., *Understanding Consumer Reactions to Assortment Unavailability*, Promotors: Prof.dr. H.R. Commandeur, Prof.dr. E. Peelen & Prof.dr. P.C. Verhoef, EPS-2006-074-MKT, ISBN: 90-5892-102-6, <http://hdl.handle.net/1765/7438>

Smit, W., *Market Information Sharing in Channel Relationships: Its Nature, Antecedents and Consequences*, Promotors: Prof.dr.ir. G.H. van Bruggen & Prof.dr.ir. B. Wierenga, EPS-2006-076-MKT, ISBN: 90-5892-106-9, <http://hdl.handle.net/1765/7327>

Sonnenberg, M., *The Signalling Effect of HRM on Psychological Contracts of Employees: A Multi-level Perspective*, Promotor: Prof.dr. J. Paauwe, EPS-2006-086-ORG, ISBN: 90-5892-119-0, <http://hdl.handle.net/1765/7995>

Speklé, R.F., *Beyond Generics: A closer Look at Hybrid and Hierarchical Governance*, Promotor: Prof.dr. M.A. van Hoepen RA, EPS-2001-008-F&A, ISBN: 90-5892-011-9, <http://hdl.handle.net/1765/357>

Stam, D.A., *Managing Dreams and Ambitions: A Psychological Analysis of Vision Communication*, Promotor: Prof.dr. D.L. van Knippenberg, EPS-2008-149-ORG, <http://hdl.handle.net/1765/14080>

Stienstra, M., *Strategic Renewal in Regulatory Environments: How Inter- and Intra-organisational Institutional Forces Influence European Energy Incumbent Firms*, Promotors: Prof.dr.ir. F.A.J. Van den Bosch & Prof.dr. H.W. Volberda, EPS-2008-145-STR, ISBN: 978-90-5892-184-0, <http://hdl.handle.net/1765/13943>

Sweldens, S.T.L.R., *Evaluative Conditioning 2.0: Direct versus Associative Transfer of Affect to Brands*, Promotor: Prof.dr. S.M.J. van Osselaer, EPS-2009-167-MKT, ISBN: 978-90-5892-204-5, <http://hdl.handle.net/1765/1>

Szkudlarek, B.A., *Spinning the Web of Reentry: ? connecting reentry training theory and practice*, Promotor: Prof.dr. S.J. Magala, EPS-2008-143-ORG, ISBN: 978-90-5892-177-2, <http://hdl.handle.net/1765/13695>

Teunter, L.H., *Analysis of Sales Promotion Effects on Household Purchase Behavior*, Promotors: Prof.dr.ir. B. Wierenga & Prof.dr. T. Kloek, EPS-2002-016-MKT, ISBN: 90-5892-029-1, <http://hdl.handle.net/1765/868>

Tims, B., *Empirical Studies on Exchange Rate Puzzles and Volatility*, Promotor: Prof.dr. C.G. Koedijk, EPS-2006-089-F&A, ISBN: 90-5892-113-1, <http://hdl.handle.net/1765/8066>

Tuk, M.A., *Is Friendship Silent When Money Talks? How Consumers Respond to Word-of-Mouth Marketing*, Promotors: Prof.dr.ir. A. Smidts & Prof.dr. D.H.J. Wigboldus, EPS-2008-130-MKT, ISBN: 978-90-5892-164-2, <http://hdl.handle.net/1765/12702>

Valck, K. de, *Virtual Communities of Consumption: Networks of Consumer Knowledge and Companionship*, Promotors: Prof.dr.ir. G.H. van Bruggen & Prof.dr.ir. B. Wierenga, EPS-2005-050-MKT, ISBN: 90-5892-078-X, <http://hdl.handle.net/1765/6663>

Valk, W. van der, *Buyer-Seller Interaction Patterns During Ongoing Service Exchange*, Promotors: Prof.dr. J.Y.F. Wynstra & Prof.dr.ir. B. Axelsson, EPS-2007-116-MKT, ISBN: 90-5892-146-8, <http://hdl.handle.net/1765/10856>

Verheul, I., *Is There a (Fe)male Approach? Understanding Gender Differences in Entrepreneurship*, Prof.dr. A.R. Thurik, EPS-2005-054-ORG, ISBN: 90-5892-080-1, <http://hdl.handle.net/1765/2005>

Verwijmeren, P., *Empirical Essays on Debt, Equity, and Convertible Securities*, Promotors: Prof.dr. A. de Jong & Prof.dr. M.J.C.M. Verbeek, EPS-2009-154-F&A, ISBN: 978-90-5892-187-1, <http://hdl.handle.net/1765/14312>

Vis, I.F.A., *Planning and Control Concepts for Material Handling Systems*, Promotors: Prof.dr. M.B.M. de Koster & Prof.dr.ir. R. Dekker, EPS-2002-014-LIS, ISBN: 90-5892-021-6, <http://hdl.handle.net/1765/866>

Vlaar, P.W.L., *Making Sense of Formalization in Interorganizational Relationships: Beyond Coordination and Control*, Promotors: Prof.dr.ir. F.A.J. Van den Bosch & Prof.dr. H.W. Volberda, EPS-2006-075-STR, ISBN 90-5892-103-4, <http://hdl.handle.net/1765/7326>

Vliet, P. van, *Downside Risk and Empirical Asset Pricing*, Promotor: Prof.dr. G.T. Post, EPS-2004-049-F&A, ISBN: 90-5892-07-55, <http://hdl.handle.net/1765/1819>

Vlist, P. van der, *Synchronizing the Retail Supply Chain*, Promotors: Prof.dr.ir. J.A.E.E. van Nunen & Prof.dr. A.G. de Kok, EPS-2007-110-LIS, ISBN: 90-5892-142-0, <http://hdl.handle.net/1765/10418>

---

Vries-van Ketel E. de, *How Assortment Variety Affects Assortment Attractiveness: A Consumer Perspective*, Promotors: Prof.dr. G.H. van Bruggen & Prof.dr.ir. A. Smidts, EPS-2006-072-MKT, ISBN: 90-5892-101-8, <http://hdl.handle.net/1765/7193>

Vromans, M.J.C.M., *Reliability of Railway Systems*, Promotors: Prof.dr. L.G. Kroon, Prof.dr.ir. R. Dekker & Prof.dr.ir. J.A.E.E. van Nunen, EPS-2005-062-LIS, ISBN: 90-5892-089-5, <http://hdl.handle.net/1765/6773>

Vroomen, B.L.K., *The Effects of the Internet, Recommendation Quality and Decision Strategies on Consumer Choice*, Promotor: Prof.dr. Ph.H.B.F. Franses, EPS-2006-090-MKT, ISBN: 90-5892-122-0, <http://hdl.handle.net/1765/8067>

Waal, T. de, *Processing of Erroneous and Unsafe Data*, Promotor: Prof.dr.ir. R. Dekker, EPS-2003-024-LIS, ISBN: 90-5892-045-3, <http://hdl.handle.net/1765/870>

Wall, R.S., *Netscape: Cities and Global Corporate Networks*, Promotor: Prof.dr. G.A. van der Knaap, EPS-2009-169-ORG, ISBN: 978-90-5892-207-6, <http://hdl.handle.net/1765/1>

Watkins Fassler, K., *Macroeconomic Crisis and Firm Performance*, Promotors: Prof.dr. J. Spronk & Prof.dr. D.J. van Dijk, EPS-2007-103-F&A, ISBN: 90-5892-138-3, <http://hdl.handle.net/1765/10065>

Weerdt, N.P. van der, *Organizational Flexibility for Hypercompetitive Markets: Empirical Evidence of the Composition and Context Specificity of Dynamic Capabilities and Organization Design Parameters*, Promotor: Prof.dr. H.W. Volberda, EPS-2009-173-STR, ISBN: 978-90-5892-215-1, <http://hdl.handle.net/1765/1>

Wennekers, A.R.M., *Entrepreneurship at Country Level: Economic and Non-Economic Determinants*, Promotor: Prof.dr. A.R. Thurik, EPS-2006-81-ORG, ISBN: 90-5892-115-8, <http://hdl.handle.net/1765/7982>

Wielemaker, M.W., *Managing Initiatives: A Synthesis of the Conditioning and Knowledge-Creating View*, Promotors: Prof.dr. H.W. Volberda & Prof.dr. C.W.F. Baden-Fuller, EPS-2003-28-STR, ISBN: 90-5892-050-X, <http://hdl.handle.net/1765/1042>

Wijk, R.A.J.L. van, *Organizing Knowledge in Internal Networks: A Multilevel Study*, Promotor: Prof.dr.ir. F.A.J. van den Bosch, EPS-2003-021-STR, ISBN: 90-5892-039-9, <http://hdl.handle.net/1765/347>



Wolters, M.J.J., *The Business of Modularity and the Modularity of Business*, Promoters: Prof. mr. dr. P.H.M. Vervest & Prof. dr. ir. H.W.G.M. van Heck, EPS-2002-011-LIS, ISBN: 90-5892-020-8, <http://hdl.handle.net/1765/920>

Wubben, M.J.J., *Social Functions of Emotions in Social Dilemmas*, Promoters: Prof. D. De Cremer & Prof. dr. E. van Dijk, EPS-2009-187-ORG, <http://hdl.handle.net/1765/1>

Yang, J., *Towards the Restructuring and Co-ordination Mechanisms for the Architecture of Chinese Transport Logistics*, Promotor: Prof.dr. H.E. Harlambides, EPS-2009-157-LIS, ISBN: 978-90-5892-198-7, <http://hdl.handle.net/1765/14527>

Yu, M., *Enhancing Warehouse Performance by Efficient Order Picking*, Promotor: Prof.dr. M.B.M. de Koster, EPS-2008-139-LIS, ISBN: 978-90-5892-167-3, <http://hdl.handle.net/1765/13691>

Zhang, X., *Strategizing of Foreign Firms in China: An Institution-based Perspective*, Promotor: Prof.dr. B. Krug, EPS-2007-114-ORG, ISBN: 90-5892-147-5, <http://hdl.handle.net/1765/10721>

Zhu, Z., *Essays on China's Tax System*, Promoters: Prof.dr. B. Krug & Prof.dr. G.W.J. Hendrikse, EPS-2007-112-ORG, ISBN: 90-5892-144-4, <http://hdl.handle.net/1765/10502>

Zwart, G.J. de, *Empirical Studies on Financial Markets: Private Equity, Corporate Bonds and Emerging Markets*, Promoters: Prof.dr. M.J.C.M. Verbeek & Prof.dr. D.J.C. van Dijk, EPS-2008-131-F&A, ISBN: 978-90-5892-163-5, <http://hdl.handle.net/1765/12703>

## DISSECTING DRAYAGE AN EXAMINATION OF STRUCTURE, INFORMATION, AND CONTROL IN DRAYAGE OPERATIONS

The term dray dates back to the 14th century when it was used commonly to describe a type of very sturdy sideless cart. In the 1700s the word drayage came into use meaning "to transport by a sideless cart". Today, drayage commonly refers to the transport of containerized cargo to and from port or rail terminals and inland locations. With the phenomenal growth of containerized freight since the container's introduction in 1956, the drayage industry has also experienced significant growth. In fact, according to the Bureau for Transportation Statistics, the world saw total maritime container traffic grow to approximately 417 million twenty foot equivalent units (TEUs) in 2006.

Unfortunately, the drayage portion of a door-to-door container move tends to be the most costly part of the move. There are a variety of reasons for this disproportionate assignment of costs, including a great deal of uncertainty at the interface of modes. For example, trucks moving containers to and from a port terminal are often uncertain as to how long it will take them to pick up a designated container coming from a ship, from the terminal stack, or from customs. This uncertainty leads to much difficulty and inefficiency in planning a profitable routing for multiple containers in one day. We study this problem from three perspectives using both empirical and theoretical techniques.

### ERIM

The Erasmus Research Institute of Management (ERIM) is the Research School (Onderzoekschool) in the field of management of the Erasmus University Rotterdam. The founding participants of ERIM are Rotterdam School of Management (RSM), and the Erasmus School of Economics (ESE). ERIM was founded in 1999 and is officially accredited by the Royal Netherlands Academy of Arts and Sciences (KNAW). The research undertaken by ERIM is focused on the management of the firm in its environment, its intra- and interfirm relations, and its business processes in their interdependent connections.

The objective of ERIM is to carry out first rate research in management, and to offer an advanced doctoral programme in Research in Management. Within ERIM, over three hundred senior researchers and PhD candidates are active in the different research programmes. From a variety of academic backgrounds and expertises, the ERIM community is united in striving for excellence and working at the forefront of creating new business knowledge.



## ERIM PhD Series Research in Management

Erasmus Research Institute of Management - ERIM  
Rotterdam School of Management (RSM)  
Erasmus School of Economics (ESE)  
P.O. Box 1738, 3000 DR Rotterdam  
The Netherlands

Tel. +31 10 408 11 82  
Fax +31 10 408 96 40  
E-mail [info@erim.eur.nl](mailto:info@erim.eur.nl)  
Internet [www.erim.eur.nl](http://www.erim.eur.nl)