Sequential Auctions for Full Truckload Allocation

Martijn Mes

This thesis is number D-105 of the thesis series of the Beta Research School for Operations Management and Logistics. The Beta Research School is a joint effort of the departments of Technology Management, and Mathematics and Computing Science at the Technische Universiteit Eindhoven and the Centre for Telematics and Information Technology at the University of Twente. Beta is the largest research centre in the Netherlands in the field of operations management in technology-intensive environments. The mission of Beta is to carry out fundamental and applied research on the analysis, design, and control of operational processes.

Dissertation committee

Chairman / Secretary	Prof. dr. ir. L. van Wijngaarden
Promotor	Prof. dr. ir. J.H.A. de Smit
Assistant Promotors	Dr. M.C. van der Heijden
	Dr. P.C. Schuur
Members	Prof. dr. ir. J.C. Fransoo
	Prof. S.S. Heragu
	Prof. dr. J. van Hillegersberg
	Prof. dr. ir. J.A. la Poutré
	Prof. dr. J. Telgen

This research has been partly funded by Transumo. Transumo (TRANsition SUstainable MObility) is a Dutch platform for companies, governments and knowledge institutes that cooperate in the development of knowledge with regard to sustainable mobility.

Ph.D. thesis, University of Twente, Enschede, the Netherlands
Printed by Wöhrmann Print Service

ISBN 978-90-365-2629-6

SEQUENTIAL AUCTIONS FOR FULL TRUCKLOAD ALLOCATION

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. W.H.M. Zijm,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op donderdag 27 maart 2008 om 15:00 uur

door

Martijn Rudolf Kornel Mes
geboren op 20 juli 1976
te Dronten

Dit proefschrift is goedgekeurd door de promotor:

prof. dr. ir. J.H.A. de Smit

en de assistent promotoren:

dr. M.C. van der Heijden
dr. P.C. Schuur

Acknowledgements

By finishing this thesis, I have completed a project and an experience that I enjoyed very much. For this, I owe my gratitude to several people who were involved in the realization of this thesis. The people below, I thank in particular.

First of all, I would like to express my gratitude to prof. Aart van Harten who offered me this Ph.D. position. I thank him for motivating scientific discussions, for his useful comments, and for allowing me the freedom to work independently. Sadly, Aart died at the age of 57 on December 13, 2006. Like all members of our department OMPL, I miss his presence deeply.

I am also grateful to prof. Jos de Smit, who was willing to become my new supervisor in the last year of my Ph.D. project. His wealth of experience and sense for clear writing has improved this work. Further, I would like to express my gratitude to my daily supervisors Matthieu van der Heijden and Peter Schuur. I enjoyed working together with them. Their critical remarks kept this research in the right direction; their careful reading and sense of structure improved the readability of this thesis.

Besides the people that were directly involved in the realization of this thesis, several people had a more indirect contribution. I thank all the current and former colleagues for creating an excellent working atmosphere. I will miss the coffee breaks, OMPL car rallies, and AIO dinners. Particularly I thank Albert Douma who often served as a first hearer, a commentator, and travel companion at two international conferences. I also thank my sister Jeanette for checking my writing and for giving valuable suggestions.

This research was partly supported by the BSIK project Transumo (TRANsition SUstainable MObility). Part of this project has been carried out in cooperation with Merba bakeries. I would like to thank the management of Merba, Jan den Hartog and Wim Boerman, for their time and support. I also would like to thank prof. Jos van Hillegersberg for his cooperation and co-authoring an article about a case study at Merba bakeries.

Last, but certainly not forgotten, is my appreciation for my beloved wife Edith for all she has endured and the support she has given me.

Martijn Mes
Enschede, March 2008

Contents

Chapter 1

Introduction

In this thesis we examine the use of sequential auctions for the dynamic allocation of transportation jobs. For all players, buyers and sellers, we develop strategies and examine their performance both in terms of individual benefits and with respect to the global logistical performance (resource utilization and delivery reliability). Section 1.1 gives the motivation for this research. In Section 1.2 we describe our research design, including the research objectives, the research questions, and the research approach. We end this chapter with an outline of the remaining part of this thesis (Section 1.3).

1.1 Research motivation

New intelligent and flexible approaches for transport planning and scheduling are needed to deal with current trends in transport and logistics. Trends in external logistics include smaller transport batch sizes, shorter lead times, higher delivery frequencies, and tighter time-windows for delivery as well as higher delivery reliability. Another important trend is the increased focus on real-time decision making as a result of continuing developments in telecommunication and information technologies. These technologies, such as Internet and Global Positioning Systems (GPS), enhance the planning capability of freight carriers and provide the necessary information to perform real-time decision making. Furthermore, the possibilities of Internet trade of products in the business-to-business area will increase the complexity of physical distribution in the near future.

These trends require new operations research techniques enabling real-time decision making. Real-time optimization techniques are required for a single company optimizing its own logistical activities (intra-company) as well as for the planning issues between different actors in a network (inter-company). In

the area of freight transportation there are generally two types of companies: shippers and carriers. *Shippers* are the owners of freight, such as manufacturers, distributors, and retailers. *Carriers* are transportation companies that provide the capacity (in our case vehicles) to transport this freight.

Shippers and carriers are continuously facing pressure to operate more efficiently. Traditionally, shippers and carriers have focused their attention on controlling and reducing their own costs to increase profitability (intra-company). Recently, shippers and carriers have shifted their attention towards controlling and reducing system-wide costs (inter-company) and sharing these costs savings to increase everyone's profit (Ergun et al., 2007). A collaborative focus will open new cost saving opportunities. For an overview of the potential benefits of different forms of collaboration in Europe we refer to (Cruijssen, 2006).

Shippers and carriers can meet under a wide variety of relational structures. These structures vary from vertical integration to spot markets (Figliozzi, 2004). Vertical integration takes place when the shipper uses a private fleet. Here the shipper has direct control of operations of equipment and drivers. In the spot market, we have a large number of shippers and carriers exchanging additional loads and excessive capacity. According to Song and Regan (2003), this is a type of competitive market force used by almost all shippers and carriers to some extent. And over the past several years, these markets moved online. An example of a European load matching site is Teleroute (www.teleroute.com), that has more than 150,000 real-time daily freight offers, and over 60,000 users per day. For a review of the practice of online logistics providers in the USA we refer to (Song and Regan, 2001).

Situated between the extreme structures are the contractual agreement structures, where stable and long term contractual agreements take place between shippers and carriers. These structures are becoming increasingly popular in the trucking industry. Many shippers have a core carrier program in which they form partnerships with a few large carriers with the intent both to reduce their carrier base and to maintain or increase the level of service provided (Song and Regan, 2003). In fact, many on-line marketplaces have shifted their focus to more private collaborative networks (Song and Regan, 2001). Instead of being open to any shipper and carrier, the private marketplace is a platform with access for only a small group of companies, allowing shippers to maintain long-term relationships with their transportation providers.

In this thesis we focus on real-time decision making in transportation problems where transportation requests arrive continuously over time. Decisions involve the allocation of jobs to vehicles and the timing of these jobs. We focus in particular on the use of sequential auctions to support the allocation decision. Hereby we aim at the whole variety of relational structures between carriers and shippers. Therefore, we make a distinction between open- and closed environments. In an *open environment*, we have many independent shippers and carriers. Shippers request transportation services through an electronic auction

and carriers bid on these jobs. In a *closed environment*, we have a limited set of players over which we have some control. Examples of closed environments are (1) factories that allocate internal transportation tasks to Automatic Guided Vehicles (AGVs); (2) shippers that control their own fleet of vehicles; and (3) private collaborative networks.

In this thesis we consider both open and closed environments where our focus is as follows. In open environments, we focus on a single player, and take the auction protocol and the behavior of other players as given; no strategic behavior of the other players. As a consequence, we are only interested in the profits of the individual player. We study the profitability of different strategies for the individual player and compare this with the average profit of the other players. In closed environments, our overall goal is to achieve an efficient allocation, i.e., to maximize the utilization of transportation resources and the quality of service. In this thesis we argue that also closed environments can benefit from auction-based allocation mechanisms. Therefore, we model the different players as agents within a so-called *multi-agent system* (MAS). The auction mechanism is then used as a cooperation protocol between the agents.

Given this focus, our research is related to the following research areas: (1) dynamic vehicle routing, (2) multi-agent systems, and (3) transportation procurement auctions. Below we describe the motivation for our research within each of these three research areas.

1.1.1 Dynamic vehicle routing problems

The technological advances in ICT have also affected the transportation and logistics sector (see Regan and Golob, 1999; Golob and Regan, 2001). Along with the increased focus on just-in-time logistics, the ability to effectively make use of real-time information has become more and more important. These trends are reflected in the scientific literature by the increased interest in so-called *dynamic vehicle routing problems* (DVRP).

The *vehicle routing problem* (VRP) is usually concerned with the matching of available vehicle capacity with transportation jobs and with the timing of these jobs. A common objective is to do this at minimum costs while maintaining a required level of service. For recent surveys on the VRP and its variants, we refer to (Desaulniers et al., 2001; Toth and Vigo, 2002; Cordeau et al., 2007). The majority of the VRP literature focuses on deterministic and static versions in which all information is known at the moment the routes are planned. Also stochastic and static versions of the vehicle routing problem (SVRP) have been widely studied. Stochasticity can be found in travel times, load characteristics, the number of jobs, and the location of jobs. In the dynamic vehicle routing problems, new transportation jobs arrive dynamically when the vehicles have already started executing their tours. This requires real-time replanning in order to include the new jobs in the vehicle schedules.

There are many applications that motivate the research on the DVRP. Examples include dynamic fleet management, vendor-managed distribution systems, courier services, rescue and repair services, emergency services, and taxi cab services (see Ghiani et al., 2003). For overviews of literature on these dynamic vehicle routing problems (DVRP) we refer to (Powell et al., 1995; Psaraftis, 1995; Bertsimas and Simchi-Levi, 1996; Gendreau and Potvin, 1998).

In this thesis we consider a generalization of the vehicle routing problem, namely the pickup and delivery problem with time-windows (PDPTW), where transportation jobs are defined by a pickup location, a delivery location, and time-window restrictions at the pickup location and/or the delivery location. As stated by Cordeau et al. (2007), these demand-responsive freight transportation systems have become increasingly popular. The literature on the PDPTW is not as extensive as that on the vehicle routing problem with time-windows (VRPTW). For surveys of the PDPTW literature we refer to (Savelsbergh and Sol, 1995; Desaulniers et al., 2001; Cordeau et al., 2007).

A variant of the PDPTW that has been frequently studied is the dial-a-ride problem (DARP). Where the PDPTW is usually thought of as a model for transporting goods, dial-a-ride problems are models for passenger transportation. Also the DARP has become increasingly popular due to the ageing of the population and the trend toward the development of ambulatory health care services (Cordeau et al., 2007). A recent survey dedicated to the DARP was presented by (Cordeau and Laporte, 2007).

Although many papers have been devoted to dynamic vehicle routing problems and dynamic pickup and delivery problems, there are still some issues that have not been addressed yet (Ghiani et al., 2003); especially with regard to look-ahead policies that incorporate the future consequences of certain decisions.

Gendreau and Potvin (1998) provide a survey of relevant work on dynamic vehicle routing problems. They conclude that future research should focus on using forecasted demands for the construction of routes. In another paper (Gendreau et al., 1999), they suggest some important extensions of their approach to dynamic vehicle routing problems. They mention that it would be interesting to integrate probabilistic knowledge about the future to improve decision making at the current time. Ghiani et al. (2003) provide a review of algorithms for dynamic vehicle routing problems and highlight some issues that have not been addressed yet. They conclude that more research is required on heuristics with some look-ahead capability. In (Yang et al., 2004) different online strategies for assigning and reassigning trucks to transportation requests are examined, as well as the value of advance information for such schemes. They conclude that future research should concentrate on the search for policies that utilize the available information about future jobs more efficiently. Giaglis et al. (2004) state that limited research has been devoted to the real-time management of vehicles during the actual execution of the distribution

schedule in order to respond to unforeseen events that often occur and may deteriorate the effectiveness of the predefined and static routing decisions. In a recent publication, Cordeau and Laporte (2007) provide an overview of the scientific literature on the dial-a-ride problem (DARP). The DARP can be seen as an application area of the pickup and delivery problem devoted to passengers where more emphasis is put on controlling user inconvenience. Cordeau and Laporte (2007) believe that this problem will gain importance in the coming years, and conclude that more emphasis should now be put on the dynamic version of the problem.

The common denominator in the proposed direction of research, as stated by the references mentioned above, is that more research is required on look-ahead policies for dynamic vehicle routing problems. In this thesis we contribute to that. In particular, we develop look-ahead pricing and scheduling strategies for the dynamic pickup and delivery problem of full truckload transportation with time-window restrictions.

1.1.2 Multi-agent Systems

The increased use of information technology within and between companies will also change their coordination mechanisms. Traditionally, operations research (OR) based optimization methods are used to construct integral transport schedules. However, one may wonder whether such centralized methods are suitable for planning and control of stochastic and dynamic transportation networks. First, system-wide optimization algorithms may require a lot of information in advance that simply may not be available. Second, these algorithms can be sensitive to information updates: a minor modification in information may have an impact on the schedules of many vehicles. Third, the time required for the algorithm may not permit timely response to unexpected events such as equipment failure and the arrival of rush jobs. Finally, flexible transportation networks may consist of multiple independent organizational units that are working in an autonomous, self-interested, and not necessarily cooperative way. Therefore, these individual players may not be willing to share sensitive information (like their cost structure, current vehicle locations, and current schedule), with the result that centralized or hierarchical approaches are no longer applicable.

An alternative that has been proposed in the computer science literature is the multi-agent system (MAS). Such a system consists of a group of intelligent and autonomous computational entities (agents) which coordinate their capacities in order to achieve certain (local or global) goals (Wooldridge, 1999). MAS, originally emerged as a sub-field of distributed artificial intelligence, has turned out to be a promising solution for controlling complex networks, providing more flexibility, reliability, adaptability, and reconfigurability. However, despite these benefits, it is unclear whether the system-wide performance is

comparable to the performance of more centralized or hierarchically organized planning systems.

In recent years, many papers on multi-agent systems for transportation problems have appeared. Examples can be found in (Bürckert et al., 2000; Zhu et al., 2000; Böcker et al., 2001; Thangiah et al., 2001; 't Hoen and La Poutré, 2004; Kozlak et al., 2004). A common approach is to represent the resources and/or tasks by goal-directed agents; for example, a job agent may focus on on-time delivery against the lowest possible costs, and a resource agent may strive for utilization and/or profit maximization.

A key characteristic of these multi-agent systems is that the plan for the system as a whole is a composite of plans produced by multiple agents. These agents have limited competence and knowledge of others. The task for the designer of this distributed planning system is to define a computationally efficient coordination mechanism. A growing number of researchers have explored the use of market mechanisms as metaphors for constructing computationally tractable solutions to difficult resource allocation problems. The allocation of scarce resources is a topic that has long been studied in economics and it is shown by several authors (see Clearwater, 1996) that market mechanisms can result in good or optimal allocation of resources. This allocation is achieved in a decentralized fashion: it emerges from the interaction of buyers and sellers. Thus, economics can act as a valuable source of terminology, inspiration, and metaphors for developing solutions for resource allocation problems.

Given this local control concept, the internal behavior of agents should be described, reacting to events and stimuli in their environments. In particular, each agent should price the resources on the supply side and all kinds of logistics service effects on the demand side. Examples of these cost drivers are (1) earliness / tardiness given the specified delivery time-windows; (2) availability of capacity; and (3) probability that another (more profitable) job arrives that requires capacity from the same resource at the same time. The latter aspect refers to the intelligence in the pricing mechanism. Prices for transportation do not have to depend solely on immediate rewards, but in some way the expectations about future rewards have to be taken into account.

Although some results on multi-agent planning and scheduling are available in the area of transportation, the level of intelligence (i.e., the ability to anticipate future events) is still limited in many cases. An extensive survey of existing research on agent-based approaches to transportation and traffic management can be found in (Davidsson et al., 2005). They conclude that some problem areas are under-studied. In particular, they mention the comparison of agent-based solutions to existing techniques. In this thesis we aim to provide such a comparison and we aim to develop methods for agents to anticipate future events.

1.1.3 Auctions

As mentioned in the beginning of this chapter, there is a growing interest in collaborative logistics. But also the way contracts are negotiated is changing by enabling demand and supply to be matched dynamically through online markets. Especially due to the development of Internet sites that match shippers' demand for transportation with the transport capacity of carriers. Lin et al. (2002) did a survey about the adoption and usage of Internet procurement tools by shippers. They indicate that 60% of the shippers use Internet to procure transportation services.

McAfee and McMillan (1987) define auctions as market institutions with an explicit set of rules determining resource allocation and prices, based on bids from the market participants. Auctions are known to be an efficient way to allocate items among agents, both in terms of process and outcome (Sandholm, 2002). Moreover, they do it in a distributed and autonomy preserving way.

Usually, auctions are considered in the context whereby human bidders compete with each other in order to purchase an item at the lowest possible price from an auctioneer who wants to sell the item at the highest possible price. However, the same principles can be used by software agents for the allocation of transportation jobs. In this case, the auctioneer (e.g. a shipper) wants to subcontract transportation jobs at the lowest possible prices and each bidder (e.g. a carrier) wants to deliver the service at the highest possible payments. This situation creates a reverse auction because the sellers (carriers) bid instead of the buyers (shippers) and prices are bid down instead of up. Obviously, models for normal auctions can be reversed and applied to reverse auctions.

There are many different types of auctions. Examples of widely applied auction protocols, both in practice and in the scientific literature, are the English auction, the Dutch auction, the first-price sealed-bid auction, and the second-price sealed-bid (Vickrey) auction (see Vickrey, 1961). These auctions are used for selling a single good. The problem of auctioning multiple goods can be difficult; especially when the valuations of combinations of items differ, or when bidders have preferences over bundles, i.e., combinations of items. This is often the case in transportation exchanges (see Sandholm, 1993; Sandholm, 1991; Sandholm, 1996). Auctions that are specifically designed to deal with multiple goods are called *combinatorial auctions*. However, these auctions involve many inherently difficult problems. As mentioned by Song and Regan (2005), we face the bid construction problem where bidders have to compute bids over different job combinations, and the winner determination problem where jobs have to be allocated among a group of bidders. In addition, (1) it may be unrealistic to bundle jobs which belong to different shippers and (2) these procedures are not directly applicable in situations where jobs arrive at different points in time.

In this thesis we focus on sequential auction procedures where the items

are auctioned one at a time. The most common procurement process for transportation services is similar to a simple sealed-bid auction (Song and Regan, 2002). Here an auctioneer (shipper) announces the bidding item (contract to serve a certain transportation job), a group of bidders (carriers) review this item, and then each of them submits a price in a sealed envelope. The auctioneer then reviews the bids and determines the winner.

Determining the winners in a sequential auction protocol is easy because this can be done by picking the lowest bidder (in case of a reverse auction) for each item separately. However, problems arise due to the introduction of a new dimension, namely time. Consequence for the *bidders* is that, in order to determine the valuation of an item, they need to guess what items they will receive in later auctions. Obviously, this requires speculation on what the competitors will bid in the future. Therefore, the bid price in a sequential auction is affected by past auctions as well as by future auctions. To be precise, a bid price is affected by (1) previous auctions because winning a job has an impact on the available capacity; (2) previous auctions because we use historical auction data to estimate the competitors' bids in future auctions; and (3) future auctions because the costs for a certain job depend on future jobs. Consequence of the time aspect for the *auctioneer* is that it has the opportunity to auction the same item repeatedly until it receives an appropriate bid. To support the bid acceptance decision, shippers may use time-dependent reserve prices. For an extensive literature survey on this topic we refer to (McAfee and McMillan, 1987).

Clearly, both shippers and carriers, face dynamic pricing problems. Carriers price their transportation resources (vehicles) dynamically, depending on their location and time availability. Shippers evaluate bids for transportation jobs, depending on the time restrictions of these jobs. These decisions are related to revenue (or yield) management. Revenue Management is an economic technique to increase revenues, by accurately matching the available capacity (or product/service availability) with the market prices, based on demand forecasting. There is a lot of literature on this topic with well-known applications in air transport (see for an overview McGill and Van Ryzin, 1999; Talluri and Van Ryzin, 2005).

In this thesis we use revenue management techniques with respect to shippers' decisions. Specifically, we use time-dependent reserve prices and decommitment penalties to minimize the costs for transportation, thereby maximizing the revenues. However, with respect to carriers' decisions, there is an important distinction between revenue management techniques and the bid pricing strategies of the carriers as proposed in this thesis. To be precise, we decided to focus merely on costs instead of revenues due to the following. First, we are often dealing with a reverse second-price auction in which the lowest bidder receives the item for the price of the second lowest bidder. As a consequence, a bid of an individual bidder does not influence its expected revenue for this job,

but only its winning probabilities. Second, Revenue Management is of especially high relevance in cases where the fixed costs are relatively high compared to the variable costs. This situation for example occurs in the passenger airline case, where capacity is regarded fixed because the route is fixed. If the aircraft departs, the unsold seats cannot generate any revenue any more. To maximize profits, the seats are sold at different prices, depending on the remaining time until departure and the number of available seats. In our problem, the item for sale is a pickup and delivery job for one vehicle. The costs for doing this job are not fixed but depend on the vehicle schedule and job characteristics. For more details on the use of Revenue Management in carriers' bid pricing decisions, we refer to Chapter 8, Section 8.2.2.

The difficulties with respect to time-dependent bid prices not only have an immediate effect on the profitability of the shippers and carriers, but also on the efficiency of the allocation of transportation jobs. In this thesis we address both issues; specifically, by developing look-ahead strategies for bid pricing and winner determination.

1.1.4 Contribution

In the logistics sector we see a growing interest in real-time decision making, collaborative planning, and online markets. As mentioned before, this interest is driven by (1) the developments in the ICT and the availability of real-time information; (2) the increased focus on just-in-time logistics where customers ask for fast and flexible fulfillment of their transportation requests; and (3) the continuous pressure to operate more efficiently.

A prerequisite for efficient real-time control in transportation markets, is the availability of reliable real-time information and the ability to respond fast to the incoming information. Techniques that provide high quality solutions within reasonable response time must be developed. A distributed approach can be beneficial here. Therefore, we study the use of multi-agent systems - and more specifically the use of sequential auctions - for real-time planning and scheduling in transportation markets. Given this research focus, our research is related to three research areas: dynamic vehicle routing, multi-agent systems, and transportation procurement auctions. Each of these areas is gaining importance.

A large body of research has been devoted to each of these three research areas. However, there are some issues that have not been addressed yet, or have received too little attention. As mentioned before, more research is required on look-ahead policies that incorporate the future consequences of certain decisions; especially in market environments where players have to calculate and evaluate bids in real-time. With respect to multi-agent systems, little is known about the performance of agent-based transportation control compared to more traditional control methods. Also, little is known about the impact of MAS

design choices on the logistical performance. These design choices include (1) the identification of agents; (2) the roles, responsibilities, and decision-making capabilities of these agents; and (3) the way they interact. The objective of this thesis is to fill these gaps. More specifically, our scientific contribution consists of the following:

- We examine the performance of a multi-agent system for real-time scheduling of full truckload transportation, and compare this performance with that of more traditional approaches that are based on fast look-ahead rules and OR algorithms.
- We show that current MAS methodologies lack a sufficient quantitative basis to select a single "best" architecture, and show how simulation can be used to support this selection process. To illustrate this approach we apply it to a real world setting.
- We develop planning and scheduling policies which exploit probabilistic knowledge about the future to improve current decision making. Here we focus on the behavior of a single player and assume a stable behavior of the other players. In that sense, we are looking at competitive procurement auctions, where we optimize the decision making capabilities of a single player and take the behavior of others as given.
  - For the carriers we propose an opportunity valuation policy where not only the direct costs of a job insertion are taken into account, but also its impact on future opportunities. These opportunity costs are used to support bid pricing decisions, scheduling decisions, and waiting decisions (where to wait and for how long).
  - For the shippers we propose two policies: a dynamic threshold policy and a decommitment policy. The idea of the dynamic threshold policy is that shippers postpone commitments for which they expect to make a better commitment in the future. So if a shipper has plenty of time to auction a certain job, it will not agree with a relatively high bid. When the time for dispatch gets closer, the price it is willing to accept will rise. The idea of the decommitment policy is that the shipper allows a carrier to decommit from an agreement against a certain penalty. These penalties are chosen such, that whenever a carrier decommits a job, they cover the expected extra costs of the shipper for finding a new carrier.

  Both policies use the potential provided by probabilistically known future events. We evaluate these policies by comparing the performance of the individual agent that exploits the look-ahead policies to the performance of agents that are using a myopic policy.

- We aim at a wider applicability than competitive procurement auctions, in particular we aim at closed environments, i.e., allocation to a closed group of trusted carriers or auction procedures that are commonly used in multi-agent systems for resource allocation. Therefore, we combine carriers' and shippers' look-ahead policies and evaluate their performance in terms of system-wide logistical costs in closed environments.

Besides the scientific contribution, there is also a social aspect of this research. The main social challenge is to develop real-time control methods which have a positive effect on the sustainability, reliability, and profitability within the logistics sector. Methods should focus on efficient use of resources so that high delivery reliability can be achieved against low costs and low energy consumption.

This research is supported by the BSIK project Transumo, which stands for TRANsition SUstainable MObility. More specifically, this research is part of the Transumo project Diploma which stands for DIstributed PLanning Of freight transport networks using Multi-Agent technology. This project focuses on the development of multi-agent systems for real-time transportation planning with multiple actors, where dynamic pricing is used as an instrument for maximization of revenues and resource utilization.

The objective of Transumo is to strengthen the competitiveness of the Dutch transport sector ('Profit'), and to preserve and improve spatial and ecological ('Planet') and social ('People') aspects of mobility. Transportation is an important task in modern society. Astronomical amounts of money are spent daily on fuel, equipment, and maintenance. In 2000 in the Netherlands, almost 50 billion Euro (12.4% of the Gross National Product) was spent on logistical costs, of which 21.1 billion Euro was for transport costs (Van der Broek-Serlé, 2005). Furthermore, transportation accounts for a large part of the greenhouse gas (GHG) emissions in the world. In the European Union, transport now accounts for 21% of total GHG emissions (excluding international aviation and maritime transport), and road transportation is by far (93% share) the largest transport emission source (European Environment Agency, 2007).

In this thesis we develop intelligent transportation planning methods which minimize the fleet sizes and empty moves (profits and planet), and increase the flexibility and reliability of transport (profits and people). Given the characteristics of the logistics sector (see above), a small increase in performance can lead to huge improvements, both in terms of costs and GHG emissions. We believe that our approach - theoretical as it may seem - is a first step towards a better practice and provides general insights that can be used in many real life situations.

1.2 Research design

In this section we successively describe our research problems and objectives, the scope of this research, and our research questions and approach.

1.2.1 Research problems and objectives

In the previous sections we have mentioned our focus on using sequential auctions for the allocation of transportation tasks. Independent of the kind of environment (open or closed), these auctions always involve multiple players. On the one hand we have the entity requesting transportation service and on the other hand the entity with available capacity to serve this request. In the closed environment, these players are agents within a multi-agent system. In the open environment, these players are shippers and carriers. Below - to unify the terminology - we speak in terms of shippers and carriers.

Essentially, our problem consists of a market with shippers and carriers. Shippers receive transportation requests triggered by an external source. These transportation requests arrive continuously over time and have different characteristics which affect the price (arrival time, origin, destination, time-windows, etc.). To procure transportation, the shippers independently start a reverse auction for each job, one at a time. Carriers bid on these jobs and shippers select carriers based on these bids. This approach raises a few questions:

1. MAS: better than centralized planning?

   The principle of multi-agent systems is elegant and has clear advantages from an ICT point of view. However, it is still unclear whether the system-wide performance will be similar or even better than the performance of more centralized and hierarchically organized planning systems. It is even not guaranteed whether and when a multi-agent system will show a stable behavior. That is, will jobs be transported, will resources be properly utilized, and will prices remain within reasonable bounds in the absence of a coordination mechanism.

2. MAS: how to design?

   In building a multi-agent system we face many design decisions. To mention a few, we have to decide about (1) which resources and/or tasks are represented by an agent, (2) the roles and responsibilities of these agents, (3) the way they make decisions, and (4) the way they interact. It is important to provide some insight into the effect of different design alternatives on the logistical performance.

3. Auctions: appropriate for the dynamic allocation of transportation jobs?

   Auctions are often considered as appropriate means for dynamic job allocation in distributed environments. However, when multiple jobs are

auctioned at different points in time, such an allocation would be less appropriate if we do not take into account the future consequences of an allocation. Especially when jobs are complementary (e.g. transportation jobs that can be served sequentially by the same vehicle) or substitutable (e.g. transportation jobs that are available at the same time) a certain allocation may become unfavorable when new jobs appear. Therefore participants of the auction have to take into account the future consequences of a certain allocation.

In this research we aim to solve the problems mentioned above, which leads us to the following research objective:

To analyze in which way and to what degree multi-agent systems can be used for real-time operational planning and control of transportation networks. Further, to develop strategies for players in sequential transportation procurement auctions, and to analyze their performance in terms of both the individual benefits for the players and the system-wide logistical costs.

Our initial focus is on closed environments. In these environments, transportation requests arrive continuously over time and have to be allocated to a fixed set of vehicles. To model this environment we use a multi-agent system (MAS) where agents meet at a virtual marketplace. Next, we extend these results to open environments where multiple independent shippers and carriers meet at a marketplace. There is a major difference between these two kinds of environments. In the closed environment, we develop strategies for all players in the system and evaluate their impact on the global logistical performance. In the open environment, we develop strategies for a single player and evaluate the profitability of this player compared to other players.

1.2.2 Demarcation

To keep the research project manageable, some choices have been made with respect to the research focus:

- We consider a vehicle routing problem where jobs are characterized by a pickup location, a delivery location, and time-window restrictions. We only consider full truckload, which means that vehicles travel from origin to destination without any intermediate stops because there is no option for consolidation.
- We are not dealing with the design of an optimal auction mechanism. The design of an auction requires the precise specification of a set of rules. These rules determine an auction model, the system by which bidding is

conducted, how information is revealed, how communication is structured between buyers and sellers, and how allocations and payments are settled. In this thesis, we choose an existing standard auction mechanism (e.g. first- and second price sealed-bid auctions).

- We do not consider strategic reasoning of players on strategies of other players. Agents take prices as given and do not attempt to compute the impact of their bid on the behavior of other agents. Instead, the dominant strategy of bidders is assumed to be bidding the true valuation (marginal cost bidding). We believe that a proper methodology to calculate the true marginal costs is a required first step before incorporating more game theoretical aspects.

1.2.3 Research questions

To reach our objective, we define a number of research questions that we have to answer. These questions also define a logical sequence of research activities. For each question, we indicate the chapter in which the specific question will be answered.

1. *How does the performance of a multi-agent system compare to traditional OR-based systems in terms of (1) effectiveness, i.e., the ability to handle jobs according to specified targets, such as delivery time windows; (2) efficiency in terms of the utilization of resources and logistic costs; and (3) robustness against fluctuations in demand in terms of variation in system effectiveness and efficiency?*

   Before elaborating on the design of a multi-agent system and the decision making capabilities of the players, it is essential to gain insight into the potential benefits of such an approach. Therefore, we make a comparison in Chapter 3 between an agent-based control system and more traditional centralized heuristics.

   We use a case study on a proposed underground transportation system at Amsterdam Airport Schiphol, the Netherlands. We refer to this application as the OLS case, which is the Dutch abbreviation for underground logistic system. This case study is a logical first test case because (1) the problem is similar to ours, (2) some traditional planning methods have been developed for this case, and (3) a simulation test environment for this case is available at the University of Twente. We use this simulation environment to compare our agent-based control system with two hierarchical look-ahead heuristics that had been developed for the OLS case. Next, we simulate and compare the different control methods in a more general transportation network.

2. *How should a multi-agent system for material sourcing and scheduling of*

physical distribution be designed in terms of tasks, competences, responsibilities, and goal-directed behavior?

After gaining insight into the performance of a multi-agent system, we elaborate on specific design choices we face in building an agent system. Designs may vary in the roles and responsibilities assigned to the agents, the level of intelligence of the agents (forecasting and learning behavior), and the interaction protocols selected. Current MAS methodologies lack a mechanism to evaluate such design-choices and provide only limited support to the designer in selecting the preferred design for implementation. Therefore, we extend current MAS methodologies by multi-agent discrete event simulations. To demonstrate and test this approach, we apply it to a real life project: the design and development of a multi-agent system for the manufacturing of biscuits at the industrial bakery Merba in the Netherlands.

3. *How can we use information on historic job patterns and auction data to improve the pricing and scheduling of vehicles participating in transportation procurement auctions?*

   In Chapters 3 and 4, we propose a multi-agent system where vehicle agents are responsible for their own routing and scheduling decisions. The assignment of jobs to vehicles is done using a sequential auction procedure. Therefore, a proper pricing mechanism is needed to optimize the system-wide performance. In Chapter 5 we propose a pricing and scheduling strategy for vehicle agents where not only the direct costs of a job insertion are taken into account, but also its impact on future opportunities. We use simulation to evaluate the proposed approach.

4. *How can we use information on historic auction data to improve the auctioning strategy of shippers to procure their transportation services?*

   To improve the allocation of jobs through sequential transportation procurement auctions, we focus on strategies for the participants. In Chapter 5 we focus on profit maximizing strategies for the carriers and their vehicles. In Chapter 6 we focus on the shipper for which we propose two options: delaying and breaking commitments. Both policies use the potential provided by probabilistically known future events. The benefits of both strategies are evaluated with simulation.

5. *What is the impact of the different pricing and scheduling strategies for carriers and shippers on the system-wide logistical performance?*

   In Chapters 5 and 6, we propose strategies for carriers and shippers in sequential transportation procurement auctions. We evaluate the strategies separately, i.e., by studying the performance of a strategy for a single player while assuming naive strategies for the other players. In Chapter 7 we focus on all players by enabling them to use the proposed look-ahead strategies of Chapters 5 and 6. We use simulation (1) to provide

insight into the possible problems that occur when we apply the look-ahead strategies to all players; (2) to compare the performance of the individual look-ahead strategies with the performance of myopic policies; and (3) to study the interrelation of the different strategies (i.e., are they complementary or substitutable).

Clearly, simulation plays an important role in this research. Simulation has its own advantages and disadvantages (see Law and Kelton, 2000). First, most complex, real-world systems with stochastic elements cannot be described accurately by a mathematical model that can be evaluated analytically. In this research, complexity can be found in stochastic job arrivals and in the interaction of many players (with possibly different behavior). Second, alternative designs can be compared via simulation to see which one meets the specified requirements. In this research we compare many alternative multi-agent systems and different levels of intelligence for players in sequential auctions.

A major disadvantage of simulation is that it produces estimates of a model's true characteristics for a particular set of input parameters. An analytical model, if appropriate, can produce the exact values of the true characteristics of that model for a variety of sets of input parameters. However, solving an analytical model for stochastic and dynamic planning problems can be difficult. Especially when there are multiple independent players involved who have the ability to learn about their environment and about the behavior of other players.

1.3 Thesis outline

The outline of this thesis is as follows. We start in Chapter 2 with the description of our basic model and our simulation framework. In Chapter 3 we provide a comparison between agent-based control and more traditional centralized heuristics. The design choices we face in building a multi-agent system are described in Chapter 4.

After Chapters 3 and 4, we know something about the design and potential of multi-agent systems. However, the decision making capabilities of the different agents are still relatively simple. In the next two chapters, we improve the decision making capabilities by exploiting probabilistic knowledge about the future to improve decision making at the current time. Here we consider open environments, i.e., we focus on the behavior of a single player and assume a stable behavior of the other players. In Chapter 5 we develop opportunity valuation policies for carriers and their vehicles. In Chapter 6 we develop dynamic threshold policies for shippers.

We combine the strategies for shippers and carriers in Chapter 7. Here we return to the closed environment where we aim at the reduction of system-wide

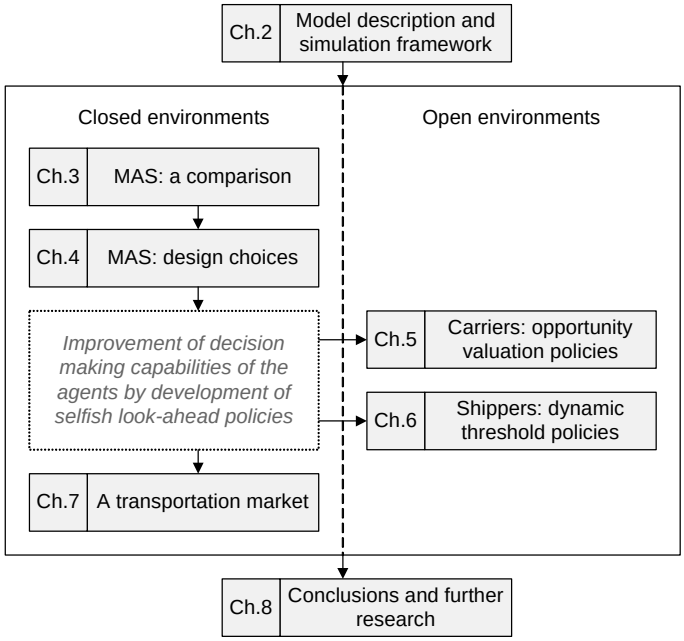

Figure 1.1: Thesis structure

logistical costs. We end the thesis with conclusions and suggestions for further research in Chapter 8. A schematic representation of the structure of this thesis can be found in Figure 1.1.

Chapter 2

Model description and simulation framework

In this chapter we introduce our model of the transportation problem and the simulation framework. Throughout this thesis, we use slightly modified versions of this model and simulation framework. These modifications are such that they suit the specific goals of the corresponding chapters. Therefore, we introduce the specific model in each chapter. In that sense, each chapter is self contained.

2.1 Model of the transportation market

In the next subsections we subsequently describe the system dynamics, the geographic area, the characteristics of the different players, the job characteristics, and the time and costs involved in our transportation problem.

2.1.1 System dynamics

The transportation network consists of independent carriers and shippers. Shippers are the beneficial owners of freight, for example, manufacturers, distributors, and retailers. Carriers are transportation companies, that provide the capacity (in our case vehicles) to transport freight. Every carrier is responsible for a set of vehicles and every shipper is responsible for a set of loads. We introduce the following sets:

$$
\begin{aligned}
\mathcal{S} &= \text{set of shippers, with } s \text{ the index of a shipper, } s \in \mathcal{S} \\
\mathcal{J} &= \text{set of jobs, with } \varphi \text{ the index of a job, } \varphi \in \mathcal{J} \\
\mathcal{J}_s &= \text{set of jobs from shipper } s, \mathcal{J}_s \subset \mathcal{J} \\
\mathcal{C} &= \text{set of carriers, with } c \text{ the index of a carrier, } c \in \mathcal{C} \\
\mathcal{V} &= \text{set of vehicles, with } v \text{ the index of a vehicle, } v \in \mathcal{V} \\
\mathcal{V}_c &= \text{set of vehicles from carrier } c, \mathcal{V}_c \subset \mathcal{V}
\end{aligned}
$$

The system dynamics is driven by the incoming jobs that are not known beforehand. Each job arrives at a shipper who then has a request for transportation. A job consists of a unit load (full truckload) which has to be transported in a geographical area (see Section 2.1.2). The job characteristics are described in Section 2.1.4.

The matching of jobs with open vehicle capacity leads to contracts between carriers and shippers. Execution of these contracts requires scheduling of the vehicles while taking the contract terms into account. Vehicle scheduling has its impact on the future availability of open capacity of vehicles and on the system dynamics and hence on the profitability of the companies. A general impression of the situation is given in Figure 2.1.

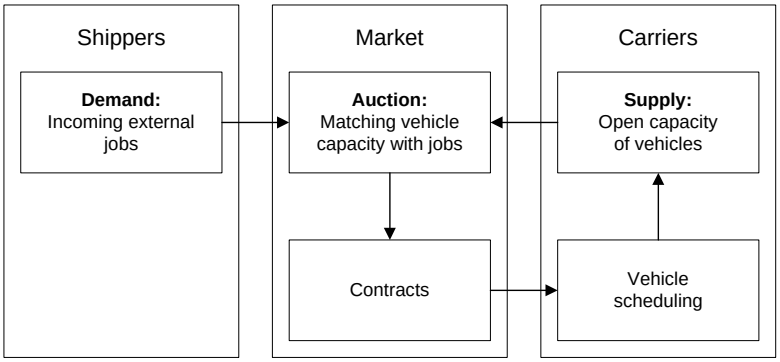

Figure 2.1: Demand driven online job assignment

The network decisions concern the assignment of jobs to vehicles, the planning of jobs in the operative schedule, and the assignment of prices to the delivered services. These decisions are taken in a decentralized manner by the different players in the network.

An important aspect in practice is that not all information is necessarily open to all parties. Due to the multiple independent shippers and carriers, we have to deal with incomplete information. Cost information for the carriers and shippers is private. Also, carriers may have incomplete information about

the network, such as information about other carriers, shippers, travel times etc. However, carriers and shippers have the opportunity to derive information about other players from historical auction data.

2.1.2 Geographic area

We consider two types of transportation networks: node networks and continuous networks. The node network is a directed graph ($\mathcal{N}$,$\mathcal{A}$), i.e., it consists of a set of nodes $\mathcal{N}$ and a set of arcs $\mathcal{A}$ connecting these nodes. In the continuous networks, transportation takes place in the Euclidian space. The origin and destination coordinates of a job are drawn randomly from the transportation area, and vehicles travel in a straight line from the origin coordinate towards the destination coordinate.

In the node networks we can control the flow between the nodes by adjusting the likelihood of being an origin or destination for all nodes. Also, for the continuous networks we consider unbalanced cases where some subsets of the transportation area are more popular than others. We indicate these subsets by regions. Of course, the continuous networks can also be regarded as node networks with an infinite number of nodes with their corresponding arcs. Working with regions in these networks can be regarded as a form of aggregation.

Because jobs arrive real-time, carriers and vehicles do not know their origins and destinations in advance. Therefore, the geographical demand pattern creates a significant amount of uncertainty for carrier decisions, such as the pricing and timing of jobs, and the routing decisions.

2.1.3 Players

Our transportation network consists of carriers and shippers. The objective of every carrier is to maximize its profits while maintaining a required level of delivery reliability. A carrier is characterized by its vehicles, its decision making capabilities, desired delivery reliability, safety margins which have to be used by its vehicles, travel prices, and other cost factors. All vehicles have capacity of a single load (full truckload). Further characteristics of vehicles are their decision making capabilities, speed, costs, current location, and schedule.

The objective of every shipper is to minimize its costs and tardiness of jobs. Shippers are characterized by their decision making capabilities and jobs. These jobs have characteristics as mentioned in Section 2.1.4. In addition, a shipper may have reserve prices for these jobs and possibly decommitment penalties (see Chapter 6).

Throughout this thesis, we consider a homogeneous fleet of vehicles. Also the characteristics of all the carriers are the same, and the same holds for the shippers. The single source of differentiation between players is the way in

which they make decisions. Given the agent-based concept, where each vehicle makes its own decisions based on its own characteristics and historic data, our approach can easily be generalized.

2.1.4 Jobs

Jobs to transport unit loads (full truckloads) arrive one-by-one according to some stochastic arrival process.

In each chapter, jobs are defined by an announcement time, an origin, a destination, and time-window constraints. The announcement time of a job is the time at which the job gets known by the shipper. The origins and destinations are either nodes or coordinates (see Section 2.1.2). The time-window constraints represent the time sensitivity of jobs and limit the flexibility of carriers and vehicles to schedule the job. These time-window restrictions differ per chapter. In Chapter 3 we have an earliest pickup time and a latest delivery time. In Chapter 4 we have an earliest delivery time, a best delivery time, and a latest delivery time. In Chapters 5 till 7, we only use a latest pickup time. Although the time-windows differ, they can easily be translated into each other, or additional time-window restrictions can be imposed. The jobs in Chapter 4 are slightly different from the rest of the thesis because each job is in fact divided into two separate jobs with their own characteristics.

The earliest pickup and delivery times are always hard restrictions. So when a vehicle arrives too early, it has to wait. The latest pickup and delivery times are always soft restrictions; carriers and vehicles are allowed to change the scheduled pickup and delivery times. Tardiness with respect to these times is penalized. In Chapter 5 we also consider a variant in which carriers have to agree upon a specific pickup time in advance. This agreed pickup time can be after the latest pickup time (in which case penalties are incurred), but the pickup time can not be changed later on.

Additional job characteristics are the handling time, the time-window length, and the contract attributes. The handling time consists of a loaded travel time between the origin and destination of a job plus the time for loading and unloading. The time-window length can be derived from the time-window restrictions mentioned above. The contract attributes describe the delivery conditions such as an agreed pickup time, and penalty costs for tardiness.

Throughout this thesis, we assume that an external job in process cannot be interrupted (no preemption). That is, a vehicle may not temporarily drop a load in order to handle a more profitable load and return later.

2.1.5 Time and costs

We distinguish the following times that result from an assignment of a vehicle to a transportation job: (1) empty travel time to reposition the vehicle to the origin of the job; (2) a handling time consisting of loading, transporting the load to its destination, and unloading; and (3) possibly an empty travel time to reposition the vehicle. We assume that all times are uniquely defined by the origin and/or destination.

Throughout this thesis, we use costs of 1 per unit travel time. The loaded travel times consist of travel times and the time needed for loading and unloading. Soft time-window restrictions are penalized with $c^p(t)$, where t is the tardiness with respect to the time-window.

In Chapter 3 the empty travel times and handling times are unknown and should be learned before using them in planning procedures. In Chapter 4 these times are given, but waiting times have to be learned. In the other chapters, we assume all times are deterministic and given.

2.2 Simulation framework

To study the implications of alternative designs and operating policies for shippers and carriers, we use simulation. Simulation enables us to explore and systematically test changes in the parameter settings for a wide spectrum of scenarios. In addition, simulation can be used to answer questions regarding systems that are far too large or complex to admit closed-form solutions to analytical models. As a consequence, system evaluation using simulation would require fewer simplifications than analytical models, which in turn has a positive effect on the validity of our findings.

We use discrete-event simulation to compare different pricing and scheduling strategies under different market settings. The basic idea of this type of simulation is that the system variables can change at only a countable number of points in time. To provide a correct comparison and significant outcomes, we perform multiple replications of each simulation run. Here we have to make a distinction between terminating simulations and steady state simulations. In Chapter 4 we have a terminating simulation in the sense that there is a natural event, namely the end of a work week, which terminates the simulation. As a consequence, we perform multiple replications of a work week. In all other chapters, we consider steady state simulations which means that we are interested in the steady state behavior. For these simulations we use a replication / deletion approach (Law and Kelton, 2000). Here we perform multiple runs of each experiment. Each run has a unique set of random number streams and a warm-up period in which we do not store the performance data. The length of each run and the number of runs are chosen such that, for our key performance

indicators, the relative width of the confidence interval compared to the mean is below a predefined adjusted relative error (see Law and Kelton, 2000). In the successive five chapters, we determine these experimental settings separately for each simulation experiment.

The auction events are assumed to take place in real time. However, computation times or delays are not taken into account in the simulation experiments. We do this because we otherwise would have to perform a real-time simulation (the simulation time is distinctly different from the CPU time). For an actual implementation in practice it is only important that the system changes are small during the computation times. In our simulation experiments, the computation times are really short compared to, for example, the time between successive job arrivals and the travel times. In addition, computation is distributed among multiple players. Also, the computationally intensive methods (see Chapter 5) can run offline, i.e., the values can be calculated before they are actually needed.

Each of the following 5 chapters requires a specific simulation model. However, most of these models fit into one general simulation framework which we describe below. Only Chapter 4 differs from the rest because here more different players are involved. Below we describe our general simulation framework, and at the end of this section we describe in what manner the model of Chapter 4 differs.

Our general simulation framework consists of six entities: (1) the transportation network, (2) an auction marketplace, (3) shippers, (4) jobs, (5) carriers, and (6) vehicles. All these entities are objects in an *object-model*. The objects are implemented in the object-oriented simulation package eM-Plant. The relation between the entities is depicted in an entity relationship diagram (ERD), see Figure 2.2.

We divide our simulation framework into four parts. First, the demand side, which consists of all shippers with their jobs. Second, the supply side, which consists of all carriers with their vehicles. Third, the market place where supply and demand are matched using an auction mechanism. Fourth, the simulation environment where the actual simulation takes place. A schematic representation of our simulation framework can be found in Figure 2.3. Here the decision modules of the different players are depicted as dashed rectangles and the local information of these players as cylinders.

Below, we describe the main features of each part.

- **Demand for transportation**

  Throughout this thesis, decisions at the demand side are mainly made by the shippers. Only in Chapter 3 decisions are also made at the job level. In this case, the bid evaluation module has moved from the shipper to the job.

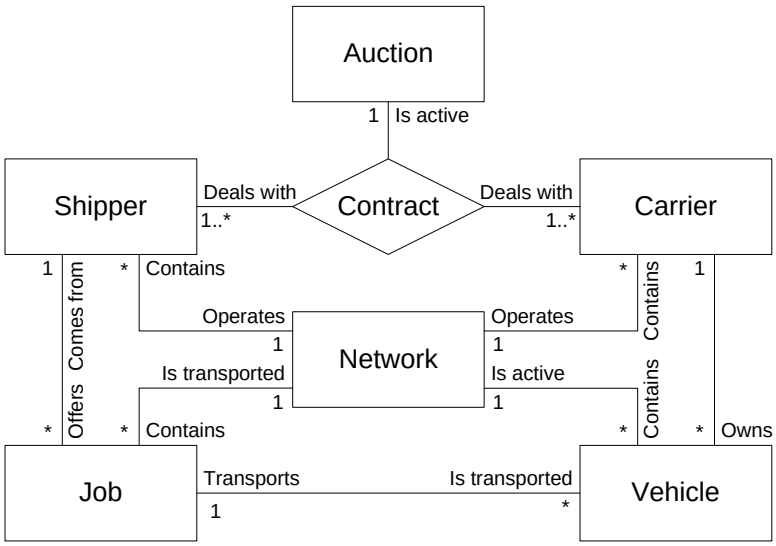

Figure 2.2: Entity relationship diagram

Information at the demand side contains all characteristics of the jobs that are not delivered yet. The learning data consists of all past jobs with their corresponding bid prices. Decisions at the demand side involve bid evaluation, trading contracts (see Chapter 3), and calculation of threshold values (see Chapters 3 and 6).

- **Supply of transport capacity**

  Both, carriers and vehicles, are able to learn data on travel times, handling times, waiting times, job characteristics, and bid prices. In this thesis we consider a decentralized control structure where the main decisions are made by the vehicles themselves. As a consequence, the vehicles should have access to all kind of learning data. However, in some cases, the carrier observes more information than a single vehicle. Therefore, carriers also learn data and share this with their vehicles. Information for the carriers and vehicles are the job characteristics, schedules, and cost settings.

  Decisions for the carriers involve the selection of one of their vehicles for a certain job. Decisions for the vehicles involve scheduling, routing, bid pricing, and opportunity costs calculation (see Chapter 5).

- **Marketplace**

  In the marketplace, an auction starts each time a shipper offers a new job. The characteristics of jobs that are not allocated to a vehicle yet, are

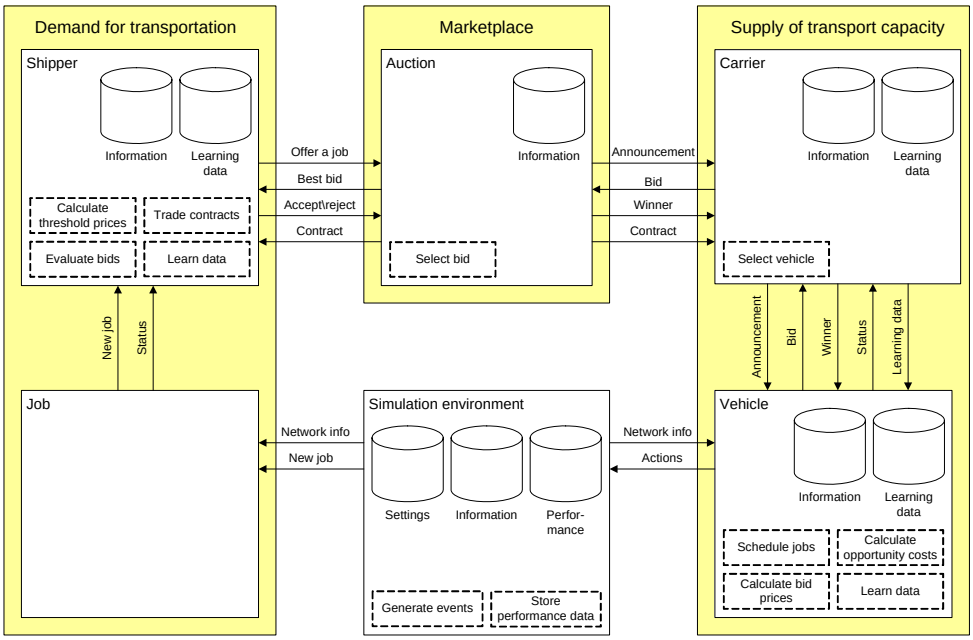

Figure 2.3: Simulation framework

stored at the marketplace. Carriers submit bids on jobs to the marketplace and the best bid for a certain job is submitted to the corresponding shipper.

- **Simulation environment**

  Here the actual simulation of the transportation network takes place. As information we have all the experimental settings, information about the state of the system, and the performance data. The events are generated here as well. These events consist of physical actions of the vehicles (movements, loading, unloading etc.) and the generation of new jobs.

As mentioned before, the simulation model of Chapter 4 is slightly different. Here the supply side only consists of Automatic Guided Vehicles (AGVs), and the demand side consists of production lines. In addition, a third part is added to the marketplace, which consists of resources which have to be visited by the AGVs.

2.2.1 Experimental factors

Given the fact that all jobs have to be transported, the main cost components are costs for repositioning of vehicles, and costs for tardiness. These costs

are mainly affected by the time-windows of jobs, the job arrival intensity, and the amount of unbalance in the network. In general, the more shipments can be accommodated, the lesser the deadheading (or average empty travel distance). The fewer shipments arrive, or the shorter the time-windows are, the more deadheading. Throughout this thesis, we use the following experimental factors:

- Job characteristics:
  - Arrival rate (average time between jobs)
  - Change in arrival rate during the day
  - Time-window length
  - Look-ahead (time between the announcement time and the earliest pickup time)
  - Contract (fixed, flexible)
- Network structure:
  - Number of nodes/regions
  - Distances
  - Degree of balance of the network (origin and destination probabilities for different nodes/regions)
  - Handling times
  - Variation in travel and handling times
- Companies:
  - Number of companies
  - Number of vehicles
  - Market structure (open, closed, virtual)
  - Market share of a company
- Control:
  - Decision structure (agent architecture)
  - Scheduling policies (Append, Insert, TSP, LocalControl, SerialScheduling)
  - Exchange of jobs between vehicles (Trade)
  - Dynamic threshold policies (Linear, Quadratic, based on a dynamic programming recursion)
  - Opportunity valuation policies (EndValues and GapValues in combination with various approximations)
  - Decommitment policies

2.2.2 Performance indicators

Relevant criteria deal with costs, service levels, and sustainability. We measure the criteria as averages over an entire simulation run, for the system as a whole as well as per individual player (e.g. vehicles, cariers, and shippers). We use the following key performance measures:

- Service measures:
  - Service level: percentage of jobs that are completed before the due time
  - Stability of the service level: the standard deviation in service level
- Sustainability measures:
  - Percentage of driving loaded, i.e., the percentage of the total distance that is not traveled empty, being an indicator for energy waste and loss of vehicle capacity
- Combined measures of service and sustainability:
  - Total costs: costs for driving loaded and driving empty plus penalties on tardiness
  - Net costs: costs for driving empty plus penalties on tardiness
  - Relative additional costs: the ratio of the net costs and the costs for driving loaded, i.e., (total costs - costs driving loaded) / costs driving loaded
  - Relative net costs of using one method compared to another
- Other performance measures:
  - Profits of vehicles and carriers, given by the income for all transportation jobs, minus the total costs for these jobs.
  - Relative profit of one player compared to other players
  - Computation time
  - Number of messages sent between the agents

As stated earlier, the model presented in this chapter contains many simplifications of a real transportation market; yet it provides the necessary features that capture the most important stochastic elements of the problem: the allocation and scheduling of jobs that arrive in real time.

In the next five chapters, we focus on different parts of the transportation market. In Chapters 3 and 4, we focus on the structure of the market

itself. To be more precise, we study the design and potential of multi-agent systems for transport planning. Here agents represent different players and/or resources, and an auction is used as a cooperation protocol between the agents. In Chapter 5 we focus on the supply side; specifically, on look-ahead pricing and scheduling strategies for the vehicles. In Chapter 6 we focus on the demand side; specifically, on look-ahead bid price evaluation strategies for the shippers. In Chapter 7 we study a closed market setting where both carriers and shippers are using opportunistic and selfish look-ahead strategies.

Chapter 3

MAS: a comparison

In this chapter[1] we consider the real-time scheduling of full truckload transportation jobs with time-windows that arrive during schedule execution. Because a fast scheduling method is required, look-ahead heuristics are traditionally used to solve these kinds of problems. As an alternative, we introduce an agent-based approach where intelligent vehicle agents schedule their own routes. They interact with job agents, who strive for minimum transportation costs, using a Vickrey auction for each incoming job. This approach offers several advantages: it is fast, requires relatively little information, and facilitates easy schedule adjustments in reaction to information updates. We compare the agent-based approach to more traditional hierarchical heuristics in an extensive simulation experiment. We find that a properly designed multi-agent system performs as well as or even better than the traditional methods. Particularly, the multi-agent approach yields less empty miles and a more stable service level.

3.1 Introduction

For operational planning and control of many transportation networks it is important to deal with uncertainties like transportation times (e.g. due to congestion), arrival of rush jobs during schedule execution, and modifications of the job characteristics. In combination with sometimes tight restrictions (e.g. time-windows) this leads to the need for a flexible, stable, and robust planning and control system. It should be *flexible* in the sense that schedule adjustments in reaction to information updates should be easy. It should be *stable* in the

[1] This chapter is based on the paper: M.R.K. Mes, M.C. van der Heijden, and 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.

sense that minor information updates (e.g. the arrival of a single rush job) should have impact on a small part of the schedule only. It should be *robust* in the sense that the overall network performance (e.g. transportation costs and on-time delivery performance) should remain acceptable under a large number of scenarios for unexpected events like rush jobs.

Traditionally, operations research (OR) based global optimization methods are used to construct integral transport schedules. However, one may wonder whether such methods are most suitable for planning and control of stochastic and dynamic transportation networks. First, most optimization algorithms require a lot of information in advance. Second, global optimization algorithms can be sensitive to information updates: a minor modification in information may have impact on the schedules of many vehicles. Third, the time required for the algorithm may not permit timely response to unexpected events such as equipment failure and the arrival of rush jobs. Finally, flexible transportation networks may consist of multiple independent organizational units that are working in an autonomous, self-interested and not necessarily cooperative way. Therefore, these individual players may not be willing to share all their information (like their cost structure, current vehicle locations, and current schedule), so that traditional centralized or hierarchical approaches are not applicable anymore.

An alternative that has been proposed within the computer science literature is the multi-agent system (MAS). Such a system consists of independent intelligent control units linked to physical or functional entities (e.g. vehicles and transportation jobs). It seems to be a promising solution for controlling complex networks, providing more flexibility, reliability, adaptability, and reconfigurability. Agents act autonomously by pursuing their own interest and interact with each other, for example, using information exchange and negotiation mechanisms. In a transportation network, each job and each resource can have its own goal-directed agent. For example, a job agent may focus on on-time delivery against the lowest possible costs, and a resource agent may strive for utilization and/or profit maximization. A key issue is how to configure agents such that their self-interested behavior yields a near-optimal solution for the network as a whole. One option is to use a market mechanism like an auction. An overall goal for the network performance can be to balance the total tardiness and the total relevant costs.

The principle of multi-agent systems is elegant and has clear advantages from an ICT point of view. However, it is unclear whether the system-wide performance will be similar to or even better than the performance of more centralized or hierarchically organized planning systems. It is even not guaranteed whether and when a multi-agent system will show a stable behavior. That is, will all jobs be transported, will resources properly be utilized, and will prices remain within reasonable bounds in the absence of a coordination mechanism?

Although many papers have appeared on multi-agents systems, also applied to logistics, literature on the performance comparison between traditional OR-based systems and multi-agents systems is scarce. In this chapter we aim to make such a comparison for a transportation network where transportation jobs (for full truckloads) with varying soft time-windows arrive during schedule execution and should be scheduled in real time. That is, a job should be assigned to a vehicle and a feasible start time should be determined. Because a fast response is required, we use local dispatch rules and serial scheduling as benchmarks, see (Van der Heijden, Ebben, Gademan and Van Harten, 2002) and (Ebben et al., 2005) for an extension of the serial scheduling method under capacity constraints. For the multi-agent system, we develop an auction mechanism with several pricing variants. To compare the agent-based approach with the two more traditional approaches, we use discrete event simulation for an extensive numerical experiment. As overall network performance criteria we focus on the average on-time delivery percentage as *service* measure, variation in the on-time delivery percentage as *robustness* measure and the empty mile percentage as *efficiency* measure. We also use total costs (transportation costs, penalty costs) to measures a combination of service and efficiency.

The remainder of this chapter is structured as follows. In the next section, we give an overview of related literature and we explain our scientific contribution. In Section 3.3 we present our model and in Section 3.4 we discuss our choice for a particular agent-based planning concept. Next, we discuss several options for agent bidding and bid evaluation in Section 3.5. In Section 3.6 we briefly present the two more traditional planning approaches that we use as benchmarks in a simulation study. We describe the experimental settings in Section 3.7 and provide the numerical results from this study in Section 3.8. We end this Chapter in Section 3.9 with conclusions.

3.2 Literature

3.2.1 Transport planning

Our problem of assigning jobs to vehicles in a transportation network is well-known in the area of vehicle routing problems (VRP) as a real-time multi-vehicle pickup and delivery problem with time-windows. Such problems arise a.o. in the transportation of elderly and/or disabled persons, shared taxi services, certain courier services and so on. We consider a variant with full truckloads, stochastic arrival of jobs, and stochastic handling- and travel times where even the probability distributions are not known in advance.

The VRP and its variants have been studied extensively; see (Laporte, 1992) and (Toth and Vigo, 2002) for a survey. It is well-known that most variants of the VRP problem are NP-hard, so that it is virtually impossible to find

an optimal solution within a short time. Most work focuses on static and deterministic problems where all information is known when the schedule has to be generated, see for example (Desrosiers et al., 1995). When the input data (travel times, demands) are stochastic and depend on time, the planning result is not a set of routes but rather a policy that prescribes how the routes should evolve as a function of those inputs that evolve in real-time (Psaraftis, 1988). Such policies have been studied in several papers, see for example (Psaraftis, 1988) and (Gendreau and Potvin, 1998). The most common approach to handle these problems is to solve a model using the data that are known at a certain point in time, and to reoptimize as new data become available. Because a fast response is required, it is common to use relatively simple heuristics or parallel computation methods, see (Ghiani et al., 2003) for an overview.

The dynamic assignment problem, as discussed in (Godfrey and Powell, 2002), also shows some similarities. Here resources (e.g. vehicles) are dynamically assigned to tasks that arrive during schedule execution. Key differences are (1) each individual vehicle schedule contains only one job at a time; (2) the price of a job is exogenous and the only issue is whether to accept this job and if so, to assign a vehicle to this job; and (3) only the most profitable jobs are accepted. Powell and Carvalho (1998) use so-called Logistics Queuing Networks (LQN) to decompose the large and complex scheduling problem by a series of very small problems. In this way, many real world details can be included in the model that cannot be dealt with using traditional approaches. Still this is a centralized planning approach in contrast to the decentralized agent-based approach that we consider in this chapter.

Closely related work can also be found in (Regan et al., 1995; Regan et al., 1996; Regan et al., 1998) who investigate the dynamic assignment of vehicles to loads for real-time truckload pickup and delivery problems. They provide relatively simple and fast local rules. Yang et al. (2004) extend this work to a formal optimization-based approach for the same problem class. They use simulation to compare this approach with the previously developed heuristics. Mahmassani et al. (2000) present a hybrid approach combining fast heuristics for initial assignment with the optimization-based approach for the off-line problem of reassigning and sequencing accepted loads. Kim, Mahmasanni and Jaillet (2002) develop several approaches for routing and scheduling in oversaturated demand situations.

3.2.2 Agent-based logistic planning

According to (Wooldridge and Jennings, 1995), an agent is a hardware or software based computer system with key properties autonomy, social ability, reactivity, and pro-activeness. A multi-agent system (MAS) is a group of agents that interact with each other to solve a complex problem. One way to achieve this interaction between agents is by using a market mechanism where resource

agents compete for jobs by dynamic pricing of these jobs. In this chapter we will use a market-based control mechanism for the allocation of vehicles to transportation jobs.

In the last years, research on multi-agent systems also has boosted in the logistics and operations research community. Particularly, several papers have appeared in the area of manufacturing scheduling and control. For example (Cardon et al., 2000) who use genetic algorithms to solve job-shop scheduling problems, and derived schedule improvements by agent negotiations. There are also some applications in material handling and inventory management (Kim, Graves, Heragu and St. Onge, 2002) and supply chain management (Ertogral and Wu, 2000). Only Dewan and Joshi (2000) compare their agent approach with an exact solution found by CPLEX. They conclude that centralized models are an unattractive choice compared to decentralized models because of computational inefficiency and degradation in the quality of the solution with increasing problem size.

Also, several papers on agent-based transport planning and scheduling have been published. In the area of railroad scheduling, Böcker et al. (2001) present a multi-agent approach for real-time coupling and sharing of train wagons. In (Zhu et al., 2000) a multi-agent solution for air cargo assignment is considered. Although this paper contains an interesting agent-based application, it does not provide detailed information on the design of a multi-agent system itself in terms of goals, behavior, pricing strategies etc. An interesting contribution comes from Fischer et al. (1996) who developed a simulation testbed for multi-agent transport planning, called MARS. They describe the information architecture and decision structure for quite generic transport planning systems, and test their model on the traditional vehicle routing problem with time-windows where all jobs are known in advance.

In ('t Hoen and La Poutré, 2004) a multi-agent system is presented for real-time vehicle routing problems with consolidation in a multi-company setting. Cargo is assigned to vehicles using a Vickrey auction. They study a decommitment strategy where trucks have the option to break an agreement in favor of a better deal if another truck from the same company can handle the cargo. They conclude that significant increases in profit can be achieved when the agents can decommit and postpone the transportation of a load to a more suitable time. The main distinctions with our model are that they consider (1) a Less-Than-Truckload (LTL) problem and (2) a next day delivery situation where the prices for loads are independent on the time at which the load is auctioned.

Another interesting contribution comes from Figliozzi et al. (2003), who present a framework for the study of carriers' strategies in an auction marketplace for dynamic full truckload vehicle routing problems with time-windows. They also use a Vickrey auction and a simple heuristic for generating bids, namely the additional costs of serving a shipment by appending it to the end

of the vehicle schedule. They focus on profit allocation rather than on the efficiency of assignment decisions. In (Figliozzi et al., 2004) they study the impact of different assignment strategies on the travel costs under various demand conditions. They consider four fleet assignment methods that are related to the agent-based approaches considered in this chapter. We explain the differences compared to our research in the next section.

3.2.3 Contribution to the literature

Although some results on multi-agent planning and scheduling are available in the area of transportation, the level of intelligence is still limited in many cases. Also, many papers deal with the design of an agent architecture rather than analyzing the relation between agent behavior and the overall network performance. Especially little is known about the performance of agent-based transportation control compared with more traditional control methods. The scientific contribution of this chapter is related to the following new issues for agent-based transport scheduling:

- A combination of soft time-windows and incomplete information (stochastic demand and random handling times).
- A study of the impact of additional intelligence of agents (both vehicle agents and shipper agents) on the overall system performance.
- A comparison of our multi-agent system to more traditional approaches for real-time transport planning based on fast look-ahead rules and OR algorithms (serial scheduling).
- An analysis of performance *robustness*, measured by the standard deviation of the daily service levels.
- An analysis of the impact of job characteristics (such as tightness of the time-window) on the overall costs.

3.3 Model, assumptions, terminology, and notation

The key issue in our research is to match available transportation capacity with jobs that arrive during schedule execution. The matching of available vehicle capacity with incoming jobs can be done using OR-based heuristics or using an agent-based approach. We make the following model assumptions:

- All transport jobs have a size of one full truckload (FTL).

- Vehicles are location aware and carriers are aware of the next node to be visited by their vehicles.
- No jobs may be rejected, even if it is clear that a job cannot be delivered in time.
- The total transportation capacity is sufficient to handle all jobs in the long run.
- A job in process cannot be interrupted (no preemption), i.e., a vehicle may not temporarily drop a load in order to handle a more profitable job and return later on.
- Communication between shippers, carriers, and vehicles is possible any time.

In the next sections we describe our transportation problem in more detail.

3.3.1 Transportation network and demand

We consider a transportation network that is inspired by a case for an automated transportation network using AGVs (Automatic Guided Vehicles) as described in (Van der Heijden, Van Harten, Ebben, Saanen, Valentin and Verbraeck, 2002). The network consists of a set of nodes and a set of arcs connecting these nodes. In the case study, the arcs represent underground tubes through which the AGVs drive between nodes (terminals). Each node has a number of docks for loading and unloading cargo. As a consequence, vehicles may face significant waiting times at the nodes. For more details we refer to Section 3.7.1.

Jobs to transport unit loads between these nodes arrive one-by-one according to some unknown stochastic arrival process. Jobs are characterized by the following parameters: the origin node i, the destination node j, the earliest pickup time r at the origin, the latest delivery time d (due time) at the destination, and the time a at which the job becomes known in the network $a \leq r$. The earliest pickup time is a hard restriction and the due time is a soft restriction. The time to handle a job from node i to node j (waiting for loading, loading, driving from node i to node j, waiting for unloading, and unloading) is a random variable and denoted by τ_{ij}^{f} . Variation in handling times may arise from traffic congestion, variation in loading and unloading times, and waiting times at the nodes. We do not consider limitations in loading and unloading capacity at the docks explicitly, but include it as a stochastic effect in the transport handling times. The time to drive empty from node i to node j is a random variable τ_{ij}^{e}. The handling times of jobs and travel times of empty vehicles are unknown and should be learned from historic data.

3.3.2 Cost structure and performance measurement

To evaluate the system performance, we use the following key performance indicators:

- Service level, i.e., the percentage of jobs that are completed before the due time.
- Stability of the service level, measured by the standard deviation in service level per simulation period.
- Percentage of driving loaded, i.e., the percentage of the total distance that is not traveled empty, being an indicator for energy waste and loss of vehicle capacity.
- Relative additional costs, defined as the ratio of the costs for driving empty and penalties and the costs for driving loaded; or (total costs - costs driving loaded) / costs driving loaded.

The relevant cost factors for vehicles are (1) travel costs $c^r(t)$ as a function of the total travel- and handling time t, both for driving loaded and driving empty and (2) penalty costs $c^p(t)$ as function of the tardiness t. In the agent-based approaches, vehicles also use waiting costs $c^w(t)$ as a function of the waiting time t, to penalize loss of vehicle capacity. We assume that the fixed costs are identical for all vehicles, so that they are not relevant for scheduling decisions.

3.3.3 Schedules

The transport schedule consists of a set of schedules per vehicle. Each vehicle has a list of jobs and a schedule to execute these jobs.

Formally, we define a vehicle schedule as a sequence of actions of the following types: (1) move loaded along arc (i, j), (2) move empty along arc (i, j), and (3) wait at node j until time t. If a job has been delivered at node i and the next job in the schedule has to be loaded at j later on, the vehicle moves immediately empty to j and waits over there. At any point in time, the first job in a vehicle schedule is in execution and cannot be interrupted. A schedule will always end with the third option at some node with $t = \infty$. Given a set of K jobs, the number of job sequences equals $K!$. Given a certain job sequence, the timing of the jobs and the corresponding empty moves should be determined.

Vehicle schedules are updated upon (1) completion of the first action in a schedule and (2) matching a new external load with available vehicle capacity. Depending on the control method, also periodical replanning is possible.

3.4 Agent-based planning concepts

In our agent-based planning concept, we assign vehicles to jobs using a market-like negotiation protocol that implicitly coordinates the agents' decisions. The definition of such an agent-based planning concept depends on three key choices: (1) which agents to distinguish with their tasks and goals, (2) which products (services) to trade, and (3) which market mechanism (auction) to define. We address these three issues below. The goal-directed behavior of each agent will be discussed in Section 3.5.

3.4.1 Agent types

To assign jobs to vehicles, we choose for an elementary structure with one agent per vehicle and one agent per job. Further, we use a shipper agent to collect and analyze auction and processing time data of all its vehicles, and to distribute the results to its vehicles when needed. In this way, the vehicle agents have access to more information than their own history only. The same applies to the shipper agents for all the jobs issued by the shipper. Hence, our multi-agent structure consists of four agent types, see Figure 3.1.

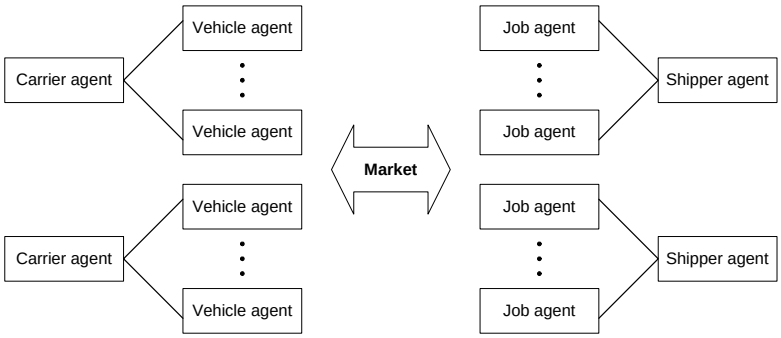

Figure 3.1: Agent structure for transportation networks

A vehicle agent has the goal to maximize its profit by deploying its capacity. A job agent has the goal to arrange transportation of the corresponding load before the due time at minimal costs. In a basic structure, all vehicle agents and job agents meet on the marketplace where they negotiate to assign jobs to vehicles. Each vehicle agent maintains its own schedule. Hence, the solution to the global scheduling problem emerges from the local scheduling and pricing decisions of the vehicle agents. In this way, one complex overall plan is replaced by many smaller and simpler plans.

The introduction of hierarchy may improve the coordination between agents. We can define hierarchy both at the job level and at the resource level. At the

job level, a shipper agent can be responsible for a set of jobs. A possible task is to reallocate the transport capacity that has been acquired such that their jobs are handled before the due times at lowest costs. For example, they may switch a job that has been scheduled but that has not been started yet with a rush job on a similar trajectory. To this end, they have full information on all jobs under their control and all transport capacity that has been acquired for these jobs. At the resource level, a carrier agent can be responsible for a subset of vehicles. If they know the positions and local schedules of all their vehicles, they can reassign vehicles to jobs to improve the profit of the fleet.

Although a hierarchical concept is interesting, we start with a fully decentralized concept. It is interesting to examine whether such a simple agent-based concept can already meet the performance of traditional OR based planning methods. However, we will use carrier agents and shipper agents to collect relevant information and to distribute it to the vehicle agents and job agents. In Sections 3.5.1 and 3.5.2, we present two extensions that require some form of hierarchical coordination.

3.4.2 Product definition

To create a marketplace, we need a product definition. We distinguish the following options:

- Transportation of a job from location i to location j, to be loaded not earlier than the release time r and to be delivered before the due time d.
- Transport capacity of a unit load that is available at node i at time t_1 to be used during a time period T. The advantage compared to the first option is that it provides the flexibility to reserve capacity for future jobs with some arbitrary destination. However, bidding is harder because not much can be said about the expected vehicle location at time $t_1 + T$.
- Transport capacity of N vehicle loads that can be used in some time interval $[t_1, t_2]$. Such a bulk trade may be advantageous for fleet management as a whole, but it is not suitable for a decentralized planning concept.
- Transport capacity of a unit load from node A to node B that has to be picked up at time t_1 and that has to be delivered at time t_2. Although this definition fits well with the job definition, it hampers flexibility for dynamic reallocation of capacity when additional (rush) jobs arrive.

We choose for the first option, because it offers both simplicity for bidding and flexibility for schedule alteration; particularly if the due time of jobs may be violated at some penalty costs as described in Section 3.3.2.

3.4.3 Auctioning mechanism

Several auction mechanisms have been proposed for distributed scheduling, see e.g. (Wellman et al., 2001). Some common auction types are:

- Bargaining, this is a one-on-one negotiation protocol where all trading partners contact each other individually.
- Sealed-bid auctions where every bidder submits his bid only once and the best bid is selected; special cases are the first-price sealed-bid auction where exactly the price offered is paid, and the Vickrey auction in which the bidder receives the price of the one but best offer (second-price sealed-bid).
- Open outcry auctions consist of multiple bidding rounds where all bids are known to each bidder. Variants are the English auction, where bidders sequentially either raise their bids or withdraw in each round until a single bidder is left; and the Dutch auction, where the price is reduced step by step starting from a high level until some bidder accepts the price.

In this chapter we select the Vickrey auction as mechanism because (1) it requires a single bidding round and (2) under some mild conditions the optimal bid is the net cost price of the bidder, who will make profit from the margin between the two best bids, cf. (Vickrey, 1961). Therefore, it provides a natural mechanism for acceptable profits. An advantage of this simple bid price is that it enables us to concentrate on the transportation control variables themselves rather than on learning and rationality issues of the agents. A drawback is that the profits may reduce to (almost) zero if the number of competitors becomes large.

We implement the market mechanism as follows. Each time a job φ arrives, the corresponding job agent starts an auction by asking all vehicles to bid. Each vehicle agent $v \in \mathcal{V}$ creates a single bid, consisting of a bid price $b(v, \varphi)$, an expected departure time, and an expected arrival time. Next, the job agent evaluates all bids and sends a grant or reject message to the vehicle agents. We allow the job agent to reject all bids if it expects to receive a better bid later on (see next section).

3.5 Bid calculation and evaluation

3.5.1 Bid calculation by vehicle agents

Let us denote the current schedule of vehicle v by Ψ_v. The acceptance of an additional job will lead to a new vehicle schedule, for which we may consider

several alternatives Ψ_v^n, where n is the index of the vehicle schedule alternative. For example, we may insert the new job at various positions in the current schedule or we may reshuffle the entire schedule to find a new optimum. Because we use a Vickrey auction, the bid price of vehicle v equals the minimum additional costs over all alternative schedules n. As mentioned in Section 3.3.2, the additional costs for vehicle v to service job φ, are given by the additional costs needed to move the load, plus the change in the total penalty costs for tardiness, plus the change the expected costs for waiting. The additional costs are therefore given by:

$$b(v,\varphi) = \min_n \left(c^r \left(\Delta T_{v\varphi}^n \right) + c^p \left(\Delta D_{v\varphi}^n \right) + c^w \left(\Delta W_{v\varphi}^n \right) \right) \tag{3.1}$$

where

$\Delta T_{v\varphi}^n$ = expected additional travel- and handling time required for vehicle v in schedule alternative n to transport the job φ;

$\Delta D_{v\varphi}^n$ = expected additional tardiness required for vehicle v in schedule alternative n to transport job φ;

$\Delta W_{v\varphi}^n$ = expected additional waiting time required for vehicle v in schedule alternative n to transport job φ.

Note that the additional waiting time may be negative if the new job can be inserted in a gap in the current vehicle schedule. Further note that we cannot simply include the difference in total tardiness in the bid price, because the penalty costs are not necessarily a linear function of the tardiness. It is obvious that a bid depends on the internal job scheduling of the vehicle agent. We consider three variants for internal vehicle scheduling.

The simplest method (called *AgentAppend*) is to add a new job to the end of the current schedule. So we have a single schedule alternative ($n = 1$). Then the change in penalty costs can only be due to tardiness for job φ because the expected arrival times of the jobs in the current schedule are not affected. The additional travel time $\Delta T_{v\varphi}^1$ equals the handling time of job φ plus the time needed to move the vehicle empty from the end location of schedule Ψ_v to the start location of job φ.

A second option is to insert the new job at any position in the existing schedule Ψ_v without altering the order of execution for the other jobs. We refer to this option as *AgentInsert*. Hence, the number of schedule alternatives equals the number of jobs in the current schedule, because the first job is in execution. For bid calculation we have to consider the cost components for the new job plus all jobs from the current schedule that will be served later on.

A third option is to construct a completely new schedule except for the job currently in execution. As this means solving a Traveling Salesman Problem (TSP), we refer to this method as *AgentTSP*. We use a depth-first, branch and bound algorithm, where we use an upper bound found with AgentInsert to test the lower bound for the remaining branch. This requires not too much

computation time because the number of jobs in a vehicle schedule is usually small (say less than ten) and AgentInsert provides a reasonable upper bound. Otherwise, we have to rely upon well-known fast heuristics for the TSP, such as tabu search, cf. (Gendreau et al., 1994).

Because of the dynamic nature of the problem it is not guaranteed that the initial assignment of a job to a vehicle remains optimal as new jobs arrive and travel time realizations become known. Therefore, we introduce an option to exchange jobs between vehicles that we call *Trade*. Whenever a vehicle - after unloading at a certain terminal i - has to travel empty to terminal j, its agent searches for another vehicle agent within the same fleet that has a job from i to j that has been released but that has not been started yet. Then the job that yields the highest savings (if positive) will be transferred to the vehicle to avoid empty traveling.

3.5.2 Bid evaluation by job agents

The job agents have to evaluate all bids; determine for each bid whether to accept or to reject it. We consider two variants for job agent behavior. In the first variant, the job agent simply accepts the best bid received from all vehicle agents. In the second variant, the job agent rejects all bids if they are all higher than a certain threshold. The idea behind this is that the job agent may expect to receive a better bid when reauctioning at a later point in time. After all, prices fluctuate over time due to changes in the available transportation capacity and in the vehicle schedules. So if the best bid is relatively high (which can be learned from history) and there is still quite some time until the latest pickup time of the job at its origin, it may be better to wait for a more attractive price. As the deadline for dispatch comes nearer, the job agent may increase the threshold to get transportation.

We assume fixed periods between reauctioning of a job that has not been assigned to a vehicle yet. We call this variant *DynamicThreshold*. The decision of the job agent is to set an initial threshold price for the first auction round and to determine the threshold prices for all further auction rounds. The fixed time between auction rounds for the same job is a parameter of the job agent. To determine the thresholds, the job agents need insight into the cost and handling times for their routes. We assume, as mentioned in Section 3.4.1, that the vehicle agents receive this information from their shipper, who keeps track of all travel times and bid prices.

The bid acceptance under DynamicThreshold works as follows. For the timing between successive auctions for the same job, we take a fixed period R. It is logical to relate the threshold price to the maximum number of auction rounds N before the job has to be transported. We have that $N = \lfloor (d - t - a)/R \rfloor + 1$ with d the due date, a the first announcement time of the job and t the expected handling time as obtained from the shipper agent. Without loss of generality,

we assume that R is such that always $N \geq 2$ (if not, the DynamicThreshold variant coincides with the first variant discussed in this section in which the lowest bid is always accepted).

The threshold prices can be based on expectations of the outcomes of future auctions. In this case, the threshold price for a certain round equals the expected price a shipper could receive in one of the next auction rounds, given a certain threshold policy. This policy is quite difficult to model because the expected price will increase in the successive auction rounds and bids in subsequent auction rounds are possibly correlated. We develop such a dynamic threshold policy in Chapter 6. Here we propose a much simpler strategy.

The threshold price α_N for the last auction round is always infinite, i.e., any offer is accepted in order to force the job to be served. The first threshold price α_1 equals a certain minimum price P_{min} and the threshold price for the second last auction round α_{N-1} equals a maximum price P_{max}. These values P_{min} and P_{max} can be based on historical data provided by the shipper agent. We consider two pricing strategies: linear and quadratic. For the linear strategy, the threshold price α_n in round n is given by:

$$\alpha_n = P_{\min} + \left(\frac{P_{\max} - P_{\min}}{N-2} \right) (n-1) \qquad \text{for } n = 1, ..., N-1 \tag{3.2}$$

For the quadratic pricing strategy we define:

$$\alpha_n = P_{\min} + \left(\frac{P_{\max} - P_{\min}}{(N-2)^2} \right) (n-1)^2 \qquad \text{for } n = 1, ..., N-1 \tag{3.3}$$

We examine the impact of DynamicThreshold in Section 3.8.

3.6 Traditional OR based heuristics as benchmark

Traditionally, heuristics from operations research are used for real-time scheduling in transport networks. We will use two of the methods from (Van der Heijden, Ebben, Gademan and Van Harten, 2002) as benchmark for our agent system, because the focus in that paper is on a similar problem as we consider here.

Both methods that we consider are hierarchical methods. At the top level, vehicles are distributed amongst nodes based on actual and expected jobs, without detailed job assignment. At the node level, vehicles are assigned to jobs, where only the vehicles can be used that are assigned to that node by the top level. The advantage of such an approach is that a complex schedule is decomposed into two simpler decisions. One of these decisions, assignment of

vehicles to jobs, should be done in real time. The other decision, distribution of vehicles amongst nodes, should be done frequently, but not necessarily real-time, because it is a higher-level decision without immediate consequences. We will use two methods that fit within this hierarchical framework, namely hierarchical coordination and integrated planning.

Under hierarchical coordination, the top level distributes vehicles using a simple priority rule, based on a central job list and a central overview of all vehicle positions and current activities. First, we calculate the latest departure time for each job as the due time minus an offset for the expected handling time (loading, transportation, unloading) and the variation in the handling time. Thereafter, we sort the job list in increasing order of latest departure times. We process the list sequentially. To each job, we assign the vehicle that can be available at the earliest point in time. If a vehicle is waiting at or driving to a different node, the top level issues an empty vehicle repositioning job with corresponding latest dispatch time to that node.

At the node level, we have a list of jobs to be dispatched (with latest departure time) and a list of empty vehicle dispatch jobs (with latest dispatch time). Every time a vehicle becomes available at the node, we choose the highest priority job from both lists. For efficiency reasons, we try to combine empty dispatch jobs with load dispatch jobs if possible. For example, if it is most urgent to dispatch a job from node A to node B, we look in the job list of node A whether there is a (lower priority) load dispatch job from A to B, and if so, the vehicle takes this load on its trip. Hence, the node level operates independently of the top level, but within the conditions set by the top level (see Van der Heijden, Ebben, Gademan and Van Harten, 2002). In the remainder of this chapter, we refer to this method by *LocalControl*.

In the integrated planning approach, we construct a better planning to distribute vehicles over nodes. To this end, we use serial scheduling (Ebben et al., 2005), where different priority rules are being used to create a sequence of jobs, which are virtually assigned to vehicles. At the node level, we still decide on the assignment of jobs to vehicles. However, to maintain the structure of the vehicle distribution planning from the top level, the node level has to handle all jobs in a sequence that has been prescribed by the top level. In that sense, we move responsibility from the node level to the top level, hoping to receive a better performance in terms of fill rate and distance traveled empty. In the remainder of this chapter, we refer to this method by *SerialScheduling*.

The aim of a hierarchical control concept as described above is to construct a more flexible and fast schedule compared to a fully centralized concept. The difference between centralized, hierarchical, and heterarchical (agent based) control structures is illustrated in Figure 3.2.

Of course, a hierarchical control concept has some advantages compared to purely central control. It requires less data exchange and is capable of reacting quicker to unexpected events because of the allocation of tasks and responsi-

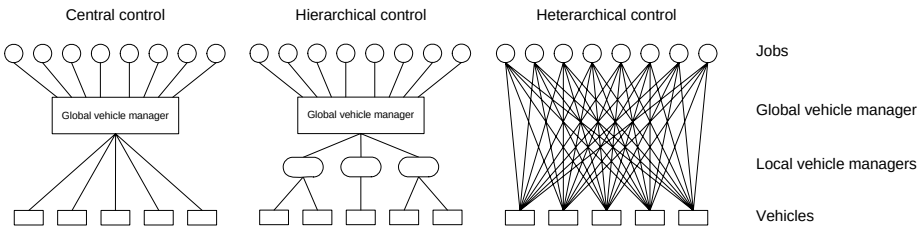

Figure 3.2: Control structures

bilities to two hierarchical levels. However, this hierarchical decomposition of control does not take into account the different roles of various independent stakeholders that negotiate on their mutual services and corresponding prices. Besides, a key difference with the agent approach is that under the hierarchical planning, all job and vehicle information should be centrally available and that a central vehicle distribution plan is constructed.

3.7 Experimental settings

In this section we discuss the experimental design. We successively describe the network characteristics (3.7.1), the fixed parameters settings (3.7.2), and the experimental factors (3.7.3).

3.7.1 Network characteristics

To test the proposed multi-agent concepts and to compare them with other control methods, we first use a network setting inspired by a case study on a proposed underground transportation system near Amsterdam Airport Schiphol, the Netherlands (Van der Heijden, Van Harten, Ebben, Saanen, Valentin and Verbraeck, 2002). We refer to this application as the OLS case, which is the Dutch abbreviation for underground logistic system. In this system, Automatic Guided Vehicles (AGVs) carry cargo between terminals that are connected by tubes. One of the proposed network layouts for the OLS-case, consists of a connection of the airport with the world's largest flower auction market in Aalsmeer (VBA) and a planned rail terminal near the Zwanenburg landing strip (RTZ). At Aalsmeer there is 1 terminal, at Schiphol Airport there are 8 terminals and there is 1 rail terminal. Besides, there is a central parking area where AGVs can wait if there are temporarily no jobs, because the parking space at (underground) terminals is limited. Each terminal has an internal track structure and consists of 4 docks where AGVs can be loaded or unloaded. The terminals are connected by tubes as illustrated by Figure 3.3.

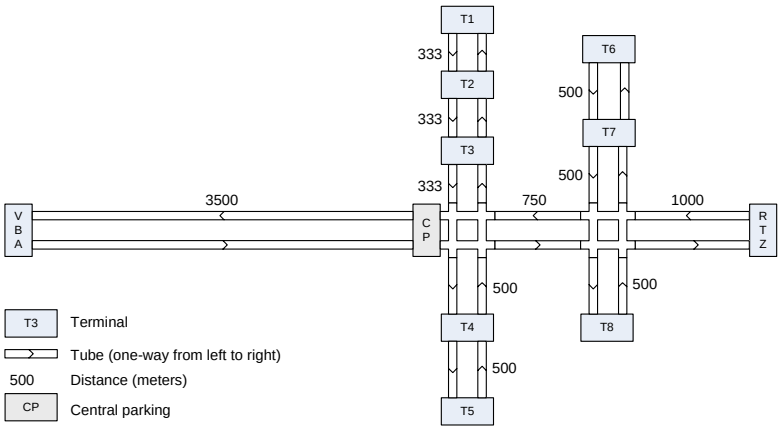

Figure 3.3: OLS network structure

Figure 3.3 shows the distances in meters between terminals, crossings, and the central parking. The times to drive through or along a terminal are significant, as a terminal may have a length up to 200 meters and AGVs drive slower within terminals for safety reasons (see next section). When an AGV enters a terminal to pickup or to deliver a job, it is assigned to a specific dock within the terminal. The distance from the terminal entrance to the dock may vary between 100 and 400 meters, so the dock assignment causes a part of the variation in the handling times of jobs.

Because this is a quite specific setting, we also consider a second network structure consisting of 20 nodes that are uniformly placed in a square region of 10x10 km. All nodes are mutually connected and the distances between these nodes are Euclidean. A central parking area is located in the centre of the square area.

One way to deal with random networks is to generate a few (2-3) random network structures and perform some replications for each scenario. This provides insight into the performance of a few configurations only. Because we are interested in the average performance of the control methods over a range of random network structures, we choose another option. That is, we start each replication with the creation of a new network, i.e., repositioning the location of all nodes. Of course, the required number of replications will increase, but we will also get a better idea of the average performance of the various control methods.

Jobs arrive according to a (non)stationary Poison process (cf. Section 3.7.3). Travel times between terminal entrances are deterministic and known in advance because they only depend on the distance and speed of vehicles. Although the distances are deterministic, the handling times show variation due to the

following:

- Variations in loading and unloading times.
- Waiting times at the terminals due to limitations on the number of AGVs on terminals.
- Waiting times at the terminals due to limitations in dock capacity.
- Dock-dependent distances on terminals.

Therefore, we treat the handling times as random variables. The mean and standard deviation of the handling times are dynamically being updated using a standard exponential smoothing procedure (see Silver et al., 1998). In case of agent-based control we use the shipper agent to keep track of all handling times and the corresponding estimates are available to all vehicles under its control.

To provide an indication of the stochasticity, we found the following values in the OLS network settings. The expected handling times range from 5 minutes (T1 to T2) with standard deviation of 40 seconds, to 25 minutes (VBA to T6) with a standard deviation of 90 seconds. Although these deviations are not very large, they are significant in the handling times.

3.7.2 Fixed parameter settings

The vehicles have a speed of 6 m/s outside the terminals and 2 m/s inside the terminals. The maximum number of AGVs allowed at a terminal is limited to 5.

For the travel cost function we take $c^r(t) = t$ and for the penalty cost function $c^p(t) = 10t$, where t is expressed in minutes. The penalty costs are such that in case of agent-based control a job agent will almost always prefer an AGV that delivers the job with minimum tardiness. The agent-based approaches also use waiting costs $c^w(t)$ in their bid prices. We set these costs equal to the historical average profit per time unit. This information is collected and distributed by the carrier agent.

We set the parameters of DynamicThreshold as follows. P_{min} is equal to the mean price for a specific route, P_{max} to the maximum price paid so far for this route, and the fixed time interval R between the auction rounds is set to 5 minutes, which equals the minimum handling time in the OLS network. The replanning period for the two hierarchical methods is set to 4 minutes.

3.7.3 Experimental factors

In this section we discuss the factors that we will vary in our simulation experiments, for both the OLS network case and the random networks.

OLS network

Table 3.1 shows the experimental factors and their settings for the OLS network.

Factor	**Values**
Demand structure	Stable, Dynamic, Highly Dynamic
Vehicle control	LocalControl, SerialScheduling, AgentAppend, AgentInsert, AgentTSP
Vehicle coordination	None, Trade
Job control	None, DynamicThreshold Linear, DynamicThreshold Quadratic

Table 3.1: Experimental factors

The experimental factor "Demand structure" refers to the variation in transportation flows over time. We distinguish three cases: Stable, Dynamic, and Highly Dynamic. In the stable demand structure, (1) the job arrival rates are identical for all origin destination pairs, (2) the job arrival rates are constant over the day, and (3) all jobs have a same time-window of 60 minutes.

In the dynamic demand structure, (1) the job arrival rates are still identical for all origin destination pairs, (2) the time between jobs vary over hours of the day according to a sinus function with period of half a day, the same mean as in the stable demand structure, and an amplitude of 3 seconds, and (3) we have three different time-windows of 30, 60, and 90 minutes that are drawn with equal probability.

The highly dynamic demand structure is similar to the dynamic demand situation, except that the job arrival rates are no longer identical for all origin destination pairs. The imbalance is given by Table 3.2. Within AAS all terminals have equal probability of being origin or destination.

From \ To	AAS	RTZ	VBA
AAS	0	26%	10%
RTZ	40%	0	4%
VBA	8%	12%	0

Table 3.2: Distribution of transportation flows

The average number of jobs per day is 1800, i.e., a time between jobs (TBJ) of 48 seconds. These jobs have to be transported by a fixed number of AGVs.

This number is chosen such that all methods are capable of handing all jobs in the long run, but not necessarily on time. In case of a stable or dynamic demand structure, we choose to use 20 AGVs and in case of a highly dynamic demand structure 22 AGVs. The announcement times a for jobs are equal to the earliest release times r. Therefore, the time-windows can be defined as the time between the first auction for a job and the due time d.

To limit the number of experiments, we test the vehicle coordination (Trade) and job control (DynamicThreshold) for the highly dynamic demand structure only. We consider four settings: Trade, DynamicThreshold Linear, DynamicThreshold Quadratic, and the combination of Trade with DynamicThreshold Linear. We omit the combination of Trade with DynamicThreshold quadratic pricing because we observed that the differences between both price functions are small (see Section 3.8.1).

Random networks

As a basic scenario we use a network consisting of 20 nodes, 20 AGVs, a time between jobs (TBJ) of 1.5 minute, and a time-window of 60 minutes. As key performance indicator we use the average relative costs per job, see Section 3.3.2. This value resembles the extra costs we make relative to the minimum costs for all jobs.

The experimental factors can be found in Table 3.3. Each of these factors will be varied, keeping the other factors equal to the basic scenario.

Factor	**Values**
Length time-windows (sec)	50, 70, 90, 110, 130
Look-ahead (min)	0, 3, 6, 9, 12
Time between jobs (sec)	84, 87, 90, 93, 96
Amplitude in deviation from mean TBJ (sec)	2, 4, 6, 8, 10
Number of nodes	12, 14, 16, 18, 20

Table 3.3: Experimental factors

We use as control methods LocalControl, SerialScheduling, AgentAppend, AgentInsert, and the combination of AgentInsert with Trade and DynamicThreshold Linear, referred to as *AgentInsertSmart*. To vary the time between jobs during a day, we describe the TBJ as a sinus with a period of half a day and a mean of 1.5 minute, and we change the amplitude.

We use a replication / deletion approach for our simulations (Law and Kelton, 2000), where each experiment consists of a sufficient number of replications (each with different seeds) of six days, each including a one-day warm-up period. The number of replications will be determined in the next sections.

3.8 Numerical results

In this section we present the results of our simulation experiments; first for the OLS network (3.8.1) and next for the random networks (3.8.2).

3.8.1 OLS network

We first determine the number of replications. To this end we consider the percentage of driving loaded (DL) and the service levels (SL) of all control methods for all three demand structures. The maximum number of replications needed with a confidence level of 99% and relative error of 5% is 10. To facilitate comparison, we use 10 replications for all scenarios.

We also performed a paired t-test on the key performance indicators SL and DL of SerialScheduling and AgentInsert using the 10 replications with a stable demand structure. Results show that, for all strategies, both differences are significant with a confidence level of 99%. The results for all control methods for the different demand structures can be found in Table 3.4.

	Stable		Dynamic		Highly dynamic	
Control	DL	SL	DL	SL	DL	SL
LocalControl	73	95.9	72	91.4	78	93.6
SerialScheduling	73	99.2	74	96.6	78	94.8
AgentAppend	82	99.7	80	95.3	82	93.9
AgentInsert	83	100	80	97.8	82	97.0
AgentTSP	83	100	80	98.1	82	97.2

Table 3.4: Simulation results - comparing control methods

We see that AgentInsert and AgentTSP both yield similar results which are better than the hierarchical methods LocalControl and SerialScheduling. Because of the dynamic system behavior, solving a TSP problem exactly has apparently little added value. Because AgentTSP is also computationally intensive, we skip this method in the remainder. We further see that AgentAppend yields lower service levels in some cases. With regard to the percentage of driving loaded we see that our agent approach always perform better than the hierarchical control methods.

The difference in service levels is larger in case of dynamic demand compared to stable demand. In case of highly dynamic demand, we even observe that decreasing the number of AGVs from 22 to 21 yields lower service levels for the agent-based methods whereas the LocalControl and SerialScheduling heuristics are simply not able to handle all demand (the job backlog steadily increases).

To gain insight into the sensitivity of the control methods to the job arrival intensity, we vary the time between jobs from 46 to 50. The impact on the percentage of driving loaded for the different control methods in case of stable

demand can be found in Figure 3.4 and the impact on the service levels in Figure 3.5.

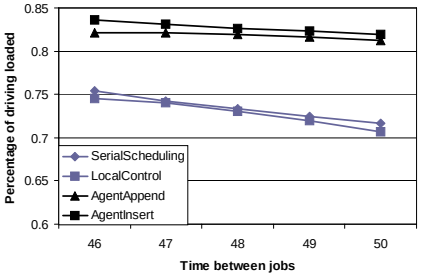

Figure 3.4: Impact of the time between jobs on the percentage of driving loaded

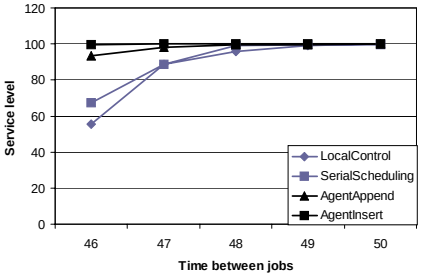

Figure 3.5: Impact of the time between jobs on the service level

From these figures we see that with increasing TBJ, the service levels go to 100% for all control methods. However, the differences in percentage of driving loaded between the heuristics and the two agent methods increase. This means that the agent methods do not lead to unnecessary empty miles if the transportation capacity is sufficient to handle all jobs in time. In case of (highly) dynamic demand situations this effective use of capacity can be used to cope with uncertainty such as rush jobs. Therefore, our agent control seems to be more robust than the two hierarchical control methods. This can also be seen from the standard deviation in service levels per replication over the ten replications for both the stable demand structure (Figure 3.6) and the highly dynamic demand structure (Figure 3.7). We see that our agent control is less sensitive to variations in demand volume, loading times, and unloading times.

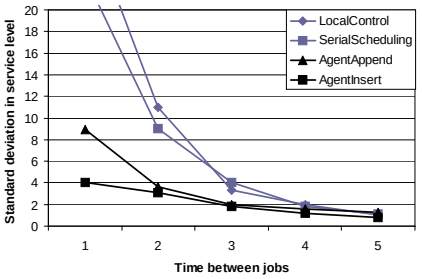

Figure 3.6: Impact of the time between jobs on the standard deviation for the stable demand structure

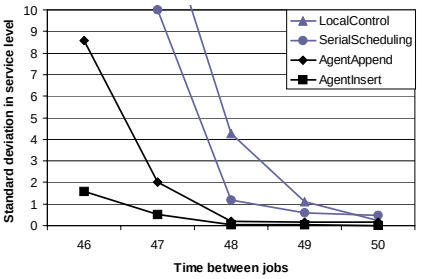

Figure 3.7: Impact of the time between jobs on the standard deviation for the highly dynamic demand structure

To examine the impact of additional agent intelligence, we show the key

performance indicators DL and SL for agent systems with or without vehicle coordination (Trade) and DynamicThreshold (both linear and quadratic) in Table 3.5.

	AgentAppend		AgentInsert		AgentTSP	
Control	DL	SL	DL	SL	DL	SL
Normal	82	93.9	82	97.0	82	97.2
Trade	83	96.2	82	98.2	82	98.3
DT-lin	83	95.5	83	97.5	82	97.9
DT-Qdr	83	95.4	82	97.6	82	97.8
TR-DT-lin	83	96.4	83	98.4	83	98.4

Table 3.5: Simulation results - additional intelligence

We see that the use of additional intelligence improves the performance; especially in case of AgentAppend. However, the improvement in service level is only significant at confidence level 98% for Trade. It might be surprising to see that additional intelligence does not significantly improves the percentage of driving loaded. The reason for this is that there is very little room for improvement. To illustrate this, we calculate a simple upper bound for the percentage driving loaded. Therefore, let us relax the problem by assuming that all jobs are known in advance, there are no time-windows, and all travel- and handling times are deterministic. Then penalty costs are not relevant and the problem reduces to the minimization of the total empty travel time under flow conservation constraints. Using the average handling times and demand data resulting from our simulation experiments for the highly dynamic case, we find an upper bound of 89% for the percentage of driving loaded. Even though this upper bound is calculated under strongly simplifying assumptions, our agent methods still achieve a percentage of driving loaded that is only 6,7% less than the upper bound. On the other hand, LocalControl and SerialScheduling lead to a percentage driving loaded that is 12,4% worse than the upper bound.

3.8.2 Random networks

Again, we first determine the number of replications based on the basic scenario. With a confidence level of 95% and a relative error of 5%, we find that we need 6 replications in case of AgentInsert and 40 replications in case of LocalControl. The differences in the required number of replications for the different control methods are higher than before. This is caused (1) by using the relative costs as key performance indicator, for which the variances are much higher and (2) by the changes in the network layout with each replication. We decided to use 20 replications because this provides enough significance to distinguish between the agent control methods and the two heuristics. This can be seen from the confidence intervals for the relative costs in the basic scenario (Table 3.6).

First, we vary the time between jobs. In Figure 3.8 we see that, for all

Control	Confidence interval
LocalControl	[59.5, 69.7]
SerialScheduling	[57.5, 63.6]
AgentAppend	[40.5, 44.1]
AgentInsert	[38.9, 41.4]
AgentInsertSmart	[39.0, 41.2]

Table 3.6: Simulation results - confidence intervals for relative costs

methods, the relative costs decrease with the time between jobs. We also see that SerialScheduling and LocalControl converge to a same level. The two basic agent methods converge to a same, but lower level. The addition of Trade and DynamicThreshold yields lower costs if the time between jobs is small. The reason is that the average length of the vehicle schedules increases. Then there are more options for load exchange (Trade). Also, it can be more beneficial to use threshold prices.

In the next experiment we keep the mean time between jobs the same for all simulation runs, but we change the deviation from the mean during the day. We see in Figure 3.9 that increasing the amplitude has negative effect on the relative costs. Not surprisingly, the method AgentInsertSmart is less sensitive to these deviations.

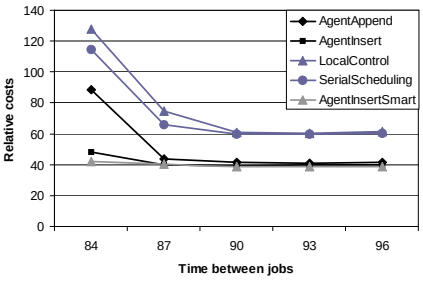

Figure 3.8: Impact of the time between jobs on the relative costs

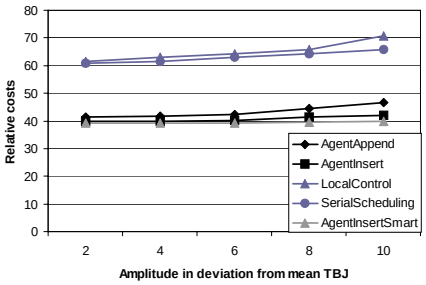

Figure 3.9: Impact of the amplitude in time between jobs on the relative costs

Second, we vary the length of the time-windows. From Figure 3.10 we see that the relative costs for all methods decrease with the length of the time-windows. The costs converge to a situation where the penalty costs are negligible. The methods LocalControl and SerialScheduling are not able to reduce their empty trips when the time-windows increase. AgentInsertSmart benefits more from increasing time-windows because the number N of possible auction rounds is higher.

Finally, we vary the number of nodes in the network keeping the rest of the parameters the same, see Figure 3.11. In case of AgentInsert the relative costs first slightly increase converging to a value of about 40%. AgentInsertSmart

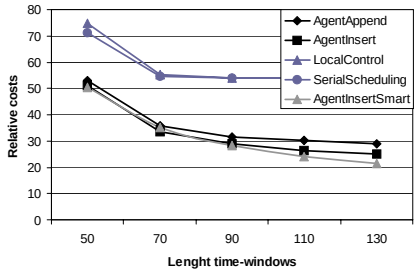

Figure 3.10: Impact of the time-window length on the relative costs

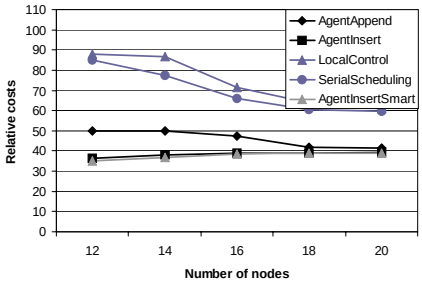

Figure 3.11: Impact of the number of nodes on the relative costs

works relative better if the network is small (less nodes) because then the average distances between nodes are higher and therefore trading jobs will be more beneficial. The relative costs for all other methods decrease in the number of nodes. Closer inspection reveals that the penalty costs always decrease with the number of nodes while the amount of empty kilometers increases. However, both cost factors converge to a stable value. Because the penalty costs are relatively small and stable for AgentInsert, the increase in costs for empty trips dominates the relative costs. The hierarchical methods, SerialScheduling and LocalControl, benefit most from an increase in the number of nodes. However, based on our simulation results, we expect that differences in relative costs will remain. Besides, the computation time of the hierarchical methods (especially for SerialScheduling) increases with the number of nodes. Therefore, we end with some notes about the computation time of the different control methods.

We implemented our methods in the simulation software eM-Plant 7.0 and we performed the experiments using a Intel Pentium 4 processor at 3.4 GHz. The computation times (milliseconds per job) for the basic scenarios can be found in Table 3.7. The computation times of the agent approaches include starting the auction, bidding by all vehicle agents (sequential whereas in practice parallel execution is plausible), and bid evaluation by the job agent. The computation times of the hierarchical methods consist of the time required for periodical replanning and local decision making. In both hierarchical methods, the computation times for local decision making per job are 2 ms on average, the computation times for periodical replanning can be found between brackets.

Control	**OLS**	**Random**
AgentAppend	2	1.6
AgentInsert	8.3	6.6
AgentTSP	19.0	8.8
LocalControl	2.9 (4.9)	3.8 (5.3)
SerialScheduling	20.2 (91.8)	78.6 (109.7)

Table 3.7: Simulation results - computation times

Obviously, the computation time of the agent methods mainly depends on the number of vehicles (participants in the auctions). The computation time of AgentInsert and AgentTSP also strongly depends on the average length of the vehicle schedules. The hierarchical methods mainly depend on the average number of open jobs, but also on the number of nodes in the network. Therefore, we see some longer computation times in the random networks with 20 nodes.

The two agent extensions result in an increase in computation times. The extension Trade results in an increase of about 0.09 ms per job exchange (about 0.5% of the jobs are exchanged in both network settings). The extension DynamicThreshold has a bigger impact, because a load requires on average 2.6 auction rounds, resulting in a proportional increase of the computation time.

3.9 Conclusions

In this chapter we propose a distributed agent-based solution to real-time, dynamic transport scheduling problems. This approach has a number of advantages. First, it is more robust in the sense that it is less sensitive to fluctuations in demand or available vehicles than more traditional transportation planning heuristics (LocalControl, SerialScheduling). Second, it provides a lot of flexibility by solving local problems locally. Third, it provides online decision-making using auction mechanisms.

From our simulation experiments, we conclude that our agent approach yields a high performance in terms of vehicle utilization and service level. When we compare the best hierarchical method (of the two considered in this chapter) with AgentInsert, we see that the differences in costs and percentage of driving loaded are always significant. With regard to the service levels, AgentInsert performs significantly better in most cases and never significantly worse.

We improve the performance of the agent-based approach by using two extensions: (1) we allow vehicle agents to exchange jobs and (2) we allow shipper agents to reject all bids and start a new auction later on. These extensions are particularly valuable if vehicle schedules contain many jobs on average.

In the next chapters, we focus on improvement of the agent behavior. For the vehicle agents, further improvement of the pricing strategy is relevant. Although vehicles can schedule multiple jobs in advance, our model is still myopic. Vehicle agents only consider the direct costs of doing certain jobs, whereas it could be better to include an opportunity loss for arriving at a terminal without a next job with low expectations for an attractive load in the near future. This topic is studied in Chapter 5. For the job agents, formal methods for the dynamic threshold policy will be developed (Chapter 6). But first (Chapter 4) we go into more detail on the design choices that we face in building a multi-agent system.

Chapter 4

MAS: design choices

In this chapter[1] we provide insight into the design choices we may face when building a multi-agent system (MAS). These design choices include (1) the identification of agents; (2) the roles, responsibilities, and decision-making capabilities of these agents; and (3) the way they interact. Current MAS development methodologies provide guidelines to help the designer with these decisions. However, qualitative design guidelines turn out to be insufficient to select the best agent architecture. Therefore, we propose to extend current MAS methodologies by multi-agent discrete event simulations.

To illustrate this approach, we consider a MAS for the logistics control of Automatic Guided Vehicles (AGVs) that are used in the dough making process at an industrial bakery. Using real-life data from the bakery, we evaluate several alternative designs. We find that architectures in which line agents initiate the allocation of transportation jobs, and AGV agents schedule multiple jobs in advance, perform best. We conclude by discussing the benefits of our MAS design approach for real-life applications.

4.1 Introduction

In recent years there has been a growing interest in distributed intelligent manufacturing due to the necessity of greater adaptability and flexibility to changes in the market demand. Agent technology is considered an approach that holds high promises for developing such systems (Jennings et al., 1998). Multi-agent systems (MAS) are believed to be particularly suited for decentralized systems in real-time and dynamic environments. Because problems are solved locally,

[1] This chapter is based on the paper: M.R.K. Mes, M.C. van der Heijden, and J. van Hillegersberg (2008). Design choices for agent-based control of AGVs in the dough making process, *Decision Support Systems* 44(4): 983-999.

these systems should (1) be able to deal with a high level of complexity, (2) require less information exchange than central control methods, (3) respond fast to unexpected events, and (4) reduce system nervousness compared to global optimization (which may lead to completely modified plans in case of minor information updates). In the previous chapter, we have described the benefits of decentralization and we have compared the performance of a basic multi-agent system with two central scheduling heuristics. Using a case study on an underground AGV system around Amsterdam Airport Schiphol, we found that a properly designed multi-agent system performs as well as or even better than central scheduling methods.

As multi-agent systems are starting to find their way from laboratory settings to real-life manufacturing, full life-cycle methodologies are needed to support MAS development. Methodologies that have recently been introduced are built upon concepts from object-oriented software engineering and artificial intelligence. These methodologies generally provide guidelines for identifying agents, their roles, responsibilities, and interaction protocols (Jiao et al., 2005). However, when applying these guidelines, several alternative designs for MAS can be derived. Designs may vary in the roles and responsibilities assigned to agents, the level of intelligence of the agents (forecasting and learning behavior), and the interaction protocols selected. Current MAS methodologies lack a mechanism to evaluate such design-choices and provide only limited support to the designer in selecting the preferred design for implementation. Therefore, we propose to extend current MAS methodologies by multi-agent discrete event simulations. These simulations provide insight into the effect of MAS design choices on system quality aspects such as logistical performance (handling and delivery times), scalability, computing time, communication cost, and robustness of the system.

We demonstrate and test this approach by applying it to a real life project; the design and development of a MAS for manufacturing biscuits at the industrial bakery Merba in the Netherlands. Merba produces a wide range of cookies for the Dutch and international market and is among Europe's largest producers of American chocolate chip cookies.

The goal of the project is the automation of the dough making process. Currently, employees collect ingredients for dough manually into barrels and move these barrels between the various processing locations. This manual process has a negative effect on the labor conditions, on the product quality, and on the traceability of ingredients. Product quality problems arise from deviations in rising times of dough and amount of ingredients. Also, human body contact with the dough is inevitable. To overcome these problems and to achieve cost reductions, Merba aims at a fully automated dough production process using Automatic Guided Vehicles (AGVs). To achieve a reliable and flexible AGV system, Merba aims at implementing MAS for scheduling transportation tasks.

During 2006, we have worked with Merba in implementing this system. In

carrying out this project, we found that current MAS development methodologies aid in creating various alternative MAS designs, but do not provide sufficient support to select the preferred design for implementation. Therefore, we applied multi-agent discrete event simulations in addition to the conceptual design stages proposed in common MAS design methodologies. In this chapter we demonstrate and evaluate this approach. We investigate in what way design choices effect logistic and system performance. We thus follow a design science approach (Hevner et al., 2004) in which the artifact (the MAS methodology) is extended by adding simulation and evaluated in a field project.

The remainder of this chapter is structured as follows. In the next section we give an overview of related literature. The Merba setting and the requirements of the project are described in Section 4.3. In section 4.4 we present our extension to current methodologies to design MAS. We describe the resulting alternative agent-based designs in Section 4.5. Next, we present our simulation experiments of the alternative designs in Section 4.6, present some extensions (Section 4.7), and end up with conclusions (Section 4.8).

4.2 Literature

Wooldridge and Jennings (1995) define an agent as a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives. An intelligent agent is further required to be proactive and social (Wooldridge, 2002).

Agent technology has been used for a vast range of applications, ranging from e-mail assistants to air traffic controllers (see Jennings et al., 1998). Recently, agent-based control architectures have been suggested as alternatives to traditional manufacturing control techniques (McDonnell et al., 1999). One of the earliest agent-based manufacturing systems, called "yet another manufacturing system" (YAMS), was developed by Van dyke Parunak (1987). He considers a hierarchical production system in which each node (factory, manufacturing cell, workstation, and machine) is represented as an agent. Each agent has a collection of plans and negotiates with lower level agents to assign production tasks. In a later paper, Van dyke Parunak et al. (2001) presented AARIA (Autonomous Agents for Rock Island Arsenal) in which the manufacturing resources (e.g. people, machines, and parts) are encapsulated as autonomous agents that are using a mixture of different scheduling techniques. This approach, to represent manufacturing resources by agents, is common in agent-based manufacturing systems. Coordination between the agents is usually achieved through negotiation or by means of an auction protocol. For example, Lin and Solberg (1992) introduce part agents and resource agents that negotiate with each other to achieve individual objectives. Maturana et al. (1999) also use resource agents and introduce mediator agents for coordination between agents. McDonnell et al. (1999) use three classes of agents:

part managers, resource managers and information managers. Coordination is achieved using an auction protocol to construct complete plans for part production. In (Maione and Naso, 2001) part dispatching decisions are made by part agents and machine agents. The proposed approach is effective and reactive to severe disturbances and changes in the manufacturing environment. Shen and Norrie (2001) present an approach that combines mediation and bidding mechanisms for agent-based dynamic manufacturing scheduling. Kim, Graves, Heragu and St. Onge (2002) consider an industrial warehouse order picking problem where goods, storage areas, and order pickers are modeled as intelligent agents. Real-time task allocation is achieved through the use of a negotiation mechanism. Their results demonstrate an increase in throughput, flexibility, robustness, and fault tolerance with respect to unforeseen events. For more references on agent-based systems in the manufacturing area we refer to (Jennings et al., 1998; Van dyke Parunak, 1999; Shen and Norrie, 1999).

Most papers on agent-based control of AGV systems focus on routing (e.g. Wallace, 2001; Frazzoli et al., 2005) and the MAS architecture (e.g. Heragu et al., 2002). Only a few papers have appeared on planning and scheduling decisions in agent-based control of AGVs. An early application can be found in (McElroy et al., 1989) where intelligent AGV agents bid for transportation of loads. Also closely related is the decentralized architecture of (Liu et al., 2002) for the coordinated control of AGVs. They compare their approach with a centralized approach, and conclude that their system provides higher utilization and is more robust to fluctuations in processing times. Boucke et al. (2004) propose a negotiation protocol for flexible and decentralized allocation of transportation tasks. This approach consists of a negotiation protocol where the allocation of tasks is continuously reconsidered until the task is actually started. Lau et al. (2003) describe AGV control for material handling in an automated warehouse. They present a self-organizing distributed system where schedules for transportation tasks arise from interactions between the AGVs.

We observe, for both manufacturing and transportation systems, that (1) the majority of research on agent-based planning and control focuses on the development of generic architectures/frameworks; (2) usually only a single architecture is given without any quantitative selection from alternative designs; and (3) case oriented research is often limited to a conceptual description in combination with simulation based on artificial data rather than real-life data.

The contributions of this chapter are (1) to show that qualitative arguments and modeling guidelines in current MAS methodologies are insufficient to select a single "best" architecture for MAS; (2) to show how simulation can be used to help in this selection process; and (3) to evaluate this approach by applying it in a real world setting.

4.3 Requirements for the agent system

Before presenting the requirements for the agent system, it is important to make a distinction between the scientific and practical interest of this research. Our scientific interest - and hence the goal of this chapter - is to provide insight into the MAS design choices, and to develop a decision support tool (in this case simulation) in order to evaluate the impact of these design choices on the system quality aspects as mentioned in the introduction. Our practical interest is related to the case study at Merba bakeries for the automation of the dough making process. In this chapter this case study serves as an illustration to support our research objectives. For this purpose it is not required to provide a full case description with all kind of details on the dough-making process.

To come up with an AGV control system we first established the system requirements by performing interviews with the management of the bakery. The main requirement is a flexible production system that can easily be adjusted to new product introductions or modifications in the bakery layout, and can react in real-time on process uncertainties like equipment failures, product quality problems, and the arrival of rush jobs. For example, if the quality of a (part of a) batch is insufficient, additional dough has to be prepared, leading to insertion of new jobs in the dough preparation schedule. Because of the first issue, the management decided to use AGVs rather than pipelines for the transportation of dough ingredients (as is common in industrial bakeries). Because of the second issue, the management prefers a multi-agent system for the logistic control of the AGVs.

The main requirement for the AGV system is that both, the movements of AGVs and the control of AGVs, are flexible. Although the specific details are beyond the scope of this chapter, we mention a few. For details on different AGV systems we refer to (Le-Anh and De Koster, 2006).

With respect to the AGV movements we can think of wireless guidance (i.e., laser or inertial), automatic battery charging or automatic battery swaps, and a flexible guide path. These guide paths should provide the AGVs enough freedom to drop their load at different locations (here locations are not always fixed points but rather areas in the factory). Here we can think of free-ranging AGVs (which means that their preferred tracks are software programmed and can be changed relatively easily) and unidirectional conventional (networked) guide paths (see Le-Anh and De Koster, 2006). Decisions regarding battery management and guide paths may have an impact on the availability of AGVs, and hence on the planning and scheduling decisions within our multi-agent system. Here we assume that this impact can be expressed indirectly through the travel times between different object in the factory. These travel times are input of our simulation environment. Besides, given the factory layout (Figure 4.1) and the estimated number of AGVs (<10), congestion is not likely to occur.

With respect to the control of AGVs, we require that the routing decisions

as well as the planning decisions are taken by the vehicles themselves. This requires close cooperation with the vendors of AGVs and of the AGV control system. Luckily, there is an increased interest among these companies in so-called smart vehicles and distributed control systems. A recent example can be found in Weyns et al. (2005), where a Belgian manufacturer of AGVs, Egemin N.V., in cooperation with the AgentWise research group, did a pilot project with agent controlled AGVs.

From the requirements mentioned above we designed several multi-agent systems. We discussed these alternative designs, together with our modeling assumptions, with the management of the bakery. To evaluate the design alternatives, we implemented a prototype in a simulation environment and collected the input data for this experiment at the bakery. We provided these alternatives together with the simulation results to the bakery management in order to make the final decision.

In the next subsections, we describe the dough preparation process at the industrial bakery, our model of the physical process with our basic assumptions, and the decisions involved in planning and control.

4.3.1 Process description

The production process at this bakery consists of three phases: (1) preparation of dough, (2) baking of cookies, and (3) packing and storage. Our focus is on the first phase, the preparation of dough. Dough is produced in barrels that are essentially the same. AGVs transport the barrels between the various locations in the dough preparation process, see Figure 4.1.

The process always starts with a dough production request generated by the Manufacturing Execution System (MES). Each dough request is restricted by an earliest- and latest delivery time of the dough at the production line. The timing of dough requests depends on the day planning and the packing department where cookies undergo a quality check.

First, we have to find a suitable barrel for the dough request. This can be a barrel that has been used before for a similar dough type, or a barrel that has been used for an incompatible dough type which has been cleaned at a special cleaning area. An AGV picks up the barrel and moves it towards a storage area consisting of three silos. Each silo may contain multiple ingredients and the silos have to be visited in a fixed order (displayed by S1 - S2 - S3 in Figure 4.1). The AGV positions its barrel below these silos to collect the ingredients. The time spent at each silo is the same for all dough types. Next, the AGV moves the barrel to a mixer. A single unique mixer is assigned to each dough type, but a mixer may process multiple dough types. During mixing, also new ingredients might be added to the dough such as decoration or ingredients that may only be added just before finishing mixing (like chocolate chips). After mixing the

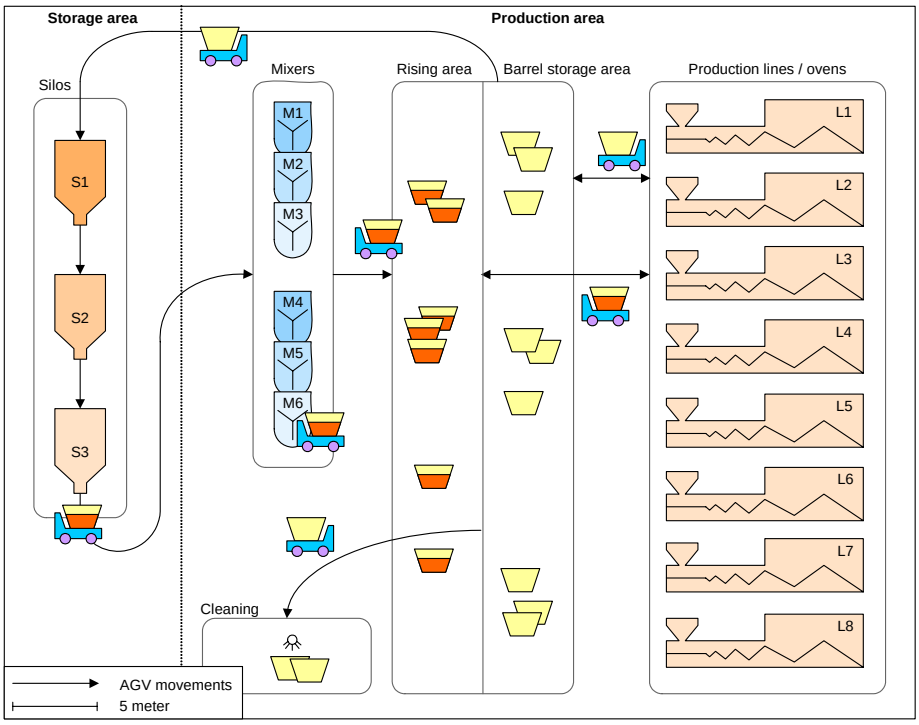

Figure 4.1: Factory layout

ingredients, the dough has to rise before it can be put into the oven. Once the rising time has passed, an AGV moves the barrel to the production line. Each dough type is assigned to a single production line, but each production line may process multiple dough types. There are no setup times for switching between dough types. At the production line, the barrel is emptied in a feeder. This feeder slowly delivers the dough to a conveyer belt moving through the oven. Because a feeder can store more than one barrel of dough, dough can be delivered some time before it is actually needed, i.e., before the earliest delivery time. The empty barrel will stay on the AGV or will be dropped at the barrel storage area.

The rising time plays an important role in planning and control of the dough preparation process. Rising starts when ingredients are mixed and stops when the dough is used at the production line. That is, rising continues if the dough is being transported by an AGV and while it is waiting in the feeder. However, each dough type has a minimum rising time at the rising area. Therefore, an AGV always waits at the rising area until the minimum rising time has passed, even if this leads to a violation of the latest delivery time. For each dough type, a best rising time is specified. The product quality depends on the deviation

between the actual rising time and the best rising time. A key problem for the planning and control system is to balance product quality (deviation from the best rising time) and tardiness (w.r.t the latest delivery time).

4.3.2 Model

Planning of the dough preparation process consists of scheduling all transportation jobs. To describe this in more detail, we discuss in this section (1) generating dough requests, (2) translating dough requests to transport jobs, (3) the overall goal function of the planning problem, and (4) the model assumptions.

Generating dough requests As mentioned in Section 4.3.1, dough requests are generated by the Manufacturing Execution System. We assume that dough requests arrive one by one according to some stochastic process (e.g. a Poisson process). Each dough request has the size of a single barrel, and is characterized by a certain dough type and time-window restrictions. Each dough type has a unique mixer, production line, and minimum- and best rising time. The time-window of a dough request consist of an earliest- and latest delivery time. The earliest delivery time is the time a line expects it can start processing the dough. If dough is delivered before this time, it has to wait in the feeder. Delivery after the latest delivery time is penalized.

Translating dough requests to transport jobs From the process description in Section 4.3.1, we can distinguish five job types for the AGVs: (1) silos - mixer, (2) mixer - rising area, (3) rising area - production line, (4) production line - barrel storage area, and (5) barrel storage area - silos. At each location, an AGV may drop the barrel in order to carry out other transportation jobs during the processing times for e.g. rising and mixing. However, it is not always practical to drop the barrel, because (1) dropping and picking up the barrel takes time and may cause waiting time for an AGV after processing is finished and (2) the AGV is needed during several processing steps for technical reasons (e.g. an AGV is needed to move the barrel from one silo to the other when collecting ingredients). For these reasons, the management decided that an AGV is not allowed to drop the barrel at the silos, the mixer, and the production lines. As a consequence, we only have two job types for the AGVs: a *preparation job* (barrel storage area - silos - mixer - rising area) and a *delivery job* (rising area - production line - barrel storage area).

A preparation job may be scheduled immediately after release of a dough request, even if the corresponding line has already released other jobs that are not yet delivered. We decided to postpone releasing the delivery job until the corresponding dough has been delivered at the rising area, because (1) the earliest- and best delivery time of the delivery job is dependent on the uncertain delivery time of the dough at the rising area and (2) rising times provide enough

flexibility to schedule the delivery job. Whenever an AGV delivers dough at the rising area, it informs the corresponding line about the actual delivery time so that it can release the delivery job.

Goal function The overall goal is to minimize costs based on two costs drivers: deviation from the best rising time and tardiness with respect to the latest delivery time. We normalize the penalties for deviation in rising time to 1 per time unit. We use the relative costs α for one minute tardiness compared to one minute deviation from the optimal rising time. The management of the bakery can influence the planning by manipulating α as the relative importance of tardiness compared to deviation from the best rising time (timeliness versus product quality). Note that these penalty functions can easily be extended towards non-linear functions. Disadvantage of this is that it will be less intuitive to the managers.

Model assumptions Throughout this chapter, we make the following assumptions:

1. Although all travel- and processing times may be stochastic, we assume that their means are known. Only the mean waiting times have to be estimated by the agents themselves.
2. We do not explicitly include traffic congestion in our model, but we can correct for traffic delays by adjusting the effective AGV speed.
3. All dough requests have to be handled, even if they are late.
4. An AGV can park at any location when it is idle.
5. We omit the processing times of dough at the production lines from our model. By using externally generated time-windows we ignore possible interdependencies between subsequent dough deliveries at the same line with limited capacity.
6. We omit batteries from our model. We assume that recharging or swapping batteries takes place during the idle times of AGVs.
7. We omit cleaning of barrels for ease of presentation. We assume that whenever an AGV delivers dough at the production line, it drops the empty barrel at the barrel storage area. For each new dough request, there is a clean barrel available at the barrel storage area.

4.3.3 Planning and control

The main functionality the AGV system should perform is to assign all transport jobs (preparation and delivery jobs) to AGVs and to schedule these jobs.

In order to do so we have to reckon with (1) the minimal and best rising time of dough and (2) limited capacity (and therefore waiting times) at the silos and mixers. Because we decompose our system into multiple agents, the main goal of the bakery has to be achieved by individual agents with individual goals. Here we face two difficulties: (1) we have to deal with multiple criteria and (2) goals of individual agents may differ from the main goal (or might even be conflicting). To deal with multiple criteria we introduced the relative costs α. An example of divergence in goals is that minimizing the costs of one dough delivery may have a negative effect on the costs for the next dough delivery. An AGV with the goal to minimize the tardiness and deviation from best rising times might incorporate extra waiting for a preparation job such that the expected rising time equals the best rising time. Because scheduling jobs has an impact on the future availability of AGVs, it might be the case that future jobs are delivered late. Therefore, we enable the individual agents to value their capacity. In the next section we describe alternative agent architectures to support the allocation and scheduling decisions.

4.4 Alternative designs for the agent system

According to Luck et al. (2003), a suitable methodology for analyzing, designing, and building multi-agent systems is a key factor for introducing agent-orientation as an engineering approach to the industry. Three well known methodologies are Prometheus (Padgham and Winikoff, 2004), Gaia (Wooldridge et al., 2000), and MaSE (Wood and DeLoach, 2000). Roughly speaking, each of these three methodologies consists of the following steps:

1. Decomposition of the system into multiple functionalities.
2. Allocation of functionalities to agents.
3. Establishing interaction protocols between the agents.
4. Designing the decision making capabilities of the agents.

The terminology may vary, but the approaches have many similarities. The first step is usually achieved by listing all system goals and grouping related goals. These related goals, together with related data, triggers, and actions, form functionalities. The main task here for the system designer is to decide among alternative decompositions. In the second step it is decided how these functionalities are allocated to agents. In the third step, we face several design choices such as the sequence of steps in an interaction protocol. In the last phase we have to design protocols for internal processing of the individual agents. This involves the way they react on triggers and incoming messages. In our approach, we call the first three steps the *architectural* design phase

and the last step the *detailed* design phase. The architectural design phase is generally supported by Agent Oriented Methodologies. For the Detailed Design Phase, support is currently lacking; especially to quantify the quality of the design. Therefore, we apply simulation as a design technique to support this phase.

Although our proposed method is independent of the specific agent design methodology used, we select the Prometheus methodology and the Prometheus Design Tool. It is a practical, rather complete and easy to understand methodology that especially provides support to our design choices in the architectural phase, which we describe below. The detailed design phase is described in Section 4.5.

4.4.1 Architectural design phase

Decomposition of functionalities (step 1)

The main goal of the bakery, balancing the deviation from best rising times and tardiness (Section 4.3.3), has to be decomposed into multiple functionalities, which can be assigned to different agents. A functionality describes a behavior, consisting of decisions and actions, together with relevant triggers and data (Padgham and Winikoff, 2004). Here we focus on the decisions and ignore physical actions (drive, pickup etc.) which are obvious. First, we create a network of connected goals (see Padgham and Winikoff, 2004). The main design choice here is to group these goals. To select reasonable groupings we use the standard software engineering criteria of coupling and cohesion. Coupling is the level of interdependency between functionalities, while cohesion is the level of uniformity of the goals in a functionality. After evaluation of different groupings we end up with three functionalities: AGV selection, dough preparation management, and dough delivery management. AGV selection is concerned with the selection of AGVs waiting in a queue before a silo or mixer. The last two functionalities are concerned with the allocation and scheduling of respectively preparation jobs and delivery jobs.

Allocation of functionalities to agents (step 2)

Next, we have to allocate the functionalities to agents, which represent physical objects in the bakery. We have the following objects: AGVs, lines, silos, and mixers. Besides an AGV agent and a line agent, we use a storage agent that combines the silos and mixers because an AGV will always visit a mixer directly after visiting the silos.

In order to assign functionalities to these agents we look at data and triggers. Functionalities may be triggered by actions of physical objects within the factory, which then form candidates for these functionalities. We also eval-

uate the data used and produced by different functionalities. This of course requires an iterative approach, because at this point we can only guess where information is located and which information is necessary for decision making. Functionalities that share the same data source form candidates for allocation to the same agent because it requires less information exchange.

The AGV selection functionality has as triggers the arrival of an AGV at a silo or mixer and finishing loading ingredients of mixing. Therefore, we allocate this functionality to the storage agent. For the dough preparation and dough delivery functionalities, we decide to investigate allocation to either the line agent or AGV agent. If we allocate these functionalities to the line agent, it will search for an AGV based on its triggers (e.g. it receives a dough request or dough has been delivered at the rising area). If we allocate these functionalities to the AGV agent, then it will search for a job at all lines based on its triggers (e.g. it becomes idle). Because both allocations seem reasonable (based on qualitative arguments), we decide to evaluate both of them using simulation. In the remainder of this chapter we refer to the case where these functionalities are assigned to the line agent by line centric (LC) and the case where they are allocated to the AGV agent by AGV centric (AC).

Interaction between agents (step 3)

Having allocated functionalities to agents, we now have to specify how agents exchange information in order to perform their given functionalities. Again, we use Prometheus by building scenarios, interaction diagrams, and protocols (see Padgham and Winikoff, 2004). Main difficulty is to establish suitable interaction sequences that describe (1) which agents communicate with which other agents and (2) the timing of communication. Given the different agent- and message types, we might end up with a large number of possible interaction sequences. Therefore, we propose a stepwise approach. First, we focus on the order in which agents are involved in an interaction sequence, which we indicate by a communication sequence. From this, we derive communication schemes that describe who communicates with whom. Next, we make a selection of suitable communication schemes. Finally, we specify communication by describing the communication protocols. This results in several agent architectures which we evaluate using simulation.

We illustrate this approach only for the dough preparation management functionality. The interaction sequences for the other two functionalities are obvious because they require only two agent types.

Communication sequences The initiator of a communication sequence is given by the agent that is responsible for the functionality under consideration. Given the two allocations of the previous section (LC and AC) and the three agents (AGV, line, storage) involved in the dough preparation manage-

ment functionality, we have 4 possible sequences. However, we also have an option to discard some agents in the decisions processes. We decided also to consider communication sequences without the storage agent, which result in 2 additional sequences (1 for each allocation).

Communication schemes An overview of all possible communication schemes for the dough preparation functionality is given in Figure 4.2. The initiator (agent at the first row) always communicates with the second agent in the sequence. The third agent, however, can be contacted either by the first or by the second agent. Each of these schemes provides a rough sketch of a possible protocol. Consider, for example, the 6th scheme: based on its triggers, the line agent is triggered by a dough request and generates a preparation job. The line agent contacts the AGV agents for offers to process this job. To generate an offer, each AGV has to decide when the job should be started. Therefore, they communicate with the storage agent about available capacity at the silos and mixers.

Selection of communication schemes In principle, each scheme from Figure 4.2 could be implemented. However, we select a few schemes for our numerical experiments using qualitative arguments. We may consider (1) the scarceness of resources in the communication sequences and (2) the required information exchange in the communication schemes. The scarceness of resources is not unambiguous here because in a dynamic environment - such as the bakery - it may occur that at different moments, different resources will be the bottleneck. Therefore, we focus on the expected information exchange. As a guideline regarding the information exchange we avoid schemes which require relatively more information exchange while decisions are based on the same information.

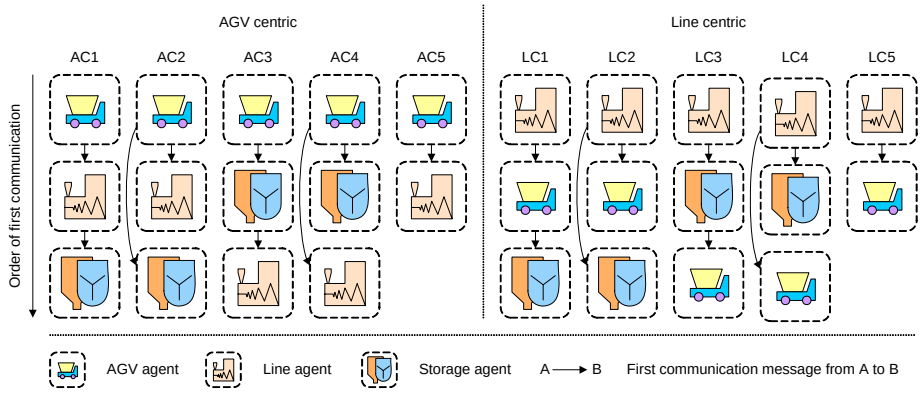

Figure 4.2: Communication schemes

Basically, the dough preparation management functionality should support the following decision: which AGV should do which preparation job at what time. The three agents involved in this decision process have the following information that might support this decision. The line agent has knowledge about dough requests and dough deliveries. The AGV agent has knowledge about its availability and expected waiting times. The storage agent has knowledge about AGV arrivals. For both allocations (LC and AC) we select one communication scheme with three agent types using the following two observations:

1. If the first agent in a scheme communicates with the other two agents, it is required that the second agent provides all necessary information to the first agent. Otherwise, the first agent has to provide all necessary information to the second agent. In the AGV centric schemes, an AGV agent only informs the other players about its idle status. So schemes AC2 and AC4 require more information exchange than schemes AC1 and AC3 respectively. In the line centric schemes, AGVs may have schedules with multiple jobs, and the storage agent may have a schedule with multiple AGV arrivals. Because the line agents inform the other players only about a single job, schemes LC2 and LC4 require more information exchange than schemes LC1 and LC3 respectively.

2. Communication with the line agent in the AGV centric schemes always involves communication with all line agents (because we want to find the most suitable job for a specific AGV), and communication with the AGV agent in the line centric schemes always involves communication with all AGV agents (because we want to find the most suitable AGV for a specific job). Therefore, it requires less information exchange if the storage agent is used as second agent instead of third. So we skip schemes AC1 and LC1.

After applying these guidelines we end up with schemes AC3, AC5, LC3, and LC5. Note that we only skipped schemes for which there is an alternative that requires less communication in order to make decisions based on exactly the same information. Therefore, this choice does not affect the logistics performance. However, it has an impact on the responsibilities and decision making capabilities of the agents. In schemes AC3 and LC3, the storage agent plays a more central role compared to the skipped schemes. A possible disadvantage, we did not take into account, is that these schemes have a single point of failure.

Agent architectures Next, we have to specify the communication protocols to be applied to the remaining communication schemes. The most common protocol between agents, in both real applications and detailed simulations, is the Contract-Net Protocol (CNP) (Van dyke Parunak, 1987). The CNP,

introduced by Smith (1980), is a high level negotiation protocol for achieving efficient cooperation. This protocol consists of four steps: (1) an initiator sends a call for proposals to a set of participants, (2) the participants respond with a proposal, (3) the initiator chooses the best proposal and awards a contract to the respective participant, and (4) the other participants are rejected. We use the CNP to support the dough preparation management and dough delivery management functionalities. For the AGV selection functionality we simply use a FCFS strategy, i.e., the storage agent simply selects the AGV that arrived first in a queue. Given the remaining communication schemes we derive four architectures. Here an architecture consists of a description of how agents react on triggers and exchange information with other agents.

In the agent centric architectures (AC3 and AC5), the job allocation process is triggered by an AGV that becomes idle. Then it might occur that a line receives a dough request while all AGVs are idle. In this case the dough will never be allocated to an AGV. Therefore, we also trigger an arbitrary idle AGV, if there is one, whenever a new job arrives.

In AC3 (Figure 4.3), the AGV informs the storage agent about its position whenever it becomes idle. In return, the storage agent sends a request to all lines to submit their job characteristics. After receiving the job characteristics, the storage agent selects the most suitable job, informs the corresponding line agent about the expected delivery time of this job, and informs the AGV agent where to move to and when.

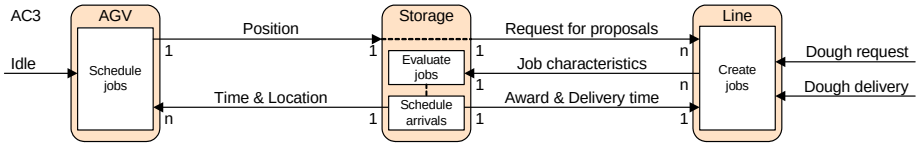

Figure 4.3: AGV centric architecture AC3

In AC5 (Figure 4.4), the AGV sends a request to all lines to submit their job characteristics. After receiving the job characteristics, the AGV agent selects the most suitable job and informs the corresponding line agent about the expected delivery time of this job.

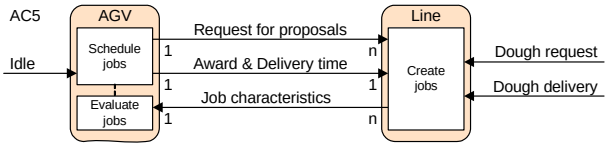

Figure 4.4: AGV centric architecture AC5

In the line centric architectures (LC3 and LC5), preparation jobs are triggered by a line agent who receives a dough request. Delivery jobs are triggered when corresponding dough has been delivered to the rising area. In LC3 (Figure 4.5), the line agent informs the storage agent about a new job. The storage agent sends a request to all AGVs. After receiving the requested information from all AGVs, the storage agent selects the most suitable AGV and informs the line agent and the AGV agent.

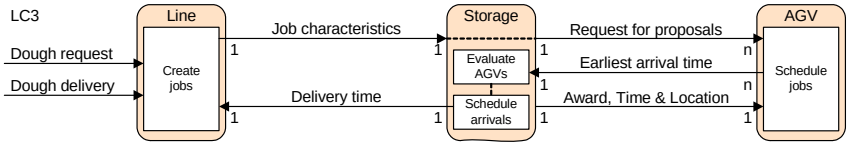

Figure 4.5: Line centric architecture LC3

In LC5 (Figure 4.6), the line agent sends the job characteristics to all AGVs. Each AGV selects the best time to start this job and calculates a price. The line agent simply selects the AGV with lowest price and informs the winning AGV.

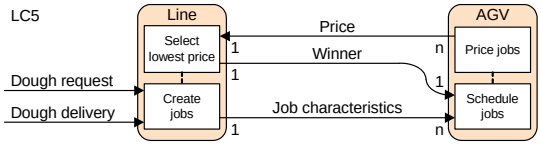

Figure 4.6: Line centric architecture LC5

The decision making capabilities of the agents (illustrated by the white rectangles in the figures above) are described in Section 4.5.

4.4.2 Summary of the architectural design phase

In the architectural design phase we go from a main system goal through a sequence of the following steps: (1) decomposition into functionalities, (2) allocation of functionalities to agents, and (3) establishing interaction protocols between the agents. The last step is achieved by (a) determining communication sequences for all allocations, (b) determining communication schemes for each sequence, and (c) determining interaction protocols for all communication lines in each scheme.

This design approach enables us to avoid overlooking promising architectures. However, when there are a lot of resources (with corresponding agents), the number of possible schemes may become very large. In our case we have 12

schemes for the dough preparation management functionality and 2 schemes for the dough delivery management functionality. Suppose we also incorporate the cleaning process of barrels. Then we have an additional functionality called barrel management and a cleaning agent. The total number of possible schemes is then 168. By going through a step-wise approach we are able to limit the number of possibilities in a structured manner. In our case we end up with 4 architectures.

4.5 Detailed design phase

In this section we describe the following decision making capabilities that are required in the four alternative architectures: create jobs (4.5.1), evaluate jobs (4.5.3), evaluate AGVs (4.5.4), schedule arrivals (4.5.5), schedule jobs (4.5.6), and price jobs (4.5.7). In addition, we introduce a waiting strategy (4.5.2) that is required for scheduling and job evaluation, and end with a section on how agents estimate parameters (4.5.8).

4.5.1 Create jobs

As mentioned before, a dough request leads to a preparation job and a delivery job for the AGVs. The characteristics of these two jobs are determined from the characteristics of the dough request. We denote the set of preparation jobs and delivery jobs by $\mathcal{L}^p$ and $\mathcal{L}^d$ respectively.

A dough requests is characterized by a dough type, a release time a, a production line p, and an earliest- and latest delivery time at the production line denoted respectively by e^d and l^d. The dough type uniquely describes the following characteristics of a dough request: the minimum rising time r_{min} at the rising area, a best rising time r_{best}, and a mixer m. For notational convenience we subtract the travel time from the mixer to the rising area from the rising time. The rising time therefore starts upon delivery at the rising area and ends at the time the line start working on this dough.

When a line receives a dough request at time a, it creates a preparation job φ with the following characteristics: a production line p, a mixer m, and a best- and latest delivery time at the rising area, respectively denoted by b^p and l^p. For the best delivery time we use $b^p = max(a, e^d - r_{best})$ because we want to deliver this job as early as possible such that there is more flexibility for the corresponding delivery job. For the latest delivery time we use $l^p = l^d - r_{best}$ because delivery after this time certainly results in penalties for the corresponding delivery job.

When a line receives a message at time t that dough has been delivered at the rising area, it creates a delivery job φ for this dough. This delivery job is

characterized by a production line p and an earliest-, best-, and latest delivery time of the dough at the production line, respectively denoted by e^d, b^d, l^d. The earliest delivery time is given by $e^d = t + r_{min} + \tau$, where τ is the travel time between the rising area and the production line of the job. The best delivery time is given by $b^d = min(t + r_{best}, l^d)$.

In the remainder we also write the above characteristics as functions of a job φ to denote the characteristics of a specific job. For example, the best delivery time of job φ is given by $b^d(\varphi)$.

4.5.2 Waiting strategy

To describe the waiting strategy we introduce a best starting time of a job. The best starting time of a preparation job provides the best time to start loading the ingredients at the first silo. The best starting time of a delivery job provides the best time to pickup the dough at the rising area. To derive the best starting time, agents have to be able to make a trade-off between loss of capacity (waiting) and the direct costs caused by deviation from the best rising time or tardiness w.r.t. the latest delivery time. Therefore, we introduce a cost factor β for the value of AGV capacity per time unit (see Section 4.5.8 for estimation of this parameter).

If ($\beta \geq 1$), then the costs for waiting are higher then the expected costs for deviation in rising times. The best starting time of a preparation job is given by the earliest arrival time of the AGV at the first silo. The best starting time of a delivery job is given by the maximum of the earliest pickup time (delivery time plus minimum rising time) and earliest arrival time of the AGV at the rising area. Otherwise, if ($\beta < 1$), then it is better to wait if this results in less deviation from the best delivery time. The best starting time of a preparation job is given by the maximum of the earliest arrival time of the AGV at the first silo, and the best delivery time b^p minus the expected time between loading the ingredients at the first silo and the time to drop the barrel at the rising area. The best starting time of a delivery job is given by the maximum of the earliest arrival time of the AGV at the rising area, and the best delivery time b^d minus the travel time from the rising area to the production line.

4.5.3 Evaluate jobs

Job evaluation is used in the agent centric architectures to determine the job which should be handled first. Therefore, we determine a priority value of each job. This value should reflect the priority of a job and the waiting time of an AGV doing this job. The total handling time is not a valid selection criterion because delivery jobs have shorter handling times than preparation jobs but may be equally important.

As input we need the characteristics of all jobs, and the expected earliest delivery times $z_{v\varphi}$ for an AGV v to deliver job φ, for all jobs. In AC3, jobs are evaluated by the storage agent. To calculate the earliest delivery time of an AGV, the storage agent receives the current location of the AGV and the costs β. In AC5, jobs are evaluated by an AGV agent. To calculate the earliest delivery time, the AGV agent estimates the waiting times before loading- and mixing ingredients. The priority value $W_{v\varphi}$ for an AGV v doing a job φ, is a measure of the distances between the best- and latest delivery times, and the expected earliest delivery time $z_{v\varphi}$. If the earliest delivery time $z_{v\varphi}$ is later than the best delivery time $b^p(\varphi)$, we add the difference to the priority value because other AGVs will do the job with even more penalties. If the earliest delivery time $z_{v\varphi}$ is earlier than the best delivery time, then we subtract the difference because another AGV may do this job better at a later moment. The same holds for the difference between the earliest delivery time and the latest delivery time. If waiting is required due to minimum- or best rising time constraints, we subtract the value of waiting given by β times the waiting time. We have the following:

$$W_{v\varphi} = \begin{cases} (z_{v\varphi} - b^p(\varphi)) + \alpha(z_{v\varphi} - l^p(\varphi)) - \beta \max(0, b^p(\varphi) - z_{v\varphi}) & \text{for } \varphi \in \mathcal{L}^p \\ \left(z_{v\varphi} - b^d(\varphi)\right) + \alpha\left(z_{v\varphi} - l^d(\varphi)\right) - \beta \max\left(0, b^d(\varphi) - z_{v\varphi}\right) & \text{for } \varphi \in \mathcal{L}^d \end{cases} \tag{4.1}$$

After calculation of all priority values, the job with highest priority will be selected.

4.5.4 Evaluate AGVs

AGV evaluation is used in LC3 to determine the AGV that should handle a specific job. We use the same approach as for job evaluation. As input we need the expected delivery times of all AGVs and the job characteristics of the job. To calculate the earliest delivery time of a delivery job, the storage agent receives the earliest arrival time at the rising area from all AGV agents. To calculate the earliest delivery time of a preparation job, the storage agent receives the earliest arrival time at the first mixer from all AGV agents. The AGV v with highest priority value $W_{v\varphi}$ for a specific job φ, calculated with (4.1), is selected.

4.5.5 Schedule arrivals

In AC3 and LC3, the storage agent maintains a schedule of AGV arrivals. This schedule consists of the following AGV handling records:

> *[AGV, Earliest arrival time, Scheduled starting time, Mixer, Arrival at mixer, Waiting time at mixer, Departure time, Best departure time, Latest departure time]*

Initially these times are random variables. For simplicity of the planning, we decide to use the expectations only. These expected times are updated at three events: (1) whenever an AGV starts loading ingredients at the silos, (2) when an AGV leaves the mixer, and (3) when a new AGV arrival is scheduled. When an AGV leaves the mixer, the corresponding record will be deleted. Whenever the storage agent or AGV agent decides about a scheduled starting time, the storage agent adds a record to its schedule and updates the times of all records. When scheduled starting times of other AGVs are changed, they are only communicated to AGV agents at the moment they schedule a new job (earlier is not necessary).

The AGV arrival schedule has two purposes: earliest arrival scheduling and best arrival scheduling. Earliest arrival scheduling is used by the storage agent in AC3 to calculate the earliest delivery time of a preparation job. Best arrival scheduling is used in architectures AC3 and LC3 to schedule the starting times of AGVs such that the expected penalties are minimized. For both purposes, the storage agent needs to know the earliest arrival time of the AGV, the costs β, and the best-, and latest departure times from the mixers. The only distinction between the two purposes is that in case of earliest arrival scheduling, we set the best departure time equal to the earliest departure time (based on the earliest arrival time and zero waiting times at the silos and mixers).

The storage agent will schedule a new AGV arrival as close as possible to the best departure time, moving some jobs earlier (without violating the earliest arrival time restrictions) and moving other jobs forward. Therefore, the storage agent evaluates the following situations: (1) for each insertion position after the first job, the new arrival is scheduled directly after the previous arrival (possibly moving further jobs forward); (2) for each insertion position before the last arrival, the new arrival is scheduled as close as possible before the next job (moving earlier jobs backwards and possibly moving further jobs forward if preceding arrivals cannot be moved further backwards); and (3) the delivery time of the new job is scheduled as close as possible to the best delivery time while moving the other arrivals.

If the current AGV arrival schedule contains n arrivals, then we have $2n+1$ possible insertion positions of the new AGV arrival. The alternative schedule with lowest costs will become the temporal schedule. In case of earliest arrival scheduling, the temporal schedule is only used to provide the earliest departure times for different jobs. In case of best arrival scheduling, the storage agent replaces the current schedule with the temporal schedule derived for the most suitable job or AGV.

4.5.6 Schedule jobs

In AC3, an AGV has only one job at a time. The storage agent determines the best starting time for this job (Section 4.5.2), calculates the loading- and mixing times in case of a preparation job, and informs the AGV where to be at what time.

Also in AC5, an AGV has one job at a time. This time the AGV determines the best starting time for a job (Section 4.5.2). Because loading- and mixing times are not communicated with the storage agent, the AGV agents estimate the expected waiting times at the first silo and the mixer. The expected waiting time is subtracted from the best starting time.

In LC3, an AGV schedule may contain multiple jobs. Each time a new job arrives, the AGV adds this job to the end of its schedule and determines the best starting time for this job (Section 4.5.2). The waiting times at the silos and mixers are calculated by the storage agent who maintains a schedule of AGV arrivals.

Also in LC5, an AGV schedule may contain multiple jobs. Again, AGVs may use a scheduling method, denoted by *append* scheduling, where new jobs are always added to the end of the schedule. However, this time it has more freedom to schedule its own jobs because they are not communicated with the storage agent. Therefore, we also consider an *insertion* scheduling method where a new job can be inserted at any position in the current schedule without altering the order of other jobs (like we did in the previous section with the AGV arrival schedule). For a given order of jobs, AGVs calculate the best starting times (Section 4.5.2). Because loading- and mixing times are not communicated with the storage agent, AGVs estimate the waiting times at the first silo and the mixer and subtract this time from the best starting time. AGVs may update their schedule at the following moments: arrival at some destination (line, silo, mixer, rising area), finishing an action (pickup barrel, drop barrel, loading, mixing), during bid calculation, and after receiving a grant.

4.5.7 Price jobs

In LC5, AGVs have to price jobs and provide this price to the line agent. The price for an AGV is given by the marginal costs of appending or inserting a new job in its current schedule. Depending on the scheduling method, the AGV agent can schedule the new job at different positions in the current schedule. We indicate the current schedule of vehicle v by Ψ_v and we write $\Psi^n_{v\varphi}$ for schedule alternative n, where the new job φ is inserted after job n ($1 \leq n \leq |\Psi_v|$). For each insertion position we also have to schedule the optimal starting times of all jobs. For example, suppose the new job is added directly after delivery of the last job in the current schedule and the new job is delivered after its due time, then we might remove unnecessary waiting times for the previous jobs.

To determine the optimal insertion position for a new job and the optimal starting times for all jobs, we solve a simple linear program for each alternative schedule. Below, we describe how this can be done.

Formally, we define the schedule Ψ_v of AGV v by an ordered list of 3-tuples $\Psi_{vm} = (\varphi_m, \omega_m, \rho_m)$ where: φ_m refers to the m^{th} job in schedule Ψ_v; ω_m to the scheduled pickup time of this job; and ρ_m to the scheduled delivery time. We write $\tau^e_{d(\varphi_m)o(\varphi_{m+1})}$ for an empty move from the destination of job φ_m to the origin of the next job. The value of a schedule is given by the deviations from the best rising times, plus α times the tardiness, plus β times the total time:

$$C(\Psi_v) = \beta\left(\rho_{|\Psi_v|} - \theta\right) + \sum_{m=1}^{|\Psi_v|} \begin{cases} |\rho_m - b^p(\varphi_m)| + \alpha\left(\rho_m - l^p(\varphi_m)\right)^+ & \text{for } \varphi_m \in \mathcal{L}^p \\ \left|\rho_m - b^d(\varphi_m)\right| + \alpha\left(\rho_m - l^d(\varphi_m)\right)^+ & \text{for } \varphi_m \in \mathcal{L}^d \end{cases} \tag{4.2}$$

Here we introduce the symbol θ to indicate the current time. The scheduled delivery time is given by $\rho_m = \omega_m + h(\varphi_m)$, where $h(\varphi_m)$ is the handling time of job φ_m. In case of a preparation job, this handling time is given by the expected time between picking up a barrel at the storage area and dropping the barrel at the rising area, including expected waiting times at the silos and the mixer. In case of a delivery job, the handling time is given by the time between picking up a barrel at the rising area and dropping it at the production line. The pickup times are scheduled such that they minimize the total costs of a schedule:

$$\begin{aligned} &\min_{\omega_m, m=2..|\Psi_v|} C(\Psi_v) \qquad (4.3) \\ &s.t. \\ &\omega_m \geq e(\varphi_m) \qquad \text{for } \varphi_m \in \mathcal{L}^d,\ m \geq 2 \\ &\omega_m \geq \omega_{m-1} + h\left(\varphi_{m-1}\right) + \tau^e_{d\left(\varphi_{m-1}\right)o(\varphi_m)} \qquad \text{for } m \geq 2 \end{aligned}$$

Here we assume that the scheduled times of the first job (which may be in execution) can not be changed. The bid price of a vehicle v for job φ is now given by the difference in costs between the cheapest alternative schedule and the current schedule:

$$b(v, \varphi) = \min_n C\left(\Psi^n_{v\varphi}\right) - C(\Psi_v) \tag{4.4}$$

4.5.8 Parameter estimation

In order to perform their tasks, AGV agents have to estimate some parameters. In all architectures they estimate the costs β per unit time. In AC5 and LC5, they also estimate the waiting times before loading and mixing ingredients.

To estimate the variables, we use an exponential smoothing procedure (Silver et al., 1998) where a learning rate γ is introduced as a weighting factor that determines the extent to which the current observation is to influence an expected value of an internal parameter. The meaning of the learning rate in this procedure is that when γ is close to one, the new forecast will be based almost exclusively on the last observation. Conversely, when γ is close to zero, the new forecast will be similar to the previous one.

The waiting times are updated after each visit at the first silo or mixer. The value of β is calculated by the AGV agent, based on the average penalties paid per time unit. The logic behind this is that if we wait an extra time unit, this AGV will be available one time unit less, which possibly results in one time unit of extra penalties. We use the exponential smoothing procedure to incorporate fluctuations in the average penalties. To avoid unstable behavior we smooth the penalties per period instead of penalties per job. An extension for learning β is discussed in Section 4.7.

4.6 Simulation

The goal of this simulation study is twofold: (1) to find out which agent architecture can be used best at the bakery and (2) to demonstrate the impact of design choices on the system performance under different parameter settings. Hence, this simulation acts as a decision support tool for the management of Merba to select the best architecture.

To design a valid simulation model we follow the approach of (Law and Kelton, 2000). Basically this consists of (1) collection of high-quality input data, (2) regular interaction with the managers, (3) keeping record of the assumptions and discussing them with the management, and (4) validation of the output. Comparison of the simulation model with the existing situation is not a valid approach here because there are large differences between the current situation and the proposed automated dough production process and there is a lack of performance data of the current manual processes. As an alternative we compare the outcomes of our simulation with the expectations from the management of Merba. These expectations are based on spreadsheet calculations using the same input data.

In this section we subsequently describe our simulation settings, the experimental factors, and the results.

4.6.1 Simulation settings

The bakery produces over 100 dough types. For ease of presentation, we aggregated these dough types into one fictive dough type per production line based

on historical data of all dough requests. These virtual dough characteristics are given in Table 4.1. Here TBJ is the average time between subsequent job arrivals. All times are given in minutes.

Line (p)	**TBJ**	**Look-ahead** $(l^d - a)$	**Min rising time** ($r_{\min}$)	**Best rising time** (r_{best})	**Mixer** (m)
L1	30	50	15	20	M1
L2	60	50	15	20	M1
L3	15	70	20	30	M2
L4	30	70	30	52	M3
L5	30	70	30	52	M4
L6	30	70	30	52	M5
L7	15	70	30	52	M5
L8	30	70	30	45	M6

Table 4.1: Dough request characteristics

Production runs 5 days per week, 24 hours per day. Every week, production starts Monday morning at 4:00 hour. The last batch is released to the dough preparation process on Saturday morning at 4:00. Because the system starts and ends empty each week, we have a terminating simulation. We consider one week as a replication for our simulation experiments. We assume that the release of dough requests follows a Poisson process with mean time between jobs per production line as given in Table 4.1. These figures have been derived from historical data on peak days of the bakery.

All AGVs have a constant speed of 1 m/s. For simplicity, we assume that AGVs always travel in a straight line (shortest distance) from one object to another. We add half a minute to all movements to incorporate the time it takes for an AGV to turn. The time to pickup or drop a barrel is 30 seconds and the loading time for ingredients is 2 minutes per silo. The time for mixing is 11.9 minutes at mixer 1, 11.6 minutes at mixer 2, and 5.3 minutes at the other mixers. The distances between all objects can be calculated from Figure 4.1.

For all experiments, we use 7 AGVs, a penalty factor $\alpha = 10$, and a smoothing factor $\gamma = 0.05$. The number of AGVs is chosen such that all dough requests can be handled (not necessarily in time). In our simulation experiments we have seen that the choice for the penalty- and smoothing factor does not affect the relative performance of the alternative architectures.

As overall performance measures we use (1) the penalty costs for job tardiness and deviation between actual and best rising time, (2) the number of communication messages, and (3) the computation time. The number of communication messages provides an indication of the network load. The computation time is measured per job assignment, taking into account parallel computation. We implemented the agent architectures in the object oriented simulation package eM-Plant 7.5 and performed experiments on a Pentium IV

processor at 3.4GHz. All performance measures are calculated as weekly averages. We choose the number of replications (weeks) needed in our simulation experiments such that a 95% confidence interval for the total costs per work week shows a relative error of at most 5%. We found that 10 replications are sufficient for all scenarios.

4.6.2 Experimental factors

The experimental factors can be found in Table 4.2. We evaluate the 4 different agent architectures, and for architecture LC5 we consider the two scheduling methods. The factor stochasticity describes the uniform deviation around the mean handling- and travel times. So a deviation of 20 will result in handling- and travel times between 0.9 and 1.1 of the normal value. We include this factor to examine the impact of uncertainty because of possible congestion effects (which architecture and planning method is most robust?). Next, we consider three fractions that describe the deviation from the standard settings (Section 4.6.1). These factors will be examined one at a time.

Factor	**Values**
Architecture	AC3, AC5, LC3, LC5
Scheduling in LC5	Append (LC5a), Insert (LC5i)
Stochasticity (%)	0, 8, 16, 24, 32, 40, 48
Fraction TBJ	0.90, 0.95, 1.00, 1.05, 1.10, 1.15, 1.20
Fraction handling times	1.00, 1.08, 1.16, 1.24, 1.32, 1.40, 1.48
Fraction look-ahead	0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4

Table 4.2: Experimental factors

The fraction TBJ provides the fraction of the mean time between subsequent job arrivals compared to the default values as given in Table 4.1. The fraction handling times describe the increase in handling time for silo 1 compared to the default value from Section 4.6.1. The handling times of the other two silos is decreased by half of this amount such that the total handling time at the storage department will be the same. In our case, a value of 1.4 means a handling time of 2 minutes and 48 seconds at silo 1 and 1 minute and 36 seconds at the other silos. We use this factor to investigate the effects of longer queues at the storage department. The fraction look-ahead is a multiplication factor for the look-ahead values from Table 4.1.

4.6.3 Results

In the first 4 experiments, we examine the performance of the different architectures in terms of penalties on tardiness and deviation from the best rising times.

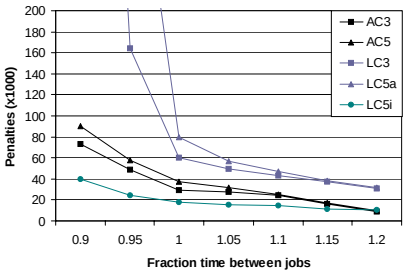

Figure 4.7: Impact of the time between jobs on the penalties

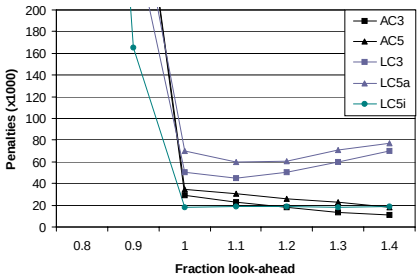

Figure 4.8: Impact of the amplitude in time between jobs on the penalties

In the *first experiment* we vary the time between jobs (Figure 4.7). We see that architectures LC3 and LC5a, where new jobs are added to the end of AGV schedules, are less robust against increasing number of jobs. Architecture LC5i performs best in most situations. However, with decreasing number of jobs, the AGV centric architectures may become in favor. In the *second experiment*, we vary the look-ahead of jobs (Figure 4.8). We see a similar behavior in which the AGV centric architectures are better with increasing look-ahead. The reason for this is that increasing look-ahead leads to longer schedules which may result in less flexibility. This is especially true in case of append scheduling, where also rush jobs have to be added to the end of the schedule. With decreasing look-ahead, the time becomes too short for AGVs to deliver the jobs on time.

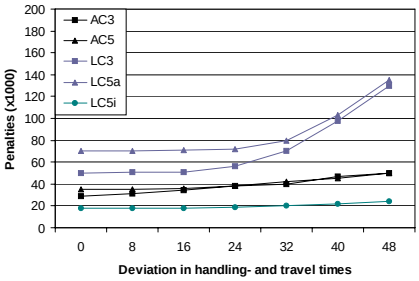

Figure 4.9: Impact of the deviation in handling- and travel times on the penalties

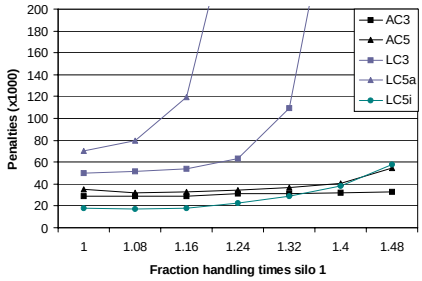

Figure 4.10: Impact of the fraction handing times for silo 1 on the penalties

In a *third experiment*, we investigate the effect of uncertainty in the handling- and travel times (Figure 4.9). As expected, penalties increase with increasing uncertainly for all architectures. We see that with increasing uncertainty, scheduling the AGV arrivals becomes less useful. In a *fourth experiment*, we investigate the effects of congestion at silo 1 (Figure 4.10). We see the performance of architecture AC3 remains the same with increasing congestion, while

the costs of all other architectures increase. We also see that with increasing congestion, it becomes more useful to scheduling the loading- and mixing times (AC3 and LC3).

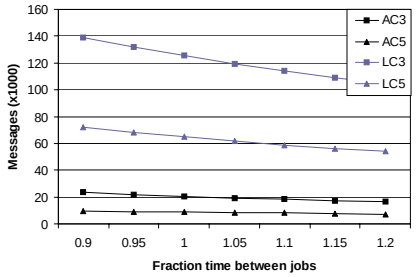

Figure 4.11: Impact of the fraction time between jobs on the number of messages

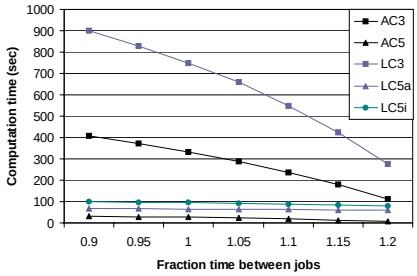

Figure 4.12: Impact of the fraction time between jobs on the computation time

Next, we consider the number of communication messages and computation time as performance indicators. In the *fifth experiment*, we vary the time between jobs and investigate the number of communication messages of the different architectures (Figure 4.11). Note that the number of communication messages is the same for both scheduling methods in architecture LC5. We see (1) the number of messages decreases with decreasing number of jobs; (2) communication with the storage agent (LC3 and AC3) requires much more communication because loading- and mixing times have to be communicated for every job with every schedule update; and (3) the line centric architectures require the most amount of communication. In a *sixth and final experiment*, we show the impact of a varying time between jobs on the computation time (seconds) in Figure 4.12. The results are obvious: (1) scheduling loading- and mixing times, increases the computation time and (2) the computation time for all architectures decreases with increasing time between jobs. Note that computation time is measured based on a parallel implementation. For architectures AC3 and AC5 this does not make a difference because we have sequential decision processes. However, in architectures LC5a and LC5i, most computation is done in parallel by all AGVs when they try to schedule a new job received by a line agent.

The data used for the experiments described above is based on real factory data in which we changed one factor at a time. We did not use a full factorial design because this would be beyond the scope of this chapter; the case study only serves as an illustration to support our research objectives (see Section 4.2). However, we also examined the effect of several other experimental factors. We found that for each architecture, there exists at least one instance in which it performs best. This certainly illustrates that qualitative arguments and modeling guidelines in current MAS methodologies are insufficient to select a

single "best" architecture for MAS. With respect to the simulated factory, we conclude that the line centric architecture LC5i where AGV agents schedule multiple jobs in advance, performs best in almost all situations, although it requires a substantial amount of communication and computation time. With increasing congestion at the silos, the architecture LC3 without the storage agent (that schedules the loading times of AGVs) will become in favor. The AGV centric architectures come in favor in case of (1) decreasing number of jobs or (2) increasing look-ahead or (3) increasing congestion at the silos. Obviously, under the latter conditions, an insertion scheduling heuristic of the AGV agents would be less beneficial and/or becomes more difficult.

4.7 Extensions

In Section 4.4.1 we mentioned the FCFS selection strategy at silos and mixers. An improvement can easily be made by using an auction protocol to select an AGV from the queue. Each AGV in a queue has to submit a bid consisting of the increase in penalties if it has to wait one turn. One turn here is the expected time until the next AGV may leave the queue. The AGV with highest bid will be handled first.

In Section 4.5.8 we calculated β as being the average penalties paid per time unit. An alternative is to calculate β as being the average revenue per time unit. In order to generate revenues we make use of a reverse Vickrey auction. In this auction, the lowest bidder wins the auction and receives the price of the second lowest bidder. This auction type has received particular attention within the multi-agent community because it possesses a dominant strategy to bid one's true valuation (Vickrey, 1961). This pricing strategy is especially suitable for architecture LC5 where AGV agents calculate a price and the AGV with lowest price is selected. The logic of calculating β, as being the average revenue per time unit, is that waiting one extra time unit will decrease the period after this job with one time unit. So expected revenues decrease with the expected revenue of one time unit. But revenues also provide valuable information about future penalties. Let us consider an AGV that just won an auction. Its revenue for the new job resembles the increase in expected penalties if not he, but the second best bidder had won this auction. Because AGVs plan jobs in the future, this information provides insight into future network pressure. In other words, the last observation of revenues provides a better estimate of future penalties than the last observation of penalties already paid.

4.8 Conclusions

During our field project at Merba bakeries, we found that current MAS development methodologies do not provide sufficient support to select the pre-

ferred design for implementation. The scientific contribution of this chapter is to provide insight into the MAS design process and to improve current MAS development methodologies to offer enhanced support in cases where multiple alternative decisions and communication scenarios exist. Scenarios vary in roles and responsibilities assigned to the agents, the level of intelligence of the agents, and the interaction protocols. Thus, our results are not restricted to the agent-based control of AGVs. In a wide range of MAS application areas where different actors and roles collaborate, such a method support will be beneficial. Also the proposed multi-agent system itself provides insights that can be generalized to other situations. Especially regarding the way agents balance different delivery criteria in the scheduling of jobs.

To illustrate the design process, we considered a simplified part of the dough production process at Merba bakeries. By using a stepwise approach - built upon existing MAS development methodologies - we already derived eight alternative designs for this part only. By using qualitative arguments, we were able to reduce this to four alternative designs. In order to select the preferred design for implementation we use multi-agent discrete event simulation.

This simulation gave us insight into the effect of our MAS design choices on the system performance in terms of delivery punctuality, product quality, robustness, amount of communication, and computation time of the different agents. It is shown by our simulation study, that qualitative arguments are not sufficient because each alternative design has its own advantages. A practical way of dealing with these results is to use a combination of different control mechanisms. Based on the system status, AGVs might use a different scheduling technique or the architecture itself may even be changed dynamically. Suppose, for example, that we are using architecture LC5i. Whenever we observe increasing congestion, we might temporarily switch to an AGV centric architecture. This adaptability of the system design is part of our future research. In addition, we want to investigate the impact of MAS design choices on a broad class of performance indicators such as flexibility, scalability, adaptability and extensibility.

Chapter 5

Carriers: opportunity valuation policies

In this chapter[1] we consider a real-time, dynamic full truckload pickup and delivery problem with time-windows where jobs should be assigned to one of a group of competing transportation companies. Our approach decomposes the problem into a multi-agent structure where vehicle agents are responsible for the routing and scheduling decisions and where the assignment of jobs to vehicles is done by using a second-price auction. We propose a pricing and scheduling strategy for vehicle agents based on dynamic programming where not only the direct costs of a job insertion are taken into account, but also its impact on future opportunities. Simulation is used to evaluate the benefit of pricing opportunities compared to simple pricing strategies in various market settings. Numerical results show that the proposed approach provides high quality solutions, in terms of profits, capacity utilization, and delivery reliability.

5.1 Introduction

Most techniques and models used in transportation planning, scheduling, and routing use a centralized solution approach with static input data. Although such techniques have successfully been implemented, they are less suitable in a dynamic environment and in an environment with multiple independent stakeholders.

[1] This chapter is based on the working paper (Mes, Van der Heijden and Schuur 2006); presented at Odysseus 2006, the Third International Workshop on Freight Transportation and Logistics, Spain; submitted to OR Spectrum.

Dynamic environments are characterized by frequent and unpredictable modifications in the relevant planning information and the ability (or even necessity) to update the planning based on this additional information. For example, carriers may know only a fraction of the shipments to be served when the first plan is constructed whereas additional (rush) transportation jobs arrive during execution of the original plan. Also, additional information on processing times (travel time updates in case of congestion) and equipment failures may arrive during execution. A proper transport planning approach should be able to construct initial plans taking into account all such uncertainties and to update the plans reacting on real-time information updates.

Particularly for real-time planning updates a central approach is not suitable, because global reoptimization may lead to a completely different plan in response to a relatively minor information update. A decentralized approach, where local problems are solved locally as much as possible to limit schedule disruption, has certainly advantages. Besides, it is known that the added value of global optimization versus local planning heuristics decreases in an uncertain dynamic environment, see e.g. (Van der Heijden, Van Harten, Ebben, Saanen, Valentin and Verbraeck, 2002). Finally, a distributed solution is required when multiple independent organizational units (multiple carriers and shippers) are working in an autonomous, self-interested, and not necessarily cooperative way. Then a distributed approach is needed to optimize the network performance (maximize profits) while reckoning with the individual competences, goals, and information access.

In the literature, our transportation problem is known as a dynamic multi-vehicle pickup and delivery problem with time-windows. We consider a variant with full truckloads and stochastic arrivals of jobs. Within the transportation network multiple shippers offer loads for transportation and multiple carriers are competing for these jobs. We propose a multi-agent system where vehicle agents are responsible for the routing and scheduling decisions of their corresponding vehicle. The assignment of jobs to vehicles is done by using an auction. Therefore, a proper pricing mechanism is needed to optimize the system-wide performance, such as the minimization of the total costs, consisting of transportation costs and penalties on tardiness.

In Chapter 3 we presented a basic multi-agent system and compared its performance with two traditional scheduling heuristics. In this chapter we aim to enhance the performance of this agent-based planning system by improving the pricing and scheduling techniques of the vehicle agents. We focus on three key questions. First, how can we use information on historic job patterns and historic auction results to improve bid pricing? Second, how can we use this information to improve planning and scheduling? Third, what is the impact of such additional intelligence on the overall system performance?

The remainder of this chapter is structured as follows. In the next section, we give an overview of related literature and we explain the scientific

contribution of this chapter. Our model is presented in Section 5.3. In Section 5.4 we present our solution method to estimate the value of a schedule using value functions. In Section 5.5 we describe how the parameters for these value functions can be estimated. In Section 5.6 we discuss some extensions of our method. Experimental settings and simulation results are presented in Section 5.7 till 5.9. We end up with conclusions in Section 5.10.

5.2 Literature

Our problem is well known in the area of vehicle routing problems (VRP). More precisely, we consider a pickup and delivery problem with time-windows (PDPTW) which is a generalization of the VRP. Here a number of vehicles have to satisfy transportation requests which are characterized by a pickup location, a delivery location, and time-window restrictions. Such problems appear in a wide variety of distribution systems, e.g. courier services, rescue and repair services, emergency services, taxi cab services, less than truckload transportation, and long-distance haulage.

The VRP and its variants have been studied extensively, for recent surveys we refer to (Desaulniers et al., 2001; Toth and Vigo, 2002; Cordeau et al., 2007). Most work focuses on static and deterministic problems in which all information is known in advance, see for example (Desrosiers et al., 1995; Fischer, 1995). However, in most real-world problems we have to deal with uncertainty, i.e., some of the parameters of the model are initially unknown or known only probabilistically. Due to recent advances in communication and information technology, these problems gain importance, and the potential savings generated by adapting routing decisions to dynamic or stochastic contexts are substantial (Psaraftis, 1995).

In the dynamic vehicle routing problems (DVRP), new transportation jobs arrive dynamically when the vehicles have already started executing their tours. This requires real-time replanning in order to include the new jobs in the vehicle schedules. Routing and scheduling in a dynamic environment has been studied by a number of authors, see for example (Psaraftis, 1988; Gendreau and Potvin, 1998; Yang et al., 2004). The most common approach to handle these problems is to solve a model using the data that are known at a certain point in time, and to re-optimize as soon as new data become available. Because a fast response is required in a real-time environment, a solution is usually achieved by using relatively simple heuristics or by parallel computation methods, see (Ghiani et al., 2003) for an overview of approaches.

Instead of just reacting to problem changes (e.g. the arrival of new customers), it may be beneficial to anticipate such events. Anticipation can take place in various types of decisions, e.g. decisions regarding the acceptance, pricing, and scheduling of jobs. In this chapter we focus on pricing, scheduling,

and repositioning decisions. To support these decisions we use probabilistic knowledge about future job arrivals. Many researches have claimed that this topic should receive more attention (for an overview, see Chapter 1, Section 1.1.1).

In the literature one can find two main approaches to cope with the problem changes caused by new customer arrivals. The two strategies are reoptimization and dispatching (Branke et al., 2005).

The *first approach* to deal with new customer arrivals is to renounce planning to a certain extent, and instead of scheduling all known tasks, only decide on a vehicle's next task. Here decisions that take into account future job arrivals, are where to wait and for how long. We indicate this type of decisions by *operational waiting decisions*.

An early example can be found in (Bertsimas and Van Ryzin, 1991). They consider a dynamic traveling repairman problem (DTRP) where service demands arrive according to a Poisson process at uniformly distributed locations in a Euclidean plane. They show, for the case of a single vehicle and light traffic conditions, that it is optimal to reposition the vehicle to the center of the service region whenever there are no customers left to be serviced. Returning to the center anticipates future customer arrivals by positioning the vehicle so that the expected distance to the next arriving customer is minimized. Similar results are shown in (Bertsimas and Van Ryzin, 1993) for the multiple-vehicle traveling repairman problem and in (Swihart and Papastavrou, 1999) for the single vehicle pickup and delivery problem.

Larsen et al. (2004) consider the dynamic traveling salesman problem. Their goal is to find a minimum-costs tour through a set of dynamic requests with soft time-windows. The issue is to relocate empty vehicles to predefined resting locations in anticipation of future demand. They consider the following online policies: (1) the vehicle goes to the nearest resting location, (2) the vehicle goes to the resting location that belongs to a region with the largest arrival rate, and (3) the vehicle goes to the resting location that belongs to a region with the highest expected number of customers within a certain time period depending on future obligations of this vehicle.

Thomas and White (2004) consider the case of a single vehicle which has to travel from a given origin to a given destination. Within the network there are a few known locations of potential customers that might issue a request while the vehicle is under way. The vehicle anticipates these possible future customer requests both by waiting at a location and by visiting noncustomer locations. They present an optimal policy for route construction by modeling their routing problem as a Markov decision process.

Ichoua et al. (2006) consider a DVRP with soft time-windows for serving customers and a hard time-window for returning at the depot. The problem consists of using probabilistic knowledge about future requests to minimize the

cost of vehicle routes. The cost of a route depends on the lateness penalties and on the total distance traveled. The original algorithm, a parallel tabu search heuristic, is extended by allowing a vehicle to wait at its current position if the probability of new, nearby requests is high.

The common problem in the papers mentioned above, is that each time a vehicle visits a certain location, it has to decide upon its next action. The main distinction with our research is that we schedule multiple jobs in advance. As a consequence, we have to anticipate future job arrivals in the sequencing and timing of jobs currently in the schedule. Another distinction with the work of (Thomas and White, 2004; Ichoua et al., 2006) is that we consider a rolling horizon instead of a single fixed period.

The *second approach* to handle real-time events in DVRP is to reoptimize whenever a new job arrives. The main decision here is to anticipate future job arrivals through scheduling of jobs. We indicate these decisions by *tactical waiting decisions*.

Mitrović-Minić and Laporte (2004) consider the uncapacitated dynamic pickup and delivery problem with time windows. They present a methodology that captures the need to take future events into consideration, even when the events are not stochastically modeled or predicted. They examine the benefits of several tactical waiting strategies on the total detour and number of vehicles, where they distinguish between the impact of decisions on the short-term and the long-term basis. Their work differs from ours because (1) we do not focus on detour but profits (i.e., probability and profitability of new job insertions) and (2) we also consider a variant in which we have to determine the pickup and delivery times of jobs at the announcement time of an auction.

Branke et al. (2005) consider a DVRP where one additional job arrives at a beforehand unknown location when the vehicles are already under way. Each vehicle has a fixed route along different locations. They optimize the waiting times of the vehicles at each location such that the probability that the new job can be inserted is maximized. The main difference with our work is that we determine the optimal pickup time for one job at a time, but evaluate multiple future jobs insertions and their impact on the profitability.

Van Hemert and La Poutré (2004) consider a DVRP where dynamically arriving less-than-truckload (LTL) jobs have to be collected and transported to one central depot within a fixed time-window. They present an evolutionary algorithm that is able to provide routing solutions taking into account the potential of fruitful regions, i.e., regions with high potential for new pickups. They show that - depending on time and capacity constraints - using the concept of fruitful regions, one may increase the number of loads that can be successfully delivered to the central depot. The proposed evolutionary algorithm is extended in a later paper (Bosman and La Poutré, 2006) by combining the evolutionary computation with learning techniques, thereby using information on predicted future loads to support current decision making. The main difference

between these two papers and our approach, is that we consider a rolling horizon full truckload (FTL) problem, where jobs have different delivery locations and latest pickup times.

Another closely related example can be found in (Yang et al., 2004). They consider and compare five on-line strategies for the dynamic pickup and delivery problem with full truckloads. Two of these policies are based on repeated reoptimization of the offline problem, while the others use simple local rules. One of these policies (called OPTUN) utilizes probabilistic knowledge about future job requests. To account for future job requests they introduce the concept of opportunity costs of serving a job. Basically, these opportunity costs are a measure of the change in degree of isolation for a vehicle ending at a new location. By taking into account the opportunity costs, remote locations are penalized and central locations are favored. They conclude that the OPTUN policy outperforms the other four policies. They further conclude that *"future research should concentrate on the search for better policies that utilize some information about future jobs more efficiently"*. In this chapter we aim to contribute to that by developing opportunity cost functions that do not depend on distance measures, but more generally on the change in expected revenues due to a new job insertion. These expected revenues mainly depend on (1) the arrival probabilities of future jobs at different locations and (2) the probability of winning these jobs (which in turn depends on the current vehicle schedules).

Besides the differences mentioned above between (Mitrović-Minić and Laporte, 2004; Van Hemert and La Poutré, 2004; Bosman and La Poutré, 2006; Yang et al., 2004; Branke et al., 2005) and this chapter, there are two additional differences. First, we consider the combination of tactical and operational waiting decisions. Second, we also focus on acceptance decisions through dynamic pricing of jobs. All these decisions are supported by the same opportunity valuation method, which therefore saves computation time.

Another related line of research is on dynamic fleet management problems (DFMP), or more generally the dynamic assignment problem. These problems ask for a dynamic assignment of resources (trucks) to tasks (loads). Truly stochastic models decompose the DFMP with respect to time periods and assess the impact of the current decisions on the future through a recourse or value function. An early example can be found in (Powell et al., 1988), who study a dynamic assignment problem where a fleet of vehicles is assigned to a set of locations with dynamically occurring demands. They show that it is advantageous to take forecasted demands into account when deciding where the vehicles should drive next, compared with a model that only reacts after new demands have arrived. More recent examples can be found in (Carvalho and Powell, 2000; Godfrey and Powell, 2002; Topaloglu and Powell, 2006). We cannot use the DFMP algorithms directly, because (1) jobs have to be accepted early to avoid loosing them to competitors and (2) jobs are scheduled in a distributed manner by the vehicle agents. Also, the price of a job is not given

externally but subject to negotiation. Moreover, the arrival intensity of jobs at a company is not described by an exogenous information process, but can be influenced by better repositioning of vehicles.

A recent development is the applicability of multi-agent systems (MAS) in the field of transportation control. There one aims at the development of robust, distributed market mechanisms. In an early paper Sandholm (1993) applied a bidding protocol, called Contract Net Protocol, to a transportation system, where dispatch centers of different companies cooperate in vehicle routing. Fischer et al. (1996) developed a system for cooperative transportation scheduling and a simulation test bed for multi-agent transport planning, called MARS. In ('t Hoen and La Poutré, 2004) a multi-agent system is presented for real-time vehicle routing problems with consolidation in a multi-company setting, where vehicles have the option to break an agreement in favor of a better deal (cf. Chapter 6). The common approach in MAS is to assign sub-problems to agents that represent resources such as shippers, carriers, and vehicles. A quite different approach can be found in (Shapiro and Powell, 2006) where a general decomposition method is described based on how decisions are actually made. For example, they describe a decomposition into regions of a resource allocation problem for large transportation companies that are regionally divided. Within the research on MAS in the field of transportation management, we focus on the intelligence of agents instead of the MAS architecture as is often subject of research. In our case, this intelligence is mainly concerned with the pricing and scheduling decisions by taking into account the future implications of new job insertions. This certainly distinguishes our approach from other research in agent-based transportation planning where agents are usually short sighted.

An interesting and closely related line of research comes from Figliozzi and co-authors. Figliozzi et al. (2003) present a framework for the study of carriers' strategies in an auction marketplace for dynamic, full truckload vehicle routing with time-windows. They focus on profit allocation rather than on the efficiency of assignment decisions. The impact of different assignment strategies on the travel costs under various demand conditions is studied in (Figliozzi et al., 2004). In (Figliozzi, 2004; Figliozzi et al., 2006) a method to quantify the opportunity costs in sequential transportation auctions is presented. In their paper, two carriers compete for transportation jobs. Each arriving job triggers an auction where carriers compete with each other to win the right of servicing the job. However, winning a job has an effect on the next auction round and, therefore, on its profit for the next job. They introduce opportunity costs to capture the consequences of accepting the current job in the auction on the expected profit of the next job to be auctioned. Therefore, they refer to this method as *one-step* look-ahead opportunity costs. Using simulation, they show that the carrier that accounts for opportunity costs can significantly improve its profitability compared to the naive carrier. The main differences of our work compared to the recent work of Figliozzi (Figliozzi, 2004; Figliozzi

et al., 2006) are: (1) we estimate the opportunity costs for individual vehicles in a decentralized planning system; (2) our opportunity costs are defined over *multiple* jobs to be acquired within a certain time horizon; (3) our opportunity costs are used to price a new load, but also to determine the optimal pickup and delivery times of jobs, and to support pro-active move decisions (moves to attractive locations if vehicles are idle); and (4) we cover estimation of the probability distribution of the lowest bid by the competitors based on incomplete information instead of assuming a known probability distribution.

5.3 Model and notation

We consider a pickup and delivery problem with full truckloads, time-windows, deterministic travel times, and stochastic arrival of jobs. To present our model we subsequently discuss the network and cost structure (5.3.1), the job characteristics (5.3.2), the market mechanism (5.3.3), and vehicle scheduling and bid price calculation (5.3.4).

5.3.1 Network description

Our transportation network is a directed graph ($\mathcal{N}$,$\mathcal{A}$), i.e., it consists of a set of nodes $\mathcal{N}$ and a set of arcs $\mathcal{A}$ connecting these nodes. In this network multiple carriers and shippers operate. The system dynamics is driven by the incoming jobs from shippers that are not known beforehand. These jobs consist of unit loads (full truckloads) which have to be transported between nodes in the network. A set of vehicles $\mathcal{V}$, not necessarily identical, belonging to the different carriers is available to transport these loads. The time to transport a load from node i to node j is given by τ^f_{ij}. This includes travel time, and the handling time to load- and unload the job. The time to drive empty from node i to node j is given by τ^e_{ij}. Both times are deterministic and vehicle independent. We use the shorthand notation τ_{ikl} for $\tau^e_{ik} + \tau^f_{kl}$.

Objective of the shippers is to minimize their costs. Objective of the carriers is to maximize their profits. We consider two cost functions, namely the travel costs $c^r(t)$ as function of the travel time t and the penalty costs $c^p(t)$ in case of tardiness ($t > 0$) with respect to the time-window restrictions (see Section 5.3.2). To cover the transportation costs, the carriers will charge the shippers for their transportation services. The total costs for a shipper are given by the sum of all prices paid to the carriers for transporting their loads. The profits for the carriers are given by their income from all transportation jobs minus the transportation costs and costs for tardiness.

Matching of jobs with open vehicle capacity is done using an auction procedure which leads to a contract between a carrier and a shipper. Execution of the resulting contracts requires scheduling of the vehicles while taking the

contract terms into account. Vehicle scheduling has its impact on the future availability of vehicle capacity and on the system dynamics and hence also on the profitability of the companies. A general impression of this situation is given in Chapter 2, Figure 2.1.

As for the dynamics and control, we assume that the system is stable in the long run, so that all jobs can be handled (not necessarily on time). As for communication, we assume that at any moment in time communication between shippers, vehicles, and carriers is possible.

5.3.2 Job characteristics

Jobs to transport unit loads (full truckloads) between nodes in the network arrive one-by-one according to a stationary Poisson process with arrival intensity λ_{kl} of jobs from node k to node l. We define a job φ by an announcement time $a(\varphi)$, an origin node $o(\varphi)$, a destination node $d(\varphi)$, and a latest pickup time $e(\varphi)$ of the load at its origin. The announcement time is the time at which the transportation request arrives at the shipper. We introduce a time-window length $z(\varphi) = e(\varphi) - a(\varphi)$ within which transportation should be started.

We assume that the carrier agrees with the latest pickup time as soft restriction with penalty costs $c^p(t)$. This penalty function is a positive non-decreasing function of the time t and may differ per job. To keep the discussion simple, we do not include an earliest delivery time in the job characteristics, which can be incorporated rather easily. We further assume that a job in process cannot be interrupted (no preemption), i.e., a vehicle may not temporarily drop a load in order to handle a more profitable load.

We consider two types of contracts, namely contracts with fixed and with flexible pickup times. In case of *fixed* pickup times, the exact pickup time is agreed upon in the contract and carriers are not allowed to change this time later on. As a consequence, penalty costs are only made when the agreed pickup time is later than the latest pickup time. In case of *flexible* pickup times, the pickup times may be modified by the carriers during schedule execution, even if this results in tardiness. Moreover, a carrier only has to decide about the next job of a vehicle whenever this vehicle becomes available.

For simplicity of reasoning, we first consider fixed contracts. In Section 5.4.4 we describe how the results can be extended to flexible contracts.

5.3.3 Market mechanism

Jobs are offered in sequential transportation procurement auctions. In these auctions, shippers typically put out a request for quotes from a set of carriers (Song and Regan, 2002). This process is similar to a simple sealed-bid auction in which each bidder submits a sealed bid for a single job. Without loss of

generality, we choose here for a reverse second-price sealed-bid auction, also known as the Vickrey auction. In this auction mechanism, every bidder submits a single sealed bid and the bidder with the highest bid receives the object at the price of the second highest bid.

The Vickrey auction has been widely used for multi-agent systems because (1) it requires a single bidding round and (2) it forces bidders - under some mild conditions (see Vickrey, 1961) - to bid their true valuation of the object, thereby avoiding many bidding problems (e.g. speculation on profit margins). However, this property no longer holds in sequential auctions where the valuation of bundles of items, acquired in separate auctions, differs from the sum of the valuations of individual items. This certainly is the case in sequential transportation procurement auctions, where bundles form efficient routes consisting of multiple pickup and/or delivery locations. This issue is illustrated in ('t Hoen and La Poutré, 2006). They show, through experiments and game theoretical analysis, that one can benefit from deviation from the true valuation, i.e., the immediate valuation of this single item. In this chapter we define the true valuation as the estimated value of a new job in the long run. Note that bidding this true valuation is not as simple as it seems as we aim to take into account opportunity costs in this chapter. As it will appear, this is not a trivial matter.

Besides the limitations of the Vickrey auction with respect to sequential auctions, there are also limitations regarding strategic behavior (e.g. bidders who deliberatively inflict losses to rivals). For more details on these limitations we refer to (Sandholm, 2000; Brandt and Weiß, 2002). In this chapter we ignore these limitations by assuming that individual bidders try to maximize their profits without caring for the profits made by the other agents. As a consequence, we assume that bidders bid their true valuation as described above.

We implement the market mechanism as follows. When a job φ arrives at some shipper, it starts an auction by sending an announcement to all carriers. All carriers respond with a bid confirming agreement with the contract terms. The shipper evaluates all bids and the winning carrier will receive a grant message while the others receive reject messages. We assume that the final price paid by the shipper is published. Of course, generalization towards a closed auction, where bidders only receive information about whether they win or lose an auction, is possible. In fact, in a closed auction bidders have censored data for the distribution of the minimum price. In principle, we can handle this using e.g. Maximum Likelihood estimation.

In this chapter we focus on the transportation side: the carriers and their vehicles. When a carrier receives a job announcement it has to calculate the expected costs for doing this job. However, complete assessment of the feasibility and the expected profit of a job in real-time is hard. In order to respond fast to these auctions, we choose for a distributed structure where every vehi-

cle calculates a bid for this job and sends it to its carrier. Therefore, vehicles are modeled as intelligent agents that determine their bidding and scheduling strategy based on historic data (experience), learning, and on expectations of future consequences of current actions. They have the opportunity of learning about the environment (travel times, job characteristics) and about other players (winning prices) with each auction. Each vehicle agent is responsible for the planning and scheduling decisions for its corresponding vehicle.

After the carrier receives all bids from its vehicles, it selects the bid with the lowest costs and sends it to the auction. When a carrier is awarded for this job, it will assign this job to the vehicle whose bid was submitted to the auction. We assume that there is no exchange of jobs between vehicles.

5.3.4 Vehicle scheduling and bid calculation

In this section we explain how vehicle agents price and schedule new jobs. Because we focus on the strategies of individual vehicle agents, we omit any subscripts indicating specific vehicles or companies.

Schedule definition

At each point in time, a vehicle has a job schedule, i.e., a list of jobs with scheduled starting times. The destination of the last job in the schedule will be referred to as *schedule destination* and the time until the expected arrival time at the schedule destination is referred to as *length of a schedule*.

Formally, we define a schedule Ψ by an ordered list of 2-tuples $\Psi_n = (\varphi_n, \omega_n)$ where φ_n refers to the n^{th} job in the schedule and ω_n to the scheduled pickup time of this job. The loaded move for job φ_n goes from $o(\varphi_n)$ to $d(\varphi_n)$, starting at time ω_n and being delivered at time $\omega_n + \tau^f_{o(\varphi_n)d(\varphi_n)}$. All times mentioned here and in the sequel are relative to the current time θ. In the remainder we denote the delivery time of job n by ρ_n and the number of jobs in a schedule by N.

A *gap* appears between two consecutive jobs whenever the pickup time of the second job is later than the delivery time of the first job. Such a gap may be used for a suitable job to be inserted later on. The actual insertion of this new job can take place before arrival at the destination of the first job. Otherwise, as we will show later on, the vehicle has an option (1) to drive immediately to the origin of the second job, (2) to wait as long as possible at the current location in anticipation of a new job insertion, and (3) to move pro-actively to another, probably more profitable, location in anticipation of a new job insertion. Obviously, these options are restricted by the pickup time of the second job.

We introduce the phrase *end-gap* for the difference between the planning

period T (which we choose to be much larger than the length of a schedule) and the length of a schedule. The gaps and the end-gap are important to value a schedule, because future jobs can only be inserted in these periods. An illustration of a possible schedule can be found in Figure 5.1. The node at the beginning of a gap will be referred to as *start-node*, and the node at the end of a gap by *end-node*.

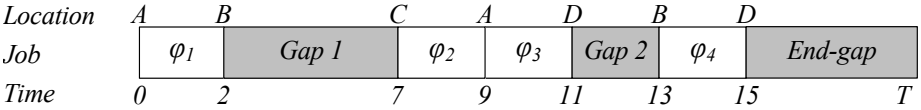

Figure 5.1: Example of a vehicle schedule

Scheduling and waiting decisions

The goal of a vehicle agent is to maximize its profits. Estimation of the expected profit of a new job is not straightforward because the destination of this job may have an effect on (1) the expected empty travel distance for the next job and (2) the likelihood of winning a next job. The same holds for the origin which has an effect on the insertion of future jobs before this job. Therefore, we focus on maximizing the profits within a certain planning horizon T. We denote the expected profits during period T given a schedule Ψ, by $V(\Psi, T)$. These profits consist of the payments for all jobs in schedule Ψ, minus the costs for these jobs, plus the expected profits for future jobs to be inserted in schedule Ψ (see Section 5.4 for a formal expression).

When an auction for a new job φ is started, each vehicle agent creates a temporal schedule $\Psi \cup \varphi$ combining its current schedule with the new job in such a way that the value $V(\Psi \cup \varphi, T)$ is maximized. Note that at this point we do not know the payment for the new job which is therefore set to zero.

In this chapter we assume that vehicle agents contemplate the insertion of a new job at any position in the current schedule without altering the order of execution for the other jobs. Since the fixed contract agreements do not permit moving already assigned jobs, a new job can only be inserted in gaps in such a way that the agreed pickup times for other jobs are not violated. If the gap is larger than the time needed to insert the job, the vehicle agent has flexibility to select the best pick-up time. One option is to schedule the new job as early as possible, but in some situations it may be advantageous to postpone. Let us illustrate that using an example referring to Figure 5.1.

Example 5.1. *Suppose a job from A to C is to be inserted in the schedule. Let* $\tau^f_{AC} = 2$ *and* $\tau^e_{BA} = 1$. *Consider Gap 1. Clearly, the new job can start at any time between* $t = 3$ *and* $t = 5$. *If we schedule the job as early as possible*

(start at $t = 3$), the vehicle will arrive in C at $t = 5$. Probably, the vehicle has to wait there until the next job can be picked up at $t = 7$. After all, the probability that another job can be inserted in this time interval is low, because an empty ride is needed anyway. If on the other hand the start of the job is scheduled at $t = 5$, the vehicle will arrive in C at $t = 7$ so that the next job can be started immediately. Then the vehicle has three time units to drive from B to A. Then it is very well possible that another job can be inserted in this interval, either from B to A or between other locations that require only little additional empty driving time.

To construct a temporal schedule, a vehicle agent evaluates all possible insertions of the new job φ in the current schedule Ψ. Because the first job of the schedule is always in execution, the number of insertion positions equals N. For a given position n in the schedule (i.e., after the n^{th} job in the current schedule), the vehicle agent has to select the most profitable pickup time $\omega(n)$ for the new job. This pickup time is bounded by the delivery time of job φ_n plus the empty travel time towards the origin of the new job, and the pickup time of job φ_{n+1} minus the loaded travel time for job φ, and the empty travel time from the destination of this job to the origin of job φ_{n+1}. We indicate each alternative schedule by $\Psi_{\varphi,n,\omega(n)}$. The temporal schedule is given by the alternative schedule with the highest profit:

$$V(\Psi \cup \varphi) = \max_{n} \left\{ \max_{\omega(n)} V\left(\Psi_{\varphi,n,\omega(n)}\right) \right\} \tag{5.1}$$

where $n = 1..N$, $\omega(n) \geq \rho_n + \tau^e_{d(\varphi_n)o(\varphi)}$ for all n, and $\omega(n) + \tau^f_{o(\varphi)d(\varphi)} + \tau^e_{d(\varphi)o(\varphi_{n+1})} \leq \omega_{n+1}$ for $n < N$. Note that the numbering n here is defined over the current schedule Ψ. The numbering in the temporal schedule $\Psi \cup \varphi$ is changed due to the insertion of the new job, i.e., if the new job is inserted after job n in the current schedule Ψ, then ω_m for $m \geq n$ in the current schedule becomes ω_{m+1} in the temporal schedule and the pickup time $\omega(n)$ of the new job becomes ω_n in the temporal schedule. We solve the maximization in (5.1) by discretization of the pickup time $\omega(n)$.

A vehicle agent updates its schedule when (1) an auction for a new job is won and (2) the first loaded move in a schedule has been completed. In the first case, the vehicle agent replaces its current schedule with the temporal schedule that had been constructed for the auction. In the second case, the vehicle agent faces an *operational waiting decision*. Suppose the vehicle is currently located in node i and has to be at node j at time t, then it has three options. First, it can drive immediately to node j and wait over there. Second, it can wait at node i until it wins another job or at last drives to node j at time $t - \tau^e_{ij}$. Third, it can drive pro-actively to another node k in anticipation of a future job and if it does not receive such a job, it can move at time $t - \tau^e_{kj}$ to node j. Operational waiting decisions are presented in Section 5.4.3.

Bid calculation

As mentioned in Section 5.3.3, we focus on bidding true valuations. The true valuation of a new job insertion is given by the decrease in expected profits during period T due to insertion of the new job, assuming we did not receive a payment for this new job. Bidding less is not optimal since then winning this job will result in loss of profit. Bidding more reduces the likelihood of winning the shipment while the expected final price remains the same (see Figliozzi, 2004; Vickrey, 1961). Therefore, the bid price $b(\varphi, \Psi)$ for a new job φ in the current schedule Ψ is given by the value of the current schedule minus the value of the temporal schedule with the new job:

$$b(\varphi, \Psi) = V(\Psi, T) - V(\Psi \cup \varphi, T) \tag{5.2}$$

As shown in the next section, this bid price not only includes the increase in transportation and penalty costs, but also the change in value of future job opportunities.

Roadmap for obtaining the value functions

To summarize, a vehicle faces three types of decisions: (1) bid price calculation, (2) scheduling of jobs, and (3) operational waiting decisions. These decisions have in common that they have to be made real-time. Bid price calculation should be done upon announcement of a new job. Scheduling of jobs should also be done upon each announcement, but in case of flexible contracts also upon delivery of a job. The operational waiting decision has to be made after each delivery.

All vehicle decisions are supported by so-called value functions that are introduced in the next sections. These value functions not necessarily have to be calculated real-time, and not necessarily separately by each vehicle. For example, in our simulation experiments (Section 5.7), we calculate the value functions either once (at the end of a learning phase) or periodically.

In the next sections we introduce the value functions and present some approaches to calculate these functions. For clarity of exposition we proceed in a number of steps (Figure 5.2). First, we present the value functions for the case of fixed contracts (Section 5.4). In Section 5.4.1 we present a recursive formulation for the value functions. Because it appears that these functions are difficult to calculate exactly, we present some approximations in Section 5.4.2. After that, (Section 5.4.3) we describe how the value functions can be used for bid pricing, scheduling, and waiting decisions. In Section 5.4.4 we present the required modifications of our approach in order to deal with flexible contracts. Next, we describe how the parameters are estimated (Section 5.5). In Section 5.6 we improve our results by relaxation of some of the assumptions mentioned in Section 5.4.2 and 5.6.

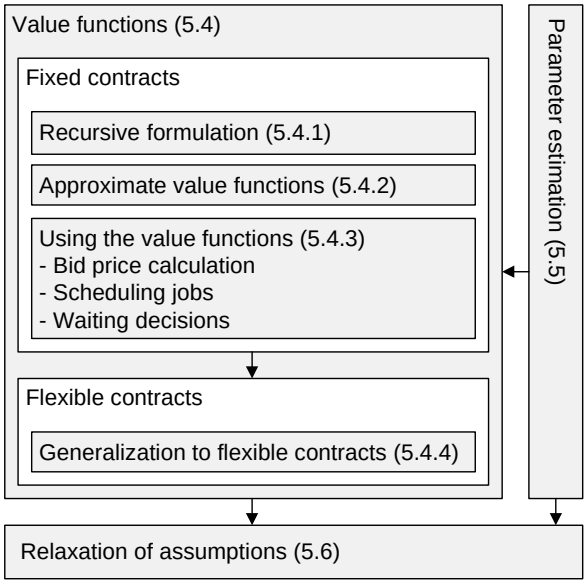

Figure 5.2: Roadmap for obtaining the value functions

5.4 Value functions

In the previous section we introduced the expected profits $V(\Psi, T)$ during a period T, given the current schedule Ψ. This profit is given by the sum of the winning prices p_φ for all jobs φ in the schedule Ψ, minus the direct costs for all jobs, plus the value of all gaps between subsequent jobs, plus the value of the end-gap.

The direct costs of a job φ with pickup time ω consist of travel costs for the loaded move and possibly penalty costs:

$$C^d(\varphi, \omega) = c^r\left(\tau^f_{o(\varphi)d(\varphi)}\right) + c^p\left((\omega - e(\varphi))^+\right) \tag{5.3}$$

The costs for empty moves are not included in the direct costs because they might be replaced by loaded moves. Instead, we include these costs in the value of gaps between subsequent jobs.

To quantify the values of gaps we introduce two value functions, namely a gap-value and an end-value. The *gap-value* $V^g(i, j, \sigma, t)$ is the expected profit of all future moves in a gap defined by start-node i, end-node j, expected time σ until arrival at node i, and gap length t. The *end-value* $V^e(i, \sigma, t)$ is the expected profit during a period $t = T - \sigma$ after arrival at schedule destination i at time σ from now on. As an example consider the schedule of Figure 5.1. The value of the first gap is given by $V^g(B, C, 2, 5)$ and the end-value is given by

$V^e(D, 15, T-15)$. In the remainder we use the word *time-to-go* to indicate the time σ from now till the arrival at the schedule destination i or the start-node i of a gap.

The expected profit $V(\Psi, T)$ during period T given schedule Ψ is now given by:

$$\begin{aligned} V(\Psi,T) &= \sum_{n=1}^{N} \left(p_{\varphi_n} - C^d(\varphi_n, \omega_n)\right) + V^e\left(d(\varphi_N), \rho_N, T-\rho_N\right) \\ &\quad + \sum_{n=1}^{N-1} V^g\left(d(\varphi_n), o\left(\varphi_{n+1}\right), \rho_n, \omega_{n+1}-\rho_n\right) \end{aligned} \tag{5.4}$$

We also use this expression to calculate the expected profit $V(\Psi \cup \varphi, T)$ of the temporal schedule with the new job φ. Because we do not know the payment for this new job φ, we set p_φ equal to zero (see Section 5.4.3).

5.4.1 Recursive formulation

In our multi-agent setting, a vehicle agent should find a sequence of decisions such that its expected trajectory of future states, within gaps or at the end-gap, yields the maximum expected reward. The values of these trajectories are given by the value functions V^e and V^g for which we derive recursive relations in this section. After discretization of the time, we obtain a Stochastic Dynamic Programming (SDP) recursion (see Section 5.4.2).

The recursive relations are described by four types of information: state space, decision set, transition probabilities, and expected rewards. We use the state variables to capture all necessary information to value the future behavior of the system to be controlled. Derived directly from the value functions at the beginning of Section 5.4, we use $\{i, j, \sigma, t\}$ to describe the state within a gap and $\{i, \sigma, t\}$ for the state in an end-gap. We present the recursion for end-values only, since the gap-values are derived in a similar - albeit slightly more complicated - manner. The necessary modifications for the gap-values will be discussed at the end of this section.

Example 5.2. *An illustration of the transition of states in the end-gap can be found in Figure 5.3. In this example, the current time is* $\theta = 9$*, the planning horizon is* $T = 14$*, and the vehicle schedule initially ends in location* C *at time* 17*; so the state at time* $\theta = 9$ *is given by* $\{i, \sigma, t\} = \{C, 8, 6\}$*. Suppose that the vehicle agent wins a next job at time* 13 *with origin* C *and destination* B*, to be picked up at time* 17 *and to be delivered at time* 19*. Then the state at the auctioning time* 13 *is given by* $\{B, 6, 4\}$*.*

As mentioned in Section 5.3.4, vehicles may create gaps in their schedule in

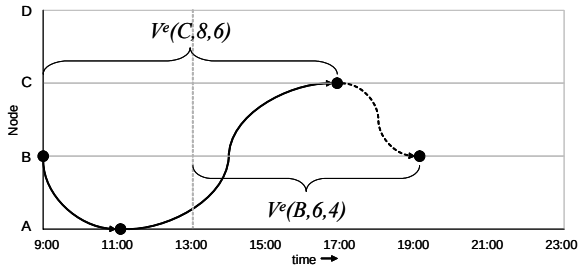

Figure 5.3: Transition of end states

anticipation of future jobs. However, for our dynamic programming recursion we assume that jobs are scheduled as early as possible. As a consequence, we have that in each iteration, the job is scheduled immediately after the job of the previous iteration. This enables us to use the state spaces as mentioned above instead of storing a complete schedule. As can be seen from our simulation experiments, this results in a slightly underestimation of the real profits.

For a recursion on the end-values, this means that a job is always added to the end of the schedule. Whenever a vehicle finishes a job and its schedule is not empty, it will drive immediately towards the origin of the next job. Whenever it finishes a job and its schedule is empty, it has to decide where to wait. Therefore, we consider the decision $\delta(i) \in \mathcal{N}$ to move pro-actively to node $\delta(i)$ directly upon arrival at node i. If $\delta(i) = i$, the decision is to wait at the current node i. In the remainder we use the shorthand notation δ instead of $\delta(i)$. Of course, a vehicle will only make this decision when it is waiting at some node ($\sigma = 0$). So if its current state is given by $\{i, 0, t\}$ then its next state, immediately after making the decision δ, is given by $\{\delta, \tau^e_{i\delta}, t - \tau^e_{i\delta}\}$.

To derive a recursion for the end-values $V^e(i, \sigma, t)$, we consider the following three cases: (1) we win a job during the time-to-go σ, (2) otherwise we end up at node i and decide to move pro-actively to node δ and we win a job during this time $\tau^e_{i\delta}$, and (3) otherwise we end up at node δ and we wait until we win the next job over there. In the remainder of this section, we show that we can express the corresponding value for each case using the same value function, which clarifies the presentation. For this purpose we introduce a *partial* value function $V^p(i, \sigma, \eta, t)$, which is defined as the expected future revenue during a period t for a vehicle ending in location i given it wins a job at time η during its time-to-go σ. The end-value is then derived by combining the three partial value functions. An illustration of the three cases can be found in Figure 5.4.

In *case* (1), the next job is won within the time-to-go σ, say at time $\theta + \eta$ ($0 \leq \eta \leq \sigma$). We define $p_{ikl}(\sigma - \eta)$ as the conditional probability that a vehicle ending in location i will have a trip from k to l as next job, given that the corresponding agent wins a job at time $\theta + \eta$. Here $\sigma - \eta$ is the time between

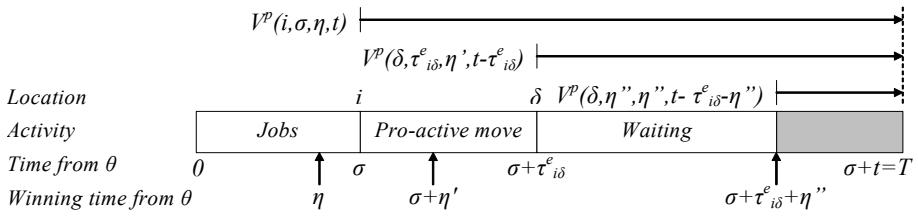

Figure 5.4: Three cases for the partial value function depending on the winning time

winning a job and the earliest time it can actually start this job. Note that the probability of winning a specific job may depend on this time because jobs have time-window restrictions (see Section 5.3.2) and these time-windows may differ per route.

We define the expected reward of a job from k to l that is won at time $\theta+\eta$ by $r_{ikl}(\sigma-\eta)$. This new job ends at time $\sigma+\tau_{ikl}$ from now on. If $\tau_{ikl} \leq t$ (the job is handled within the time horizon $T=\sigma+t$), then we include the full profit in the value function. Otherwise, we include a fraction t/τ_{ikl} corresponding to the percentage of the job that is completed within the time horizon. The next end-value is the value at the new end-node l at time $\sigma+\tau_{ikl}$, so the time-to-go from η on is $\sigma+\tau_{ikl}-\eta$ and the remaining time horizon is $\max\{t-\tau_{ikl},0\}$. By summation over all possible routes kl we get the following partial value function:

$$V^p(i,\sigma,\eta,t) = \sum_{k,l\in\mathcal{N}} p_{ikl}(\sigma-\eta) \left[\begin{array}{c} \alpha_{ikl}(t)\, r_{ikl}(\sigma-\eta) + \\ V^e(l,\sigma+\tau_{ikl}-\eta,t-\tau_{ikl}) \end{array} \right] \tag{5.5}$$

Here we use the boundary conditions $V^e(i,\sigma,t)=0$ and $V^p(i,\sigma,\eta,t)=0$ for $t\leq 0$. When $t<\tau_{ikl}$ we include only a fraction $\alpha_{ikl}(t)$ of the profit in the value function. This fraction is given by:

$$\alpha_{ikl}(t) = \min\left\{\frac{t}{\tau_{ikl}},1\right\}, t>0 \tag{5.6}$$

We put $\alpha_{ikl}(t)=0$ for $t\leq 0$. By weighing over the time η at which the next job is won - which we describe using a probability density function $f_{i\sigma}(\eta)$ and corresponding distribution function $F_{i\sigma}(\eta)$ - we find that the first part of the value function for the end-gap is given by $\int_0^\sigma f_{i\sigma}(\eta)\, V^p(i,\sigma,\eta,t)\, d\eta$. Note that $f_{i\sigma}(\eta)$ is an exponential density because we assumed Poisson arrivals, see Section 5.3.2.

In *case* (2) *and* (3), we do not win a job during the time-to-go σ. This happens with probability $1-F_{i\sigma}(\sigma)$. Then the vehicle is at location i at time

$\theta + \sigma$, and we update the current time θ by making it the arrival time at node i. Now we have to find the best option for the pro-active move to location δ, which takes a time $\tau^e_{i\delta}$ and which costs $c^r(\tau^e_{i\delta})$ (if $\delta = i$ we wait at node i without costs for a pro-active move). Therefore, we compute the expected revenues and costs if we move pro-actively to δ and we select the option which maximizes the revenues in our recursion:

- In case (2), we win the next job at time $\theta + \eta'$ before arrival at node δ ($0 \leq \eta' \leq \tau^e_{i\delta}$). The remaining time horizon directly after arrival at the end-node δ is given by $t - \tau^e_{i\delta}$. Therefore, we find that the partial value function is given by $V^p(\delta, \tau^e_{i\delta}, \eta', t - \tau^e_{i\delta})$, which we have to weigh over the time at which the next job is won, having density function $f_{\delta\tau^e_{i\delta}}(\eta')$.
- In case (3), we update the current time θ by making it the arrival time at node δ to which we have moved pro-actively. We win the next job at time $\theta + \eta''$. The new time-to-go is therefore also η'' and the remaining time horizon after winning this new job is $t - \tau^e_{i\delta} - \eta''$. The partial value function for this case is given by $V^p(\delta, \eta'', \eta'', t - \tau^e_{i\delta} - \eta'')$. Again, we have to weigh this function over the time at which the next job is won, having density function $f_{\delta 0}(\eta'')$. Note that this probability density function is slightly different because we do not have the condition that we win during the time-to-go σ which is zero in this case.

By combining the value functions for the three cases, we find the following relation for the end-value:

$$V^e(i,\sigma,t) = \int_0^\sigma f_{i\sigma}(\eta)\, V^p(i,\sigma,\eta,t)\, d\eta + (1 - F_{i\sigma}(\sigma)) \max_\delta \left\{ \begin{array}{r} -c^r(\tau^e_{i\delta}) + \int_0^{\tau^e_{i\delta}} f_{\delta\tau^e_{i\delta}}(\eta')\, V^p(\delta, \tau^e_{i\delta}, \eta', t - \tau^e_{i\delta})\, d\eta' \\ + \left(1 - F_{\delta,\tau^e_{i\delta}}(\tau^e_{i\delta})\right) \int_0^\infty f_{\delta 0}(\eta'')\, V^p(\delta, \eta'', \eta'', t - \tau^e_{i\delta} - \eta'')\, d\eta'' \end{array} \right\} \tag{5.7}$$

Again, we use the boundary conditions $V^e(i,\sigma,t) = 0$ and $V^p(i,\sigma,\eta,t) = 0$ for $t \leq 0$. By combining (5.7) with (5.5) we have a recursive formulation that can be used to calculate the end-values.

The gap-values $V^g(i,j,\sigma,t)$ are derived in a similar manner with three exceptions. First, the state is given by (i,j,σ,t) (see beginning of Section 5.4). This implies that the node j at the end of the gap has to be passed to the next iteration. Second, we use another boundary condition $V^g(i,j,\sigma,t) = -\infty$ if $t < \tau^e_{ij}$, meaning that we have to arrive on time at the gap destination j. Third, because transitions and corresponding revenues are dependent on the restrictions at the end of the gap, we also have to include the end-node j and remaining gap time t in the transition functions $p_{ikl}(\sigma - \eta, j, t)$, the revenue functions $r_{ikl}(\sigma - \eta, j, t)$, and in the winning time distribution $F_{ij\sigma t}(\eta)$.

5.4.2 Value function approximations

Even with perfect knowledge of the states, solving recursion (5.7) using (5.5) is a complex and time-consuming process. Especially because it can not be solved with backward dynamic programming (even not if we discretize time), because a value function depends on other value functions with both a larger and smaller time-to-go σ. Another complicating factor is that the state space can be very large. Therefore, we apply the following approximations: (1) discretization of time and (2) replacement of the time-to-go σ with its expectation.

For clarity of exposition we decided to perform two additional approximations: (3) we approximate the gap-values by using the same parameters for the gap-values as for the end-values and (4) we use a time-to-go of zero. This way we avoid introducing too many parameters and keep the algorithms compact. Besides the notational convenience, these approximations also reduce the computation times. As we show in our simulation experiments (Section 5.7) these approximations still provide accurate results. Improvement by relaxation of the last two approximations is straightforward as we will see in Section 5.6.

First, we present the approximation of the end-values. Next, we address the approximation of the gap-values.

End-value approximation

As a first approximation we discretize time, from the current time θ on, into intervals of length ε. For ease of notation, we assume that the time dimension is chosen such that $\varepsilon = 1$. Then we use a discrete probability density $q_{i\sigma}(\eta)$ instead of the continuous density $f_{i\sigma}(\eta)$, where $q_{i\sigma}(\eta)$ denotes the probability that the vehicle will receive a job in the time interval $(\theta+\eta, \theta+\eta+1]$, given schedule destination i and time-to-go σ. Similarly, we use $Q_{i\sigma}(\eta)$ instead of $F_{i\sigma}(\eta)$.

In the next approximation we replace the time-to-go σ with $\bar{\sigma}$. Because the state is now independent on the time-to-go σ, we can reduce this state to $\{i, t\}$ and we are able to derive the approximate value functions recursively. We denote the approximate end-values by $\tilde{V}^e$. This new value function is given by:

$$\tilde{V}^e(i,t) = \sum_{\eta=0}^{\bar{\sigma}} q_{i\bar{\sigma}}(\eta)\,\tilde{V}^p(i,\bar{\sigma},\eta,t) + \left(1 - Q_{i\bar{\sigma}}(\bar{\sigma})\right) \max_{\delta} \left\{ \begin{array}{c} -c^r\left(\tau^e_{i\delta}\right) + \sum_{\eta'=0}^{\tau^e_{i\delta}} q_{\delta\tau^e_{i\delta}}(\eta')\,\tilde{V}^p\left(\delta, \tau^e_{i\delta}, \eta', t-\tau^e_{i\delta}\right) \\ + \left(1 - Q_{\delta\tau^e_{i\delta}}\left(\tau^e_{i\delta}\right)\right) \sum_{\eta''=0}^{\infty} q_{\delta 0}(\eta'')\,\tilde{V}^p\left(\delta, \eta'', \eta'', t-\tau^e_{i\delta}-\eta''\right) \end{array} \right\} \tag{5.8}$$

where $\tilde{V}^p$ is the approximate partial value function. This function is exactly the same as in (5.5), with the only exception that the end-value $V^e(i,\sigma,t)$

is replaced by the approximate end-value $\tilde{V}^e(i,t)$ where the time-to-go σ is replaced with its expectation $\bar{\sigma}$.

The approximate end-values can be obtained using a simple backward stochastic dynamic programming recursion. At each single (discrete) point in time, starting with a planning horizon $t = 1$ until $t = T$, we calculate $\tilde{V}^e(i,t)$ for all schedule destinations $i \in \mathcal{N}$. However, as mentioned in the beginning of this section, we decided to use $\bar{\sigma} = 0$ in our recursions. This simplifies our presentations, especially regarding the gap-value functions. As a result, a vehicle never expects to win a job during a certain time-to-go. So at each point in time, we calculate the probability that we win an auction during this time unit. If we do not win an auction, we make a proactive move (ending multiple time-units ahead) or we wait a single period at this node. The backward recursion for the end-values is given in Algorithm 5.1.

init:
  given a planning horizon T
  $\tilde{V}^e(i,t) = 0 \quad \forall i \in \mathcal{N}$ with $t \leq 0$
for $t = 1$ to T do
  for $\forall i \in \mathcal{N}$ do

$$\tilde{V}^e(i,t) = Q_{i0}(1)\,\tilde{V}^p(i,t) + (1 - Q_{i0}(1)) \max_\delta \Big\{ -c^r(\tau^e_{i\delta}) + \tilde{V}^e(\delta, t - \max(\tau^e_{i\delta}, 1)) \Big\}$$

$$\tilde{V}^p(i,t) = \sum_{k,l \in \mathcal{N}} p_{ikl}(0) \Big[\alpha_{ikl}(t)\, r_{ikl}(0) + \tilde{V}^e(l, t - \tau_{ikl}) \Big]$$

  end;
end;

Algorithm 5.1: Calculating the approximate end-values

We may expect that this recursion for the approximate end-values $\tilde{V}^e(i,t)$ underestimates the realized average profits within a period of length t after arrival at node i, simply because we ignored the time-to-go σ. Therefore, we also present two extensions (Section 5.6) that provide a more precise treatment of the time-to-go σ.

Gap-value approximation

We approximate the gap-values analogously to the end-values. First, we replace the gap-value function $V^g(i,j,\sigma,t)$ by $\tilde{V}^g(i,j,t)$, where time is discretized and the time-to-go σ is replaced by $\bar{\sigma}$.

As mentioned at the beginning of Section 5.4.2, we further apply two approximations for clarity of presentation. First, we use a time-to-go of zero. Second, we approximate the transition probabilities $p_{ikl}(\sigma - \eta, j, t)$ by $p_{ikl}(\sigma - \eta)$,

the revenues $r_{ikl}(\sigma-\eta,j,t)$ by $r_{ikl}(\sigma-\eta)$, and the winning time distribution $f_{ij\sigma t}(\eta)$ by $f_{i\sigma}(\eta)$. A drawback of the latter approximation is that we may overestimate the winning probabilities, especially if we face a transition that involves a significant risk that we will violate the restrictions at the end of the gap. Then it is possible that we make a non-profitable transition, e.g., if we had taken into account the restrictions at the end of the gap we would not have made a certain transition. To overcome this, we multiply the transition probabilities with a decision variable δ^a_{kl}. This variable equals 1 if we accept the transition from k to l and otherwise it is zero. The logic behind this decision variable is that a vehicle always has the option to wait a single time unit if this seems to be more profitable than making this non-profitable transition. The approximate gap-values are calculated using the recursion given in Algorithm 5.2.

init:
  given an end-node j and gap length t
  $\tilde{V}^g(i,j,s)=-\infty \quad \forall i\in\mathcal{N}/j$ with $s\leq 0$, and $\tilde{V}^g(j,j,0)=0$
for $s=1$ to t do
  for $\forall i\in\mathcal{N}$ do

$$\tilde{V}^g(i,j,s)=\max_{\delta^a_{kl}|k,l\in\mathcal{N}}\left\{\begin{array}{r} Q_{i0}(1)\,\tilde{V}^p(i,j,t)+(1-Q_{i0}(1)+u(i))\times \\ \max_\delta\left(-c^r(\tau^e_{i\delta})+\tilde{V}^g(\delta,j,s-\max(\tau^e_{i\delta},1))\right)\end{array}\right\}$$

$$\tilde{V}^p(i,j,t)=\textstyle\sum_{k,l\in\mathcal{N}}\delta^a_{kl}p_{ikl}(0)\left[r_{ikl}(0)+\tilde{V}^g(l,j,t-\tau_{ikl})\right]$$

$$u(i)=Q_{i0}(1)\left(1-\textstyle\sum_{k,l\in\mathcal{N}}\delta^a_{kl}p_{ikl}(0)\right)$$

  end;
end;

Algorithm 5.2: Calculating the approximate gap-values

Here $u(i)$ is the probability that we did not accept a transition for a job won after node i. Note that the fraction $\alpha_{ikl}(t)$ is not used in the approximate gap-values because the vehicle always has to be at the end-node j before the end of the gap length t.

5.4.3 Using the value functions

As mentioned in Section 5.3.4, vehicles face three types of decisions: (1) bid price calculation, (2) scheduling jobs, and (3) operational waiting decisions. These decisions are supported by the end-values and gap-values. In this section we explain how this works.

Bid price calculation

In (5.2) we introduced the bid price of a new job φ in the current schedule Ψ, as the difference between $V(\Psi, T)$ and $V(\Psi \cup \varphi, T)$. After using (5.4), we see that the insertion of a new job φ at position n with pickup time ω in the current schedule Ψ, does not affect the winning prices and loaded travel costs of other jobs in the schedule Ψ. It only leads to additional direct costs $C^d(\varphi, \omega)$ and a decrease in one of the value functions in (5.4).

Suppose that at current time θ an auction is started for a new job φ. A vehicle agent evaluates all insertion positions n of the new job in the current schedule Ψ. An example is depicted in Figure 5.5 where a new job is inserted either in a gap or in the end-gap.

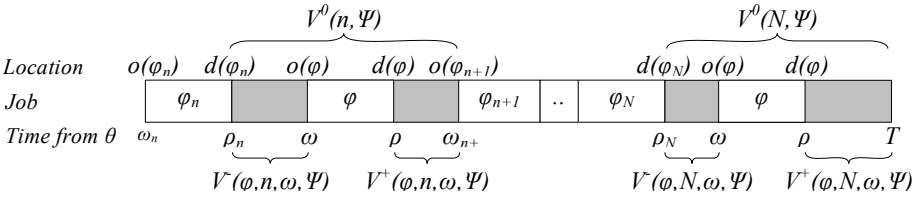

Figure 5.5: Change in value functions due to a job insertion

As can be seen, insertion of a new job implicates that a gap in the current schedule is replaced by two new gaps. We indicate the value of the old gap by V^0, the value of a new gap before the new job by V^- and the value of a new gap after the new job by V^+. Obviously, a gap with length zero has a value of zero. For a given insertion n and pickup time ω of the new job φ in the current schedule Ψ, these values are as follows:

$$V^0(n, \Psi) = \begin{cases} V^g\left(d(\varphi_n), o\left(\varphi_{n+1}\right), \rho_n, \omega_{n+1} - \rho_n\right) & if\ n < N \\ V^e\left(d(\varphi_N), \rho_N, T - \rho_N\right) & else \end{cases} \tag{5.9}$$

$$V^-(\varphi, n, \omega, \Psi) = V^g\left(d(\varphi_n), o(\varphi), \rho_n, \omega - \rho_n\right) \tag{5.10}$$

$$V^+(\varphi, n, \omega, \Psi) = \begin{cases} V^g\left(d(\varphi), o\left(\varphi_{n+1}\right), \rho, \omega_{n+1} - \rho\right) & if\ n < N \\ V^e\left(d(\varphi), \rho, T - \rho\right) & else \end{cases} \tag{5.11}$$

As mentioned before, a new job insertion will result in additional direct costs and a decrease in one of the value functions. We denote this loss in one of the value functions by an opportunity cost function $OC(\varphi, n, \omega, \Psi)$. The opportunity costs of a given insertion position n and pickup time ω are given by the value of the current gap or end-gap minus the value of the new gaps:

$$OC(\varphi, n, \omega, \Psi) = V^0(n, \Psi) - V^-(\varphi, n, \omega, \Psi) - V^+(\varphi, n, \omega, \Psi) \tag{5.12}$$

So the bid price of (5.2) can be rewritten as:

$$b(\varphi, \Psi) = \min_{n,\omega}\left(C^d(\varphi, \omega) + OC(\varphi, n, \omega, \Psi)\right) \tag{5.13}$$

The opportunity costs describe the loss in expected future revenues due to a new job insertion, taking into account the stochastic job arrival process. That is, the vehicle agent does not know which jobs will arrive, when they arrive, and which auctions will be won. However, the agent has information about past jobs and auctioning processes that can be used to estimate the attractiveness of a specific time slot at a specific location for the vehicle.

Scheduling jobs

As mentioned in Section 5.3.4, a vehicle contemplates the insertion of a new job at any position in the current schedule without altering the order of execution for the other jobs. To do this, the vehicle simply evaluates all possible insertion positions. In case of fixed contracts, the vehicle also has to decide about the optimal pickup time ω. The optimal pickup time ω is calculated using (5.3), (5.13), and (5.12). For a given insertion position n, this pickup time is the time at which the sum of the values for the two new gaps is maximal:

$$\omega = \arg\max_{\omega'} \left\{ V^{-}(\varphi, n, \omega', \Psi) + V^{+}(\varphi, n, \omega', \Psi) - c^{p}\left((\omega' - e(\varphi))^{+} \right) \right\} \tag{5.14}$$

To reduce the computational burden for calculating the optimal pickup time ω, we decided to use the approximations $\tilde{V}^{e}$ and $\tilde{V}^{g}$. When we calculate the approximate value functions for a given gap length t, we automatically obtain the value of all gap lengths $s < t$ (because we iterate on the remaining gap length). The idea is that we calculate the approximate value functions for extreme lengths of the two gaps resulting from a new job insertion (i.e., the gap before and after the new job). The optimal pickup time ω can then be calculated rather easily by using all intermediate values. Let us illustrate this with an example.

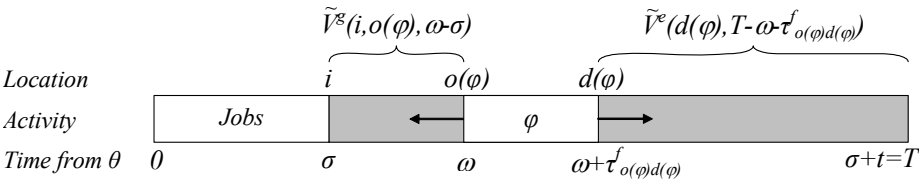

Figure 5.6: Optimal pickup time for appending a job

Example 5.3. *Consider a vehicle with schedule destination i and time-to-go σ (see Figure 5.6). Suppose this vehicle wants to add a new job φ to the end of its current schedule. Then the earliest pickup time is $\sigma + \tau^{e}_{io(\varphi)}$. The maximum length of the end-gap is therefore $T - \sigma - \tau_{io(\varphi)d(\varphi)}$. The maximum length of the gap before this job, given that the job has to be delivered within*

the planning horizon T, is $T-\sigma-\tau^{f}_{o(\varphi)d(\varphi)}$. The approximate value functions for the extreme gap lengths are thus given by $\tilde{V}^{e}\left(d\left(\varphi\right),T-\sigma-\tau_{io(\varphi)d(\varphi)}\right)$ and $\tilde{V}^{g}\left(i,o\left(\varphi\right),T-\sigma-\tau^{f}_{o(\varphi)d(\varphi)}\right)$. When we calculate these functions, we know the value of all intermediate gap lengths that can occur due to this new job insertion. We derive the optimal pickup time ω, by using these intermediate gap values, as follows:

$$\omega=\arg\max_{\omega'}\left\{\begin{array}{c}\tilde{V}^{g}\left(i,o\left(\varphi\right),\omega'-\sigma\right)+\tilde{V}^{e}\left(d\left(\varphi\right),T-\omega'-\tau^{f}_{o(\varphi)d(\varphi)}\right)\\ -c^{p}\left(\left(\omega'-e\left(\varphi\right)\right)^{+}\right)\end{array}\right\} \tag{5.15}$$

Waiting decisions

Whenever a vehicle becomes idle, it has to decide upon a pro-active move. This decision can simply be found by using the parts of the value functions for which we assume we have to wait. Using (5.8), we find that the best waiting decision $\delta^{*}\left(i\right)$ at node i in case of an end-gap (based on $\bar{\sigma}$) is given by:

$$\delta^{*}\left(i\right)=\arg\max_{\delta\in\mathcal{N}}\left\{\begin{array}{c}-c^{r}\left(\tau^{e}_{i\delta}\right)+\sum_{\eta'=0}^{\tau^{e}_{i\delta}}q_{\delta\tau^{e}_{i\delta}}\left(\eta'\right)\tilde{V}^{p}\left(\delta,\tau^{e}_{i\delta},\eta',t-\tau^{e}_{i\delta}\right)\\ +\left(1-Q_{\delta\tau^{e}_{i\delta}}\left(\tau^{e}_{i\delta}\right)\right)\times\\ \sum_{\eta''=0}^{\infty}q_{\delta0}\left(\eta''\right)\tilde{V}^{p}\left(\delta,\eta'',\eta'',t-\tau^{e}_{i\delta}-\eta''\right)\end{array}\right\} \tag{5.16}$$

If we are at the end of our schedule and all jobs have a stationary arrival process, then this decision has to be taken only once. If there is a more profitable node δ than the current node i, then the vehicle will move directly towards this node, otherwise it will wait at node i until the next job.

In case of a gap, this decision has to be taken at every moment in the gap for which the vehicle is not active. However, an optimal waiting strategy can be found in advance using the approximate gap-values which are already calculated with Algorithm 2. In this recursion we established the best waiting decision for each node i and remaining gap length δ. Searching these values will provide the time at which the vehicle has to move to the end-node of the gap, given that it did not receive another job in this gap.

5.4.4 Extension to flexible contracts

The approach in case of flexible contracts is quite different from the fixed contracts. In case of flexible contracts, vehicles no longer have to agree on the pickup times of jobs. Moreover, it is not necessary for the vehicle agent to maintain a detailed schedule of all pickup and delivery times of all jobs. However, to calculate the costs of a new job insertion, and to use the gap- and

end-value functions, we need some kind of schedule. For this purpose we use a tight schedule in the sense that, given a certain order of jobs, all these jobs are scheduled as early as possible, while keeping in mind the required time for empty moves from the destination of a job to the origin of the next job. Note that this schedule is only used to support the bid pricing decisions. A vehicle is not restricted by these pickup times, but can simply decide to insert new jobs or to wait at some node after delivery of a job. To avoid excessive computation times, we also consider the insertion scheduling heuristic for the case of flexible contracts.

When using flexible contracts, we have to change our approach on five aspects: (1) the temporal schedule, (2) the direct costs, (3) the end-values, (4) the gap-values, and (5) the waiting strategy.

The temporal schedule

Whenever a vehicle receives a job announcement, it only has to decide upon the best insertion position and no longer upon the pickup time ω (because each job is scheduled as early as possible). Therefore, we use an alternative schedule $\Psi_{\varphi,n}$ for inserting job φ after the n^{th} job in the current schedule Ψ. Again, the temporal schedule is given by an alternative schedule with the highest profits:

$$V(\Psi \cup \varphi) = \max_{n} \{V(\Psi_{\varphi,n})\} \tag{5.17}$$

The direct costs

Another consequence of using the flexible contracts, is that the penalty costs are not fixed in advance. Because a vehicle no longer has to agree on a certain pickup time, it has the possibility of delaying the pickup time of a job against predetermined penalties. Instead of calculating the penalties for the new job φ, we now have to calculate the increase in penalties for all jobs in the temporal schedule compared to the current schedule, cf. (3.1) in Chapter 3. As a consequence, the direct cost function of (5.3) is no longer applicable. Instead, we now use the original equation (5.2) in combination with (5.4) to calculate the bid price.

The end-value

For the end-values we use the approximate value function $\tilde{V}^e(i,t)$, with i the destination of the last job and t the remaining planning horizon after the scheduled delivery time of the last job in the tight schedule. Although the actual time-to-go σ is not used in the approximate end-values, we want to point out that the time-to-go σ is somewhat different in case of flexible contracts because

the timing of jobs is not fixed, and as a consequence we do not know the time until delivery of the last job in the schedule.

The gap-values

Also for the gap-values, we use the approximate value function $\tilde{V}^g(i,j,t)$, with i the start-node and j the end-node. However, the meaning of t is different here. We define t_n as the flexibility of the n^{th} gap (the gap after job n). This flexibility consists of an empty travel time τ^e_{ij} from the start-node i to the end-node j, plus a time slack s_n. This time slack is the maximum amount of time that this job can be postponed without causing an increase in penalties for this job or one of the succeeding jobs. We calculate the gap flexibilities recursively as shown in Algorithm 5.3.

init: $s_N = \infty$
for $n = N-1$ down to 1 do
$\quad s_n = \min\left(\left(e\left(\varphi_{n+1}\right) - \omega_{n+1}\right)^+, s_{n+1}\right)$
$\quad t_n = s_n + \tau^e_{d(\varphi_n)o(\varphi_{n+1})}$
end;

Algorithm 5.3: Calculating the gap flexibilities

Note that the gap-value $\tilde{V}^g(i,j,t)$ is only defined for $t \geq \tau^e_{ij}$. For future reference we put $\tilde{V}^g(i,j,t) = -\infty$ for $t < \tau^e_{ij}$.

A major implication of the flexible contracts for insertion of a new job in a gap, is that it reduces the value of succeeding gaps and of the end-value. For the dynamic programming recursion this means that (1) we have to subtract these values in every iteration and (2) the gap-values are defined recursively. Regarding the latter, we have to calculate the value of the end-gap first, then the value of the gap before the last job, then working backwards until the gap after the first job. We denote the decrease in end-value due to a delay t of the scheduled delivery time of the schedule destination d by $\Delta\tilde{V}^e(d,t)$. We use $\Delta\tilde{V}^g(n,t)$ to denote the sum of the decreases in gap-values for all gaps after the n^{th} gap, given the scheduled pickup time of the job $n+1$ is postponed a time t. To calculate the gap-values for flexible contracts, we use a similar recursion as in Section 5.4.2, but with the two distinctions mentioned above. The approximate gap-values are calculated using Algorithm 5.4.

To approximately calculate the decrease $\Delta\tilde{V}^e(d,t)$ in end-value, we do not have to know the time-to-go σ until arrival at the schedule destination d. One can argue that the expected profit on day t equals the expected profit on day $t+1$, for t large enough (see Chapter 7). Therefore, we calculate the change in end-value as follows:

init:
  given a schedule destination d
  $t_N = \infty$
for $n = N-1$ down to 1 do
  given $j = o\left(\varphi_{n+1}\right)$, and t_n (the end-node and flexibility of the n^{th} gap)
  $\tilde{V}^g(i,j,s) = -\infty \quad \forall i \in \mathcal{N}/j$ with $s < \tau^e_{ij}$, and $\tilde{V}^g(j,j,s) = 0 \quad \forall s$
  for $\forall i \in \mathcal{N}/j$ do
    for $s = \tau^e_{ij}$ to t_n do (only if $\tau^e_{ij} \leq t_n$)

$$\tilde{V}^g(i,j,s) = \max_{\delta^a_{kl} | k,l \in \mathcal{N}} \left\{ Q_{i0}(1)\, \tilde{V}^p(i,j,s) + (1 - Q_{i0}(1) + u(i)) \times \max_{\delta \in \mathcal{N}} \begin{pmatrix} -c^r\left(\tau^e_{i\delta}\right) + \tilde{V}^g\left(\delta, j, s - \max\left(\tau^e_{i\delta}, 1\right)\right) \\ -\Delta\tilde{V}^e\left(d, \max\left(\tau^e_{i\delta}, 1\right) + \tau^e_{\delta j} - \tau^e_{ij}\right) \\ -\Delta\tilde{V}^g\left(n, \max\left(\tau^e_{i\delta}, 1\right) + \tau^e_{\delta j} - \tau^e_{ij}\right) \end{pmatrix} \right\}$$

$$\tilde{V}^p(i,j,s) = \textstyle\sum_{k,l \in \mathcal{N}} \delta^a_{kl} p_{ikl}(0) \left\{ \begin{pmatrix} r_{ikl}(0) - \Delta\tilde{V}^e\left(d, \tau_{ikl} + \tau^e_{lj} - \tau^e_{ij}\right) \\ -\Delta\tilde{V}^g\left(n, \tau_{ikl} + \tau^e_{lj} - \tau^e_{ij}\right) + \tilde{V}^g(l, j, s - \tau_{ikl}) \end{pmatrix} \right\}$$

$$u(i) = Q_{i0}(1)\left(1 - \textstyle\sum_{k,l \in \mathcal{N}} \delta^a_{kl} p_{ikl}(0)\right)$$

    end;
  end;
end;

Algorithm 5.4: Calculating the approximate gap-values with flexible contracts

$$\Delta\tilde{V}^e(d,t) = \tilde{V}^e(d,T) - \tilde{V}^e(d,T-t) \tag{5.18}$$

To calculate the decrease $\Delta\tilde{V}^g(n,t)$ in gap-values, we have to recalculate the gap flexibilities t_m for all gaps $m \geq n+1$. We denote the updated flexibilities by t'_m. These flexibilities are calculated using Algorithm 5.3, with the only exception that we increase all times ω_m and ρ_m for $m \geq n+1$ with time t. We calculate the change in gap-values as follows:

$$\Delta\tilde{V}^g(n,t) = \sum_{m=n+1}^{N-1} \begin{pmatrix} \tilde{V}^g\left(d\left(\varphi_m\right), o\left(\varphi_{m+1}\right), t_m\right) \\ -\tilde{V}^g\left(d\left(\varphi_m\right), o\left(\varphi_{m+1}\right), t'_m\right) \end{pmatrix} \tag{5.19}$$

A major disadvantage of the dependencies between gaps, is that we can not simply calculate the value of all possible gaps in advance. Therefore, we also propose a variant in which we ignore the dependencies in gaps. Here we treat each gap in the schedule as if it is the last gap. So each new move that

is inserted in this gap only reduces the value of the end-gap. Because we can calculate the end-values in advance for all possible schedule destinations, we can also calculate these flexible gap-values for all possible gaps in advance. As we will show in Section 5.7, this reduces the computation time and still yields reasonable results.

Example 5.4. *As an illustration, let us consider the schedule of Figure 5.7. The free flexibility of the third, second, and first gap are respectively given by:* $t_3^p = \min\left((14-12)^+, \infty\right) = 2$, $t_2^p = \min\left((10-7)^+, 2\right) = 2$, $t_1^p = \min\left((6-5)^+, 2\right) = 1$. *The gap flexibilities are now given by:* $t_3 = 2+(12-9) = 5$, $t_2 = 2$, *and* $t_1 = 1+(5-3) = 3$. *To calculate the value of this schedule, we subsequently calculate* $\tilde{V}^e(C, T-15)$, $\tilde{V}_3^g(A, D, 5)$, $\tilde{V}_2^g(C, C, 2)$, *and* $\tilde{V}_1^g(B, D, 3)$.

Now consider the value $\tilde{V}_1^g(B, D, 3)$ *for the first gap. For* $s < 2$, *the value* $\tilde{V}_1^g(B, D, s)$ *is* $-\infty$ *because the time* τ_{BD}^e *required for the empty move is longer than the available flexibility. For* $s = 2$, *the value* $\tilde{V}_1^g(B, D, s)$ *is* $-c^r(2)$ *because the available flexibility if just enough to do the empty move. For* $2 < s \leq 3$, *there is a probability that the empty move is replaced by one or more loaded moves. Then every iteration will decrease the flexibility of the subsequent gaps by* $s-2$. *So we have to subtract the following values:*

$$\begin{aligned}
\Delta\tilde{V}^e(C, s-2) &= \tilde{V}^e(C, T) - \tilde{V}^e(C, T-(s-2)) \\
\Delta\tilde{V}^g(1, s-2) &= \tilde{V}_2^g(C, C, 2) - \tilde{V}_2^g(C, C, 2-(s-2)) + \\
&\quad \tilde{V}_3^g(A, D, 5) - \tilde{V}_3^g(A, D, 5-(s-2))
\end{aligned}$$

This computation can be done very fast because these values are already calculated given the backward iteration over all gaps and time lengths.

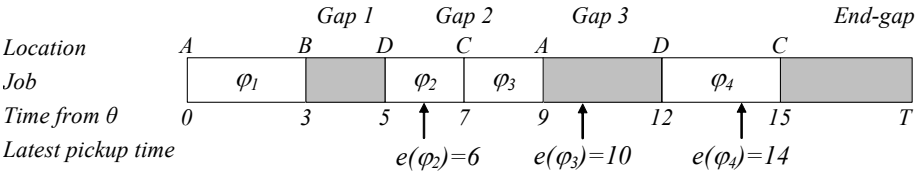

Figure 5.7: Flexible gap values

The waiting strategy

A final implication of the flexible contracts is their affect on the waiting strategy. With the flexible contracts we only have an operational waiting decision. Upon delivery of a job we have an option (1) to wait at the current location, (2) to

drive immediately to the origin of the next job, or (3) to move proactively to another location and wait over there. Again, the optimal decision $\delta^*(i)$ in node i can be found in the dynamic programming recursion:

$$\delta^*(i) = \arg\max_{\delta \in \mathcal{N}} \begin{pmatrix} -c^r(\tau^e_{i\delta}) + \tilde{V}^g(\delta, j, s - \max(\tau^e_{i\delta}, 1)) \\ -\Delta\tilde{V}^e\left(d, \max(\tau^e_{i\delta}, 1) + \tau^e_{\delta j} - \tau^e_{ij}\right) \\ -\Delta\tilde{V}^g\left(n, \max(\tau^e_{i\delta}, 1) + \tau^e_{\delta j} - \tau^e_{ij}\right) \end{pmatrix} \tag{5.20}$$

In the previous sections, we presented dynamic programming recursions for the gap-values and end-values, for both the fixed and flexible contracts. Before calculating the gap-values and end-values we need to estimate the required parameters. In the next section we describe how this can be done.

5.5 Parameter estimation

As mentioned in Section 5.4.2, we use the same parameters for the gap-values as for the end-values. In this section we describe how these parameters can be estimated in order to calculate the (approximate) value functions. In Section 5.6 we describe the required changes to this section in order to use intrinsic parameters for the gap-values.

Because we have deterministic travel times, we assume that all vehicles are aware of the travel times τ^e_{ij} and τ^f_{ij} for all routes $i, j \in \mathcal{N}$. This leaves us with the following parameters that we need to estimate:

- The average time-to-go $\bar{\sigma}$.
- The conditional probability $p_{ikl}(\sigma - \eta)$ that a vehicle ending in location i will have a trip from k to l as next job, given that the corresponding agent wins a job at time $\theta + \eta < \theta + \sigma$.
- The expected reward $r_{ikl}(\sigma - \eta)$ of a job from k to l that is won at time $\theta + \eta < \theta + \sigma$ with start-node i.
- The probability $q_{i\sigma}(\eta)$ that we win a new job in the time interval $(\theta + \eta, \theta + \eta + 1]$ if the schedule ends at node i.
- The distribution function $Q_{i\sigma}(\sigma)$ that we win a new job during the time-to-go σ if the schedule ends at node i.

We estimate these parameters and functions based on historic data. A possibility is to store all historic waiting times, revenues, travel times, and transition percentages for all possible states. Even when these parameters do not depend on the time-to-go σ, we must store a lot of information. Therefore,

we propose to estimate these parameters based on auction data for certain routes and the job arrival intensity for these routes. The auction data can be used to estimate (1) the sample mean $\bar{x}_{ij}$ and sample variance s^2_{ij} of all observations of the winning price for all routes $i, j \in \mathcal{N}$; (2) the average job arrival intensities λ_{ij} for all routes $i, j \in \mathcal{N}$; (3) the average time-window length z_{ij} for jobs on all routes $i, j \in \mathcal{N}$; and (4) the average time-to-go $\bar{\sigma}$. The time-window lengths z_{ij} describes the time between the announcement of a job and the latest pickup time. The expected length of a time-window might be dependent on a lot of characteristics such as a specific shipper. In this chapter we focus on certain routes and therefore we assume that these time-window lengths can be estimated based on the route. The arrival intensities λ_{ij} are estimated based on the job announcement characteristics by taking the average of past observations. The average time-to-go $\bar{\sigma}$ is estimated as the average time between winning a job and picking up this job. Note that when the auction data is only visible by the carriers, we assume that they inform their vehicles about these parameters.

In the next section we describe how the vehicles estimate the distribution of the lowest bid using the sample mean and variance provided by their carrier. In Section 5.5.2 we describe how the vehicles calculate the transition probabilities $p_{ikl}(\sigma - \eta)$ and expected revenues $r_{ikl}(\sigma - \eta)$. Calculation of the winning probabilities $q_{i\sigma}(\eta)$ and distribution $Q_{i\sigma}(\sigma)$ of winning moments is given in Section 5.5.3.

5.5.1 Distribution of lowest bids

Consider a route k, l. The estimation of the distribution parameters of the lowest bids for jobs on this route, depends on the structure of the auction. Here we use a second-price auction (see Section 5.3.3) where only the winning price, i.e., the one but lowest bid, is published. To estimate the distribution of the lowest bid, carriers can use information about all winning prices together with their own bid history. Because your own bids provide little information about the bids of your competitors, we only use information about the winning prices. This causes a problem, because we need the probability distribution of the *lowest* price whereas we only have observations of the *one but lowest* price. In this section we discuss how we deal with this problem.

The choice for a certain distribution function which is appropriate to describe the lowest bid, depends on the transportation network under consideration. It depends, among others, on (1) whether transportation takes place in a continuous area or between nodes, (2) the number of vehicles, and (3) the bid pricing behavior of these vehicles. Consider, for example, a triangular network with equal distances between the three nodes. Suppose further that the bid price of a vehicle, for transporting a load between these nodes, is given by the increase in travel distance of the cheapest insertion. One can easily see that

we then only have four possible bid prices. An empirical distribution would be most appropriate in this case. Of course the number of price classes will increase when (1) bids also include penalty costs and opportunity costs and (2) we have a network with a lot of nodes. Without loss of generality, we decided to use a continuous distribution function. For clarity of presentation, we use the same distribution function in our simulation experiments, even though network instances with only three nodes are involved.

We use the theory of the so-called Extreme Value Distributions (EVD), being a class of probability distributions for the order statistics of a large set of random observations from the same (arbitrary) distribution (m^{th} order statistic is the m^{th} smallest value). Particularly, the EVD cover the minimum and maximum value, but also a limiting distribution for the one but lowest (highest) observation is known. These limiting distributions have the same set of distribution parameters. This enables us to estimate the parameters of the limiting distribution for the one but lowest bid, and use these parameters for the limiting distribution of the minimum bid. Below, we elaborate on this approach in formulas.

Suppose that the bids b_c for a single job from competitor c ($c = 1..C$) are independent and identically distributed with a cumulative distribution function $H(x)$. We denote the probability distribution functions of the corresponding order statistics by $H_m(x)$. The probability distributions of the first- and second order statistic are respectively given by:

$$H_1(x) = 1 - \Pr(b_1 > x, b_2 > x, ..., b_n > x) = 1 - (1 - H(x))^n \quad (5.21)$$

$$H_2(x) = nH^{n-1}(x)[1 - H(x)] + H^n(x) \quad (5.22)$$

Except for special cases, it is not possible to express these distributions as closed form expression with parameters that can easily be estimated. It is shown in (Gumbel, 1958), that for any well-behaved initial distribution (i.e., $H(x)$ is continuous and has an inverse), limiting distributions for $n \longrightarrow \infty$ can be derived. In this chapter we use the Gumbel distribution which only requests that the tail of the distribution $H(x)$ declines exponentially (normal, log-normal, exponential, and gamma). The Gumbel distributions $G_1(x)$ for the lowest value and $G_2(x)$ for the second lowest value are given by:

$$G_1(x) = 1 - e^{-e^{\frac{x-\alpha}{\beta}}} \quad (5.23)$$

$$G_2(x) = 1 - \left(e^{-e^{\frac{x-\alpha}{\beta}}}\left(1 + e^{\frac{x-\alpha}{\beta}}\right)\right) \quad (5.24)$$

where α and $\beta > 0$ are the location and scale parameters (Reiss and Thomas, 1997). We use G_1 and G_2 to approximate H_1 and H_2 respectively. We estimate the parameters α and β using observations from $G_2(x)$ and insert these parameters in $G_1(x)$. Various statistical methods can be used to estimate α and β, depending on the observed data. In case of an open auction we can

simply use the Method of Moments. From the moments of the standard Gumbel distribution $G_m(x)$ (see Reiss and Thomas, 1997), it is straightforward to derive the location and scale parameter of $G_2(x)$:

$$\beta_{kl} = \frac{s_{kl}}{\sqrt{\frac{1}{6}\pi^2 - 1}} \tag{5.25}$$

$$\alpha_{kl} = \bar{x}_{kl} + \beta_{kl}(\gamma - 1) \tag{5.26}$$

Here $\gamma = 0.577216...$ is Euler's constant, and $\bar{x}_{kl}$ and s_{kl} are respectively the sample mean and sample standard deviation of all historical winning prices (the second-lowest bids) for jobs on route k, l. In the remainder we indicate the distribution of the lowest bid by $H_{kl}^{\min}(x)$, which in fact describes the probability that a vehicle will lose an auction given its bid price x.

Note that we used the assumption that bids in successive rounds are independent and identically distributed (iid) random variables. We may question whether that assumption is realistic, because the system state (state of the vehicles) will be quite similar in successive auctioning rounds unless the job arrival frequency is very low. However, Reiss and Thomas (1997) state that even if the distributions $H(x)$ are not exactly known or the iid condition of the bids fails, then H_1 may still be an accurate approximation of the actual distribution of the minimum. Because H_2 can be expressed as a function of H_1, the same holds for the distribution of the second lowest bid.

5.5.2 Estimating revenues and transition probabilities

To estimate the revenues and transition probabilities, the vehicle agent uses so-called winning intensities. The winning intensities provide for all routes the intensity at which the carrier expects to win jobs given a certain state. We define the winning intensity $\xi_{ikl}(\sigma)$ as the mean number of winning jobs per time unit from k to l after arrival at node i with time-to go σ. The winning intensities are given by:

$$\xi_{ikl}(\sigma) = \lambda_{kl} \cdot p_{ikl}^{win}(\sigma) \tag{5.27}$$

where $p_{ikl}^{win}(\sigma)$ is the probability of winning a job from k to l with start-node i and time-to-go σ, given by:

$$p_{ikl}^{win}(\sigma) = 1 - H_{kl}^{\min}(b_{ikl}(\sigma)) \tag{5.28}$$

where $b_{ikl}(\sigma)$ is the expected bid price for the vehicle, given a job on route kl, location i, and time-to-go σ. This bid price consists of direct costs $c_{kl}(\sigma) = c^r\left(\tau_{kl}^f\right) + c^p\left((\sigma + \tau_{ik}^e - z_{kl})^+\right)$ and opportunity costs OC_{ikl}:

$$b_{ikl}(\sigma) = c_{kl}(\sigma) + OC_{ikl} \tag{5.29}$$

This bid price is derived from (5.13). We modified the notation because we do not have to decide about an insertion position or pickup time in our recursions since we assume that subsequent jobs are scheduled immediately after each other (see Section 5.4.1). The opportunity costs, however, are not known yet. On the contrary, this whole approach is focused on finding them. Therefore we use an estimate based on previous opportunity costs charged in this state (an extension is discussed in Section 5.6). In case of end-values, we approximate the opportunity costs by:

$$OC_{ikl} = \tilde{V}^e(i,T) + c^r(\tau^e_{ik}) - \tilde{V}^e(l, T - \tau_{ikl}) \tag{5.30}$$

Here we used the approximate value functions to reduce computation time. A difference with the original opportunity cost function (5.12) is that the value of the gap before the new job is replaced by the costs for an empty move. The reason for this is that the new job from node k to node l is scheduled as early as possible, i.e., a time τ^e_{ik} after arrival at node i.

In the end-value function (5.7) we integrate the expected revenues over all possible winning moments η. Given that we win a job during time-to-go σ at time η, the remaining time-to-go is $\gamma = \sigma - \eta$. The expected revenue of a vehicle as a function of the winning time-to-go γ, is given by the difference between the expected lowest bid of its competitors (given that its own bid is lower) and the direct costs:

$$r_{ikl}(\gamma) = \frac{1}{1 - H^{\min}_{ikl}(b_{ikl}(\gamma))} \int_{x=b_{ikl}(\gamma)}^{\infty} x dH^{\min}_{ikl}(x) - c_{kl}(\gamma) - c^r(\tau^e_{ik}) \tag{5.31}$$

In our simulation experiments (see Section 5.7) we calculate this function by numerical integration. The conditional probability $p_{ikl}(\gamma)$ that the winning job has origin k and destination l given location i is given by:

$$p_{ikl}(\gamma) = \frac{\xi_{ikl}(\gamma)}{\sum_{k'l'} \xi_{ik'l'}(\gamma)} \tag{5.32}$$

When time-windows, penalty costs, or travel costs differ per route, then also the transition probabilities will be time dependent. So at different points in time, different jobs will have the highest winning probability. We illustrate this by means of an example.

Example 5.5. *Suppose we only have three possible routes AB, AC, and AD. The arrival intensities of jobs on these routes are* $\lambda_{AB} = 0.2$, $\lambda_{AC} = 0.4$, *and* $\lambda_{AD} = 0.6$. *The time-window length for each job is zero and the travel costs are 50. For all possible destinations* $i \in \{B, C, D\}$, *the winning intensities* $\xi_{AAi}(\sigma)$ *are calculated using (5.27), where the* $H^{\min}_{AAi}(b)$ *is the Gumbel distribution with parameters* $\alpha_{Ai} = 100$ *and* $\beta_{Ai} = 5$. *The conditional probabilities* $p_{AAi}(\gamma)$ *are calculated using (5.32), with* $i \in \{B, C, D\}$.

First, consider a linear penalty function of 1 *per time unit for all jobs. The winning intensities and the conditional probabilities are depicted in Figure 5.8 in the upper left corner and upper right corner respectively. We see that all jobs have equal winning probability, independent on the winning time-to-go* γ*. Now suppose the penalty costs differ per job. We use a linear penalty function with costs of* 1, 1.5, *and* 2 *for jobs on route AB, AC, and AD respectively. The winning intensities and the conditional probabilities are depicted in Figure 5.8 in the bottom left corner and bottom right corner respectively. Clearly at different moments in time, different jobs have the highest probability of being won.*

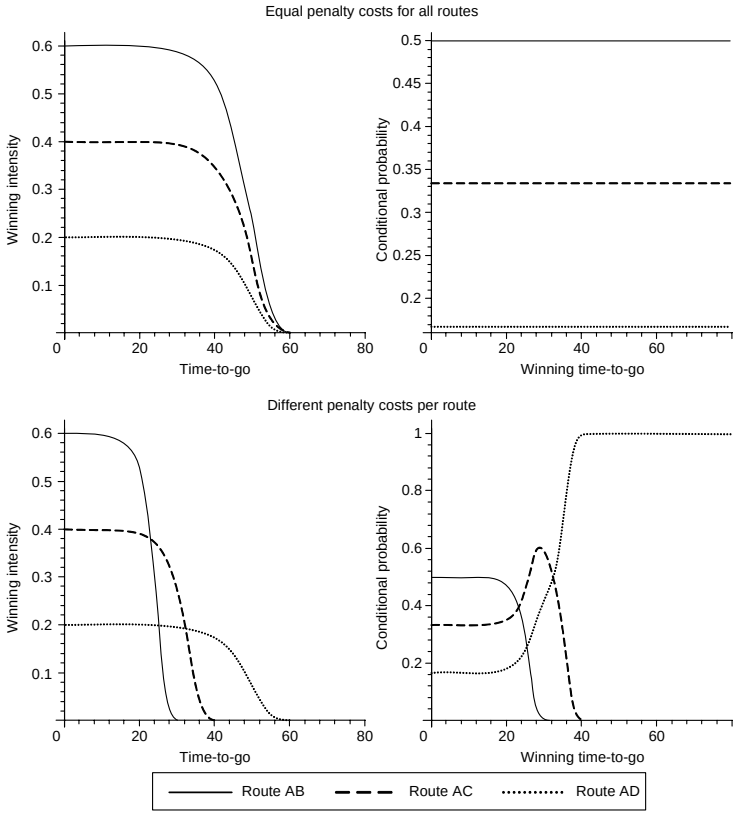

Figure 5.8: Time-dependent conditional winning probabilities

5.5.3 Distribution of winning moments

We described the winning moments η by a distribution function $Q_{i\sigma}(\eta)$. Difficulty is that the winning moments follow a so-called non-homogeneous Poisson process (NHPP) (see Ross, 2003). To ease our notation we introduce the following definition:

$$\Lambda_{i\sigma}(t) = \int_0^t \sum_{kl} \xi_{ikl}(\sigma - u)\, du \tag{5.33}$$

The value $\Lambda_{i\sigma}(\eta)$ is called the rate function of the nonhomogeneous Poisson process because the counting process of winning jobs can be described by a Poisson process with mean $\Lambda_{i\sigma}(\eta)$. The distribution function of the random amount of time η until the first time we win an auction is given by:

$$Q_{i\sigma}(\eta) = 1 - e^{-\Lambda_{i\sigma}(\eta)}, \quad 0 \leq \eta \leq \sigma, \sigma > 0 \tag{5.34}$$

Note that when $\sigma < z_{kl} - \tau^e_{ik}$ for all k, then the penalties in the expected bid $b_{ikl}(\sigma)$ of (5.29) are zero. As a consequence, the winning moments follow a homogeneous Poisson process $Q_{i\sigma}(\eta) = 1 - e^{-\eta \sum_{kl} \xi_{ikl}(\sigma)}$. This situation generally occurs in our simulation experiments.

5.6 Relaxation of assumptions

Throughout this chapter, we made several assumptions regarding the calculation of the value functions. In this section we relax some of these assumptions. First, we provide two relaxations that account for a more precise treatment of the time-to-go σ. Next, we present an alternative way to calculate the expected opportunity costs of (5.30). Finally, we describe how we can use custom parameters for the gap-values.

5.6.1 Relaxation 1: precise first uncertain move

In Section 5.4.2 we decided to approximate the time-to-go σ either by zero, or by its expectation $\bar{\sigma}$. We made this approximation to derive a backward recursion for the value functions. However, an option is to calculate the first recursive step more precisely by using the actual time-to-go σ. Therefore, we distinguish between the first uncertain move and all further uncertain moves (cf. Powell et al., 1988). The first uncertain move occurs directly after arrival at node i, which is the schedule destination or the start-node of a gap. All further uncertain moves occur after that. The idea is that the first uncertain move (1) has the largest impact on the expected profits and (2) can still be calculated more precisely because we can do it outside the dynamic programming recursion. We illustrate this process with an example.

Example 5.6. *Consider the end-value of the schedule depicted in Figure 5.9. The numbers in black squares refer to the three cases for winning a new job, as mentioned in Section 5.4.1. The current time is 9:00, and the destination of the schedule is node C with a time-to-go of 4 hours. Because we are looking at the end-value, all moves before 13:00 are treated as certain moves. There is a probability (case 1) that we receive a next job before arrival at our current schedule destination. In this case we will have a first uncertain move directly after 13:00 consisting of a full move and possibly proceeded by an empty move. Otherwise (case 2), we will make a pro-active move towards node B and receive a job during this pro-active move. Otherwise (case 3), we wait at node B until we win a job. After the first uncertain move we approximate all further possible moves, indicated by the grey dotted lines.*

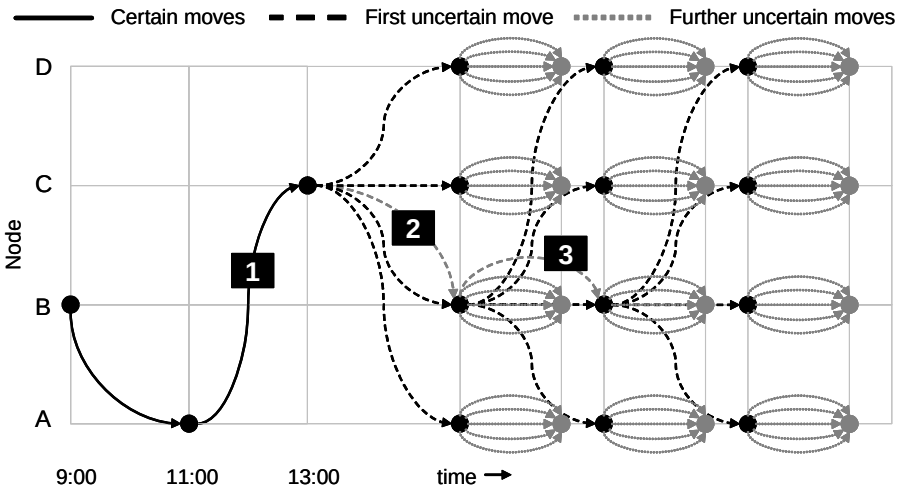

Figure 5.9: Approximation of end-values

In Section 5.4 we introduced the approximate value functions $\tilde{V}^e$ and $\tilde{V}^g$, for the end-value and the gap-value respectively. Now we use these approximations to describe the value of all moves after the first uncertain move. To describe the value of all uncertain moves, including the first uncertain move, we introduce the value function $\hat{V}^e(i,\sigma,t)$ which is given by:

$$\hat{V}^e(i,\sigma,t) = \sum_{\eta=0}^{\sigma} q_{i\sigma}(\eta)\,\tilde{V}^p(i,\sigma,\eta,t) + (1-F_{i\sigma}(\sigma))\max_{\delta}\left\{ \right. \tag{5.35}$$

$$\left. \begin{array}{c} -c^r(\tau^e_{i\delta}) + \sum_{\eta'=0}^{\tau^e_{i\delta}} q_{\delta\tau^e_{i\delta}}(\eta')\,\tilde{V}^p(\delta,\tau^e_{i\delta},\eta',t-\tau^e_{i\delta}) \\ +\left(1-F_{\delta\tau^e_{i\delta}}(\tau^e_{i\delta})\right)\sum_{\eta''=0}^{\infty} q_{\delta 0}(\eta'')\,\tilde{V}^p(\delta,\eta'',\eta'',t-\tau^e_{i\delta}-\eta'') \end{array} \right\}$$

where $\tilde{V}^p$ is the approximate partial value function as introduced in Section 5.4.2. The gap-values $\hat{V}^g(i,j,\sigma,t)$ can be derived in a similar manner. The advantage of this approach is that we can incorporate the actual time-to-go σ rather easily.

5.6.2 Relaxation 2: average time-to-go

In Section 5.4.2 we replaced the time-to-go σ with its expectation $\bar{\sigma}$. However, to simplify the presentation, we used a time-to-go of zero in our dynamic programming recursions. This way, we do not consider the case that jobs are won in advance, i.e., during a certain time-to-go or during a pro-active move.

If we use the average time-to-go $\bar{\sigma}$, then we have to incorporate the three cases for winning a new job: (1) during the time-to-go $\bar{\sigma}$, (2) during a pro-active move $\tau^e_{i\delta}$, and (3) after arrival at node δ. The recursion for the approximate end-values based on the average time-to-go $\bar{\sigma}$ is given in Algorithm 5.5. Here we use (5.8) to describe the value of the three cases. The algorithms for the approximate gap-values, both fixed and flexible, are derived in a similar manner.

init:
  given a planning horizon T
  $\tilde{V}^e(i,t) = 0 \quad \forall i \in \mathcal{N}$ with $t \leq 0$
for $t = 1$ to T do
  for $\forall i \in \mathcal{N}$ do

$$\tilde{V}^e(i,t) = \sum_{\eta=0}^{\bar{\sigma}} q_{i\bar{\sigma}}(\eta)\, \tilde{V}^p(i,\bar{\sigma},\eta,t) + (1 - Q_{i\bar{\sigma}}(\bar{\sigma})) \max_\delta \left\{ \begin{array}{r} -c^r(\tau^e_{i\delta}) + \sum_{\eta=0}^{\tau^e_{i\delta}} q_{\delta\tau^e_{i\delta}}(\eta)\, \tilde{V}^p(\delta, \tau^e_{i\delta}, \eta, t - \tau^e_{i\delta}) + \\ \left(1 - Q_{\delta\tau^e_{i\delta}}(\tau^e_{i\delta})\right) \sum_{\eta=0}^{\infty} q_{\delta 0}(\eta)\, \tilde{V}^p(\delta, \eta, \eta, t - \tau^e_{i\delta} - \eta) \end{array} \right\}$$

$$\tilde{V}^p(i,\bar{\sigma},\eta,t) = \sum_{k,l \in \mathcal{N}} p_{ikl}(\bar{\sigma} - \eta) \left[\begin{array}{c} \alpha_{ikl}(t)\, r_{ikl}(\bar{\sigma} - \eta) + \\ \tilde{V}^e(l, t - \tau_{ikl}) \end{array} \right]$$

  end;
end;

Algorithm 5.5: Calculating the approximate end-values using the average time-to-go $\bar{\sigma}$

5.6.3 Relaxation 3: expected opportunity costs

In Section 5.5.2 we mentioned that in order to calculate the opportunity costs, we already need the opportunity costs, see (5.29). Because we estimate all parameters (see Section 5.5) before calculating the dynamic programming recursions for the approximate value functions, we propose to use the opportunity

costs that are calculated earlier for this state (e.g. in an earlier auction round or period). So the opportunity costs are defined recursively. An alternative is to perform multiple iterations of the dynamic programming recursions. At the beginning of each iteration we calculate all parameters, using the results of the previous iteration to calculate the opportunity costs. Again we can start the first iteration using the value functions calculated in the previous auction round or period. We can stop the iteration when the difference in value-function is sufficiently small. In our simulation experiments we show some convergence results.

5.6.4 Relaxation 4: custom parameters for the gap-values

Instead of using the same parameters for the end-values as for the gap-values, we now use the intrinsic parameters for the gap-values. To do this, we have to (1) make all functions of Section 5.5 also dependent on the end-node j and the gap flexibility t and (2) use another opportunity cost function (5.30). In case of fixed contracts, these opportunity costs are given by:

$$OC_{ikl}\left(j,\sigma,t\right)=\tilde{V}^{g}\left(i,j,t\right)+c^{r}\left(\tau_{ik}^{e}\right)-\tilde{V}^{g}\left(l,j,t-\tau_{ikl}\right) \tag{5.36}$$

In principle, we can use the approximate gap-values that are calculated in an earlier auction round or period, or perform multiple iterations of the dynamic programming recursion (see previous section). Of course, we then have to store the gap-values $\tilde{V}^{g}\left(i,j,t\right)$ for all combinations of $i,j\in\mathcal{N}$ and $t=\tau_{ij}^{e}..T$. However, in case of flexible gaps, this approach is less appropriate because the value of a gap is dependent on other gaps in the schedule. As a consequence, we would have to store all possible gap-values for all possible schedules. Therefore we propose another method for both the fixed gap-values and flexible gap-values.

The basic idea is that we explicitly incorporate the estimation of opportunity costs in the dynamic programming recursion itself. More precisely we estimate all required parameters at each stage s of the dynamic programming recursions, where s is the remaining gap flexibility, by using the gap-values that are calculated in earlier stages $t<s$. For a given gap flexibility s we can estimate the opportunity costs as follows:

$$OC_{ikl}\left(j,\sigma,s\right)=\tilde{V}^{g}\left(i,j,s-1\right)+c^{r}\left(\tau_{ik}^{e}\right)-\tilde{V}^{g}\left(l,j,s-1-\tau_{ikl}\right)$$

These opportunity costs are calculated using the gap-value functions $\tilde{V}^{g}\left(i,j,t\right)$ for $t<s$, which were already calculated in earlier steps of the dynamic programming recursion.

5.7 Simulation

In the next sections we discuss the settings and numerical results of a concise simulation study. The goal of this study is to provide insight into (1) the performance of opportunity based bid pricing and scheduling and (2) the effect of different approximations. We compare the opportunity based bid pricing and scheduling strategy with a naive strategy where only the direct costs of a job insertion are taken into account. Because this naive strategy is not able to value opportunities, we use a simple waiting strategy where new jobs are scheduled as early as possible at a certain place in the schedule. For a broader comparison we refer to Chapter 3, where this naive pricing and scheduling strategy is compared with two centralized heuristics.

We consider three simplified market structures. First, an *open market* where we apply our opportunity based bid pricing and scheduling approach to a single vehicle, while other vehicles (indicated by external market) are using the naive strategy. Second, a *virtual market* which is a special form of open market where bids of the external market are generated from a given distribution function. Third, a *closed market* where all vehicles are using the same pricing and scheduling strategy. The closed market structure can be seen as an internal application of our approach. For example if we apply our approach to a closed consortium of carriers or even to a single carrier where the vehicles compete with each other.

The opportunity based bid pricing approach is developed for open markets where we apply this strategy to an individual vehicle and assume that it does not affect the behavior of the other vehicles. This is a basic assumption of our approach because current decisions (bid price calculations) are based on historical observations of auction data. For a closed market, we expect that when all players update the value functions at the same time based on the same observations, the average prices for jobs will increase. In Chapter 7 we go into more details about this.

5.8 Experimental settings

We consider unbalanced transportation networks, in the sense that some nodes are more popular than others. We consider two network settings, a 3 node network and a 9 node network. In the 3 node network, the nodes are spanning up an equilateral triangle. The distances between the nodes are 50 km. We label the nodes A, B, and C. The probability of being an origin node is 0.1, 0.3, 0.6 for node A, B, C respectively. In the 9 node network, the nodes are the grid points of a 2x2 square grid. Distances are Euclidian and such that the horizontal and vertical distances between adjacent nodes are 25 km. The probability of being an origin for nodes on row 1, is 5 times higher than row 2

and 25 times higher than row 3. Per row all probabilities are equal. In both network settings, the destination node is drawn randomly from all nodes other than the origin node.

We use 10 vehicles, each having a travel speed of 50 km/hour (both empty and loaded). The travel cost function is given by $c^r(t) = t$ and the penalty cost function by $c^p(t) = 10t$. The loading- and unloading times are 5 minutes each. For the dynamic programming recursions, we discretize time into periods of 1 minute and use a planning horizon of 12,000 minutes. Jobs have a time-window length of 10 hours and arrive according to a Poisson process. In the 3 node network, the mean interarrival time of jobs is 900 seconds. In addition, we also vary the mean time between jobs and the time-window length. For the time between jobs we use [600,700,800,900,1000] seconds. For the time-window length we consider [300,400,500,600,700] minutes. In the 9 node network we consider two settings for the mean interarrival time between jobs: quiet (1200 seconds) and busy (800 seconds).

We evaluate the approximate end-values and gap-values in combination with the relaxations of Section 5.6. We consider the following value functions:

VE	End-values based on $\sigma = 0$
VEA	End-values based on $\sigma = \bar{\sigma}$
VG	Gap-values based on $\sigma = 0$
VGC	Gap-values with custom parameters based on $\sigma = 0$
VGAC	Gap-values with custom parameters based on $\sigma = \bar{\sigma}$
VGS	Gap-values calculated recursively over all gaps in the schedule based on $\sigma = 0$
VGAS	Gaps-values calculated recursively over all gaps in the schedule based on $\sigma = \bar{\sigma}$

Relaxation 1 of Section 5.6, the precise first uncertain move, is not added as an experimental factor. As a consequence, all policies are based on the approximate values $\tilde{V}^e$ and $\tilde{V}^g$. We omitted this factor because we have seen, from our simulation experiments, that the added value of this relaxation is relatively small in all cases, i.e., a reduction in net costs per job (consisting of costs for empty moves and penalties on tardiness) of <4% in the open markets and <1% the closed markets.

For the virtual market we use a Gumbel G_1 distribution to generate the lowest bid of the external market. The parameters of this distribution are estimated based on a learning phase of 500 days. In this learning phase we have 10 vehicles using the naive pricing and scheduling strategy. At each auction round, we store the lowest bid of a fixed group of 9 vehicles (which represent the external market). At the end of the learning phase we estimate the parameters of the Gumbel distribution from the mean and standard deviation of the lowest bid data. Also, the single vehicle calculates the different value functions in advance at the end of the learning phase, except for the flexible gap-values which are defined over all gaps in a schedule.

In the open market, we have a single vehicle using opportunity based bid pricing and scheduling, and 9 vehicles using the naive strategy. Although prices from the external market are possibly influenced by the behavior of the individual vehicle, we decided to use the same value functions as in the virtual market. As a consequence, we have to calculate the value functions only once for the virtual or open market.

In the closed market, we have 10 vehicles using opportunity based bid pricing and scheduling. Because now all players can change their bid pricing behavior, we can not simply calculate all value functions in advance. Instead, we calculate them periodically, each 50 days, using the observations of the last 50 days.

We use the 3 node network to evaluate all policies in all combinations of the virtual and open market, and the fixed and flexible contracts. We further investigate the impact of varying time-window lengths and varying time between jobs, using the flexible contracts. In the 9 node network we test a selection of policies using the flexible contracts. In this network we investigate the virtual and open market, but also the closed market setting.

As performance indicators we consider the percentage of the time vehicles are driving loaded (DL), the service level (SL) being the percentage of jobs that are picked up before the latest pickup time, and the relative profit (RP). The definition of relative profit differs per market structure. In the virtual market, the relative profit of a certain policy is given by the realized profit under this policy compared to the profit of using the naive policy. In the open market, the relative profit of a certain policy is given by the realized profit under this policy compared to the average profit of the 9 vehicles that are using the naive policy. In the closed market, the relative profit of a certain policy provides the change in total net costs compared to the situation in which all vehicles are using the naive policy. Here the total net costs consist of costs for traveling empty and penalty costs. The loaded move costs are not included because they always have to be made and can not be reduced.

We use a replication / deletion approach for our simulations, cf. (Law and Kelton, 2000), where each experiment consists of a number of replications (each with different seeds) and a warm-up period. The length of each simulation run for the closed market setting is 710 days, including a warm-up period of 210 days. The length of each simulation run for the virtual and open market consists of 510 days including 10 days as a warm-up period. Here the learning phase is not included in the run length, but is done separately for multiple experiments in advance. To determine the number of replications, we consider all three performance indicators of all experiments. The maximum number of replications needed with a confidence level of 95% and a relative error of 5% is 5. To facilitate comparison, we use 5 replications for all experiments. Only the learning period consists of one replication.

5.9 Numerical results

First, we present our simulation results for the 3 node network. We subsequently present: the prices and behavior of the value functions (Section 5.9.1), the results for the virtual market (Section 5.9.2), the results for the open market (Section 5.9.3), and the effect of the input parameters (Section 5.9.4). After that, we present the results for the 9 node network in Section 5.9.5, and provide some insight into the computation times of our methods in Section 5.9.6.

5.9.1 3 node network - Learning phase

During the learning phase all vehicles use the naive pricing and scheduling policy, and we keep record of the lowest bid from a group of 9 vehicles. The mean and standard deviation of these observations can be found in Table 5.1. In case of fixed contracts, prices mainly depend on the origin node because new jobs are always appended to the end of a schedule. Prices for jobs with origin node A are the lowest because most jobs end at node A. In case of flexible contracts, prices depend both on the origin node and the destination node, because new jobs are inserted in a schedule. Therefore, the prices for jobs with origin node A (the most popular origin node) or destination node C (the most popular destination node) are relatively lower.

	Flexible contracts		Fixed contracts	
Route	Mean	St.dev	Mean	St.dev
AB	62.89	19.37	70.01	1.30
AC	23.10	24.80	70.04	1.80
BA	78.42	20.97	80.79	23.06
BC	49.47	35.80	80.84	22.80
CA	122.44	19.89	110.51	28.09
CB	118.73	24.03	110.86	27.96

Table 5.1: Mean and standard deviation of lowest bids for jobs on different routes during the learning phase

Next, we calculate the approximate value functions for the flexible contracts. To calculate the end-values we perform 10 iterations (see relaxation 3). First, we calculate the end-values based on a time-to-go of zero (Algorithm 5.1). After that, we calculate the end-values based on an average time-to-go $\bar{\sigma} = 180$ minutes (Algorithm 5.5). The results can be found in Figure 5.10 and Figure 5.11.

From these figures we draw the following conclusions:

- For a longer planning horizon, we see a linear increase in end-values while the absolute differences between end-values for the different schedule destinations remain the same. This makes sense because, for t large, the

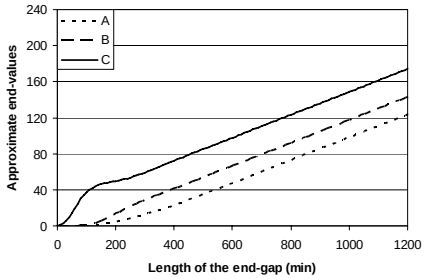

Figure 5.10: Approximate end-values for the flexible contracts using an average time-to-go of zero

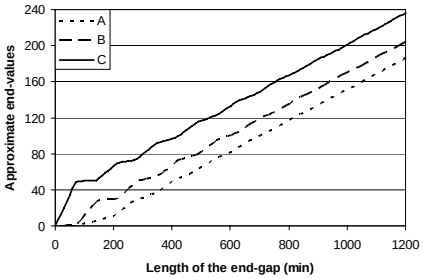

Figure 5.11: Approximate end-values for the flexible contracts using an average time-to-go of 180 minutes

expected profits at period $[t, t+1]$ are independent on the start-node at $t = 0$ because you visited many nodes afterwards. The absolute differences in end-values for various schedule destinations, emerge at small t because the initial increase depends on the popularity of the schedule destination (see next point).

- Initially, the increase in end-value for start-node C is higher then the linear slope mentioned above (and the opposite for start-node A). We have the following explanation. The value of a node is given by the expected profit of the first job we won, plus the value of the node where the new job ends. For start-node C, the expected waiting time is small and the probability of an empty move is low. So the expected profit of the first job is relatively high. However, this new job ends at node A or node B, which has a lower value. The initial slope is higher because we only consider the short-term profits, i.e., without taking into account the value of the node where the new job ends.

- The end-values based on an average time-to-go (Figure 5.11) have more visible fluctuations. These fluctuations are cyclic and appear to be flattening out. The cycle period depends on the time to move loaded from one node to another. To illustrate this, let us consider the start-node C. The waiting time at this node is almost zero, and therefore profits immediately increase with increasing gap length. At a gap length of 70 minutes, we expect to do precisely one job from C to A or B. The expected profit of this job is approximately 50. The next 70 minutes, the expected profits do not increase much, because the expected waiting times and the probability of empty moves are higher at the new start-node.

- The end-values are much higher when we take into account the average time-to-go of 180 minutes. The explanation is that the expected waiting times are much lower because, on average, jobs can be won 3 hours in advance. The slope with $\bar{\sigma} = 0$ and $\bar{\sigma} = 180$ is approximately 0.127 and

0.193 respectively. As we will show later on, the latter provides a better match with the realized profits. However, an interesting result here is that the absolute differences in end-values of the different start-nodes are more or less equal in both situations. As a consequence, the differences in end-values for the various locations are the same with both figures. The advantage of using $\bar{\sigma} = 0$ in this case is that we do not have to estimate the average time-to-go and it requires less computation time (see Section 5.9.6).

To provide insight into the convergence of end-values, we calculate the slope of the approximate end-values in each iteration. Here the slope is calculated as the average difference between the values of the largest end-gap and the one but largest end-gap: $\sum_{\forall i \in \mathcal{N}} \tilde{V}^e(i,T) - \tilde{V}^e(i,T-1)$. The results can be found in Figure 5.12. Clearly, the value functions converge very fast, even though we started with end-values of zero.

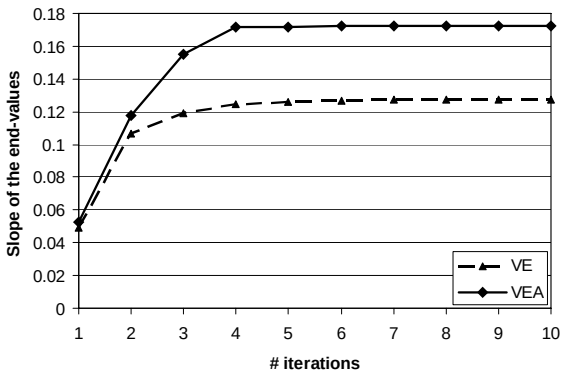

Figure 5.12: Convergence in the slope of the end-values

Next, we calculate the approximate gap-values for the flexible contracts. We calculate the gap-values for all possible gaps, skipping those for which the start-node equals the end-node because these values are always zero. To illustrate the value functions, we ignore the dependencies between gaps, so in fact we only consider gaps for which there are no subsequent gaps. The shape of the gap-value functions for gaps earlier in a schedule is more or less the same; the absolute values are slightly lower because new job insertions in these gaps reduce the value of subsequent gaps, see (5.19).

First, we calculate the gap-values using the same parameters as for the end-values. After that, we calculate the gap-values using their own parameters (see relaxation 4). The results can be found in Figure 5.13 and Figure 5.14.

From these figures we draw the following conclusions:

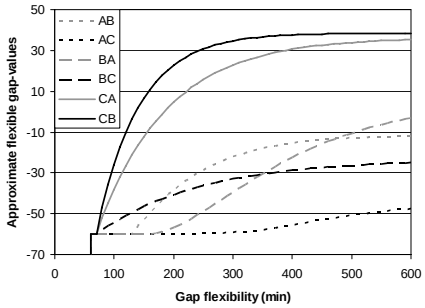

Figure 5.13: Approximate gap-values for the flexible contracts using the same parameters as for the end-values

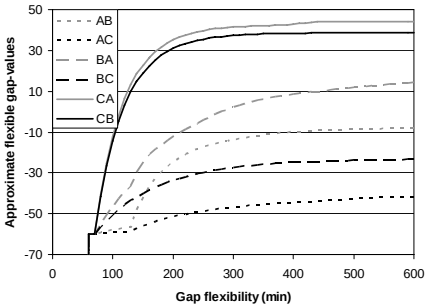

Figure 5.14: Approximate gap-values for the flexible contracts using custom parameters

- The gap-values are minus infinity (see Section 5.4.4) if the gap flexibility is lower than the time required for an empty move from the start-node towards the end-node (60 minutes).
- The gap-values are equal to the costs for an empty move (60) when the gap flexibility is just large enough to make an empty move towards the end-node.
- For a large gap flexibility, the order of gap-values for the different start-nodes equals the order of end-values.
- All gap-values converge to some constant value. The explanation is that when we use more flexibility (because we insert a new job, wait at this start-node, or move pro-actively towards a node other than the end-node) the value we gain with this increase is wiped out by the resulting decrease in end-values.
- The gap-values are slightly higher when we use custom parameters for the gap-values. As mentioned in Section 5.6, the custom parameters for the gap-values only affect the opportunity costs in (5.30). Given the second price auction, these opportunity costs do not affect the prices we receive, but only the probabilities of winning certain jobs. If we use the same parameters for the gap-values as for the end-values we sometimes win less profitable jobs, resulting in lower gap-values.
- Following the argumentation of the previous point, we see that the ordering of gap-values in Figure 5.13 is not always logical. For example, the gap CB appears to be better than the gap CA, while more jobs end at node A. The reason for this is that we are using the same parameters for the gap-values as for the end-values. This means that we are working with bid prices that do not take into account the gap restrictions, see (5.29).

Because the end-value of the start-node B is higher than the end-value of the start-node A, the opportunity costs for a job towards node A are also higher. This reduces the possibility of winning a job towards node A, and hence the gap-value of gap CB is higher. If we use custom parameters for the gap-values (Figure 5.14) this ordering is as can be expected.

- An interesting result here is that some gap-values can become positive. So for a certain node i, it is sometimes better to insert a job with origin not equal to i (because a gap ii has value zero). For example, let us compare the gap CC with a gap CA. If we have a gap CC, then we always pickup the new load at the end-node C, immediately after arrival at the start-node C. This way we have the longest possible end-gap. If we have a gap CA, then we possibly have to move empty from the start-node C towards the end-node A. This empty move will costs 60, but it also reduces the end-gap with 60 minutes. But still we see that the gap CA, with a gap flexibility of 600 minutes, has a positive value of 44. The explanation is as follows. Most jobs depart from node C and most jobs end at node A. So the probability that the empty move CA can be replaced by one or more loaded moves is very high. But also in the end-gap, the probability of a job ending at node A is very high. Once we arrive at node A (the most unpopular origin node), it is likely that we have to wait or have to make an empty move. However, this is not the case in the gap CA, because once we arrive at a node A, we already have a job with origin node A that immediately can be started. So basically, a job with a relative unpopular origin node serves as an escape option for cases at which you end at such a node.

Next, we calculate the approximate end-values for the fixed contracts. The results can be found in Figure 5.15 and Figure 5.16. Again, the slope is higher if we use the average time-to-go $\bar{\sigma}$, but the absolute differences are approximately the same.

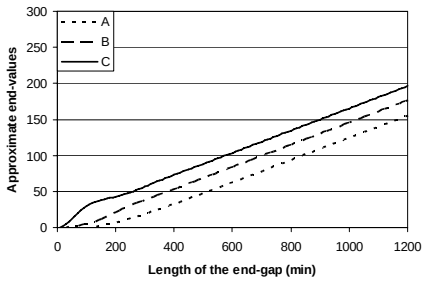

Figure 5.15: Approximate end-values for the fixed contracts using an average time-to-go of zero

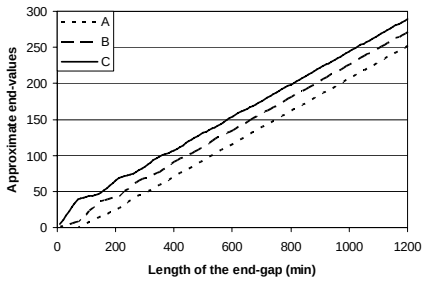

Figure 5.16: Approximate end-values for the fixed contracts using an average time-to-go of 180 minutes

We also calculate the approximate gap-values for the fixed contracts. First, we calculate the gap-values using the same parameters as for the end-values. After that, we calculate the gap-values using their own parameters (see relaxation 4). The results can be found in Figure 5.17 and Figure 5.18.

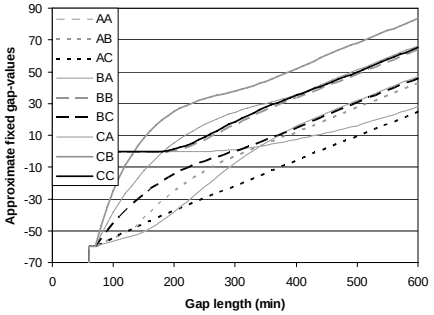

Figure 5.17: Approximate gap-values for the fixed contracts using the same parameters as for the end-values

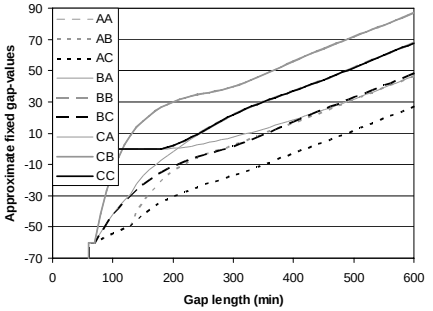

Figure 5.18: Approximate gap-values for the fixed contracts using custom parameters

We see some similarities between the fixed and flexible gap-values. The major difference is that for larger gaps, the gap-values increase linearly. The slope of this function equals the slope of the end-values. The explanation for this is that for a large gap, the increase in profit due to an increase in gap length, is less sensitive to the gap restrictions. The behavior of the gap-values is therefore similar to the end-values. We further see that the nine gap-value functions tend to four linear trends. The absolute differences between these linear trends are almost the same as the difference in end-values for different start-nodes. However, it appears that this behavior is not inherent to the fixed gap-values; it is the result of using observations of the learning period (see Table 5.1) in combination with the continuous distribution to describe the lowest bid. In Chapter 7 we go into more details on this.

For both fixed and flexible contracts, we also calculate the gap-value functions based on a average time-to-go $\bar{\sigma} = 180$. These figures (not depicted here) have more visible fluctuations and higher values compared to the gap-value functions based on a zero time-to-go. The differences between the lines are more or less the same. The latter is caused by the fact that in this 3 node network, a vehicle never decides to make a pro-active move. The pro-active move decisions can easily be determined from the given figures. For example, consider the end-values if Figure 5.16. If we make a pro-active move from the most unpopular node (node A) to the most popular node (node C), we go from an end-value $\tilde{V}^e(A, 1200) = 252.9$ to $\tilde{V}^e(C, 1140) = 276.2$, but we lose a value of 60 for the empty move. So even for this case we will lose 36.7.

As mentioned before, the error in value functions due to the zero time-to-go not necessarily means that we also make the wrong decisions. For example, if

we have to weigh the value of two possible schedule destinations, the end-values based on a zero time-to-go provide almost the same opportunity costs as those based on the average time-to-go. However, things go wrong when we weigh the value of a certain period against the costs for an empty move. Because the expected profits based on a zero time-to-go are much lower, we often prefer to wait for a new job instead of making an empty move. Especially in case of fixed contracts this is an issue. For example consider Figure 5.18. The optimal gap lengths can easily be determined from Figure 5.18 by subtracting a linear function with a slope given by the end-values (see Figure 5.16) from all gap-values. The results can be found in Figure 5.19. For all gaps with start-node different from the end-node, it appears to be optimal to postpone the pickup time for the next job (with origin equal to the end-node of the gap) with a time longer than the time required for a loaded move. Clearly, the zero time-to-go approximation is not appropriate in case of fixed contracts, as we will show in the next two subsections.

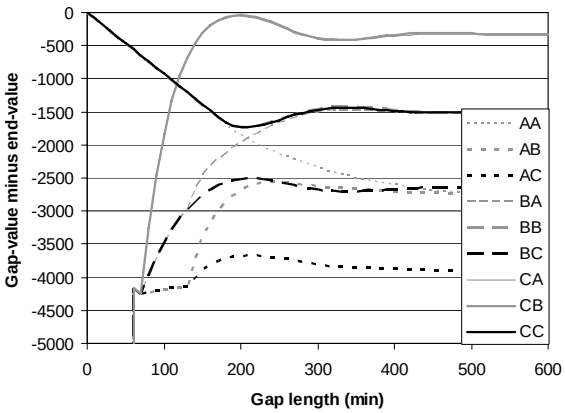

Figure 5.19: Value of various gap lengths

5.9.2 3 node network - Virtual market

Based on the data from the learning phase, we now generate the lowest bid from the competitors of the individual vehicle. For both contracts, fixed and flexible, we evaluate various policies and approximations. The results can be found in Table 5.2.

From these results, we draw the following conclusions:

- The opportunity valuation policies always lead to an increase in performance with respect to the three performance indicators.

Contract	Policy	DL	SL	RP
Flexible	Naive	79.5	95.1	0
	VE	89.4	99.0	56.9
	VE+VG	91.5	99.0	70.0
	VE+VGC	91.8	99.0	71.3
	VE+VGS	91.9	99.0	72.9
	VEA	89.4	99.8	57.6
	VEA+VGAS	92.0	99.3	74.1
Fixed	Naive	74.9	96.9	0
	VE	89.8	99.5	86.2
	VE+VG	81.7	98.0	2.6
	VE+VGC	82.4	98.4	4.2
	VEA	91.6	99.7	87.2
	VEA+VGAC	92.0	99.7	89.1

Table 5.2: Simulation results for the virtual market in the 3 node network

- In case of flexible contracts, a more precise policy (adding gap-values, an average time-to-go, custom gap-parameters, gap dependencies) always leads to a better performance, with the only exception that the service level with the policy VEA is the highest.
- In case of fixed contracts we see a similar behavior, although using the gap-values based on a zero time-to-go, has a negative impact on the performance, especially with respect to the relative profit. The reason for this is that the individual vehicle underestimates its revenues. The consequence, as already mentioned in Section 5.9.1, is that the pickup times of jobs are postponed more than they should be.
- A remarkably result is that the relatively simple policies perform very well. In both cases, the policy VE results already in a major increase in performance. In case of flexible contracts, the policy VE+VG performs very well. The advantage of these approximate policies is that they work much faster, see Section 5.9.6.

To explain the performance of the individual vehicle in case of flexible contracts, let us consider the route between C and B. Theoretically, 15% of the jobs from a vehicle go from B to C and 30% from C to B. Under the naive policy we see that for the individual vehicle, 15.4% of the jobs go from B to C and 24.5% go from C to B. Under the policy VEA+VGAS, these numbers are 37.3% and 39.4% respectively. Obviously, this results in fewer empty miles for the individual vehicle and hence in higher profits.

Also in case of fixed contracts, the individual vehicle aims at the more profitable jobs. For example, with the policy VE, 68.9% of the jobs from the individual vehicle are on the route between B and C. If we use the gap-values,

then most 'opportunistic gaps' are created with C as start-node, and A or B as end-node. With opportunistic gaps we mean that the pickup time of the new job, given a certain insertion position, is not scheduled as early as possible. Instead, the pickup time is postponed in anticipation of new loaded moves which may replace the empty move. Using the policy VE+VG, then 24.3% of the jobs are scheduled with opportunistic gaps, and 24.7% of the jobs are inserted in these gaps. So in some opportunistic gaps, more than one new job is inserted. Using the policy VEA+VGAC, then 7.3% of the jobs are scheduled with opportunistic gaps, and 6.9% of the jobs are inserted in these gaps. Clearly, less opportunistic gaps are created when we use the average time-to-go. In fact, it appears that almost all opportunistic gaps are filled with a new job which replaces the empty move by a loaded move.

5.9.3 3 node network - Open market

Here we compare the performance of the individual vehicle with the other 9 vehicles. For the performance indicators DL and SL, we first show the results of the individual vehicle, and after the slash we show the average result of the other 9 vehicles. We further use the relative profit to denote the profit of the individual vehicle compared to the average profit of the other 9 vehicles. The results can be found in Table 5.3.

Contract	Policy	DL	SL	RP
Flexible	naive	69.7 / 70.0	99.7 / 99.8	−3.2
	VE	86.4 / 67.9	99.1 / 99.9	560.8
	VE+VG	85.6 / 67.5	98.9 / 99.8	726.2
	VE+VGC	86.4 / 67.3	98.8 / 99.8	730.3
	VE+VGS	86.8 / 67.3	98.9 / 99.9	762.7
	VEA	88.5 / 67.8	100.0 / 99.8	575.7
	VEA+VGAS	90.6 / 67.6	98.2 / 99.8	769.7
Fixed	naive	68.3 / 68.6	99.9 / 99.9	−0.2
	VE	87.3 / 65.7	98.2 / 99.9	545.4
	VE+VG	81.4 / 66.5	97.7 / 99.9	230.4
	VE+VGC	81.9 / 66.5	97.8 / 99.9	214.5
	VEA	94.4 / 64.8	99.6 / 99.9	545.9
	VEA+VGAC	94.8 / 64.3	98.7 / 99.9	577.3

Table 5.3: Simulation results for the open market in the 3 node network

The results are similar to those of the virtual market. However, the percentage of driving loaded of the individual vehicle is slightly lower, and the relative profit is much higher. The latter can be explained because now not only the individual vehicle is better off, but the others are worse off because the most profitable jobs are taken out by the individual vehicle. This can also be seen from the average percentage of driving loaded of the other 9 vehicles, which

decreases when the individual vehicle is using an opportunity valuation policy instead of using the naive policy.

The difference between the different policies and approximations is almost the same. Again, the approximations perform very well, but the more precise formulations perform better. In case of flexible contracts, we see that the profit of the individual vehicle is up to 9 times higher than the average profit of the other 9 vehicles. So the profit of the individual vehicle is almost equal to the total profit of the other 9 vehicles.

The explanation for the performance of the individual vehicle is similar to that given in case of the virtual market. The major distinction is that now the other 9 vehicle suffer from the individual vehicle picking out the most profitable jobs. For example, with the policy VE, 68.5% of the jobs from the individual vehicle are between B and C (the two most popular origin nodes), whereas the other 9 vehicles have on average 41.8% of their jobs on the route between B and C, which is less then the average total of 45% of jobs on this route.

5.9.4 3 node network - Varying some parameters

In these experiments we vary the average time between jobs and the time-window length of jobs. For different settings, we study the relative profit of the individual player using opportunity based bid pricing and scheduling compared to the naive pricing strategy. In the virtual market setting, this profit is relative to the profit under the naive policy. In the open market setting, this profit is relative to the average profit of the 9 other vehicles, which are using the naive policy.

First, we show the results for the virtual market settings, see Figure 5.20 and Figure 5.21. Clearly, the relative profit increases with an increasing number of jobs (smaller time between jobs) and with increasing time-window length. The reason is that (1) with increasing number of jobs, the individual vehicle has more chances to select the most profitable jobs and (2) with increasing time-window length the individual vehicle has more scheduling flexibility to reduce empty moves. Furthermore, we see that the added value of using the gap-values increases with increasing number of jobs and increasing time-window length.

Next, we show the results for the open market settings, see Figure 5.22 and Figure 5.23. Again, we see a similar behavior for varying time-window lengths. However, the relative profit as a function of the time between jobs is quite different. In fact, we see an opposite effect because relative profits decrease with an increasing number of jobs. The reasons that the results are different than those of the virtual market are (1) the behavior of the individual vehicle has an effect on the 9 other vehicles and (2) all jobs have to be transported by the 10 vehicles. The decrease in relative profit of the individual vehicle with decreasing time between jobs is caused (1) by the increase in penalties

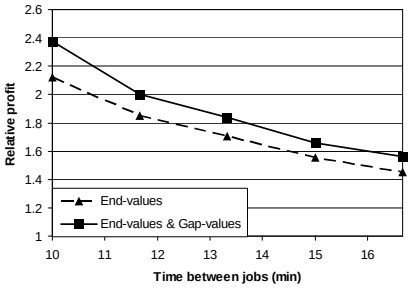

Figure 5.20: Relative profit for varying time between jobs in the virtual market setting

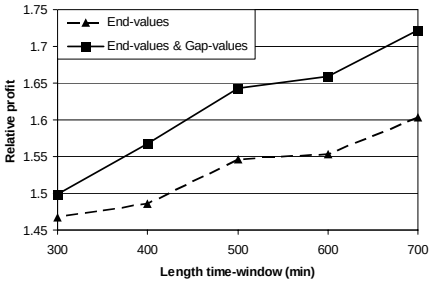

Figure 5.21: Relative profit for varying time-window lengths in the virtual market setting

which may dominate the opportunity costs and (2) by the increase in vehicle utilization for the other 9 vehicles. Also, fewer jobs means lower profits for all vehicles. However, because the individual vehicle picks out the most profitable jobs from the few jobs offered, the 9 other vehicles suffer even more from this. Consider, for example, the largest time between jobs (1000 seconds). Here the profit for the individual vehicle is more than 5% higher than the sum of the profits for the 9 other vehicles.

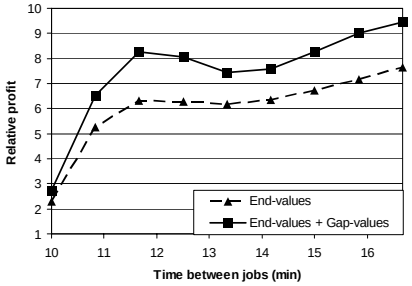

Figure 5.22: Relative profit for varying time between jobs in the open market setting

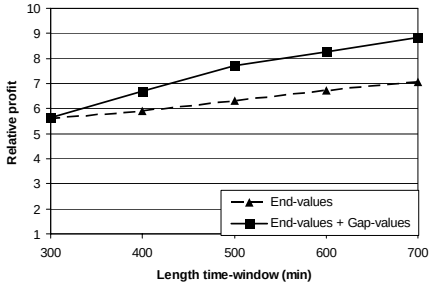

Figure 5.23: Relative profit for varying time-window lengths in the open market setting

5.9.5 9 node network

Next, we consider the 9 node network using the virtual market, the open market, and also the closed market setting. In the closed market setting, all 10 vehicles are using the same policy. We consider the end-values and gap-values based on the average time-to-go. Because these value functions require considerably more computation time in the 9 node network, we decided to ignore the dependencies between gaps and therefore use the gap-values VGAC. The

results of using these value functions in the 9 node network can be found in Table 5.4.

Market	Policy	Network	DL	SL	RP
Virtual	Naive	Quiet	63.8	94.3	0
		Busy	64.0	92.9	0
	VEA	Quiet	75.5	100.0	79.5
		Busy	78.3	99.9	96.2
	VEA+VGAC	Quiet	75.7	100.0	98.6
		Busy	78.5	100.0	103.4
Open	VEA	Quiet	69.6 / 61.5	99.7 / 98.8	395.0
		Busy	71.1 / 62.2	99.5 / 98.1	254.7
	VEA+VGAC	Quiet	71.9 / 60.9	99.6 / 99.0	822.1
		Busy	73.8 / 61.8	99.5 / 98.4	498.0
Closed	Naive	Quiet	62.6	98.4	0
		Busy	63.0	97.7	0
	VEA	Quiet	64.8	99.7	10.1
		Busy	64.8	99.5	8.9
	VEA+VGAC	Quiet	65.0	98.7	11.1
		Busy	65.0	98.5	10.1

Table 5.4: Simulation results for the 9 node network

The results for the virtual and open market are more or less the same as in the 3 node network. The opportunity valuation policies always lead to an increase in performance. In some cases, the profit of the individual vehicle is more than the sum of the profits for the 9 other vehicles. We further see that the percentage of driving loaded is most of the time higher in the busy networks because there are more opportunities to avoid empty moves. The service levels are most of the time lower in the busy networks. With respect to the relative profit of the individual vehicle, we see that in the virtual market the relative profit is higher in the busy network whereas in the open market the relative profit is higher in the quiet networks. This corresponds with our observations in the 3 node network (cf. Figure 5.20 and Figure 5.22).

In the closed network settings, the opportunity valuation policies also perform very well. The total costs for driving empty and tardiness are reduced by 11.1% by using the policy VEA+VGAC in the quiet network setting. In the virtual and open networks, the benefits of opportunity valuation are merely caused by selecting the most profitable jobs. This no longer holds in closed environments because all vehicles use the same policy and all jobs have to be transported. So the only explanation is that vehicles are scheduling the jobs more efficiently. To illustrate this, let us consider the policy VE+VGC. Here jobs from the first row to the last row, are scheduled 5.5% later in time, and jobs from the last row to the first row 19.7% earlier in time. So jobs with a higher probability of an empty move afterwards are scheduled later in time,

increasing the probability that the empty move is replaced by a loaded move. We also see that the average schedule length (difference between the expected delivery time of the last job and the current time) is decreased by 9.3%. This in turn provides more flexibility to schedule jobs on time.

5.9.6 Computation times

The computation times required for the insertion scheduling heuristic and marginal costs calculation for bid pricing are very small. The major concern here is the computation time required to calculate the approximate value functions. At most, these value functions have to be calculated upon each announcement and each time we make a pro-active move decision. However, we can also calculate these functions periodically; this certainly saves a lot of time. Particularly because the end-values already have to be calculated for all possible schedule-destinations and the gap-values for all possible start-nodes. If we calculate the gap-values periodically, we only face extra computation time because we have to calculate them for all possible end-nodes and for the longest possible gap (e.g. the longest time-window length). Only the values VGAS have to be calculated upon each announcement because they depend on the whole schedule.

Also the two relaxations, the average time-to-go and the custom parameters for the gap-values, have an impact on the computation times. First, consider the average time-to-go. If we compare Algorithm 5.5 with Algorithm 5.1, we see that we have to calculate the partial value function TN^2 times in Algorithm 5.1 whereas we have to do it approximately $3T^2N^2$ times in Algorithm 5.5. Next, consider the custom parameters for the gap-values. If we use custom parameters, we have to calculate them for each remaining gap length s. So the computation time for parameter estimation will be increased by T.

For our experiments we used the simulation software eM-Plant 7.5 and an Intel Pentium 4 processor at 3.4 GHz. The computation times required to calculate the value functions for the different policies are given in Table 5.5. The end-values are calculated for the whole planning horizon of 12,000 minutes. The gap-values are calculated in advance for all possible end-nodes and a maximum gap length of 600 minutes.

Policy	3 nodes	9 nodes
VE	0.35	15.19
VEA	1.12	129.21
VG	0.05	8.79
VGA	0.16	75.52
VGC	0.15	23.46
VGAC	0.46	200.47

Table 5.5: Computation times in seconds

Clearly, the computation time increases drastically when going from 3 nodes to 9 nodes. However, this increase is much smaller in the approximations where we are using a time-to-go of zero. In a real-time application with a large number of nodes, we can overcome the relatively large computation times by offline calculation of the value functions, like we did in our simulation experiments. Given the insertion scheduling strategy, we are now able to calculate bids and schedule jobs within milliseconds.

5.10 Conclusions

In this chapter we presented an opportunity based bidding concept. This method makes it possible to determine the value of a schedule, more specifically, the value of gaps in the schedule and the value of a schedule destination. This enables us to calculate the opportunity costs, which are defined as the loss in expected future revenues due to a new job insertion. By including this value in our bid price, we not only cover the direct costs of a new job insertion, but also its future implications. Moreover, by using this value in our scheduling decisions, we prevent less profitable moves and increase our opportunities by better prepositioning of vehicles.

From our simulation experiments, we conclude that an individual player using opportunity based bid pricing and scheduling performs significantly better than other players who use a naive pricing strategy. For example, in an open environment with 10 vehicles we showed that the profit of the individual vehicle is in some cases higher than the total profit of the 9 other vehicles. Besides the profitability of opportunity based bid pricing and scheduling, we also see an increase in vehicle utilization and service levels. For closed environments (internal use of our approach), we showed that opportunity based bid-pricing and scheduling can reduce the system-wide logistical costs (an average a reduction of 10% in the costs for empty moves and penalties on tardiness). We further have seen that the approximations (a time-to-go of zero and gap-values using the same parameters as the end-values) perform remarkably well in most cases. The advantage of using the approximations is that they require less computation time and are better scalable to larger systems.

We explain the benefits of opportunity based bid pricing and scheduling from the following behavior. First, the vehicle agents tend to schedule unattractive jobs later in time, thereby increasing the probability of combining these job with other jobs. Second, in case of flexible contracts, the average schedule length is reduced such that there is more flexibility of scheduling jobs on time. Third, in case of fixed contracts, the vehicle agents tend to create gaps before unattractive jobs, thereby reducing the possibilities of empty moves. Fourth, in case of open markets, the individual vehicle using opportunity based bid pricing and scheduling tend to select out the most profitable jobs.

In this chapter we focused on the carriers and their vehicles. In the next chapter, we focus on strategies for the shippers. After that, we combine these two chapters in Chapter 7, by considering a transportation market where both shippers and carriers are using look-ahead strategies for scheduling, bid pricing, and bid evaluation.

Chapter 6

Shippers: dynamic threshold policies

In this chapter[1] we consider a transportation procurement auction consisting of shippers and carriers. Shippers offer time sensitive pickup and delivery jobs and carriers bid on these jobs. We focus on revenue maximizing strategies for shippers in sequential auctions. For this purpose we propose two strategies, namely delaying and breaking commitments. The idea of delaying commitments is that a shipper will not agree with the best bid whenever it is above a certain reserve price. The idea of breaking commitments is that the shipper allows the carriers to break commitments against certain penalties. We evaluate the benefits of both strategies with simulation. In addition, we provide insight into the distribution of the lowest bid, which is estimated by the shippers.

6.1 Introduction

The procurement of transportation is an important task for shippers because it greatly affects their costs and service levels. In practice, a procurement process includes carrier screening, carrier assignment, load tendering, and performance review. During the last few years, this procurement has moved from telephone to web based services (Song and Regan, 2001). In this chapter we consider an automated transportation procurement auction where shippers offer time sensitive pickup and delivery jobs and carriers bid on these jobs. Each auction is initiated by a shipper and ends with commitment between the shipper and a carrier.

[1] This chapter is based on the working paper (Mes, Van der Heijden and Schuur, 2007); presented at TRISTAN VI, the Sixth Triennial Symposium on Transportation Analysis, Thailand; conditionally accepted for publication in Transportation Research Part C.

Auctions are often considered as appropriate means for dynamic job allocation in distributed environments. Desirable properties include limited required information exchange and, in some cases, Pareto efficient outcomes (McAfee and McMillan, 1987; Wellman et al., 2001). However, when multiple jobs are auctioned at different points in time (sequential auctions), such an allocation would be less appropriate if we do not take into account the future consequences of an allocation. Also, when jobs are complementary (e.g. transportation jobs that can be served sequentially by the same vehicle) or substitutable (e.g. transportation jobs that are available at the same time), a certain allocation may become unfavorable when new jobs appear. To overcome this, several authors (e.g., Caplice and Sheffi, 2003) suggested the use of *combinatorial auctions* for transportation procurement in which a carrier can bid on multiple jobs. However, combinatorial auctions involve many inherently difficult problems. As mentioned by Song and Regan (2005), we face the bid construction problem, where bidders have to compute bids over different job combinations, and the winner determination problem, where jobs have to be allocated among a group of bidders. In addition, (1) it may be unrealistic to bundle jobs which belong to different shippers and (2) these procedures are not directly applicable in situations where jobs arrive at different points in time.

To improve the allocation of jobs, we take the sequential transportation procurement auction as given, and focus on strategies for the participants. In the previous chapter, we focused on profit maximizing strategies for the carriers. We proposed a bid pricing strategy where the arrivals of future jobs are taken into account through the use of opportunity costs. In this chapter, we focus on strategies for the shipper. We propose two options, namely delaying and breaking commitments.

The idea of delaying commitments is that a shipper postpones commitments for which it expects to make better commitments in the future. So if a shipper has plenty of time to auction a certain job, it will not agree with a relatively high bid. When the time for dispatch becomes nearer, the price it is willing to accept will rise. We denote this mechanism by *dynamic threshold policy*. The idea of breaking commitments is that the shipper allows a carrier to decommit from an agreement against a certain penalty. These penalties are chosen such, that whenever a carrier decommits a job, they cover the expected extra costs for finding a new carrier. We denote this mechanism by *decommitment policy*.

There is some strategic equivalence between the dynamic threshold policy and the decommitment policy. Suppose a carrier wins a certain job. After two days the carrier decommits from the contract and pays a certain penalty. One can imagine that this penalty equals the expected costs for the shipper for auctioning this job two days later, which in turn has some strong connections with the dynamic threshold policy. We develop a dynamic programming algorithm that can be used for both, the dynamic threshold policy and the decommitment policy.

The goal of this chapter is threefold. First, to derive a dynamic threshold policy and a decommitment policy. Second, to evaluate their benefits in terms of (1) costs savings for the shippers and (2) system-wide logistical costs. Third, to study their relation, i.e., are they complementary or substitutable?

6.2 Literature

Our problem is related to several research areas, such as operations management (transportation), economic theory (optimal auctions), and mathematical theory (optimal stopping). In the next sections we describe the relation of this chapter with these research areas and describe our contribution.

6.2.1 Transportation

The dynamic allocation of transportation jobs belongs to the large class of dynamic fleet management problems. A few representative examples of this stream include (Carvalho and Powell, 2000; Godfrey and Powell, 2002; Yang et al., 2004). This research mostly focuses on real-time vehicle routing strategies for a single carrier. Here we focus on the shipper, and more specifically on shippers that procure transportation services using auctions. A transportation procurement auction consists of three steps: (1) bid preparation by the shipper (who may bid on what), (2) bid pricing by the carriers, and (3) bid analysis by the shipper. The majority of research on transportation procurement auctions focuses on the bid preparation step (Caplice and Sheffi, 2003). Others focus on bid pricing, scheduling, and routing decisions of the carriers (see Figliozzi et al., 2003). Less attention has been paid to the shipper.

Traditionally, a shipper allocates transportation jobs to carriers one-by-one, i.e., through sequential auctions. Such a system ignores the interdependencies between subsequent jobs. A significant portion of the trucking industry costs is due to the repositioning of empty vehicles from the destination of one load to the origin of a subsequent load (Song and Regan, 2002). Interdependencies occur because serving one job is greatly affected by the opportunity to serve another job. To cope with these dependencies, Caplice and Sheffi (2003) suggested to use combinatorial auctions. As demonstrated by Ledyard et al. (2002), the benefits of combinatorial auctions to shippers can be significant. A survey on combinatorial auctions for the procurement of transportation services can be found in (Sheffi, 2004). For reasons as mentioned in the introduction, we choose here for sequential one-shot auction procedures. To cope with the interdependencies among jobs, we propose the dynamic threshold policy and the decommitment policy.

6.2.2 Optimal auctions

The design of auction mechanisms that maximize the seller's expected revenue, called optimal auctions, received a great deal of attention. For an extensive literature survey on this topic we refer to (McAfee and McMillan, 1987). Part of this work focuses on reserve prices in sequential auctions. As shown by Myerson (1981), the reserve price increases the expected revenue of the seller by preventing the object from being sold at a low price. Closely related is the work of (McAfee and Vincent, 1997) who study the optimal reserve-price path in a sequence of first- and second- price auctions. In particular, the auctioneer puts the same object for sale repeatedly, until it is sold. At each round he chooses a reserve price according to his (increasingly pessimistic) beliefs about the buyers' valuations.

A crucial assumption in the optimal auction literature is that each bidder's valuation is known to be drawn from a common distribution (Bose et al., 2006). We do not make any assumptions on the valuation functions of the individual bidders, but instead estimate the distribution of the lowest bids. In addition, we incorporate correlations in bids, together with a regression on time and several job characteristics. Another difference is that we consider reverse (procurement) auctions.

Another line of research within the economic auction literature focuses on decommitment. In automated negotiation systems, contracts have traditionally been binding. Such contracts do not allow agents to efficiently deal with future events when contracts might become unfavorable. To overcome this, a leveled commitment protocol is introduced in (Sandholm and Lesser, 2001). Here an agent can decommit (for whatever reason) simply by paying a decommitment fee to the other agent. It is shown, through game-theoretic analysis, that this leveled commitment feature increases the Pareto efficiency of contracts and can make contracts more beneficial for both parties. The efficiency of such protocols depends heavily on how the penalties are decided. Therefore, (Sandholm et al., 1999) developed algorithms for optimizing contracts in terms of prices and penalties. Penalty functions for sequences of multiple leveled commitment contracts are studied in (Andersson and Sandholm, 2001). They conclude that penalties as a percentage of the contract price, increasing in decommitment time, perform best.

Another example can be found in ('t Hoen and La Poutré, 2004) who apply the decommitment concept to a multi-agent transportation setting. They conclude that significant increases in profit can be achieved when the agents can decommit and postpone the transportation of a load to a more suitable time. Their setting differs from ours in that we consider a full truckload problem where bid prices - and hence the decommitment penalties - are dependent on time. Another difference is that in their setting exchange of jobs is only allowed between vehicles of the same carrier, so carriers are not allowed to decommit from a contract with a shipper. In fact, upon decommitment, a virtual auction

is held at the carrier to find a new vehicle for the decommited job.

In this chapter we provide a formal expression for time-dependent decommitment penalties by drawing a parallel with the reserve prices.

6.2.3 Optimal stopping

The choice of whether to accept the lowest bid in a sequential auction is related to the so-called optimal stopping problems. A famous example of an optimal stopping problem is the secretary problem. In the classical secretary problem, a decision-maker has to hire one applicant out of a pool of n applicants who appear sequentially. The decision-maker must decide immediately upon seeing an applicant whether to hire him. For a historical overview of the classical secretary problem we refer to (Ferguson, 1989). The classical secretary problem is called a no-information problem in which the distribution of offers is unknown. The stopping decision is only based on the relative ranks of the observations and not on their actual values. This is obviously different from our case because shippers are able to learn the pricing behavior.

A number of authors have established the existence and properties of optimal stopping policies when the offer distribution is known in advance, and the offer rate is periodic. For example, Karlin (1962) studied the problem of selling an asset. Our approach differs from this line of research in the sense that we consider (1) historic auction information to update the offer distribution, (2) time-dependent offers, (3) correlation between subsequent offers, and (4) the finite horizon problem as a special case. For more information on optimal stopping problems we refer to (Chow et al., 1971).

6.2.4 Contributions

To summarize the previous sections, our contribution consists of the following:

1. We develop cost minimizing strategies for shippers in transportation procurement auctions. To avoid combinatorial complexities, we propose a dynamic threshold policy that enables the shippers to strategically delay commitments or set decommitment penalties for the carriers.

2. We determine optimal reserve-price paths for reverse auctions using probability distribution functions for the lowest bid based on historical data. These distributions do not depend on the iid assumption of bids or on the bid price distributions of individual bidders.

3. We provide a formal expression for time-dependent decommitment penalties by drawing a parallel with the reserve price paths.

4. We aim at a wider applicability than competitive procurement auctions. In particular, we aim at closed environments, i.e., allocation to a closed group of trusted carriers, or for auction procedures that are commonly used in multi-agent systems for resource allocation. Therefore, we provide a performance evaluation, using simulation, not only in terms of individual benefits for the shippers, but also in terms of the system-wide logistical performance.

6.3 Model

We consider a transportation market consisting of shippers and carriers. Shippers offer jobs by starting a transportation procurement auction and carriers bid on these jobs. One job involves the transport of a unit load (full truckload). We define a job by an announcement time a, an origin i, a destination j, and as a soft restriction a latest pickup time l of the load at the origin. Tardiness with respect to the latest pickup time is penalized with c per time unit. We introduce a soft time-window length $\sigma = l - a$ within which transportation should be started. We introduce d as the distance between the origin and destination of a job, also indicated by job length. Note that for notational convenience we omit job indices.

Objective of a shipper is to minimize the price paid to a carrier for transporting a certain load. Shippers consider the prices as random variables with probability distributions which can be estimated based on historic data. These distributions are characterized by a time-dependent mean and standard deviation, and by correlations in prices between subsequent auction rounds.

In a transportation procurement auction, shippers typically put out a request for quotes from a set of carriers (Song and Regan, 2002). This process is similar to a simple sealed-bid auction in which each bidder submits a sealed bid for a single item. We choose here for a reverse first-price sealed-bid auction in which the lowest bidder receives his bid amount, given the shipper does not reject all bids.

We implement the market mechanism as follows. When a job arrives at some shipper, it starts an auction by sending an announcement to all carriers. In return, each carrier responds with a bid. Without a dynamic threshold policy, the shipper sends a grant message to the carrier with the lowest bid while the others receive a reject message. When using a dynamic threshold policy, a shipper might expect to receive a better bid in the future. After all, prices fluctuate over time due to changes in the available transportation capacity and in the transportation schedules. So if the best bid is relatively high (which can be learned from history) it might be better to wait for more attractive prices. In this case, the shipper only selects a winner whenever the lowest bid is below a certain threshold level. Otherwise, the auction stays open (Section

6.4.1) or the shipper will start a new auction some period later (Section 6.4.2). As the deadline for dispatch comes nearer (or is already reached), the shipper increases its threshold to get transportation. Basic assumption here is that we consider only one job at a time. So threshold prices of jobs are independent of the current set of open jobs.

In case of decommitment, a vehicle is allowed to decommit from a job by paying some penalty to the shipper (Section 6.5). In this case the shipper immediately starts a new auction for this job.

6.4 Dynamic threshold

First, we develop a continuous time threshold function (Section 6.4.1). This function can be used in a continuous auction where a shipper waits until a bid drops below the threshold price (bidders' take-it-or-leave-it prices). Next, we develop a basic policy for repeated auctions in discrete time (Section 6.4.2). In this structure, the shipper will start a new auction a fixed auction period later when the best bid is above its threshold price. This structure can be considered as an approximation for the continuous case if the auction period is small. In Section 6.4.3 we incorporate possible correlation in bid prices between subsequent auction rounds.

6.4.1 Basic threshold policy for continuous auctions

To illustrate the theoretical benefits of a dynamic threshold policy, we consider a continuous time model under some simplifying assumptions.

Whenever a shipper becomes aware of a new job, it will start an auction for this job. All vehicles immediately bid on this job, but may update their bid at any time. The shipper, of course, is only interested in the lowest bid. We assume independent and identically distributed lowest bids which can be described by a continuous distribution function $F(b)$. The time between subsequent updates of the lowest bid is exponentially distributed with rate λ. At each update of the lowest bid, the shipper has to decide whether to accept the current lowest bid. We only consider the period before the latest pickup time. If the job is not sold before this time, the shipper will face costs $Z(\tau) = \beta + c\tau$, where β is a constant, c the penalty costs per time unit, and τ the tardiness with respect to the latest pickup time (see Appendix for a formal derivation of this cost function).

We indicate the time until the latest pickup time by a time-to-go t. We introduce the value function $V(t)$ as the minimum expected price a shipper has to pay eventually, given a time-to-go t. For $t < 0$, the value $V(t)$ is given by the value $Z(-t)$ of auctioning the job after its latest pickup time. The

recursive relationship which characterizes the finite horizon minimum expected price, as a function of the time-to-go t, is given by:

$$V(t) = E[\min(B, V(t-Y))] \tag{6.1}$$

with B the stochastic variable for the lowest bid and Y the exponentially distributed time until the next bid update.

Karlin (1962) showed that there exists an optimal policy that accepts the first bid B whose value satisfies $B > E[V(t-Y)]$. So we only accept a bid whenever it is lower than the minimum expected price at the expected time of the next update of the lowest bid. We rewrite this policy by using a threshold function $\alpha(t)$, which is given by:

$$\begin{aligned} \alpha(t) &= E[V(t-Y)] \\ &= E[\min(B, \alpha(t-Y))] \end{aligned} \tag{6.2}$$

Integration over all bid prices B and time between bid updates Y gives the following:

$$\begin{aligned} \alpha(t) = &\int_0^t \left(\int_0^{\alpha(t-y)} b dF(b) + \alpha(t-y) \int_{\alpha(t-y)}^{\infty} dF(b) \right) \lambda e^{-\lambda y} dy + \\ &\int_t^{\infty} Z(y-t) \lambda e^{-\lambda y} dy \end{aligned} \tag{6.3}$$

Following an approach similar to (Karlin, 1962), we rewrite this function in the form of a differential equation. Partial integration of the first integral yields:

$$\alpha(t) = \int_0^t \left(\alpha(t-\xi) - \int_0^{\alpha(t-\xi)} F(b)\, db \right) \lambda e^{-\lambda \xi} d\xi + \left(\beta + \frac{c}{\lambda} \right) e^{-\lambda t} \tag{6.4}$$

Next, we replace ξ by $t-x$, and multiply both sides by $e^{\lambda t}$:

$$\alpha(t) e^{\lambda t} = \lambda \int_0^t \left(\alpha(x) - \int_0^{\alpha(x)} F(b)\, db \right) e^{\lambda x} dx + \left(\beta + \frac{c}{\lambda} \right) \tag{6.5}$$

Differentiation by t, together with the boundary condition $\alpha(0)$, yields:

$$\begin{aligned} \alpha'(t) &= -\lambda \int_0^{\alpha(t)} F(b)\, db, \text{ with } t \geq 0 \\ \alpha(0) &= \beta + \frac{c}{\lambda} \end{aligned} \tag{6.6}$$

We can solve this differential equation numerically for different distributions $F(b)$ of the lowest bid. Since a uniform distribution allows an analytical

derivation, let us assume that $F(b) = \frac{b}{\omega}$, for $0 \leq b \leq \omega$. We further take $\omega = \beta + \frac{c}{\lambda}$, so at the latest pickup time the shipper expects to pay the highest possible bid. Then we have the following threshold function:

$$\alpha(t) = \frac{2\omega}{\lambda t + 2} \tag{6.7}$$

If we get only one chance to auction a job, the expected price equals $\omega/2$. The relative savings $s(t)$ depending on the time-to-go t, by using the threshold policy, are therefore given by:

$$s(t) = \left(1 - \frac{4}{\lambda t + 2}\right) \cdot 100\% \tag{6.8}$$

To illustrate the savings take $\lambda = 1$, then for $t = 2, 8, 18, 38$ the savings are respectively given by $0\%, 60\%, 80\%, 90\%$.

Although these results are promising, they only hold under the assumption of independent and identically distributed lowest bids. In many cases, this assumption is not realistic. In the next two subsections we extend our results to time-dependent and correlated bid prices.

6.4.2 Basic threshold policy for repeated auctions

Here we incorporate a more realistic pricing behavior, by taking into account the time-dependency of bids and the correlation between bids. To do so, let us discretize time. The resulting dynamic threshold function may then be used as an approximation for the continuous case. However discretization may also be part of the market structure because basically the continuous auction is replaced by a repeated auction. In a repeated auction, we have a series of auctions used to sell the same object in subsequent periods.

For the timing between successive auction rounds we take a fixed period R. In contrast with the previous section, we express the timing of an auction in terms of auction round numbers instead of the time-to-go. Numbering starts at the first auction round at the announcement time of a job.

In each auction round, the shipper has to decide whether to accept the lowest bid. Before rejecting all bids, the shipper has to calculate the probability of receiving a better bid in the future. To do so, we assume that the lowest bid b as a function of the auction round n can be described by a continuous distribution function $F_n(b)$. The shipper estimates this distribution based on historical auction data. In Section 6.6 we illustrate this estimation process for a specific network instance which we also consider in our simulation experiments (Section 6.8).

We only consider auction rounds between the announcement time a and the latest pickup time l. Therefore, the maximum number of auction rounds

is given by $N = \lfloor (l-a)/R \rfloor + 1$. Again, we use a cost function $Z(\tau)$ (see Appendix) for the expected price after the latest pickup time. We introduce the shorthand notation Z to denote the expected price of the first auction round after the latest pickup time.

Depending on the auction period R, there is a probability that the lowest bid remains the same in the next auction round. Therefore, we introduce an update probability q^u that describes the probability that the lowest bid is updated (at least one time) between two successive auction rounds. So for a Poisson updating process with rate λ (see previous section), this probability is given by $q^u = 1 - e^{-\lambda R}$. We assume that the shipper is able to estimate this update probability, independent of the underlying updating process.

We introduce the value function $V_n(b_n)$ as the expected price a shipper has to pay eventually (in this or one of the remaining auction rounds) given a lowest bid b_n in the current auction round n. We use B_n as the stochastic variable for the lowest bid in auction round n. To get an optimal strategy, we deduce the optimum decision numbers by working backward from the last possible auction round. We get the following:

$$\begin{aligned} V_N(b_N) &= \min\{b_N, Z\} \\ V_n(b_n) &= \min\{b_n, E[V_{n+1}(B_{n+1}|B_n = b_n)]\} \end{aligned} \tag{6.9}$$

In the last auction round we have to choose between the current bid b_N and the expected price Z for auctioning the job after the latest pickup time. In all other rounds we have to choose between accepting the current bid b_n or reject it and expect a price $E[V_{n+1}(B_{n+1}|B_n = b_n)]$ later on. The price we expect to accept later on depends on the current lowest bid, because there is a probability $1 - q^u$ that we receive the same lowest bid. Therefore, we have to take into account the current lowest bid in the threshold prices. The threshold price in auction round n equals the expected price we accept in auction round $n+1$ or later, given the current lowest bid b_n:

$$\begin{aligned} \alpha_n(b_n) &= E[V_{n+1}(B_{n+1}|B_n = b_n)] \\ &= (1-q^u)\min\{b_n, \alpha_{n+1}(b_n)\} + q^u \int_0^\infty \min\{b, \alpha_{n+1}(b)\}\, dF_{n+1}(b) \end{aligned} \tag{6.10}$$

We solve this equation by backwards dynamic programming. To do so, we have to include the bids in the state space. Therefore, we discretize the bids, add the current lowest bid to the state space, and iterate on the auction round n:

$$\begin{aligned} \alpha_n(b_n) &= (1-q^u)\min\{b_n, \alpha_{n+1}(b_n)\} + \\ &\quad q^u \sum_{b=0}^{L} P_{n+1}(b)\min\{b, \alpha_{n+1}(b)\} \end{aligned} \tag{6.11}$$

where $P_n(b)$ is a discretization of $F_n(b)$ and L is chosen large enough.

To illustrate the behavior of the threshold prices, we calculate (6.11) for some parameter settings. We use time-to-go $\sigma = 10$, a Poisson updating rate $\lambda = 1$ (so $q^u = 1 - e^{-R}$), and we take $Z = \infty$ (so in the last auction round we always accept the lowest bid). For the probability $P_n(b)$ of the lowest bid, we discretize a normal distribution to integer values for the bids b. We calculate the expected price $E[V_n(B)] = E[\min\{B, \alpha_n(B)\}]$ as a function of the auction round n. We perform two numerical experiments: in Example 6.1 we evaluate the impact of the mean and standard deviation and in Example 6.2 we evaluate the impact of the auction period.

Example 6.1. *Here we use an auction period $R = 1$ and evaluate three scenarios for the mean and standard deviation of the lowest bid as a function of the auction round n:*

- *Constant mean of 100 and standard deviation of 50*
- *Increasing mean of 5 per auction round, starting at 50 in the first round, and constant standard deviation of 50*
- *Constant mean of 100 and increasing standard deviation of 3 per auction round, starting at 20 in the first round*

The results can be found in Figure 6.1. We see that prices increase with increasing auction rounds. Starting with a low mean results in lower threshold prices, which then increase relatively faster. Starting with low standard deviation results in higher threshold prices because we are less able to profit from the variation in bid prices. But because variances increase, this threshold function also increases relatively slower. In the last auction round we always accept the lowest bid. Therefore, in the last auction round, all three prices equal 100.

Example 6.2. *Here we evaluate the threshold prices for varying lengths of the auction period R. For the distribution of the lowest bid we use a constant mean and standard deviation of respectively 100 and 50. Going from $R = 1$ to $R = 0.2$ results in 5 times more auction rounds. Going from $R = 0.2$ to $R = 0.04$ again results in 5 times more auction rounds. From Figure 6.2 we conclude that the added value of extra auction rounds clearly decreases, which is in fact a good argument to use this threshold policy as an approximation for the continuous case.*

6.4.3 Incorporating correlation

Bid prices of carriers fluctuate due to new job arrivals. In the previous section we have developed a basic dynamic threshold policy that takes advantage of

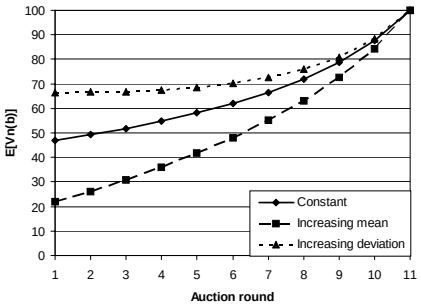

Figure 6.1: Impact of the time between jobs on the expected threshold price

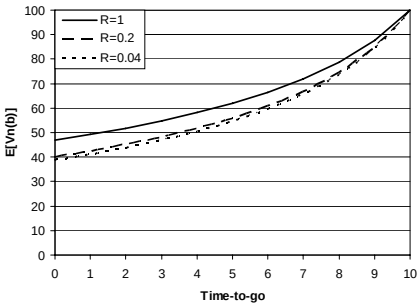

Figure 6.2: Impact of the auction period on the expected threshold price

these fluctuations. We have seen that, for given job characteristics, these prices depend on the auction round and the time between subsequent auction rounds. Another important aspect, which we did not consider so far, is correlation between subsequent lowest bid updates. Suppose the current lowest bid is relatively high, indicating that all carriers are busy at the moment. When the shipper re-auctions the job a short time period later, and it receives a new lowest bid, it is likely that this bid is also relatively high. In this section we deal with these correlations between subsequent auction rounds.

Because a part of the correlation is caused by the time dependency of bids, we consider correlation between the deviations from expected lowest bids. We introduce $\delta_n = b_n - E[B_n]$ for the price deviation in auction round n. We only consider correlations between two successive auction rounds (no time-lags) and assume that the correlation between two successive auction rounds n and $n+1$ does not depend on the number n (so the correlation between round 1 and 2 is similar to the correlation between rounds 5 and 6). We further assume that the deviations can be described by a linear trend with coefficient ϕ:

$$\delta_{n+1} = \phi\delta_n + \epsilon_{n+1} \tag{6.12}$$

where ϵ_{n+1} is the error term.

To incorporate the impact of a price deviation in auction round n on the lowest bid in auction round $n+1$, we simply increase the mean price $E[B_{n+1}]$ in auction round $n+1$ by $\phi\delta_n$. Using (6.11), we derive the following function for the threshold prices with correlated bids:

$$\begin{aligned}\alpha_n(b_n) &= (1-q^u)\min\{b_n, \alpha_{n+1}(b_n)\} + \\ &\quad q^u \sum_{b=0}^{L} P_{n+1}(b)\min\{b+\phi(b_n - E[B_n]), \alpha_{n+1}(b+\phi(b_n - E[B_n]))\}\end{aligned} \tag{6.13}$$

The derivation of ϕ is presented in Section 6.6. To illustrate the impact

of correlation on the dynamic threshold policy, we perform another numerical experiment.

Example 6.3. *Here we use the same settings as in Example 6.2, but now with a constant auction period $R = 1$ and varying correlation factors ϕ. From Figure 6.3 we conclude that a large correlation has a negative impact on the threshold prices. After all, 'free' fluctuation decreases, resulting in a similar situation as in Example 6.1 with smaller variances.*

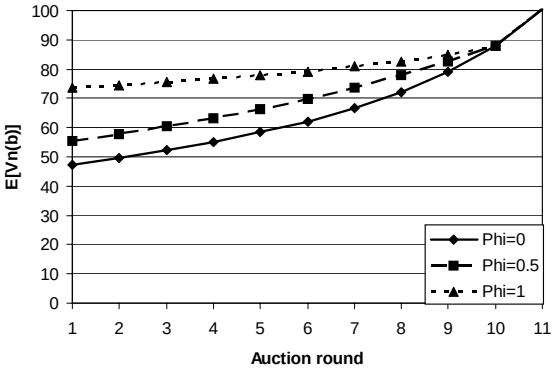

Figure 6.3: Impact of the correlation factor

As mentioned in the beginning of Section 6.4.2, the threshold policy for repeated auctions can be used as an approximation for continuous auctions. To determine the continuous threshold price $\alpha_t(b)$, depending on a time-to-go t and current bid b, we use linear extrapolation between the threshold price in the last auction round before t and the first auction round after t. If there are no remaining auction rounds after t, we simply use the threshold value Z. Otherwise ($t \geq \sigma - (N-1)R$), we use the following continuous threshold value:

$$\alpha_t(b) = \alpha_n(b) + \left(\frac{\sigma - t}{R} - \left\lfloor \frac{\sigma - t}{R} \right\rfloor \right) (\alpha_{n+1}(b) - \alpha_n(b)) \tag{6.14}$$

Here $\alpha_n(b)$ refers to the original threshold price in auction round n given the current bid b, where n is given by $n = \left\lfloor \frac{\sigma - t}{R} \right\rfloor + 1$. The auction period R should be small enough, such that the probability of more than one lowest bid update within this period is 'low'. For $t < 0$, i.e., auctioning the job after its latest pickup time, $\alpha_t(b)$ is given by $Z(-t)$, see Appendix.

6.5 Decommitment

A second trick that can be used by shippers is to allow carriers to break commitments against certain penalties. The decommitment penalties are set by the shipper and are publicly available to the carriers at all times. These penalties should cover the extra costs for a shipper to find a new carrier. A shipper will face extra costs because the time until latest pickup time is shorter now. Here we assume that shippers are risk neutral in the sense that they only calculate the expected extra costs.

If the shipper uses the threshold prices (repeated auctions), then these extra costs are given by the difference in threshold prices as presented in Section 6.4. The decommitment penalty $D_{s,t}$, for a job committed at a time-to-go s (so a time s before the latest pickup time) and decommitted at a time-to-go t, is given by:

$$D_{s,t} = E\left[\alpha_t\left(B_t\right)\right] - E\left[\alpha_s\left(B_s\right)\right] \tag{6.15}$$

where $t < s$, and B_t is the stochastic variable for the lowest bid with a time-to-go t.

A carrier decommits from a job whenever its expected revenue of inserting a new job and removing the decommitted job, is higher than the current expected revenue plus the decommitment penalty. Note that in case of decommitment, a carrier will not receive its bid price for the decommitted job.

When the shipper is not using the threshold prices, it simply uses one auction round to find a new carrier for the decommited job. Then the decommitment penalty simply equals the expected difference in price between the initial commitment time s and the current time t:

$$D_{s,t} = E\left[B_t\right] - E\left[B_s\right] \tag{6.16}$$

With (6.15) and (6.16) we have derived a formal expression for the leveled commitment penalties, only by using the threshold prices. The applicability, however, is quite different because now the decision has to be made by the carriers.

6.6 Parameter estimation

In the previous sections we illustrated the behavior of the dynamic threshold policy using numerical experiments. Obviously, the circumstances of these experiments are ideal in the sense that the shipper has perfect knowledge about the bid price distribution $F_n(b)$, the update probability q^u, and the correlation ϕ. In practice, the shipper has to estimate these parameters, for example, using historical auction data. This estimation is a major issue, especially in

experimental settings that are often considered in the vehicle routing literature (i.e., visitors are drawn randomly from a square area), as we shall illustrate in Section 6.8.

In this section we describe how a shipper can estimate the parameters for the dynamic threshold policy. We proceed in a goal oriented way because we feel that it is not possible to estimate the parameters without having some knowledge about the application area. To this end we make the following assumptions:

1. The origin and destination of jobs are chosen randomly from the Euclidian plane.
2. Carriers use a bid function that is linearly dependent on the increase in travel distance and the increase in penalty costs that are required for the new job.

Throughout this chapter, we use a single distribution to describe the lowest bid. However, given the second assumption, bids consist of two cost components: transportation costs and penalty costs. These cost factors differ in their dependence on the time-to-go t and distance d, in the update probability, and in the correlation between subsequent auction rounds.

The most elegant way to deal with these two cost factors is to explicitly incorporate them in the dynamic programming recursion. We then use multiple (linear) regression, and a separate update probability and correlation factor for both costs components. After all, a shipper should have knowledge on both cost factors regardless whether or not they are included in the bid prices. In addition, we may treat the tardiness as left censored data and perform Tobit regression (Tobin, 1958) to estimate the tardiness and derive the expected penalty costs from this.

We propose an alternative method based on the observation that once a shipper receives a bid with penalties, it is very unlikely that it will receive a lower bid in the future. Therefore, we remove the penalties from the bids b and include them separately in our dynamic threshold policy. So from this point on, b represents only the transportation costs in the lowest bid. The distribution of these costs is given by $F_n(b)$ and $F_t(b)$, for repeated and continuous auctions respectively. In this section we choose for the continuous representation. Obviously, a variable depending on the time-to-go t can easily be translated into to a variable depending on the auction period n (see Section 6.4.3). Further note that we use the subscript d to indicate the dependence on the distance d. For clarity of presentation we omit this subscript in the dynamic threshold function $\alpha_t(b)$ and distribution function $F_t(b)$.

We determine the distribution $F_t(b)$ of the transportation costs in the lowest bid using the method of moments. We describe the first two moments of this

distribution by a linear trend as a function of the time-to-go t and distance d. Given assumption 2, the transportation costs depend linearly on the distance d. It is also reasonable that there is a monotone increase in transportation costs in case of a decreasing time-to-go. If carriers are, for example, using an insertion heuristic, a decreasing time-to-go will mean that the lowest bid is derived from a smaller set of possible insertion positions (of all vehicles from all carriers). Therefore, we also assume a linear dependency on the time-to-go t. But also the variance in transportation costs depends on the time-to-go t and distance d. In our experiments (see Section 6.8) we show that also the variances can be properly expressed as a linear function of the time-to-go and distance. The mean μ^w_{td} and the standard deviation σ_{td} for the distribution $F_t(b)$ are then given by:

$$\mu^w_{td} = \alpha_w + \beta_w t + \gamma_w d \tag{6.17}$$

$$\sigma_{td} = \alpha_\sigma + \beta_\sigma t + \gamma_\sigma d \tag{6.18}$$

To determine the linear trend for the standard deviation, we divide the time-to-go t and distance d in discrete blocks, and calculate the residual variance in each of these blocks. To incorporate heteroscedasticity, we use weighted least squares to estimate the mean transportation costs. Here we give points with lower variance a greater statistical weight. To be more precise, we multiply each residual with a weight equal to the inverse of the variance σ^2_{td}.

The penalty costs are mainly affected by the time-to-go. In our experiments we have seen that this dependency can be described by a linear trend. In fact one can argue that for decreasing time-to-go, penalties approach a linear function with slope c: the penalty costs per time unit. Therefore we also describe the mean penalties μ^p_{td} by a linear trend:

$$\mu^p_{td} = \alpha_p + \beta_p t + \gamma_p d \tag{6.19}$$

We determine this trend by least squares where we only use observations in which penalties are greater than zero.

To separate the penalties from the transportation costs, we introduce q^p_t as the probability of having a lowest bid with non-zero penalties, given a time-to-go t. Once we have penalties, we simply accept the lowest bid because we expect that bid prices only increase from this point on. To determine the probability q^p_t of non-zero penalties, we divide the time-to-go t in discrete blocks and estimate the probability q^p_t for each of these blocks. Next, we fit a continuous function through these probabilities. From our simulation experiments we have seen that the Weibull survival function ($q^p_t = e^{-(t/\lambda)^k}$) is a good candidate. After applying the log function two times over the survival function, we are able to estimate the parameters using ordinary least squares.

We can incorporate the different costs components into the threshold function, simply by using the probabilities q^p_t of non-zero penalties. To illustrate

this we return to the case of repeated auctions for a given job length. So we use the index n instead of t, and omit the index d. Using (6.13), we derive the following function for the threshold prices:

$$\begin{aligned}\alpha_n(b_n) &= q_n^p(b_n+\mu_n^p)+(1-q_n^p)\left\{(1-q_n^u)\min\{b_n,\alpha_{n+1}(b_n)\}\right. \\ &\quad \left. +q^u\sum_{b=0}^{L}P_{n+1}(b)\min\left\{\begin{array}{c} b+\phi(b_n-\mu_n^w), \\ \alpha_{n+1}(b+\phi(b_n-\mu_n^w)) \end{array}\right\}\right\} \quad (6.20)\end{aligned}$$

To illustrate the impact of penalties on the dynamic threshold policy, we perform another experiment.

Example 6.4. *Here we use the same settings as in Example 6.2, but now with an auction period* $R = 1$. *The expected penalties in the last auction round are* 100 *and they decrease with* 20 *in each auction round. We consider varying parameters for the Weibull survival function. The case with* $\lambda = 0$ *coincides with the case without any penalties. Obviously, as can be seen from Figure 6.4, penalties have a negative impact on the threshold prices.*

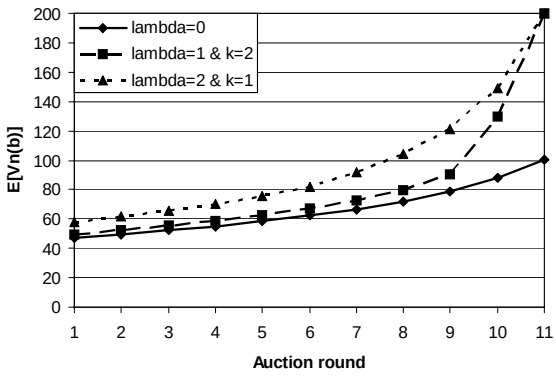

Figure 6.4: Impact of the parameters λ *and* k *from the Weibull distribution for the probability of having non-zero penalties on the expected threshold price*

To incorporate the correlation in price deviation between subsequent auction rounds, we model the trend in price deviations by an AR(1)-process. Here the expected deviation in auction round n can be expressed as a linear function of the deviation in auction round $n-1$ (see Section 6.4.3). Since we treat each pair δ_n, δ_{n-1} equally for all n, we also assume that the variance in price deviation remains the same for all auction rounds n. Then the coefficient ϕ

equals the correlation coefficient:

$$\phi = \frac{Cov\left(\delta_n, \delta_{n-1}\right)}{Var\left(\delta_n\right)} \tag{6.21}$$

Given the dynamic programming recursion for the threshold prices and the methodology to estimate the required parameters, we now ready to evaluate the impact of delaying and breaking commitments in a simulation experiment.

6.7 Experimental settings

We simulate a transportation procurement market where shippers offer transportation jobs to a set of carriers. Selection of a carrier for a certain job is only based on price and delivery time. Transportation takes place in a square area of 100x100 km, and we have 10 vehicles which drive at a speed of 50 km/hour. At the start of each simulation, these vehicles are placed at a random position in the square area.

Because our focus is on profit maximizing strategies for a shipper, we use a simple pricing and scheduling strategy for the carriers. In the remainder we speak in terms of individual vehicles and ignore the carriers. Each vehicle maintains a list of jobs. These jobs are carried out as soon as possible, and a job in process cannot be interrupted. Upon announcement of a new job, the vehicle evaluates the insertion of this job in its job list, i.e., without altering the relative ordering of jobs already in the list (cf. the insertion scheduling heuristics mentioned in Chapters 3 till 5). The bid price for this job is given by the marginal costs of the cheapest insertion. These costs are 1 per hour travel time and 10 per hour tardiness. If a vehicle wins an auction, it uses the cheapest insertion position for the new job.

In case of decommitment, a vehicle also evaluates the impact of inserting the new job while decommitting a job already in its job list. To avoid combinatorial difficulties, we only consider decommitment of a single job. For all jobs coming from a shipper that allows decommitment, the vehicle temporarily removes the job and evaluates the insertion of the new job in its reduced job list. Again, the bid price of a vehicle is given by the marginal costs of the cheapest insertion, including possible decommitment penalties.

The simulation study consists of three parts. In the first part we examine the distribution of the lowest bid. In the second part we evaluate how well the statistical model of Section 6.6 can be used to describe the lowest bid. After these two experiments we are able to describe the distribution of the lowest bid, as a function of the time-to-go and distance. We use this distribution in the third part to calculate the threshold prices. Therefore, the first two parts can be regarded as the learning phase for the third part. In the third part we evaluate a transportation market where a single shipper uses the dynamic

threshold policy and the decommitment policy. We compare the performance of this shipper to other shippers, referred to as the external market. To describe the relative size of the individual shipper compared to the external market, we introduce the notion of market share. The market share describes which portion of the incoming jobs belongs to the individual shipper. Jobs that come from the external market are always rewarded to a vehicle in the first auction round and decommitment is not possible for these jobs.

It is important to note that delaying and breaking commitments will have an effect on the market prices. To be precise, the allocation decision for one job will have effect on the future bids for other jobs (given we are dealing with a limited number of vehicles). Therefore, we have to be careful with how we generate jobs during the learning phase. To this end we decided to make a distinction between learning jobs and regular jobs. The regular jobs are exactly the same as those from the external market in the third part of this simulation study. These jobs are auctioned under a naive policy and the origin and destination of these jobs are chosen randomly from the square area. The learning jobs are only used to gain insight into the distribution of the lowest bid. To avoid influencing the market prices, we do not award these jobs to vehicles. Both job types (regular jobs and learning jobs) appear according to a Poisson process and have a time-window σ of 10 hours.

In the next sections we describe the three parts of this simulation study in more detail.

6.7.1 Part 1: price distribution

Here we want to gain insight into the distributional form of the lowest bid as a function of the job length and the time-to-go. Therefore, we consider fixed lengths for the learning jobs which we auction at several time instances. The lengths of these jobs are chosen randomly from {40,60,80,100,120} km. To determine the distribution for a specific job length and time-to-go, we have to fix the position of the job in some way. Therefore, we place each job on a diagonal and center it on the middle. To be more precise, the origin and destination coordinate of the job are located on the same diagonal, and the distance between each of these coordinates and the center equals half the predefined job length. The direction and diagonal are chosen randomly. The first auction moment of each job is 10 hours before the latest pickup time. The auction period between successive auction rounds is 2 hours and the latest auction round is 2 hours after the latest pickup time. So we auction each job 8 times. After the latest auction round, we remove the job without rewarding it to a vehicle. We use the lowest bid data for the learning jobs to evaluate the distribution characteristics for the lowest bid. The arrival rate of both job types (learning jobs and regular jobs) is 6 per hour.

6.7.2 Part 2: price evolution

Here we want to investigate how the distribution of the lowest bid depends on the job length and time-to-go. Therefore, we consider various lengths and auction moments for the learning jobs. The origin and destination coordinates are chosen randomly from the square area, just like the regular jobs. However, we use two auction rounds for each learning job. The first auction round starts at a random time between zero and ten hours before the latest pickup time. The second round starts a random time between zero and 1 hour later. After the second round, the job is removed. We use the lowest bid data for the learning jobs to estimate the functions of Section 6.6. For the arrival rate of both job types, we consider 5 per hour (quiet), 5.5 per hour (normal), and 6 jobs per hour (busy).

6.7.3 Part 3: transportation market

To calculate the threshold functions, we use the distribution function and time dependent first moments found in the previous experiments. For the arrival rate of jobs we use the same values as in part 2 of our simulation study. The auction period R is 0.5 hour; so given the time-window of 10 hour, a shipper may use up to 21 auction rounds for a single job.

This third part of our simulation study consists of five subparts.

1. Evaluation of the theoretical benefits of the dynamic threshold policy using an offline numerical experiment. With offline we mean that we do not consider vehicles bidding their marginal costs, but instead we draw a lowest bid according to the given distribution function. In fact, this is an ideal situation where the shipper has complete knowledge about the distribution of the lowest bid. In all the other experiments mentioned below, the bid prices are not given in advance but are based on the marginal costs of a job insertion in a vehicle schedule.

2. Evaluation of the impact of the dynamic threshold and the decommitment policy on the logistical costs. We apply these policies both separately and in combination. We consider a market share of 1% and 10%.

3. Evaluation of the impact of the dynamic threshold and the decommitment policy on the computation time per job. Here we use the same settings as in subpart 2.

4. Evaluation of the impact of the input data on the logistical costs by using the dynamic threshold policy. We investigate changes to the following parameters that are estimated at the end of the learning phase: the correlation factor ϕ, the update probability q^u, the constant transportation

costs α_w, and the constant penalty costs α_p (see Section 6.6). We multiply each parameter with the following factors $[0.8, 0.9, 1.0, 1.1, 1.2]$. We use a normal job arrival rate (5.5 per hour) and a market share of 1%.

5. Evaluation of the dynamic threshold and the decommitment policy in closed environments. In a closed environment we have one shipper with its own fleet of vehicles (so a 100% market share). Again, we use the estimated data from the first two experiments. However, as mentioned before, the estimated data from the learning phase is not representative for closed environments because market prices are affected by the use of the dynamic threshold and the decommitment policy. Therefore, we also evaluate an alternative where the estimates are updated periodically. To this end we introduce learning periods. At the end of each learning period, the shipper estimates all parameters using the historical data of at most 3 learning periods. Again, the parameters are estimated using the methodology of Section 6.6.

6.8 Simulation

Here we present the numerical results corresponding with the three experimental settings described in the previous section. We end in Section 6.8.4 with a summary of the simulation results.

6.8.1 Part 1: price distribution

To gain insight into the behavior of the lowest bid, we consider distribution parameters such as the mean, standard deviation, and skewness of the lowest bid. In this section we are mainly interested in the skewness. The mean and standard deviation of the lowest bid as a function of the time-to-go and distance, are investigated in Section 6.8.2.

To describe the skewness of the lowest bid, we perform a simulation experiment consisting of 10 replications with different seeds. In each replication we generate 100,000 jobs, including a warm-up period of 100 jobs. The number of replications corresponds with a confidence level of 95% with a maximum relative error of 5% for the skewness. The results can be found in Figure 6.5 for the transportation costs and in Figure 6.6 for the penalty costs.

From these figures we see that penalties always have a positive skew and skewness decreases with decreasing time-to-go. In case of a large time-to-go, the transportation costs for short jobs have negative skew while long jobs have positive skew. After the latest pickup time all jobs have negative skew for the transportation costs.

The skewness of the lowest bid distribution can partly be explained using

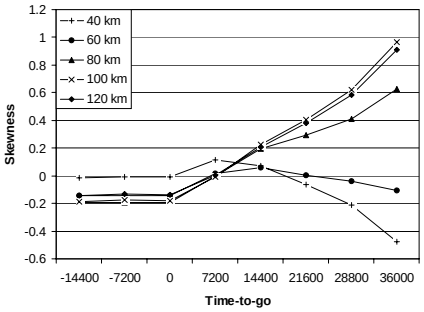

Figure 6.5: Skewness in transportation costs

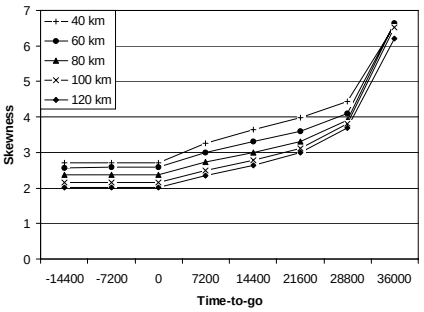

Figure 6.6: Skewness in penalty costs

the extreme value theory (EVT). This theory describes the order statistics (k^{th} order statistic is the k^{th} smallest value) of a large set of random observations from the same (arbitrary) distribution (cf. Chapter 5, Section 5.5.1). In our case we are interested in the distribution of the lowest bid out of n bids (with $n = 10$, which is of course not a large set). Extreme value theory states (see Fisher and Tippett, 1928) that for $n \to \infty$, the distribution of the lowest bid is (1) Weibull whenever the parent distribution is bounded from below and (2) Gumbel whenever the parent distribution declines exponentially. From our experiments we see that small jobs have a higher probability of insertion. Therefore, prices of these jobs have a very long left tail, going to zero if the job can be nicely inserted. The limiting distribution for the lowest bid on these jobs is the Gumbel distribution with a negative skew. As can be seen in Figure 6.5 this is indeed the case for small jobs with long time-to-go.

Most of the time, long jobs have to be added to the end of a schedule (otherwise the cheapest alternative is an insertion in a schedule that currently requires a large empty move, which is unlikely in situations where both vehicles and shippers are looking for the cheapest alternative). The price for appending a job consists of the costs for driving empty towards the origin and the costs for the loaded move. Hence, these prices are bounded from below at a value equal to the costs of the loaded move. The limiting distribution for the lowest bid on these jobs is a Weibull distribution with a positive skew. As can be seen in Figure 6.5 this is indeed the case for long jobs with long time-to-go.

Although extreme value theory can be used to explain the skewness of our observations, we can not use one of the extreme value distributions (Gumbel, Fréchet, Weibull) to describe the lowest bid, since we are not dealing with large samples. After fitting several distribution functions to the lowest bid data, we found that only small jobs (40 km and 60 km) can be fitted nicely. Suitable distributions are the Normal- and Logistic distribution, and for a long time-to-go the Weibull distribution. Possible candidate distributions for the longer jobs are Gamma, Beta, Lognormal, and the Gumbel distribution for the

maximum. However, for longer jobs, none of these distributions provide a good fit. The reason is that these distributions always have one peak whereas our data suggest two peaks.

The peaks in the distribution of the lowest bid are caused by the shape of the transportation area. To gain insight into this phenomenon, let us consider the bid price of a single vehicle for adding a new job at the end of its job list. This bid price depends on the loaded travel distance for the new job, but also on the empty travel distance towards the origin of the new job. The likelihood of a certain empty travel distance r can be described by the circumference of a circle with radius r around the origin of the new job. More precisely, by the part of the circumference that falls within the square area. To illustrate this, consider a job of length 80km, located on the diagonal and centered on the middle. After using standard trigonometry, we derive the total length of the intersection of the circle with the square area, as a function of the radius r. The results can be found in Figure 6.7. Clearly, the length of the intersection has two peaks. As a consequence, also the bids of individual vehicles exhibits two peaks, and the same holds for the distribution of the lowest bid (given the limited number of bidders). The distribution of the lowest bid differs from the bid price distribution of an individual bidder in the sense that (1) the peaks are moved to the left (corresponding with lower costs) and (2) the first peak is relatively higher because it is likely that the lowest bid comes from the neighborhood of the first peak.

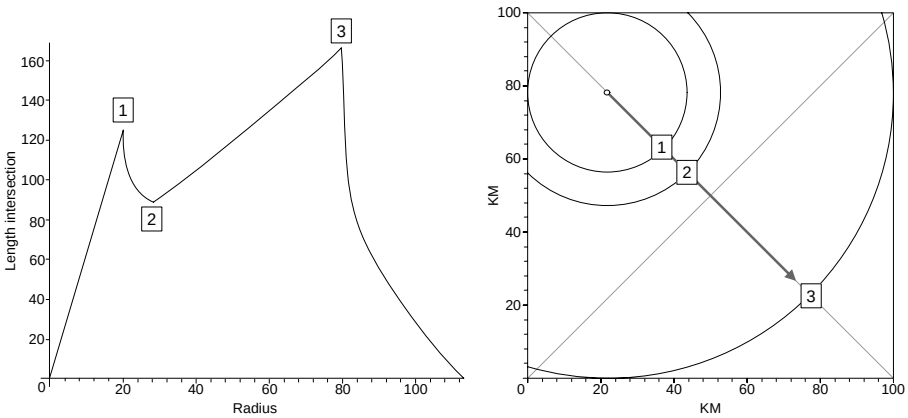

Figure 6.7: Length of the intersection of a circle (with fixed center point on the diagonal) with a square as a function of the radius

Although we only present the results based on an insertion scheduling method, similar results hold for (1) a scheduling method where every new job is appended to the end of a schedule and (2) a scheduling method which allows complete rescheduling (cf. the policies AgentAppend and AgentTSP in

Chapter 3). The results in case of append scheduling are similar to those of insertion scheduling for long jobs because long jobs have a low probability to be inserted. The results in case of complete rescheduling are similar to those of insertion scheduling with short jobs because there is more scheduling flexibility and the bounds are therefore less tight.

To summarize this section, it is impossible to come up with a single distribution function that can be used effectively to describe the lowest bid for all job types. For example, the skewness of the distribution ranges from high negative values to high positive values. To overcome this, one could use separate distribution functions, depending on the time-to-go and distance. However, to keep the results reproducible without providing an enormous amount of different distribution functions, we decided to use a single distribution function. We choose for the normal distribution because it has zero skewness and provides a good fit for intermediate job lengths.

6.8.2 Part 2: price evolution

For all three network settings (quiet, normal, busy) we perform least squares on the transportation costs and penalty costs of the lowest bid. Therefore, we generate 100,000 jobs for all three network settings.

First, we estimate the transportation costs. To estimate the standard deviation σ_{td} in transportation costs, we divide both, the time-to-go t and distance d, in 10 classes and use least squares on the standard deviation per class (see Section 6.6). To estimate the mean transportation costs μ^w_{td} we use weighted least squares, considering the heteroscedasticity described by σ_{td}. The results can be found in Table 6.1.

Network	Mean and standard deviation of transportation costs	R^2
Quiet	$\mu^w_{td} = 1742.42 - 138.68t + 85.33d$	0.75
	$\sigma_{td} = 1266.53 - 68.34t + 8.24d$	0.71
Normal	$\mu^w_{td} = 1874.45 - 164.93t + 87.32d$	0.74
	$\sigma_{td} = 1369.07 - 84.76t + 9.42d$	0.88
Busy	$\mu^w_{td} = 2061.72 - 200.68t + 90.49d$	0.73
	$\sigma_{td} = 1590.51 - 105.92t + 9.07d$	0.84

Table 6.1: Estimation of the transportation costs

Clearly, prices increase with decreasing time-to-go or increasing distance. The reason for this is that with decreasing time-to-go, there is less flexibility to schedule a job, so the probability of a cheap insertion decreases. The same holds for longer transportation jobs. We also see that variances increase with decreasing time-to-go or increasing distance, just like the expected prices.

Next, we estimate the penalties, consisting of the mean penalty costs μ^p_{td}, and the shape and scale parameters of the Weibull survival function (see Section

6.6), respectively given k and λ. The results can be found in Table 6.2.

Network	Mean penalty costs	k	λ
Quiet	$\mu^p_{td} = 15912.13 - 3218.62t + 52.25d$	1.70	1.09
Normal	$\mu^p_{td} = 16339.40 - 3204.27t + 69.75d$	1.03	1.56
Busy	$\mu^p_{td} = 16307.84 - 2640.34t + 151.22d$	1.69	1.19

Table 6.2: Estimation of the penalty costs

Finally, we consider the correlation coefficient between subsequent auction rounds and the update probability q^u_t depending on the time t between successive auction rounds. To calculate the correlation coefficient, we only consider subsequent auction rounds in which the lowest bid is updated. To determine the update probabilities, we divide all data in 20 bins, based on the time between subsequent auction rounds. We then fit an exponential distribution to derive the update probability $q^u_t = 1 - e^{-\lambda t}$ depending on the time t between subsequent auction rounds. The results can be found in Table 6.3.

Network	Correlation coefficient	Rate update probability
Quiet	0.774	0.911
Normal	0.776	0.934
Busy	0.771	0.982

Table 6.3: Correlation and update probability

The correlation in deviation from expected bid prices decreases with increasing number of jobs. Note that these correlation coefficients are quite high. Therefore, we may expect that ignoring these correlations in the dynamic threshold policy will have a negative impact on the performance, as we show later on (Section 6.8.3). The probability of an update of the lowest bid increases with increasing number of jobs.

6.8.3 Part 3: transportation market

Here we present the results for (1) the offline numerical experiment, (2) delaying and breaking commitments, (3) varying input data, and (4) the closed environment. As a performance indicator we consider the average net costs per job. These net costs consist of empty travel costs and penalty costs. The loaded travel costs are excluded because these costs have to be made and can not be reduced. Because the costs of a job are determined by the accepted bid price, we subtract the loaded travel costs from the bids. Given a speed of 50 km/hour and travel costs of 1 per hour, the loaded travel costs for a job with distance d are $d/50$.

Offline numerical experiment

To evaluate the theoretical savings of the dynamic threshold policy, we consider a job with average length. The average distance between 2 points in a square network of 100x100 km is approximately 52.082 km. For a job with this length, we calculate the expected price $E[B_1]$ in the first auction round and the expected price $E[V_1(B)]$ of following an optimal threshold policy: $E[V_1(B)] = E[\min\{B, \alpha_1(B)\}]$. As a performance indicator we consider the relative savings in net costs for using the threshold policy compared to the naive policy. Therefore, we subtract the loaded move costs from both values $E[B_1]$ and $[V_1(B)]$. The results can be found in Table 6.4.

Network	Relative savings for a 52km job
Quiet	29.0
Normal	27.5
Busy	23.4

Table 6.4: Theoretical savings for the three network cases

We conclude that the relative savings decrease with increasing number of jobs.

Delaying and breaking commitments

In this experiment we investigate the performance of the dynamic threshold policy (DT) and the decommitment policy (DC). We use the following performance indicators:

RD Relative difference in net costs per job from the individual shipper compared to the external market. A value of 25.5% means that the net costs per job for the individual shipper are 25.5% lower than the average net costs per job for the external market.

RTC Relative difference in the net total costs. A value of 0.2% means that the net total costs for all jobs (from the individual shipper and the external market) are reduced with 0.2% by using one or both policies.

FR The percentage of jobs that are accepted in the first auction round. We use this indicator in combination with the dynamic threshold policy.

TGA The average time-to-go when jobs are accepted, for jobs that are not accepted in the first auction round. A value of 6.2 means that jobs that are not accepted in the first auction round, on average are accepted 6.2 hours before the latest pickup time. We use this indicator in combination with the dynamic threshold policy.

DE The average relative difference in costs for jobs that are not accepted in the first auction round, compared to the threshold price in the first auction round of these jobs. We use this indicator in combination with the dynamic threshold policy to compare the threshold price in the first auction round with the realized costs.

DJ The average number of decommitments per jobs. A value of 49.9 means that each job is on average decommitted 49.9% of the time. We use this indicator in combination with the decommitment policy.

TGD The average time-to-go when jobs are decommitted. A value of 8.4 means that on average, decommitted jobs are decommitted 8.4 hours before the latest pickup time. We use this indicator in combination with the decommitment policy.

AR The average number of auction rounds per job for the individual shipper.

Each experiment consists of 10 replications with different seeds. The number of replications for the 1% and 10% market share (MS) are 50,000 and 10,000 jobs respectively. The number of replications corresponds with a confidence level of 95% with a maximum relative error of 5% for the performance indicators RD and RTC in the normal network setting.

Before presenting our simulation results we mention an important observation that explains some of our findings. The market prices (average costs per job) when using one of the policies are higher than expected. The reason for this is that the expected costs are based on the learning period where we auction each job as soon as possible. When allowing delaying or breaking commitments, some jobs are auctioned later. This results in less scheduling flexibility of the vehicles, and therefore higher costs. This effect is more visible in case of 10% market share and in the busy networks because here vehicles tend to have longer schedules.

MS	Network	RD(%)	RTC(%)	FR(%)	TGA	DE(%)	AR
1%	Quiet	25.5	0.2	22.2	6.2	−0.8	6.9
	Normal	30.6	0.4	24.6	6.5	0.9	6.2
	Busy	23.1	0.6	31.9	6.8	−1.8	5.4
10%	Quiet	24.1	1.6	21.3	6.2	−1.5	6.9
	Normal	24.9	1.8	24.3	6.5	−0.9	6.4
	Busy	19.5	4.3	30.8	6.8	−1.9	5.4

Table 6.5: Simulation results for the dynamic threshold policy

First, we consider the dynamic threshold policy. From the results of Table 6.5 we draw the following conclusions.

1. The relative difference (RD) in net costs per job is the highest in the normal networks. An explanation for this is that, in the quiet networks,

the update probabilities are lower and therefore the individual shipper has less potential for receiving a lower bid. On the other hand, in the busy networks, the probability of penalties is higher so that the individual shipper tends to accept more jobs in the first auction round. We see that with increasing number of jobs, the number of jobs that are accepted in the first auction round (FR) increases and on average the other jobs are accepted earlier (TGA). The reduction in total costs (RTC) is always larger in the busy networks.

2. If we compare the 10% market share with the 1% market share, we see that (1) the total costs are always lower and (2) the costs advantage for the individual shipper when using the dynamic threshold policy is smaller. The latter is caused by the fact that the difference in expected prices (based on the learning phase) and the actual market prices, increases with increasing market share. Despite the estimation error, the total costs are reduced. This provides an indication that with increasing market share, so towards the application of an internal allocation mechanism, the total costs will be reduced even further. Of course, we then have to focus on updating procedures for parameter estimation, see Section 6.8.3.

3. The differences between the realized costs for jobs not accepted in the first auction round are very close to their expectation, i.e., the threshold price in the first auction round (DE). Most of the time, the realized costs are slightly higher than the expected costs, which is caused by the increase in market prices.

4. In case of 10% market share, relatively fewer jobs are accepted in the first auction round. Again, this is caused by the increase in market prices because the threshold functions and the learning data is the same in both market share settings.

5. Another interesting aspect is that the relative differences in costs for the individual shipper compared to the external market, are comparable to the theoretical savings from Table 6.4. There are three explanations for the differences:

   (a) Assumptions in our dynamic threshold policy, such as the time independent update probability and the use of an AR(1)-process to describe the correlation in bid prices.

   (b) Estimation errors in the data from the learning phase. For example, the normal distribution is not suitable to describe the lowest bid for all job types.

   (c) The external market also profits from an individual shipper using the dynamic threshold policy. After all, the total costs go down as a result of a better vehicle utilization, which provides the vehicles more flexibility to schedule jobs.

A remarkable result is that the relative difference (RD) in the normal network with 1% market share is higher than the theoretical saving (30.6 > 27.5). After closer inspection it appears that this is mainly caused by the increase in market prices. To be more precise, the lowest bid in the first auction round with a normal network and 1% market share is on average almost 10% higher than expected.

MS	Network	RD(%)	RTC(%)	DJ(%)	TGD	AR(%)
1%	Quiet	−21.5	0.2	49.9	8.4	1.5
	Normal	−24.3	0.3	59.3	8.5	1.6
	Busy	−78.3	2.2	79.7	8.3	1.8
10%	Quiet	−21.7	1.2	48.7	8.4	1.5
	Normal	−26.0	1.7	57.7	8.4	1.6
	Busy	−55.7	4.3	70.7	8.4	1.7

Table 6.6: Simulation results for the decommitment policy

Next, we consider the decommitment policy. From the results of Table 6.6 we draw the following conclusions.

1. The net costs per job for the individual shipper are higher compared to those of the external market. A reason for this is that we are only working with cost prices. For the shipper this means that it determines the decommitment penalties such that the expected costs with or without decommitment are the same. However, the market prices are higher than expected.

   Another reason for the relative difference in case of decommitment is that the external market will benefit from the decommitment option for jobs of the individual shipper. A vehicle only decommits from a job whenever this results in savings for inserting the new job. With probability of 99% (in case of 1% market share) this new job comes from the external market which obviously benefits from the decommitment option for jobs of the individual shipper.

2. Decommitment works especially well in the busy network. In the busy network with 1% market share, we see that the decommitment option for 1% of the jobs decreases the total net costs with 2.2%. Clearly, the costs for jobs from the external market decreases. An explanation for the performance of the decommitment policy is that it appears to be saver, with respect to the latest pickup time, than the dynamic threshold policy. First, there are more jobs not accepted in the first auction round in the dynamic threshold policy, than jobs decommitted with the decommitment policy. Second, decommitment takes place on average 8.4 hours before the latest pickup time compared to the 6.5 hours before acceptance in case of the dynamic threshold policy. After decommitment, the individual

shipper immediately starts a new auction and accepts the lowest bid. Therefore, we can say that on average, decommitted jobs are accepted 2 hours earlier than delayed jobs (jobs that require more than 1 auction round in the dynamic threshold policy).

3. The absolute reduction in total costs increases with increasing market share. However, the relative reduction, with respect to the percentage of jobs that allow decommitment, decreases with increasing market share. A similar conclusion is drawn in ('t Hoen and La Poutré, 2004), although they consider a slightly different network setting (see Section 6.2).

MS	Network	RD(%)	RTC(%)	FR(%)	TGA	DE(%)	DJ(%)	TGD	AR(%)
1%	Quiet	11.6	0.6	23.2	5.9	−0.5	32.3	6.8	9.6
	Normal	6.5	0.5	25.2	6.0	−0.3	38.5	7.0	9.6
	Busy	−68.6	0.5	33.6	6.3	−2.8	52.3	7.2	9.0
10%	Quiet	13.5	2.5	22.8	5.9	0.7	32.8	6.8	9.7
	Normal	−23.1	−0.8	25.7	6.1	−0.1	38.5	6.9	9.4
	Busy	−69.0	−13.4	32.0	6.4	0.1	50.4	7.1	8.8

Table 6.7: Simulation results for the combination of the dynamic threshold policy and the decommitment policy

Next, we consider the combination of the dynamic threshold policy and the decommitment policy. From the results of Table 6.7 we draw the following conclusions:

1. In the quiet and normal network with 1% market share, the total costs are further reduced (RTC).

2. In the normal and busy network with 10% market share, there is an increase in total costs. The reason is that with the combination of both policies, jobs are accepted much later.

Here we summarize the results for the 1% market share which is representative for open environments (see 6.8.3 for closed environments). With respect the relative difference in net costs per job for the individual shipper compared to the external market, we prefer the dynamic threshold policy (savings of 23-31%). With respect to the reduction in total net costs, we prefer the dynamic threshold policy in relatively quiet networks (savings 0.2-0.4%) and the decommitment policy in relatively busy networks (savings 2.2%). These savings in total costs are significant given that only 1% of the jobs allow decommitment.

The costs can be reduced even further by using both policies in combination. However, in the busy networks, the combination of both policies becomes more sensitive to estimation errors. In addition, the combination of both policies requires a lot more computation time as we will see in the next section.

Computation times

Here we evaluate the required computation times for the different policies. The computation time for a simulation experiment consists of the sum of time required for (1) bid pricing and scheduling of the vehicles, (2) saving performance data, (3) network animation, and (4) calculation of the threshold prices and decommitment penalties. Here we are only interested in the later. Besides, computation times for the vehicles are negligible in this case because they use a simple insertion heuristic, bid their marginal costs, and are allowed to evaluate only one job decommitment at a time.

The threshold prices have to be calculated whenever the shipper has to decide whether to accept the lowest bid. The decommitment penalties have to be calculated at each auction round, for all jobs of the individual shipper that are not yet picked up. The threshold prices and decommitment penalties are calculated using (6.20). Here we iterate on the auction round $n = 1...N$, and in each iteration we have to calculate the threshold price $\alpha_n(b)$ for all possible bids $b = 1..L$. To calculate $\alpha_n(b)$, we also have to evaluate all possible bids $b = 1..L$ for the next auction round. The complexity of the dynamic programming recursion is therefore $O\left(NL^2\right)$, where N is the number of remaining auction rounds and L the number of possible bid prices.

For our experiments we used the simulation software eM-Plant 7.5 and an Intel Pentium 4 processor at 3.4 GHz. The dynamic threshold policy is programmed in Delphi 7 as a dynamic link library which we included in our simulation environment. We found that there is no significant difference in running time between the three network settings. The extra time required for the decommitment penalties (taking the expectation over all possible bids) is negligible (<1ms). We further evaluate the running time for different number of remaining auction rounds N and number of possible bids L. First, we evaluate the affect of different remaining auction rounds $N = 1..20$ using $L = 200$. The running time can be fitted by a linear trend $27.595N - 28.795$ with $R^2 = 0.998$. The running time at $N = 1$ is close to zero because the threshold price simply equals the expected price in the last auction round (see Section 6.4.2). Second, we evaluate the affect of different values for the number of possible bids $L = \{100, 200, ..., 2000\}$ using $N = 20$. The running time can be fitted by a power function $0.0136L^{1.9921}$ with $R^2 = 1.000$.

With this information we can estimate the computation time per job under the different policies. The computation time per job using only the decommitment policy is negligible because we only have to calculate the expected prices using (6.16). The computation time C_{DT} per job using the dynamic threshold policy is given by the average number of rounds per job (AR) times the average computation time per round, which in turn depends on the average number of

remaining auction rounds:

$$C_{DT} = AR\left(27.595\left(20 - \frac{AR-1}{2}\right) - 28.795\right) \tag{6.22}$$

So given the average number of auction rounds from Table 6.5, this computation time lies between 2.5 and 3.0 seconds.

To calculate the computation time for the combination of both policies, we add the time required to calculate the decommitment penalties. These decommitment penalties have to be calculated at two events:

1. When the individual shipper accepts the lowest bid for a certain job, it has to calculate the expected costs $E[\alpha_s(B)]$. The average number of times a job is accepted is given by $1 + \frac{DJ}{100}$. The average number of remaining auction rounds upon acceptance is $2 \cdot TGA$. The computation time C_1 for this event is then given by:

$$C_1 = \left(1 + \frac{DJ}{100}\right)(27.595\,(2 \times TGA) - 28.795) \tag{6.23}$$

2. At each auction round, for jobs from the individual shipper as well as from the external market, the individual shipper has to calculate the decommitment penalties for its open jobs, i.e., the jobs that are allocated to a vehicle but not yet picked up. We denote the average number of open jobs by ANO and the average number of remaining auction rounds for the open jobs by ANRA. At each auction, on average a fraction of $MS/100$ (MS = market share) of the open jobs belongs to the individual shipper. The individual shipper calculates the decommitment penalties for these jobs using (6.15). The expected costs $E[\alpha_s(B)]$ at the time of commitment are already calculated. Hence, the computation time C_2 for this event is given by:

$$C_2 = \frac{MS}{100} ANO\,(27.595 \times ANRA - 28.795) \tag{6.24}$$

The average total computation time C_{BOTH} per job using both policies is given by $C_{BOTH} = C_{DT} + C_1 + C_2$. This computation time, together with the explanatory variables, are shown in Table 6.8 for the different network settings.

Clearly, these computation times are huge. In a simulation experiment with 1% market share we generate 5,000 jobs (1% market share times 50,000 jobs run length times 10 replications) for the individual shipper. For the busy network setting, the total computation time is than almost 20 hours (5,000x13.8 sec).

MS	Network	TGA	DJ(%)	AR(%)	ANO	ANRA	C_{BOTH} (sec)
1%	Quiet	5.9	32.3	9.6	0.2	13.3	11.0
	Normal	6.0	38.5	9.6	0.2	13.1	12.6
	Busy	6.3	52.3	9.0	0.3	11.8	13.8
10%	Quiet	5.9	32.8	9.7	2.2	13.2	11.6
	Normal	6.1	38.5	9.4	2.8	12.6	13.2
	Busy	6.4	50.4	8.8	3.4	12.1	14.6

Table 6.8: Simulation results for the combination of the dynamic threshold policy and the decommitment policy

Sensitivity to the input data

Here we vary some input parameters that are estimated at the end of the learning phase. The impact on the relative difference and reduction in total costs can be found in Figure 6.8 and Figure 6.9 respectively.

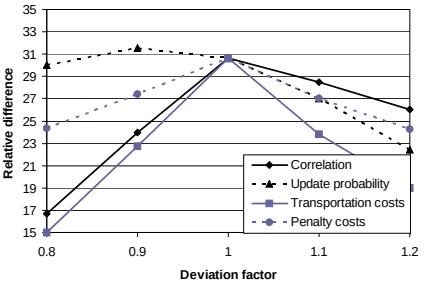

Figure 6.8: Impact of the input data on the relative difference

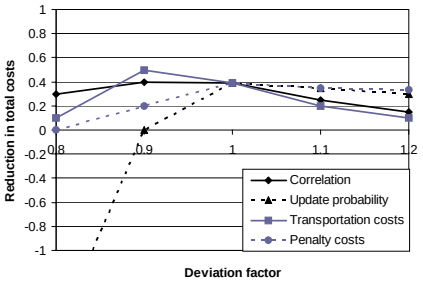

Figure 6.9: Impact of the input data on the reduction in total costs

From Figure 6.8 we conclude that, with one exception, the maximum relative difference is achieved using the exact values of the learning period. Except for the update probability, it appears to be better to use a slightly lower value. A lower update probability will result in higher threshold values. This corresponds with our observation from Table 6.5 where the realized costs are slightly higher than the current threshold values.

Also for the reduction in total costs, we see that the maximum is achieved using the exact data from the learning period. Only a slightly lower value of the transportation costs will result in a larger reduction. However, a lower value of the transportation costs will decrease the threshold values, which in turn have an opposite affect on the relative difference. The reason for this is that the prices for the external market are affected by the single shipper using the dynamic threshold policy.

Closed environments

In this last experiment we use the dynamic threshold policy in a closed environment. For the 3 network settings (quiet, normal, busy) we compare the system performance under the naive policy, the dynamic threshold policy, and the decommitment policy. We decided not to use the combination of both policies given the high computation times (see Section 6.8.3). We consider the following performance indicators:

- The percentage of time the vehicles are traveling loaded
- The service level defined as the percentage of jobs that are delivered before the latest pickup time
- The reduction in total costs compared to the situation in which we use the naive policy

First, we perform a similar experiment as in Section 6.8.3, with the only distinction that we are using a 100% market share. The results can be found in Table 6.9. We see that both policies have a positive affect on the percentage of driving loaded. The dynamic threshold policy has a negative affect on the service level. Of course, delaying an arbitrary job will increase the probability of tardiness for this job. Apparently, the dynamic threshold policy, using the input data of Section 6.8.2, is not able to capture this effect appropriately. In case of a busy network this even results in an increase in total costs. The decommitment policy works remarkably well. It even leads to an increase in service level under all network settings.

Control	Indicator	Quiet	Normal	Busy
Naive	Driving loaded (%)	74.4	74.8	75.3
	Service level (%)	99.4	99.2	98.7
Threshold	Driving loaded (%)	77.1	77.2	77.1
	Service level (%)	97.4	97.7	96.5
	Reduction in total costs (%)	4.7	6.2	−3.9
Decommit	Driving loaded (%)	75.6	76.0	76.3
	Service level (%)	99.8	99.8	99.6
	Reduction in total costs (%)	6.6	7.6	11.6

Table 6.9: Theoretical savings for the three network cases

Next, we introduce an updating procedure for the estimation of parameters as described in Section 6.6. Therefore, we introduce a learning period of 10 days. Every 10 days, the shipper estimates the required parameters for the threshold policy using at most the data of the last 30 days.

Each experiment consists of 10 replications with different seeds. Each replication consists of 100 days with 50 days as a warm-up period. The number

of replications corresponds with a confidence level of 95%, with a maximum relative error of 5% with respect to the percentage of driving loaded and the service level. The results can be found in Table 6.10.

Control	Indicator	Quiet	Normal	Busy
Naive	Driving loaded (%)	74.2	74.5	74.9
	Service level (%)	99.4	99.2	99.1
Threshold	Driving loaded (%)	76.1	76.7	75.9
	Service level (%)	98.1	98.4	98.7
	Reduction in total costs (%)	8.4	9.1	9.3
Decommit	Driving loaded (%)	75.5	76.0	76.8
	Service level (%)	99.8	99.8	99.7
	Reduction in total costs (%)	12.9	14.5	15.5

Table 6.10: Theoretical savings for the three network cases

From these results we see that it is important to use an updating procedure in a closed environment. In all cases, the dynamic threshold policy and the decommitment policy yield a higher reduction in total net costs compared to the results of Table 6.9. Again, the relatively simple decommitment policy outperforms the dynamic threshold policy.

Another important observation here is that the reduction in net costs is much lower than the theoretical values found in Table 6.4. As mentioned before, the theoretical values are calculated using estimated market prices based on a situation in which all jobs are auctioned as early as possible. Auctioning one job later will have an effect on prices for other jobs. In the experiments with a market share of 1%, we have seen (Table 6.5) that on average 75% of the jobs requires more than one auction round. So market prices will certainly change and on average they will increase. Therefore, it becomes more difficult to achieve the theoretical savings. But still, we are able to reduce the net costs with more than 10%.

6.8.4 Summary of the simulation results

In the first two parts of this simulation study we have seen that the estimation of parameters can be quite difficult. Despite that, we still see promising results in the third part of this simulation study.

To test the dynamic threshold policy, we compared the realized savings of an individual shipper with 1% market share, with the theoretical savings that can be achieved in open environments. We have seen that the difference between the theoretical and realized savings is less then 4%. This difference is caused by (1) estimation errors in the lowest bid and (2) the effect of the dynamic threshold policy used by the individual shipper on the net costs of the external market. Given the large number of assumptions we had to make in

the estimation of parameters, this difference is surprisingly small. To test the model itself, we compared the estimated costs for not accepting a job in the first round (the threshold value of the first round) with the average realized costs. These differences are around 1%, with a largest error of 2.8%.

For an individual shipper in an open environment, the dynamic threshold policy is able to achieve a reduction in net costs of 20-30% compared to the external market. In closed environments, the decommitment policy performs best. In fact, it achieves a reduction in the total net costs of 13-16%.

6.9 Conclusions

In this chapter we presented a dynamic threshold policy and a decommitment policy. The dynamic threshold policy enables shippers to postpone commitments for which they expect to make a better commitment in the future. The decommitment policy allows carriers to decommit from an agreement with a shipper against a predefined penalty. From our simulation experiments, we conclude that both policies reduce the total costs. Also, the costs per job for a shipper using the dynamic threshold policy are significantly lower than those who did not use such a threshold policy (20-30% reduction in net costs per job). The decommitment policy, as we derived it without profit margins, results in relatively higher costs per job for the individual shipper, compared to the other shippers. We also considered closed environments (private fleets or collaborative carriers) where all jobs are auctioned using reserve prices or allow decommitment. We found that both policies were able to reduce the system-wide logistical costs, especially the decommitment policy which yields savings of 13-16% in total net costs.

In Chapter 5 we focused on profit maximizing strategies for the carriers. We proposed a bid pricing strategy where the arrival of future jobs are taken into account through the use of opportunity costs. In this chapter we evaluated profit maximizing strategies for the shipper. If both parties (carriers and shippers) are using intelligent strategies, it might be the case that they compete against each other and no party will be better of. It might even be the case that this has a negative effect on the system-wide performance. This combination of profit maximizing strategies for shippers and carriers is the subject of the next chapter.

6.10 Appendix

Here we derive the expected costs $Z(\tau)$ to auction the job a time τ after its latest pickup time. After the latest pickup time, this job can only be scheduled with penalties. These penalties are c per time unit. We model this by saying

that if we do not accept the current lowest bid, and the time until the next bid update is Y, then the shipper faces extra penalties cY, which it has to pay immediately (obviously in our application these penalties are included in the future bid prices). We introduce a value function $V(\tau)$ which reflects the minimum expected price the shipper has to pay eventually, given a time τ after the latest pickup time. This price excludes penalties paid so far. So we write:

$$Z(\tau) = V(\tau) + c\tau \tag{6.25}$$

The minimum expected price a shipper has to pay, as a function of the time τ, is given by:

$$V(\tau) = \min(B, E[V(\tau + Y) + cY]) \tag{6.26}$$

with B the stochastic variable for the lowest bid and Y the exponentially distributed time until the next bid update. The threshold function in time τ equals the expected price at the next update of the lowest bid:

$$\begin{aligned} \beta(\tau) &= E[V(\tau + Y) + cY] \\ &= E[\min(B, \beta(\tau + Y)) + cY] \end{aligned} \tag{6.27}$$

Now, we only accept an offer b at time τ if it is lower than $\beta(\tau)$. So we get the following:

$$\beta(\tau) = \int_0^\infty \left(\int_0^{\beta(\tau+y)} b dF(b) + \beta(\tau + y) \int_{\beta(\tau+y)}^\infty dF(b) + cy \right) \lambda e^{-\lambda y} dy \tag{6.28}$$

Basically $\beta(\tau + y)$ does not differ from $\beta(\tau)$ because we consider an infinite horizon problem. Note that if you receive a bid update a time Y after the previous bid update, and you decide not to accept it, this bid 'disappears' and you already lost an amount cY. It is just like starting the problem over again, i.e., the problem is invariant of time. So we write:

$$\beta = \int_0^\infty \left(\int_0^\beta b dF(b) + \beta \int_\beta^\infty dF(b) + cy \right) \lambda e^{-\lambda y} dy \tag{6.29}$$

$$= \int_0^\beta b dF(b) + \beta \int_\beta^\infty dF(b) + \frac{c}{\lambda} \tag{6.30}$$

From this we get:

$$c = \lambda \int_0^\beta (\beta - b) dF(b) \tag{6.31}$$

Solving this equation for β yields a stationary threshold policy after the latest pickup time: we accept the first offer below β. The expected costs $Z(\tau)$,

given we receive a bid update τ time units after the latest pickup time, are then given by:

$$Z(\tau) = \beta + c\tau \tag{6.32}$$

As an example, suppose $F(b)$ is $U[0, \omega]$, the uniform distribution on the interval $(0, \omega)$. Then we get the following:

$$\begin{aligned} c &= \lambda \int_0^{\beta} \frac{(\beta - b)}{\omega} db, \text{ if } \beta \leq \omega \\ c &= \lambda \int_0^{\omega} \frac{(\beta - b)}{\omega} db, \text{ if } \beta > \omega \end{aligned} \tag{6.33}$$

Equating to c, we find:

$$\begin{aligned} \beta &= \sqrt{\frac{2\omega c}{\lambda}}, \text{ if } c \leq \frac{\omega\lambda}{2} \\ \beta &= \frac{c}{\lambda} + \frac{\omega}{2}, \text{ if } c > \frac{\omega\lambda}{2} \end{aligned} \tag{6.34}$$

So for penalty factors $c > \frac{\omega\lambda}{2}$, we have a threshold price larger then ω. This means that we accept the first bid after the latest pickup time, because it is always lower then ω. The expected price will then be $\omega/2$, and the expected penalties c/λ because we expect to receive this bid update a time $1/\lambda$ after the latest pickup time.

For the discrete case, we replace the expected time $1/\lambda$ between successive bid updates by the auction period R.

Chapter 7

The interaction of carrier and shipper strategies

In this chapter we study a closed transportation market where shippers offer time-sensitive full truckload pickup-and-delivery jobs through sequential auctions, and where a fixed set of vehicles compete with each other to service these jobs. We model this as a multi-agent system consisting of shipper agents and vehicle agents that both apply profit maximizing look-ahead strategies. We study the impact of applying these local strategies - in the absence of any form of central coordination - on the system-wide logistical costs.

For the shipper agent we propose two auction strategies, namely the use of reserve prices and decommitment penalties (see Chapter 6). For the carrier agent we focus on pricing and scheduling decisions where not only the direct costs of jobs are taken into account, but also the impact on future opportunities (see Chapter 5). We use simulation (1) to compare the performance of the individual look-ahead strategies with the performance of myopic policies and (2) to study the interrelation of the different strategies. We conclude that the individual strategies reduce the system-wide logistical costs in almost all cases, depending on the network characteristics. We also conclude that the shippers' and vehicles' policies are complementary: the joint effect of two policies is larger than the effect of an individual policy. We further provide insight into possible problems we may face when multiple players use look-ahead strategies that require the individual players to learn each others' behavior to enhance their decision making capabilities. Therefore, we end this chapter with a proposal for further research that may overcome some of these problems.

7.1 Introduction

During the last decade there has been a growing interest in collaborative logistics due to the ever increasing pressure on shippers and carriers to operate more efficiently. Cooperation among transportation agencies takes place on various organizational and institutional levels, and in various forms. These forms range from spot markets to private fleets. In the spot markets, a large number of shippers and carriers exchange loads and vehicle capacity. In the private fleets, a shipper has exclusive and direct control over a fleet of vehicles. Situated between these extremes are the contractual agreement structures which become increasingly popular in the trucking industry. Here contractual agreements take place between shippers and carriers. These relations are stable and often long-term. Many large shippers have a core carrier program in which they form partnerships with a few large carriers, with the intent both to reduce their carrier base and to maintain or increase the level of service provided (Song and Regan, 2003). In fact, many on-line marketplaces have shifted their focus to more private collaborative networks (Song and Regan, 2001). Instead of being open to any shipper and carrier, the private marketplace is a platform with access only for a small group of companies. This model allows shippers to maintain long-term relationships with their transportation providers.

In this chapter we focus on private collaborative networks and private fleets. In these structures, the problem arises of how to allocate jobs, possibly coming from various shippers, to a fixed set of transportation agencies within one organizational association. In earlier chapters, we refer to this situation as a closed environment. Examples of such an environment can be found in Chapters 3 and 4, where internal transportation tasks have to be allocated to a set of automatic guided vehicles. In these systems, the overall goal is to achieve an efficient allocation, i.e., to minimize the logistical costs within an affiliated group of transportation resources.

Traditionally, the allocation and scheduling decisions in a closed environment are supported by operations research (OR) based optimization methods. Throughout this thesis, we argue that such a central approach is less suitable for real-time and dynamic environments (see e.g. Section 1.1.2 and Section 3.1 for an argumentation). As an alternative we propose to use a multi-agent system (MAS). Such a system consists of a group of intelligent and autonomous computational entities (agents) which coordinate their capacities in order to achieve certain (local or global) goals (Wooldridge, 1999). For the described environments this means that we represent each organizational unit by an agent. In Chapter 3 we presented such a system, where we introduced shipper agents that are responsible for finding transport capacity and carrier agents that are responsible for the routing and scheduling decisions of their vehicles. The main decisions here are (1) the allocation of full truckload transportation jobs to a fixed group of carriers and (2) the timing of these jobs. The allocation of jobs is done using a sequential auction procedure: for each incoming job, the shipper

agent starts an auction and carrier agents bid on these jobs.

The proposed agent approach may cause some difficulties. First, the individual goals of agents may be conflicting and even may deteriorate the system-wide logistical performance. However, in Chapter 3 we compared the performance of the agent approach with two central scheduling heuristics. Using a case study on an underground AGV system around Amsterdam Airport Schiphol, we found that a properly designed multi-agent system performs as well as or even better than the central scheduling methods. A second problem is that jobs arrive real-time, so an optimal allocation of transportation jobs to vehicles can only be derived afterwards, i.e., when all jobs are known. This means that a certain allocation may become unfavorable when new jobs appear. To cope with this, we improved the decision making capabilities of carriers and shippers in Chapters 5 and 6 respectively. This improvement consists of the introduction of look-ahead, where agents take into account future job arrivals in their current decisions (e.g. bid pricing, job scheduling, and bid evaluation). These look-ahead strategies are developed for so-called open environments in which we focus on the strategies of an individual agent and ignore the impact of others. To evaluate the approach, we compare the profitability of the individual agent with the other agents that are using a myopic strategy (such as described in Chapter 3).

In the present chapter, we apply the look-ahead strategies of Chapters 5 and 6 to a closed environment. As a consequence, we are less interested in the profitability of individual agents, but rather in the reduction of system-wide logistical costs. Applying the look-ahead strategies to all players in a closed environment does result in some problems. The look-ahead strategies require the agents to estimate the behavior of the other agents. As a consequence, it is assumed that the behavior of other agents remains the same. Obviously, this no longer holds when the other players also use strategic learning policies.

The goal of this chapter is threefold. First, to provide insight into the possible problems that occur when we apply the look-ahead strategies to all players in a closed environment. Second, to study the impact of these strategies on the system-wide logistical costs, and to study the interrelation of the different strategies. Third, to take a few steps towards a new research approach that might overcome some of the problems caused by multiple agents that learn each others' behavior.

7.2 Literature

Because we focus on multiple participants in a transportation marketplace, our problem is related to several research areas. Our focus on vehicle strategies is related to Dynamic Vehicle Routing Problems (DVRP) and Dynamic Fleet Management Problems (DFMP). Our focus on shipper strategies is related

to research on optimal auctions and optimal stopping problems. In the next sections we describe the relation of this chapter with these research areas and describe our contribution.

7.2.1 Vehicle routing and fleet management

Vehicle routing and scheduling in a dynamic environment has been studied by a number of authors (see Psaraftis, 1988; Gendreau and Potvin, 1998; Yang et al., 2004). The most common approach to handle these problems is to solve a model using the data that are known at a certain point in time, and to re-optimize as soon as new data become available. Because a fast response is required in a real-time environment, a solution is usually achieved by using relatively simple heuristics or by parallel computation methods, see (Ghiani et al., 2003) for an overview of approaches. In this chapter we decompose the problem into a structure where vehicles themselves are responsible for pricing and scheduling decisions.

The dynamic allocation of transportation jobs belongs to the large class of dynamic fleet management problems (DFMP). A few representative examples of this stream of research include (Carvalho and Powell, 2000; Godfrey and Powell, 2002; Yang et al., 2004). We cannot use the DFMP algorithms directly, because (1) jobs have to be accepted early to avoid losing them to competitors and (2) jobs are scheduled in a distributed manner by vehicle agents. Also the price of a job is not given externally but subject to negotiation. Moreover, the arrival intensity of jobs at a company is not described by an exogenous information process, but can be influenced by better repositioning of vehicles.

Closely related research on opportunity valuation in bid pricing can be found in (Figliozzi et al., 2006). Examples of related research on opportunity valuation in scheduling decisions can be found in literature on the dynamic fleet management problems (e.g. Godfrey and Powell, 2002) and waiting strategies (e.g. Thomas and White, 2004; Mitrović-Minić and Laporte, 2004; Ichoua et al., 2006). For more references we refer to Chapter 5 where we proposed an opportunity valuation method that supports the pricing, scheduling, and waiting decisions of vehicles.

7.2.2 Transportation procurement auctions

From the shipper's perspective, our focus is on the transportation procurement auction. The design of auction mechanisms that maximize the seller's expected revenue, called optimal auctions, has received a great deal of attention. For an extensive literature survey on this topic we refer to (McAfee and McMillan, 1987).

Traditionally, a shipper allocates transportation jobs to carriers one-by-one,

i.e., through sequential auctions. Such a system ignores the interdependencies between subsequent jobs. A significant portion of the trucking industry costs is due to the repositioning of empty vehicles from the destination of one load to the origin of a subsequent load (Song and Regan, 2002). Interdependencies occur because serving one job is greatly affected by the opportunity to serve another job. To cope with these dependencies, Caplice and Sheffi (2003) suggested to use combinatorial auctions. As demonstrated by Ledyard et al. (2002), the benefits of combinatorial auctions to shippers can be significant. A survey on combinatorial auctions for the procurement of transportation services can be found in (Sheffi, 2004). In this chapter we consider sequential auctions. To cope with the interdependencies among jobs, we use reserve prices and decommitment penalties.

As shown by Myerson (1981), the reserve price increases the expected revenue of the seller by preventing the object from being sold at a low price. Closely related is the work of (McAfee and Vincent, 1997), who study the optimal reserve-price path in a sequence of first- and second- price auctions. In particular, the auctioneer puts the same object for sale repeatedly, until it is sold. At each round he chooses a reserve price according to his (increasingly pessimistic) beliefs about the buyers' valuations. The choice of whether to accept the lowest bid in a sequential auction is also related to the so-called optimal stopping problems, see (Chow et al., 1971) for more details. Our approach differs from this line of research in the sense that we consider (1) historic auction information to update the offer distribution, (2) time-dependent offers, (3) correlation between subsequent offers, and (4) the finite horizon problem as a special case.

Decommitment penalties are introduced in (Sandholm and Lesser, 2001). Here an agent can decommit (for whatever reason) simply by paying a decommitment fee to the other agent. It is shown, through game-theoretic analysis, that this leveled commitment feature increases the Pareto efficiency of contracts and can make contracts more beneficial for both parties.

For more references on the use of reserve prices and decommitment penalties we refer to Chapter 6.

7.2.3 Contributions

In the previous sections we have described our contribution on the separate perspectives of shippers and carriers. Our main contribution, however, lies in the combination. To the best of our knowledge, the interaction between carriers and shippers, each using look-ahead profit maximizing strategies, has not been studied before. To summarize, our contribution consists of the following:

1. We evaluate the impact of the individual look-ahead policies on the system-wide logistical costs. Here we focus on closed environments in the

sense that we consider the allocation of transportation jobs to a given set of vehicles (e.g. collaborative carriers or private fleets).

2. We evaluate the interaction between the different policies and evaluate their benefits in terms of system-wide logistical costs. Specifically, we evaluate whether the policies are complementary, i.e., if the joint effect of two policies is larger than the effects of the individual policies.

3. We provide insight into the possible problems that occur when we apply the look-ahead strategies to all players in a closed environment and present methods to deal with this.

7.3 Model of the transportation market

Jobs to transport unit loads (full truckload) arrive one-by-one. These jobs are characterized by the following parameters: an origin i, a destination j, a latest pickup time e of the load at the origin, and a time a at which the job becomes known in the network $a \leq e$.

We consider two types of transportation networks:

1. Node networks: transportation takes place between a discrete set of nodes. Hence, the origin and pickup locations represent nodes in a directed graph $(\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ the set of arcs connecting these nodes.

2. Region networks: transportation takes place between a continuous set of locations. The origin and pickup locations are represented by (x, y) coordinates in the Euclidean plane. However, to estimate the value of certain areas within the network, we discretize the network into disjoint regions. We denote the set of regions by $\mathcal{N}$.

Within the network, all jobs have to be transported by a fixed set of available vehicles (not necessarily identical). For this purpose we assume that the total transportation capacity is sufficient to handle all jobs in the long run. We further assume that a job in process cannot be interrupted (no preemption); i.e., a vehicle may not temporarily drop a load in order to handle a more profitable job and return later on. The overall goal is to minimize the costs. We consider two cost functions, namely the travel costs $c^r(t)$ as function of the travel time t and the penalty costs $c^p(t)$ in case of tardiness ($t > 0$), which is a positive non-decreasing function of the time t. The time to transport a load from node i to node j is given by τ_{ij}^f. This includes travel time, and the handling time to load- and unload the job. The time to drive empty from node i to node j is given by τ_{ij}^e. Both times are deterministic and vehicle independent.

To model the transportation market we use a multi-agent system. Such a system consists of intelligent individuals that are situated in a dynamic and uncertain environment, interact with each other, and improve their decision making capabilities by learning about their environment. We represent each player by an agent that acts as a decision maker. Following an approach similar to the previous chapters, we introduce vehicle agents and shipper agents. The shipper agents receive externally generated transportation jobs. To match these jobs with transport capacity, they offer these jobs to the vehicle agents through sequential transportation procurement auctions. Vehicle agents bid on these jobs and maintain a schedule of the jobs they have won. Without loss of generality, we choose here for a reverse first-price sealed-bid auction in which the lowest bidder receives its bid amount (given the shipper agrees with this bid). For clarity of exposition we further assume that there is only one shipper agent that receives and auctions all jobs.

Our objective is to minimize the system-wide logistical costs, which consist of costs for empty moves and penalty costs (the costs for loaded moves are not included because all jobs have to be transported). This global objective has to be achieved by individual agents with conflicting goals. Objective of the shipper agent is to minimize the costs for transportation given by the sum of all prices paid to the vehicles for transporting their loads. Objective of the vehicle agents is to maximize their profits given by the income from all transportation jobs minus the costs for doing these jobs. Optimizing this system basically comes down to optimizing the auction-based allocation. In the next section we deal with this issue.

7.4 Improving auction-based allocations

A job is allocated to a vehicle whenever the vehicle agent wins the auction for this job. After the arrival of new jobs, it may appear that the job assignment is not optimal anymore. Especially when jobs are complementary (e.g. transportation jobs that can be served sequentially by the same vehicle) or substitutable (e.g. transportation jobs that are available at the same time), a certain allocation may become unfavorable when new jobs appear.

To improve the allocation, we may reallocate all jobs that are not yet transported. However, this is often not realistic in practice because it may require a lot of computation time. Another option is to use auction protocols that are specifically designed to deal with complementary goods. For example, simultaneous auctions (or parallel auctions) where bidders participate in multiple auctions at the same time and combinatorial auctions where bidders may bid on combinations of items. However, combinatorial auctions involve many inherently difficult problems. As mentioned by Song and Regan (2005), we face the bid construction problem, where bidders have to compute bids over different job combinations; and the winner determination problem, where jobs have

to be allocated among a group of bidders. In addition, (1) it may be unrealistic to bid on a bundle of jobs which belong to different shippers and (2) these procedures are not directly applicable in situations where jobs arrive at different points in time.

To improve the allocation of jobs, we take the sequential transportation procurement auction as given, and focus on strategies for the participants. We consider the following options:

1. Opportunity valuation
2. Delay commitments
3. Break commitments

In the first option, vehicle agents do not only take into account the direct costs of doing a certain job, but also its impact on the opportunity costs. These opportunity costs are used to capture the loss in expected future revenue of a vehicle due the acceptance of a new job. Job characteristics, such as the destination of a new job, affect the opportunity costs. But also the order and timing of jobs in a schedule have an effect on the opportunity costs. Therefore, we use the opportunity costs in the bid pricing and scheduling decisions (see Section 7.4.2).

In the last two options (delaying and breaking commitments), we use repeated one-shot auction procedures which combine features of both simultaneous and combinatorial auctions. We get repeated auctions because we (1) allow the shipper agents to delay commitments by repeatedly starting an auction for the same job and (2) permit vehicle agents to break a commitment, i.e., the allocation for a specific job will be reconsidered by starting a new auction for this job.

The idea of delaying commitments is that a shipper uses threshold prices in each auction round. When all bids are higher than this threshold price, the shipper rejects them and starts a new auction later on. This way, shippers are able to postpone commitments (an allocation of a job to a vehicle) for which they expect to make a better allocation in the future. So if the shipper has plenty of time to auction a certain job, it will not agree with a relatively high bid. When the time for dispatch becomes nearer, the price it is willing to accept will rise. We call this a *dynamic threshold policy.*

The idea of breaking commitments is that the shipper allows a vehicle to decommit from an agreement against a certain penalty. These penalties are chosen such, that whenever a vehicle decommits a job, they cover the expected extra costs for finding a new vehicle. When a vehicle decommits a job, the shipper has to re-auction the job in order to find a new vehicle that is willing to do this job. We call this a *decommitment policy.* Note that such a policy is only reasonable in case of private fleets or collaborative networks.

In the next two subsections we describe the three policies in more detail. Apart from some minor modifications due to specific model details (i.e., first-price auction and region networks), this description is a recapitulation of the Chapters 5 and 6. However, we limit ourselves to the basic concepts that are required for understanding the numerical results presented in this chapter. After that (Section 7.4.3), we present the resulting system dynamics of using these policies in our closed transportation market.

7.4.1 Shipper agent: dynamic threshold policy

In the dynamic threshold policy we assume that a shipper has the opportunity to auction a job multiple times. After each auction, the shipper agent has to decide whether to accept the lowest bid. To support this decision we use a value function $V^r_{i,j,d}(t,b)$ (see Chapter 6) as the expected price a shipper has to pay in the remaining period t before the latest pickup time, for a job from i to j (which represent nodes or regions), having distance d, given a lowest bid b in the current auction. We added the current bid b in the state space because sequential bids for the same jobs are correlated. When the current lowest bid is relatively high, it is likely that the lowest bid at the next auction round will also be relatively high. We assume that the time between subsequent auction rounds is fixed; we denote this time by R. In Chapter 6 we show that the optimal policy is to accept the current bid b, for a job from i to j having distance d and remaining time t until the latest pickup time, only when this value b is below a threshold value $\alpha_{i,j,d}(t,b)$. This threshold value, cf. (6.10) in Chapter 6, equals the expected value function at the next auction round:

$$\alpha_{i,j,d}(t,b) = E_{B_{t-R}}\left[V^r_{i,j,d}(t-R, B_{t-R}|B_t = b)\right] \tag{7.1}$$

with $t-R$ the remaining time after the next auction round and B_t the stochastic variable for the lowest bid at remaining period t.

To calculate the threshold values, Chapter 6 proposes a dynamic programming recursion. For this purpose, shippers should have some knowledge about the time dependency of bids for various job characteristics. To be precise, a shipper has to estimate:

1. Mean value of the lowest bid excluding penalties.
2. Variance of the lowest bid excluding penalties.
3. Mean penalty costs in the lowest bid.
4. Probability of having non-zero penalties in the lowest bid.
5. Probability that the lowest bid is updated between subsequent auction rounds.

6. Correlation in transportation costs in the lowest bid between subsequent auction rounds.

These parameters are estimated by the shipper based on historical observations of the lowest bid. The first three parameters are estimated using multiple linear regression on the remaining time t and distance d (see Section 6.6). The fourth and fifth parameter are estimated as a function of the remaining time t using a continuous distribution function (see Section 6.6).

In Chapter 6 the origin and destination of a job are not taken into account in the estimation of the lowest bid. This is appropriate in case of balanced networks such as considered in Chapter 6. However, in this chapter we also consider unbalanced networks where some nodes/regions are more popular than others. To deal with this, we incorporate the origin and destination in the multiple linear regression functions (for the first three parameters), that is, for both, the origin and destination, we add $|\mathcal{N}| - 1$ indicator functions.

In the decommitment policy, the shipper always accepts the lowest bid. However, vehicles are allowed to decommit from an agreement with a shipper against a predetermined time-dependent penalty. These decommitment penalties, for a certain job as a function of the remaining time until the latest pickup time, are calculated by the shipper directly at the start of an auction for this job. The penalties should cover the extra costs for a shipper to find a new carrier. A shipper will face extra costs because after decommitment, the remaining time until the latest pickup time is shorter and the vehicles have less flexibility to schedule this job on time. This also means that the shipper faces variation in the prices, i.e., sometimes the decommitment penalties are insufficient to cover the increase in costs. In principle, the shipper should price this risk. Here we assume risk neutral shippers.

When the shipper uses the decommitment policy in combination with the dynamic threshold policy, the decommitment penalties are given by the difference in threshold prices $V^r_{i,j,d}(t, b)$ between the initial commitment time and the decommitment time. Otherwise, this value is simply given by the expected lowest bid at the decommitment time minus the expected lowest bid at the initial commitment time. The expected lowest bid for given job characteristics and remaining time t can be calculated from the parameters mentioned before (mean value of the lowest bid excluding penalties plus the probability of having non-zero penalties times the expected penalties). Whenever a vehicle decommits, (1) it will not receive the agreed price for the decommitted job, (2) it has to pay the shipper the time-dependent decommitment penalty, and (3) the shipper immediately starts a new auction for this job.

In Chapter 6 we simulate the dynamic threshold policy and the decommitment policy in an open environment. Here only a small percentage, denoted by market share, of the jobs are auctioned under such a policy. For these jobs, we show that the average costs are much lower than the average costs of jobs

that are auctioned under a naive policy (no reserve prices or decommitment penalties). We further show that if we increase the market share, the savings per job auctioned under the dynamic threshold policy are relatively lower. As an explanation we mentioned that with increasing market share, it gets more difficult to estimate the lowest bid. As a consequence, shippers need more time to learn the right parameter settings for the dynamic threshold policy and decommitment policy. In this chapter we focus on closed environments which resemble the 100% market share. This causes some problems as we show in Section 7.5.

In the remainder we denote the use of a dynamic threshold policy by RES (reserve prices) and the use of a decommitment policy by DEC.

7.4.2 Vehicle agents: opportunity valuation policies

At each point in time, a vehicle v has a job schedule Ψ_v, i.e., a list of jobs with scheduled pickup times. These pickup times are scheduled as early as possible, taking into account the required times for empty moves. In the remainder we denote (1) the number of jobs in a schedule by N, (2) the destination of the last job in the schedule Ψ by *schedule destination* $d(\Psi)$, and (3) the time until the expected arrival time at the schedule destination by *length of a schedule* $g(\Psi)$.

Vehicles use this schedule to support their job sequencing decisions and bid pricing decisions. Vehicles are not restricted by the scheduled pickup times, but can simply decide to insert new jobs or to wait at some node after delivery of a job (denoted by flexible contracts in Chapter 5). The vehicles use an insertion scheduling heuristic. Here a vehicle contemplates the insertion of a new job at any position in the current schedule without altering the order of execution for the other jobs.

Given the insertion scheduling method, new job insertions take place either in a period between two successive jobs or after the schedule destination. The idea is that we value these periods in order to capture the impact on future opportunities. As before, we use the phrase *end-gap* for the difference between a planning horizon T (which we choose to be much larger than the length of a schedule) and the length of a schedule. We use the term *gap* to indicate the potential time between two consecutive jobs that can be used for future job insertions.

The gaps and the end-gap are important to value a schedule, because future jobs can only be inserted in these periods. To quantify the values of these periods, we introduce two value functions in Chapter 5, namely a gap-value and an end-value. In this chapter we only consider the end-values because the combination with gap-values appears to have little added value (see Chapter 5).

The *end-value* $V^e(i, \sigma, t)$ is defined as the expected profit during a period t after arrival at schedule destination i, given a time-to-go σ until arrival at node i. In Chapter 5 we calculate the end-values using a Stochastic Dynamic Programming (SDP) recursion. The recursive relations are described by four types of information:

- State space (i, σ, t) with schedule destination i, time-to-go σ, and planning period t.
- Decision $\delta(i)$ to move to location $\delta(i)$ after arrival at location i.
- Transition probabilities $p_{ikl}(\sigma)$ that a vehicle ending in location i receives a job from k to l as next job within the time-to-go σ.
- Expected reward $r_{ikl}(\sigma)$ of a job from k to l that is won within the time-to-go σ and that is scheduled after arrival at node i.

For clarity of exposition, we slightly modified the definition of the transition probabilities and expected rewards compared to Chapter 5. Originally, see Section 5.5, these functions also depend on the time at which a vehicle wins the job, and we integrate them over all possible winning moments. For the transition probabilities this means that it consists of two parts: (1) the probability of winning a job during a certain time period and (2) the conditional probabilities that the winning job foes from k to l.

To calculate the end-values using the SDP recursion, the vehicle agents have to estimate the transition probabilities and expected rewards as a function of the route ikl and the time-to-go σ. To do this, vehicles use historical observations of the lowest bid for various job characteristics (see Chapter 5). In this Chapter we use an approximation $\tilde{V}^e(i, t)$ where the time-to-go σ is replaced either by an average time-to-go or by a time-to-go of zero (see Chapter 5). In the remainder we denote the use of end-values based on an average time-to-go by VEA and the use of end-values based on a zero time-to-go by VE0. Note that with the policy VEA, vehicles also have to estimate the average time-to-go.

The end-value $\tilde{V}^e(i, t)$ provides an indication of the attractiveness of a schedule destination i. With attractive we mean that the expected waiting time until winning a new job after arrival at this node is relatively short. As a consequence, the expected revenue during a period t after arrival at this node is relatively high. The end-values are used by the vehicle agents (1) to calculate a bid price for a new job, (2) to choose an appropriate insertion position for a new job, and (3) to support so-called *pro-active move decisions*. Below we elaborate on this.

The bid price of vehicle v, for inserting a new job φ in its current schedule Ψ_v, is given by the marginal costs of this insertion plus opportunity costs. Given the vehicle has currently N jobs in its schedule, it can schedule the new

job in N possible ways. We write $\Psi^n_{v\varphi}$ for schedule alternative n, where the new job φ is inserted after job n. The bid price $b(v,\varphi)$ of vehicle v for job φ, is given by:

$$b(v,\varphi) = \min_{n=1..N} \left(c^r \left(\Delta T^n_{v\varphi} \right) + c^p \left(\Delta D^n_{v\varphi} \right) + OC \left(\Psi^n_{v\varphi} \right) \right) \tag{7.2}$$

where

$\Delta T^n_{v\varphi}$ = expected additional travel time required for vehicle v in schedule alternative n to transport job φ;

$\Delta D^n_{v\varphi}$ = expected additional tardiness for vehicle v in schedule alternative n due to accepting job φ;

$OC\left(\Psi^n_{v\varphi}\right)$ = the opportunity costs of adding job φ to the schedule of vehicle v using alternative schedule n. These opportunity costs are given by the difference in end-value of the alternative schedule $\Psi^n_{v\varphi}$ compared to the current schedule Ψ_v:

$$OC\left(\Psi^n_{v\varphi}\right) = \tilde{V}^e\left(d\left(\Psi_v\right), T - g\left(\Psi_v\right)\right) - \tilde{V}^e\left(d\left(\Psi^n_{v\varphi}\right), T - g\left(\Psi^n_{v\varphi}\right)\right) \tag{7.3}$$

Note that the bid price (7.2) is similar to (3.1) in Chapter 3, with the only exception that we now use the opportunity costs of a new job insertion instead of the change in waiting time.

We denote the alternative schedule $\Psi^n_{v\varphi}$ with the lowest costs by the temporal schedule $\Psi^*_{v\varphi}$. A vehicle agent updates its schedule when (1) an auction for a new job φ is won and (2) the first loaded move in a schedule has been completed. In the first case, the vehicle agent replaces its current schedule Ψ_v with the temporal schedule $\Psi^*_{v\varphi}$. In the second case, the vehicle agent has to decide upon its next move. Here we assume that if the vehicle schedule is not empty, it will drive immediately to the origin of the next job. Otherwise, the vehicle faces a pro-active move decision, i.e., it has to decide whether to stay or to move pro-actively to another node. The decision to move to node $\delta(i)$, given the current node i, is given by the node δ that maximizes the revenue within the remaining planning horizon $T - \tau^e_{i\delta}$ after arrival at node δ, minus the cost for this empty move:

$$\delta(i) = \arg\max_{\delta \in \mathcal{N}} \left(-c^r\left(\tau^e_{i\delta}\right) + \tilde{V}^e\left(\delta, T - \tau^e_{i\delta}\right) \right) \tag{7.4}$$

Note that more complicated decisions are involved when vehicles not always start the next job as early as possible. For details on this we refer to Chapter 5.

7.4.3 System dynamics

We implement the market mechanism as follows. When a job arrives at the shipper, it starts an auction by sending an announcement to all vehicles. In

return, each vehicle calculates a bid considering the marginal costs of doing this job and its impact on future opportunities (7.2). Next, the shipper has to decide whether to accept the lowest bid (7.1). A shipper may decide to reject all bids and start a new auction later on. Otherwise, the winning vehicle is informed and all vehicles receive information on the lowest bid. If the shipper allows decommitment, it also calculates the time-dependent decommitment penalties for the new job and sends this to the winning vehicle.

The winning vehicle implements the temporal schedule upon which its bid is calculated. If the winning vehicle decided to decommit from another job, then this decommitment is announced to the shipper, which in turn immediately starts a new auction for this job. Finally, after each auction, both the shipper and the vehicles store information of the lowest bid together with the job characteristics. The vehicles also make pro-active move decisions (7.4). These decisions only have to be made after delivery of the last job in their schedule. A general impression of the situation is given in Figure 7.1. Here the arrow *network information* indicates that vehicles are informed about their position and status with respect to loading, unloading, and traveling.

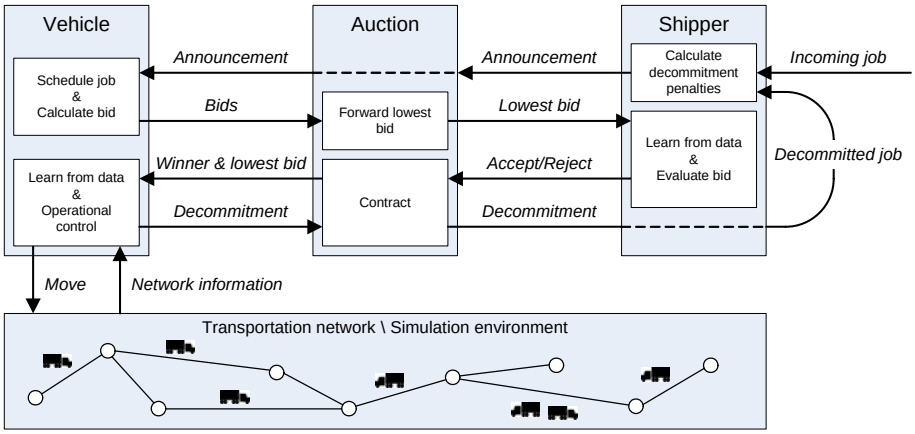

Figure 7.1: Transportation procurement market

Apart from these system dynamics, the shipper and the vehicles periodically update their beliefs about other players by using the auction results of the last period. Specifically, the vehicle agents estimate the distribution of winning moments, the transition probabilities, the expected rewards, and the average time-to-go (required for the policy VEA). They use this information to calculate the end-values $V^e(i,t)$ for $\forall i \in \mathcal{N}$ and $t \leq T$ for the next period in advance. The shipper agent estimates the six parameters mentioned in Section 7.4.1. This situation, where all players learn form each other, causes some problems as we will show in the next section.

7.5 Interaction effects

In Chapters 5 and 6, we evaluated the performance of profit maximizing strategies for shippers and carriers in an open environment. In this environment, we focused on the behavior of a single agent and assume stationary behavior of all other players. We compared the profit of the individual player using a look-ahead strategy with the average profit of the other players that are using a myopic policy. If all players are using look-ahead strategies, and therefore learn about each other's behavior, this certainly leads to some undesired interaction effects. We present these difficulties in the next two subsections for the opportunity valuation policies and dynamic threshold policies respectively.

7.5.1 Opportunity valuation policies

With the opportunity valuation policies, vehicles include opportunity costs in their bid pricing and scheduling decisions. The resulting performance is influenced by (1) other vehicles that are doing this and (2) the shipper that employs reserve prices or allows decommitment of jobs.

First, we consider the impact of other vehicles that are using opportunity costs. When all players use exactly the same end-values that are updated periodically (or all players calculate them separately at the same time based on the same observations), we may expect an increase in prices which we explain below. Specifically, updating the end-values has an effect on the expected rewards and transition probabilities that are used in the dynamic programming recursion:

1. **Expected rewards.** A vehicle estimates the expected reward of a job by taking the difference between the expected lowest bid of its competitors, given this bid is higher than its own expected bid, minus its own expected travel and penalty costs (its bid minus the opportunity costs). The opportunity costs of this vehicle are not subtracted from its expected reward because these costs are expected to be made later on (see Section 5.5.2). The opportunity costs of jobs are on average positive because doing a new job reduces the available time within the planning period T to do other jobs. Therefore, bids prices on average increase. Given that the transition probabilities remain the same, because all jobs have to be served, the expected rewards of jobs increase and therefore also the value functions. In turn, the opportunity costs will be higher and we end up with a continuous increase in bid prices.

   An example can be found in Figure 7.2, where vehicles calculate the end-values periodically starting with zero values in the first period. Consider a vehicle that calculates the end-values at the end of period 2. To do so, it has to calculate (1) the transition probabilities for jobs on all routes

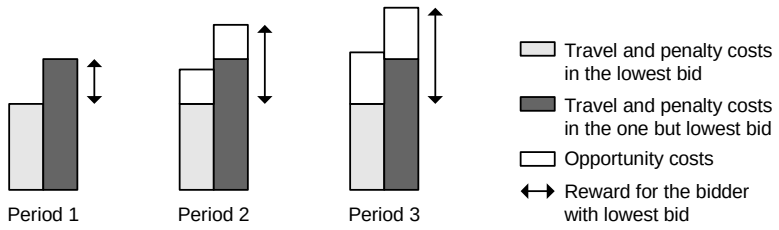

Figure 7.2: Illustration of increasing rewards

and (2) the expected rewards of these jobs. To calculate the transition probabilities, the vehicle uses observations of competitors' bids in period 2 and its own expected bid based on opportunity costs charged in period 2 (see next point). As a consequence, the expected transition probabilities in period 3 will be more or less the same as the realized transition frequencies in period 2 (which can be expected in a closed environment). The expected rewards for this vehicle are (approximately) given by the lowest bid of its competitors (given that this bid is higher than its own), minus its travel and penalty costs. The travel and penalty costs for a given state (i, σ, t) remain the same in each learning period. However, the lowest bid in period 2 from the competitors includes opportunity costs. So compared to the first period, the expected rewards for a given job are increased by the opportunity costs of this job. Given the fact that the transition probabilities remain the same, the total expected revenue in a given planning horizon increases.

2. **Transition probabilities.** If all players learn from the same data (which is the case when all players see the price received by the winning bidder), then their bid pricing and scheduling behavior is also the same. So if a single vehicle wants to estimate its transition probabilities based on historical auction data, it has to be careful which opportunity costs to use. For example, if it uses multiple iterations of the approximate end-value recursion (see Chapter 5, Section 5.6.3), it estimates the transition probabilities based on 'old' observations from the competitors' bids in combination with its 'new' opportunity costs. This will change its estimated transition probabilities, whereas in reality all other vehicles also change their opportunity costs, so that the transition probabilities should remain the same. If we do not take this into account, we may expect an even larger increase in expected rewards as mentioned above.

To avoid the increasing rewards we propose some policy adjustments. With respect to the transition probabilities, we let the vehicles update their value functions at the beginning of each period, where they calculate the transition probabilities based on the opportunity costs they charged at the previous pe-

riod. With respect to the expected rewards, we let the vehicles reason with profits as being the difference between the second lowest bid, given this bid is higher then their own, and their own bid. In other words, we do not include opportunity costs in the calculation of expected rewards. To be precise, we set the opportunity costs in the expected bid price in the dynamic programming recursions equal to zero, see (5.29) in Chapter 5. We indicate this adjustment by NOC (no opportunity costs).

In addition, we consider another policy adjustment where we apply some revenue optimization principles to the calculation of end-values. Here we apply a similar trick as in Chapter 5 (Section 5.4.2), where we introduced a decision variable δ^a for the gap-value approximation. Earlier, we assumed that we derive the bid prices in an optimal way, thereby avoiding less profitable moves. However, given that it takes some time to learn the right parameter settings, it might be the case that in the recursion, the expected revenue of waiting one time unit is higher than the expected revenue of accepting a new job. Therefore, we now add an acceptance decision for each transition: we do not accept the transition when the resulting expected revenue is lower than the value of waiting a single time unit. We expect that this policy converges faster to a steady state behavior. We indicate this adjustment by ACC (acceptance decision).

Next, consider the impact of the shippers' look-ahead strategies on the performance of the opportunity valuation policy. The opportunity valuation policy is affected by (1) the option to decommit because we have an option to switch to another job and (2) the reserve prices because the use of reserve prices affects the winning probabilities. In this chapter we ignore these dependencies by using precisely the same dynamic programming recursions as described in Chapter 5. In other words, vehicles do not explicitly incorporate the reserve prices and decommitment penalties in their bid pricing and scheduling decisions. The results are evaluated with simulation, see Section 7.6 till 7.8.

7.5.2 Dynamic threshold policy

The results of using the dynamic threshold policy or decommitment policy are influenced by the number of jobs that are auctioned under such a policy and by the opportunity costs charged by the vehicles. The latter aspect is not a problem here because we enable the shipper to continuously update its estimation of the lowest bid, including the opportunity costs. With respect to the first aspect we distinguish the following problems:

1. Delaying or breaking commitments has an impact on the vehicle schedules and hence on the bid prices for future jobs. The more jobs that are auctioned under such a policy, the larger this impact is. As a consequence, it will take some time for the shippers to learn the right parameter settings

for their policies. Obviously, this is not a desired property in fast changing environments.

2. Updating expectations on the lowest bid for various job types has an effect on the calculated threshold values. As a consequence, the reserve prices and decommitment penalties change. This in turn has an effect on future observations. For example, if the reserve prices are currently relatively high, most jobs will be accepted early, and hence there will be fewer observations with a small time-to-go. As a result, estimation of the lowest bid for jobs with a small time-to-go is less reliable.

3. In addition to the previous point, the number of observations with a certain time-to-go also affects the estimated value of the lowest bid for jobs with this time-to-go. To illustrate this, suppose that within a learning period only a small percentage of jobs are auctioned with a small time-to-go. Then the estimated value of the lowest bid for jobs with a small time-to-go is relatively lower than it would have been when relatively more jobs where auctioned with a small time-to-go within the learning period.

To cope with the second problem, we introduced so-called learning jobs in Chapter 6. These jobs are auctioned at several points in time without allocating them to a vehicle. They are only used to gain insight into the time dependency of bids. To cope with the other problems, the shipper has to update its beliefs about the distribution of the lowest bid continuously and has to update its policies accordingly. We illustrate the benefits of this policy in our simulation experiments.

7.6 Experimental settings

The goal of this simulation study is (1) to provide insight into the problems mentioned in Section 7.5 and (2) to evaluate the impact of various combinations of shipper's and vehicles' strategies on the system-wide logistical costs. These two goals are worked out in two separate parts. The specific settings of these parts are presented in Section 7.7 and 7.8 respectively. Below we present the general settings.

We consider four network configurations (see Figure 7.3):

4N Here we have 4 nodes that form the corner points of a square. The horizontal and vertical distances between the nodes are 50km.

4R Here we have 4 square regions that span a 2x2 grid within a square area of 100x100 km.

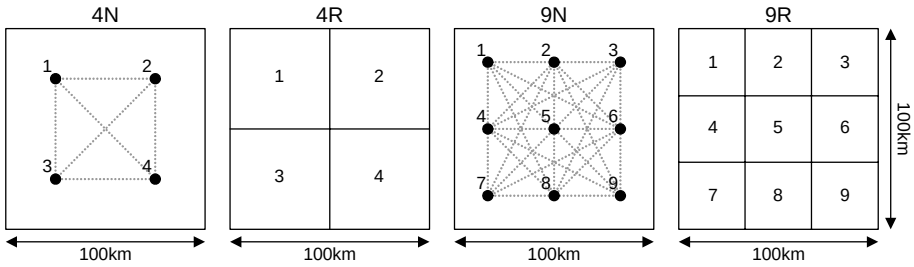

Figure 7.3: Network configurations

9N Here we have 9 nodes located as grid points on a 2x2 grid. The horizontal and vertical distances between adjacent nodes are $33\frac{1}{3}$ km.

9R Here we have 9 square regions that span a 3x3 grid within a square area of 100x100 km.

The nodes/regions are numbered consecutively per row, starting in the upper left corner and ending in the lower right corner. We consider unbalanced transportation networks, where some nodes/regions are more popular than others. To adjust the transportation flow in the node networks (4N and 9N), we set for each node an origin probability, which is the probability that this node becomes the origin of a new job. For a given job, we first draw an origin node using the given origin probabilities and next draw a destination node randomly from the remaining nodes. In the region networks (4R and 9R), we adjust the origin probabilities per region and draw a destination region randomly from all regions (possibly the same as the origin region). Within a given origin/destination region, we draw a (x,y) coordinate randomly from the square area. The different origin probabilities for the 4 node/region networks are shown in Table 7.1.

Degree of balance	Origin probabilities for node/region i ($i = 1..4$)
Balanced	$\frac{1}{4}(1 + (i-1) * 0.0)$
Slightly unbalanced	$\frac{1}{7}(1 + (i-1) * 0.5)$
Unbalanced	$\frac{1}{10}(1 + (i-1) * 1.0)$
Highly unbalanced	$\frac{1}{13}(1 + (i-1) * 1.5)$

Table 7.1: Origin probabilities

For the 9 node/region networks we only consider an unbalanced network structure, where the origin probabilities are given by $i/45$ for $i = 1..9$.

We use 10 vehicles, each having a travel speed of 50 km/hour. The travel cost function is given by $c^r(t) = t$ and the penalty cost function by $c^p(t) = 10t$, where t is measured in minutes. The loading- and unloading times are 5 minutes

each. For the dynamic programming recursions on the end-values we discretize time into periods of 1 minute and use a planning horizon T of 12,000 minutes. Jobs arrive according to a Poisson process.

In our simulation experiments we evaluate the different look-ahead policies of the vehicle agents and the shipper agent. To use these policies, the agents learn about the behavior of others. Here learning takes place periodically (1) by estimation of all required parameters based on the past period and (2) by updating their policy in accordance with this (see Section 7.4.3). We call such a period a *learning period*. The number of learning periods varies per experiment. The length of a learning period is always 10 days.

We consider the following policies:

NA The vehicle agents and the shipper agent use naive policies (no end-values, no reserve prices, and no decommitment penalties).

VE0 Vehicle agents use the end-values in their bid pricing, scheduling, and waiting decisions. The end-values are based on a time-to-go of zero.

VEA Vehicle agents use the end-values in their bid pricing, scheduling, and waiting decisions. The end-values are based on the average time-to-go that is estimated using observations of the last learning period.

DEC The shipper agent allows decommitment of jobs.

RES The shipper agent uses reserve prices.

In addition, we study the policy adjustments NOC and ACC of the opportunity valuation policies (see Section 7.5.1) and study combinations of the opportunity valuation policies with either DEC or RES. We decided not to consider the combination of DEC and RES given the high computation times (see Chapter 6, Section 6.8.3).

As mentioned in Section 7.3, we consider a first-price auction. However, the opportunity valuation methods are developed for a second-price auction (cf. Chapter 5). In Chapter 5 vehicles observe the second lowest bid and use this to estimate the lowest bid. For this purpose they use the Gumbel distribution G_1 for the lowest bid and use the distribution parameters of the Gumbel distribution G_2 for the second lowest bid. Here we simply estimate the distribution parameters of the Gumbel G_1 distribution. However, in the first-price auction, the expected rewards of the vehicles are zero and hence all end-values would be zero. Therefore, we let the vehicles reason with profits as being the difference between their bids and the lowest bid of their competitors (similar to a second-price auction).

As primary performance indicator we consider the average net costs per job which consists of empty travel costs and penalty costs. The loaded travel

costs are excluded because they do not depend on the decisions to be taken. In addition, we consider the relative savings of a certain policy which are defined as the relative difference in average net costs of this policy compared to that of the naive policy. For example, a relative savings of 10% for the policy VE0 means that the average net costs per job are 10% lower than those of the policy NA.

7.7 Simulation study on the impact of learning

The main goal of this simulation study is to evaluate the impact of learning. To be precise, to study the impact of updating the policies on the behavior of the value functions and on the average net costs per job. To illustrate the behavior of the value functions we use the 4R network. We choose this network setting because (1) strange behavior becomes more visible with fewer nodes/regions, and (2) the node networks behave quite chaotically as we will see later on. We further use a time-window of 10 hours and a time between jobs of 800 seconds.

We divide this simulation study in three subparts in which we subsequently answer the following questions:

1. How do the end-values $\tilde{V}^e(i,T)$ behave (constant, fluctuate, increase) in subsequent learning periods using the policies VE0 and VEA in combination with the adjustments NOC and ACC?
2. How do the estimated parameters behave (constant, fluctuate, increase) in subsequent learning periods under the policies DEC and RES?
3. What is the impact of the number of learning periods on the net costs per job?

7.7.1 Part 1: opportunity valuation policies

Here we study the changes in end-values that occur after each learning period. We evaluate various opportunity valuation policies: the policies VE0 and VEA in combination with the adjustments NOC and ACC. To illustrate the impact of learning, we show the results in figures based on one simulation run (the use of multiple replications would level out some details). Within these figures, we show the following values as a function of the number of learning periods:

- The average end-value $\frac{1}{4}\sum_{i=1..4}\tilde{V}^e\left(i,T\right)$ for the planning horizon T.
- The difference $\tilde{V}^e\left(4,T\right)-\tilde{V}^e\left(1,T\right)$ in end-value between the most popular origin region and the most unpopular origin region.

- The realized profits within the planning horizon T (see Appendix for an explanation of how the realized profits are measured).

Note that the end-values and the realized profits are calculated at the end of each learning period. As a consequence, we show for each period t, the average realized profit within a planning horizon T during this period t and the expected profits within a planning horizon T during period $t+1$ as calculated by the end-values. To distinguish between the realized and calculated profit we add, respectively, the suffix -R or -C to the abbreviated policy name.

First, we show the end-values in subsequent learning periods for the policies VE0 and VEA. The results for the average end-values for the whole planning horizon T are given in Figure 7.4. The difference in end-values of region 1 and 4 are given in Figure 7.5.

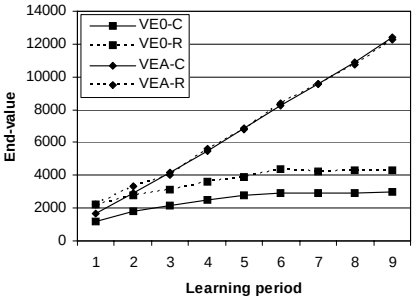

Figure 7.4: Calculated end-values per learning period

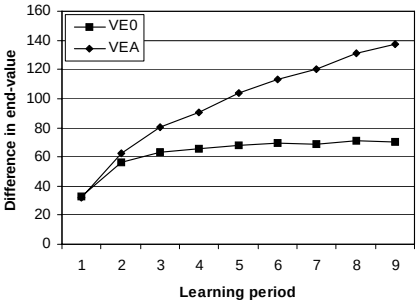

Figure 7.5: Calculated difference in end-values per learning period

For both policies, we see (1) an increase in average end-values and (2) an increase in difference in end-values of region 1 and 4. However, with the policy VE0, this increase is partly wiped out by the overestimation of waiting times. In addition, the increasing end-values may have a negative impact on the estimated transition probabilities which in turn reduces the expected rewards. We further see from Figure 7.4 that the realized profits in period t with the policy VEA are close to the expected profits of period $t+1$. With the policy VE0 we see a larger difference. This is caused by the assumption that we do not win a job in advance; the actual waiting times are therefore lower than expected.

Even though we are dealing with artificial prices and revenues, the increase in bid prices is an undesirable behavior. To avoid this behavior, we apply the policy adjustment NOC where the opportunity costs are excluded from the expected rewards during the calculation of the end-values (see Section 7.5.1). Then the winning revenues of a vehicle are given by the lowest bid of its competitors, minus its direct costs, minus the opportunity costs charged by this vehicle. Now, if bids rise due to the inclusion of opportunity costs, the expected

rewards remain the same. The results for the average end-values for the whole planning horizon T are given in Figure 7.6. The difference in end-values of region 1 and 4 are given in Figure 7.7.

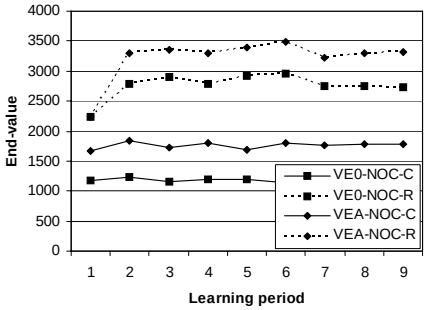

Figure 7.6: Calculated end-values per learning period using the NOC adjustment

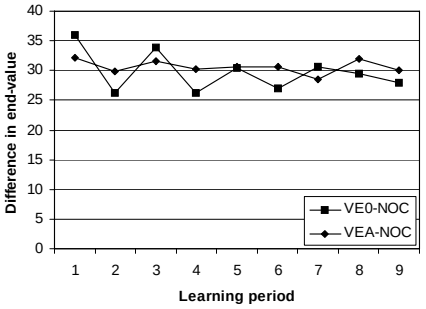

Figure 7.7: Calculated difference in end-values per learning period using the NOC adjustment

In these figures we see a rather stable level of the end-values. Obviously, there is a large difference between the expected profits and the realized profits. This difference is caused by ignoring the opportunity costs. We also see that the differences in end-values for both policies are close to each other, which corresponds with our observations in Chapter 5 (Section 5.9.1). We further see alternating behavior of the difference in end-values (Figure 7.7). We encountered these fluctuations in various experiments, of which only some are displayed here. After thorough investigation, we found two causes for these fluctuations: (1) using the policy adjustment NOC and (2) estimation errors of the lowest bid. We explain these causes in the next two subsections.

Fluctuations caused by the NOC adjustment

First, consider the fluctuations that are inherent to the NOC adjustment. Except for some special network instances (see next subsection), it appears that these fluctuations flatten out: so after a few learning periods these fluctuations are negligible. In Figure 7.7 this is not directly visible because we have much noise caused by the single replication. Later on, in Figure 7.17, we show the result for multiple iterations. There it is clearly visible that the fluctuations are flattening out. The fluctuations are even getting larger when we also apply the policy adjustment ACC (in fact in some cases, as we will show later on, these fluctuations never flatten out). We illustrate this with the policy VE0. The results for the average end-values for the whole planning horizon T are given in Figure 7.8. The difference in end-values of region 1 and 4 are given in Figure 7.9.

We explain the alternating behavior as follows. Directly at the end of the

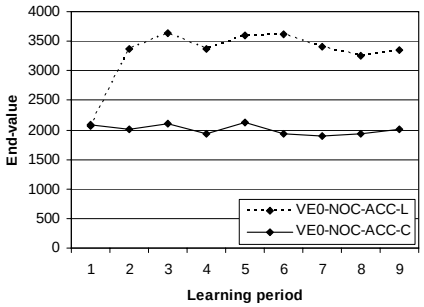

Figure 7.8: Calculated adapted end-values per learning period using the adjustments NOC and ACC

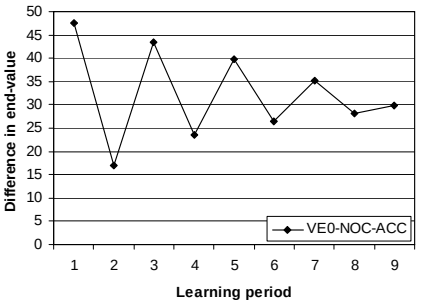

Figure 7.9: Calculated difference in adapted end-values per learning period using the adjustments NOC and ACC

first period we see large differences between the various regions. As a consequence, the end-values under an optimal policy are relatively high. In the second period we are therefore working with relatively high opportunity costs. Now consider a job from the most popular region and one from the most unpopular region.

1. For a job from the most *popular region*, vehicles that have to come from another region will bid relatively less than those that are already present in the most popular region; because the latter group will charge higher opportunity costs. As a consequence, the difference in bid prices for these jobs decreases. Because profits with the NOC adjustment are just given by the difference in bids, the expected profit also decreases.

2. For a job departing from the most *unpopular region*, vehicles that are already located in this region will charge less opportunity costs than vehicles that are located in another region. As a consequence, the expected difference in bid prices increases and the resulting revenues are therefore higher.

Given these two observations, the most popular region becomes slightly less profitable and the reverse holds for the unpopular regions. The opportunity costs of the next period will therefore be lower. Hence, we get some kind of alternating behavior in end-values.

Fluctuations caused by estimation errors

A second cause for the fluctuations is the error in estimation of the lowest bid. Particularly we mention here the use of a continuous distribution function to describe the lowest bid (see Chapter 5). In networks with a few nodes this is

not appropriate because there exists only a limited set of possible bid prices. As a consequence, it can happen that a vehicle, at the end of some learning period, estimates a certain winning probability whereas in reality it will never win. At the end of the next learning period, the vehicle updates the distribution to describe the lowest bid. It is likely that this distribution is different because of the estimation errors in the past period.

To illustrate the fluctuations, we consider the 4N network in combination with the NOC adjustment. The results for the average end-values for the whole planning horizon T are given in Figure 7.10. The difference in end-values of region 1 and 4 are given in Figure 7.11. Clearly, the fluctuations in Figure 7.11 are larger than in Figure 7.7.

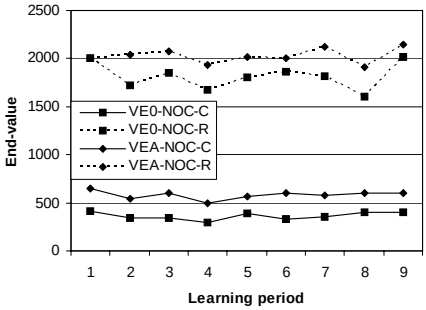

Figure 7.10: Calculated end-value per learning period in the 4N network

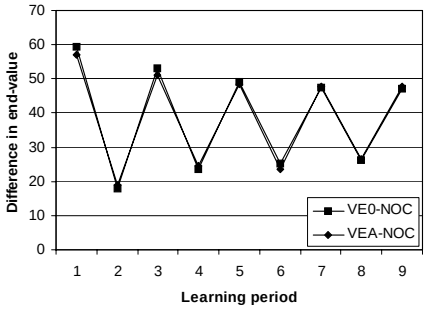

Figure 7.11: Calculated difference in end-value per learning period in the 4N network

In fact, in many cases these fluctuations never flatten out, especially when we also consider the ACC adjustment or perform multiple replications of the dynamic programming recursion on the end-values. An example can be found in Figure 7.12 and Figure 7.13 where we consider the 4N network with the adjustments NOC and ACC. We see that the combination of the two policy adjustments results in (1) increasing end-values and (2) increasing fluctuations in the difference in end-values.

Although we focus here on closed environments, we performed similar experiments for open environments. Here we did not encounter the problems of increasing prices or fluctuations. However, we see that it takes some learning periods to reach steady state behavior. To provide an indication, we see that after the three learning periods, the values remain within a range of 5% around a stable value whereas the values of the first period are 50% away from this value (for the policy VE0 and VEA). To speed up this process, we propose in Chapter 5 to perform multiple iterations of the dynamic programming recursion for end-values. In that case we see that it takes only one learning period to stay within a 5% bound.

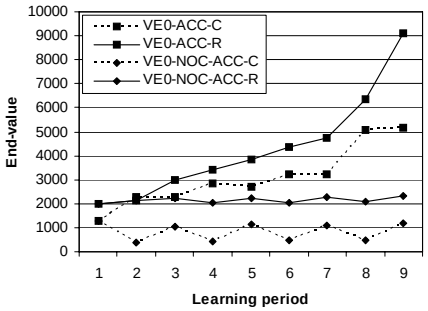

Figure 7.12: Calculated end-values per learning period in the 4N network

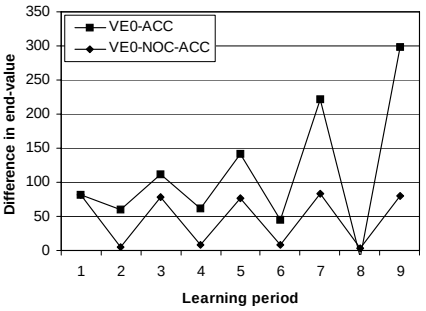

Figure 7.13: Calculated difference in end-values per learning period in the 4N network

7.7.2 Part 2: dynamic threshold policies

Here we study the changes in regression parameters that occur after each learning period, using the policies DEC and RES. Note that we already provide a comparison between the expected prices and the realized prices in Chapter 6.

Learning of the shipper consists of updating its beliefs about the distribution of the lowest bid for various job characteristics. This distribution is characterized by a number of time dependent and distance dependent parameters (see Section 7.4.1). We observe a lot of fluctuations in these parameters with each update. An example can be found in Figure 7.14 where we show the results for (1) the parameter of the mean transportation costs in the lowest bid that describes the dependency on the remaining time t until the latest pickup time; (2) the parameter of the mean transportation costs in the lowest bid that describes the dependency on the distance d; (3) the constant value for the mean transportation costs in the lowest bid; and (4) the probability that the lowest bid changes between subsequent auction rounds.

In general we see an initial increase in reserve prices which stabilizes after some learning periods. This result can be explained as follows. In the first learning period, all jobs are auctioned in one auction round without the use of reserve prices or decommitment penalties. Because all jobs are auctioned as early as possible, the vehicle schedules are relatively flexible in the sense that new job insertions can take place in many ways without causing additional tardiness. As a consequence, the prices for learning jobs with a short remaining time until the latest pickup time are relatively low. After the first learning phase, jobs are auctioned under the dynamic threshold policy or decommitment policy. As a consequence, the average remaining time between auctioning a job and its latest pickup time decreases and the vehicle schedules will become less flexible. Therefore, the price of learning jobs with a short remaining time until the latest pickup time will be relatively higher.

To illustrate this, we consider the decommitment policy and study the parameters during 25 learning periods. To reduce the fluctuations we decided to use exponential smoothing (Silver et al., 1998) with a smoothing parameter of 0.05. The results, using the same parameters as for Figure 7.14, can be found in Figure 7.15.

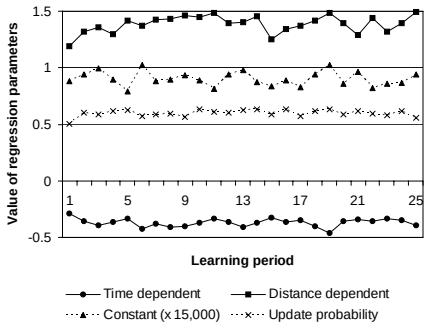

Figure 7.14: Value of regression parameters as a function of the learning period without exponential smoothing

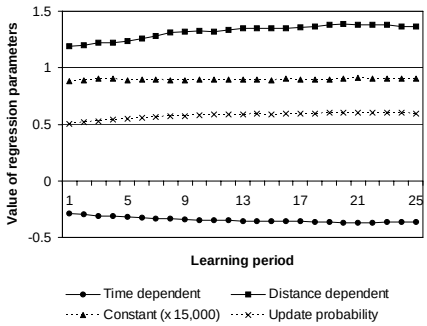

Figure 7.15: Value of regression parameters as a function of the learning period using a smoothing parameter of 0.05

For most parameters, we see that it takes some learning periods to converge. For the time dependent parameter we observe a decrease, so it is relatively better to auction a job early in time. For the distance dependent parameter we observe an increase, so prices increase, especially for jobs with larger distance. Among the parameters that are not depicted here, we observe an initial increase for the variance of the mean transportation costs in the lowest bid and for the correlation of bids. For all the other parameters (see Section 7.4.1), we observe a more stable level.

In case of the dynamic threshold policy, it takes even more learning periods for the estimated parameters to converge. To provide an indication: where the policy DEC requires 20 learning periods to stabilize (see Figure 7.15) it will take 35 learning periods with the policy RES to achieve a same level of stability. The reason for this is that with the dynamic threshold policy, on average more jobs are auctioned with shorter remaining time until the latest pickup time (cf. Chapter 6, Section 6.8.3) and hence the lowest bids for these jobs increases. Clearly, the dynamic threshold policy and decommitment policy require more learning periods than the opportunity valuation policies (see Section 7.7.1). If we are dealing with a stable system, this is not a problem. However, in a fast changing environment, we are not able to adjust these policies in time.

7.7.3 Part 3: impact of the learning period

Here we evaluate the impact of the number of learning periods (1 till 9) on the net costs per job. For this purpose we use a replication / deletion approach for our simulations (see Law and Kelton, 2000) where each experiment consists of a number of replications (each with different seeds) and a warm-up period. The warm-up period consists of the number of learning periods times the length of a learning period (10 days). The length of each simulation run, excluding the warm-up period, is 100 days. We use 5 replications, which is sufficient for a confidence level of 95% with a relative error of 5% with respect to the net costs per job.

First, we evaluate the performance of using the policies VE0 and VEA in combination with the policy adjustment NOC. The average relative savings as a function of the number of learning periods can be found in Figure 7.16. The average difference in end-values over the 5 replications can be found in Figure 7.17.

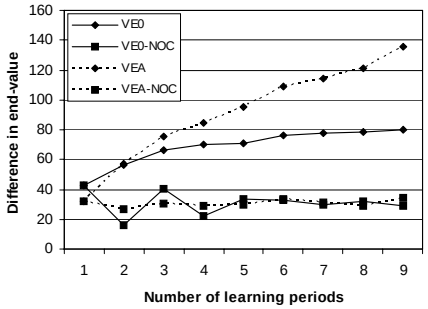

Figure 7.16: Relative savings for various end-values

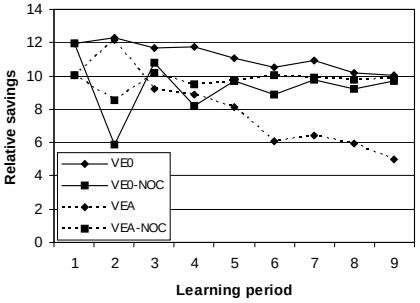

Figure 7.17: Calculated difference in end-value per learning period

For the original end-values (VE0 and VEA), we see increasing differences between the regions (cf. Figure 7.5). The initial relative savings of these end-values are relatively high. However, with an increasing number of learning periods we see a decrease in relative savings. Both policies using the NOC adjustment seem to converge to the same stable levels, both regarding the difference between the nodes and the relative savings. However, the initial fluctuations of the policy VE0-NOC are higher than those of the policy VEA-NOC.

A remarkable result here is that after some learning periods, the policies VE0-NOC and VEA-NOC yield approximately the same relative savings. We explain this as follows. The end-values are used to calculate the opportunity costs. The opportunity costs consist of two components: (1) a value dependent on the decrease in length of the end-gap and (2) a value dependent on the change in schedule destination. The second part will be almost the same with

both policies because the difference in end-value between the regions is almost the same (cf. Figure 7.7). The first component will certainly be different because the absolute value of the end-values differs between these policies (cf. Figure 7.6). However, because all vehicles include this value in their bids, the ordering of bids remains approximately the same, and hence it is likely that the new job will be allocated to the same vehicle under both policies. The advantage of using VE0 is that (1) we do not have to estimate the average time-to-go and (2) it requires less computation time (see Chapter 5, Section 5.9.6).

In the remainder of this chapter, we choose the policy VEA-NOC as default opportunity valuation policy. We abbreviate this policy by OV. Next, we evaluate the performance of using combinations of the policy OV, the dynamic threshold policy (RES), and the decommitment policy (DEC). The results can be found in Figure 7.18.

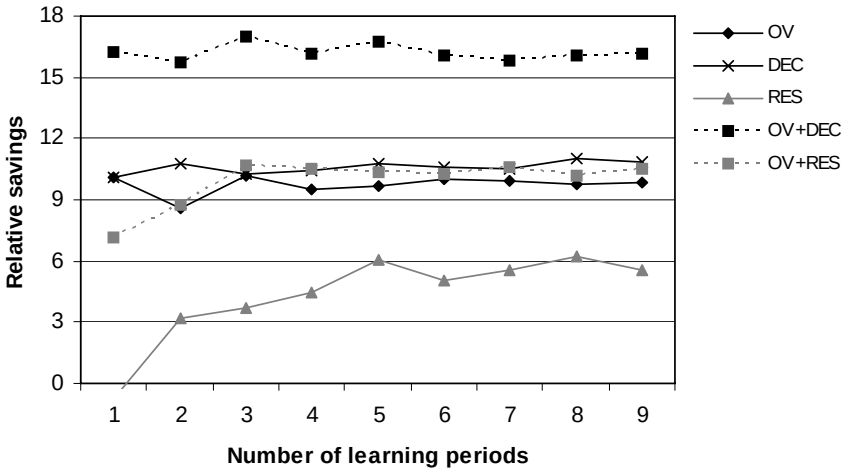

Figure 7.18: Relative savings as a function of the number of learning periods

From this figure we see that the individual policies OV and DEC do not need many learning periods, i.e., one period seems to be enough. The major advantage of this is that they are suitable for nonstationary environments (e.g. the industrial bakery mentioned in Chapter 4). For the policy RES we see that it takes some time to come up with reasonable relative savings; with one learning period we even see that the net costs per job increase compared to the naive policy.

In the Appendix (Table 7.3) we show the results in terms of percentage of driving loaded (percentage of the total driving distance that is traveled loaded) and the service level (percentage of the jobs that are picked up before

the latest pickup time). We see that the policies OV and DEC always yield a better performance than the naive policy NA, with respect to both performance indicators. However, the policy RES always results in a lower service level. This corresponds with our conclusion in Chapter 6 (Section 6.8.3) that the dynamic threshold policy has a negative effect on the service levels.

7.8 Simulation study on the combination of strategies

The goal of this simulation study is to evaluate the impact of various combinations of shipper's and vehicles' strategies on the system-wide logistical costs, measured by average net costs per job. For this purpose we consider the experimental factors as shown in Table 7.2. As mentioned in the previous section, we use the abbreviation OV to denote the policy VEA-NOC.

Factor	**Values**
Policies	NA, OV, DEC, RES, OV+DEC, OV+RES
Degree of balance	balanced, slightly unbalanced, unbalanced, highly unbalanced
Time-window length (min)	300, 400, 500, 600
Time between jobs (seconds)	700, 800, 900, 1000
Network	4N, 4R, 9N,9R

Table 7.2: Experimental factors

A full factorial experiment with respect to these factors would require 6x4x4x4x4=1536 experiments. For clarity of exposition, and to reduce computation time, we consider the following combinations:

1. All combinations of the factors Policies, Degree of balance, and Time-window length; with as fixed settings the 4R network and a time between jobs of 800 seconds.
2. All combinations of the factors Policies, Degree of balance, and Time between jobs; with as fixed settings the 4R network and a time-window length of 600 minutes.
3. All combinations of the factors Policies, Time-window length, and Network; with as fixed settings the unbalanced network and a time between jobs of 800 seconds.
4. All combinations of the factors Policies, Time between jobs, and Network; with as fixed settings the unbalanced network and a time-window length of 600 minutes.

As a consequence, we consider 4x6x4x4=384 experiments. For all experiments we perform 5 replications, which is again sufficient for a confidence level of 95% with a relative error of 5% with respect to the net costs per job. Here we always use a warm-up period consisting of 5 learning periods. From the results in the previous section (see Figure 7.18) we see that this number is sufficient for most policies to converge to a relatively stable performance. The results can be found in Figure 7.19.

From these figures we draw the following conclusions:

- The combination of shipper and carrier intelligence never reduces the performance compared to one of the individual policies. Moreover, the highest savings are always achieved with a combination of two policies. In almost all cases the best combination is OV+DEC. Only with fewer jobs (large time between jobs) the combination OV+RES comes in favor.

- With respect to the individual policies, the policy OV performs best in unbalanced networks. The policy DEC performs the best in balanced networks. The policy RES works well with long time-window length or long time between jobs.

- With increasing time-window lengths, we observe increasing relative savings for almost all methods. However, with the unbalanced networks (unbalanced and highly unbalanced) we only observe an increase with the policy RES. The reason for this is that the naive policy NA also performs better with increasing time-window length. Because the policy RES results in relatively lower service levels (see Table 7.3 in the Appendix), this policy benefits the most from an increase in time-window length. In addition, an increasing time-window length also gives the opportunity to use more auction rounds for a job.

  If we look at the absolute values (see Figure 7.21 in the Appendix) we see that for all policies, the average net costs per job decrease with increasing time-window length. However, with the reserve price policies (RES and OV+RES), these costs decrease relatively faster.

- With increasing time between jobs (fewer jobs), we observe that each policy acts differently. The policy OV remains relatively stable. Only in the case with many jobs in the highly unbalanced network we see relatively large savings. These savings are caused by the fact that the naive policy works very bad in this case. The policy DEC works best with a large number of jobs in the balanced networks and in all other cases with an average number of jobs. The performance of the policy RES increases with increasing time between jobs.

  If we look at the absolute values (see Figure 7.21 in the Appendix), we see a rather stable behavior for all policies in the balanced networks (balanced and slightly unbalanced). Only with the policy RES we see

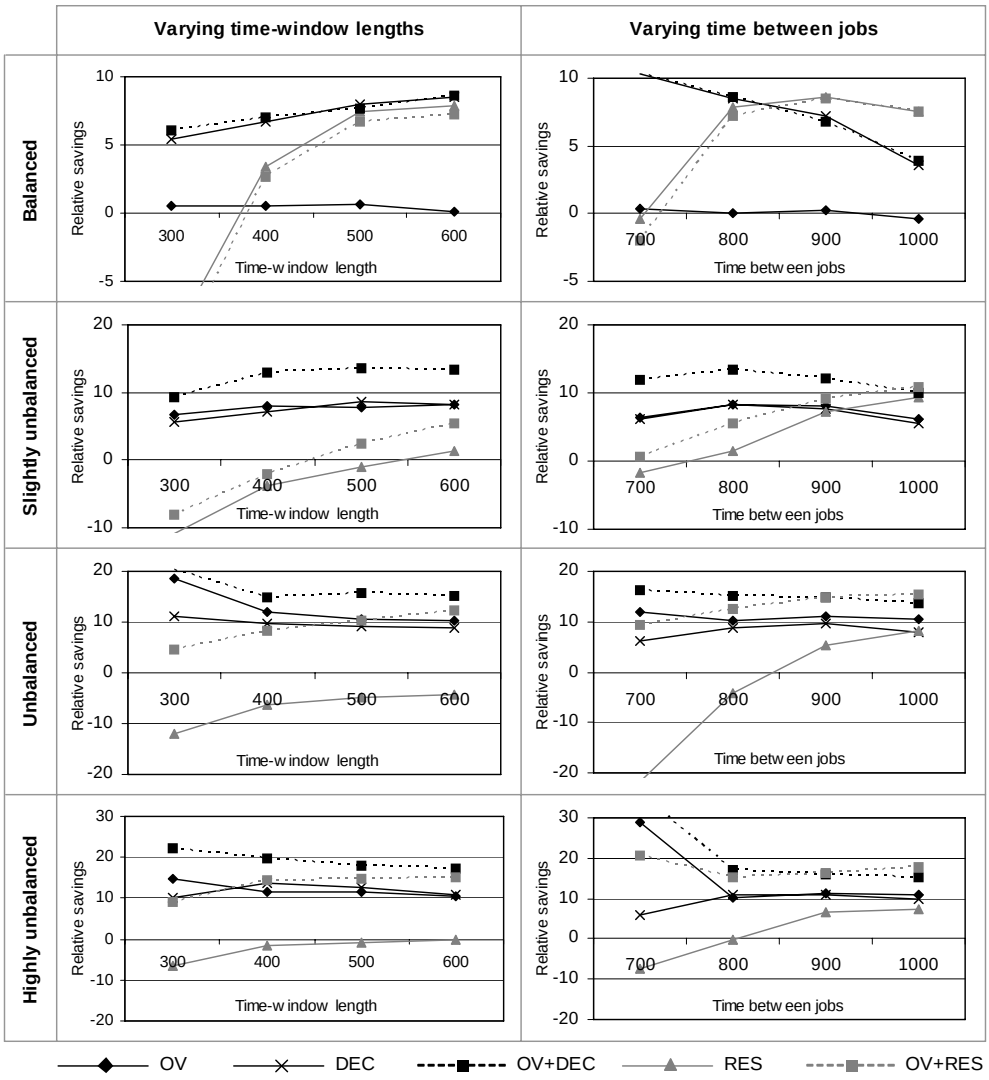

Figure 7.19: Relative savings for different values of the factors degree of balance, time-window length, and time between jobs

decreasing net costs with increasing time between jobs. The reason for this is that the penalties (which are the highest with the policy RES) are reduced. In unbalanced networks we see relatively high costs with many jobs. However, we also see that with some of the policies, the costs increase with fewer jobs. This is caused by the fact that there are fewer possibilities for combining these jobs. This behavior is especially visible with the policy DEC.

- With increasing imbalance we see increasing relative savings for most of the policies. With increasing imbalance, the difference between RES and OV+RES increases. In other words, the added value of combining these policies increases. A remarkable result is that the differences between the unbalanced network and the highly unbalanced network are small. This is caused by the fact that within the highly unbalanced network, the majority of transport takes place within one region. This region can be regarded as balanced because the origin and destination coordinates are drawn randomly within this region.

  However, if we look at Figure 7.21 (see Appendix), we see that both the absolute values, as well as the absolute savings (difference with the naive policy) always increase with increasing imbalance, even in the highly unbalanced networks.

Next, we evaluate the performance of the various policies in the unbalanced network settings for the configurations 4N, 4R, 9N, and 9R. The results can be found in Figure 7.20. We draw the following conclusions:

- The results with respect to the dependency of the time-window length and time between jobs are similar to those given above. Again, the best performance in achieved with a combination of vehicle and shipper strategies.

- Going from a 4 node/region network to a 9 node/region network results in higher absolute values (see Figure 7.22 in the Appendix) which is simply caused by the fact that there are more empty moves required. Although we can not simply compare the 4 node/region networks with the 9 node/region networks (because the imbalance differs) we mention some results. Going from 4N to 9N, we see an increase in relative savings, especially for the opportunity valuation policies (OV, OV+DEC, OV+RES). The reason for this is that all policies, except the individual policy DEC, suffer from estimation errors in the 4N network because a continuous distribution is used to describe the lowest bid. Obviously, this results in estimation errors in a network with few nodes because there are only a limited number of possible bid prices. Going from 4R to 9R, we see that the absolute savings remain more or less the same for all policies (see Figure 7.22 in the Appendix) and therefore the relative savings decrease.

- Going from a node network to a region network results in higher absolute values (see Figure 7.22 in the Appendix) which is simply caused by the fact that there are more empty moves required. For the policies RES and DEC we see that the absolute savings compared to the naive policy NA remain approximately the same. As a consequence, the relative savings are slightly lower. A remarkable result is that the relative savings are much higher for all opportunity valuation policies (OV, OV+DEC, OV+RES) in the 4R network compared to the 4N network. The reason for this is that with the opportunity valuation policies, vehicles estimate the lowest bid from their competitors using a continuous distribution function. This results in estimation errors when there are only a limited number of possible bids, which is the case in the 4N network (cf. Section 7.5.1). The relatively poor behavior of the opportunity valuation policies can also be seen from (1) only in the 4N network the policy OV performs worse than the policy DEC and (2) the added value of OV to the policy RES is much higher in the 4R network than in the 4N network.

To summarize the results, we have seen that the combination of vehicle and shipper strategies always improves the performance compared to one of the individual policies. In almost all cases, the combination of the opportunity valuation policy and the decommitment policy works best. Only in settings with long time-windows or few jobs, the combination of the opportunity valuation policy and the dynamic threshold policy comes in favor. In almost all cases the relative savings of these policies lie between 10% and 20%. Although these results are promising, we also mentioned some problems in Section 7.5. In the next section we propose a direction for future research that might overcome some of these problems.

7.9 A promising research direction

In this chapter we mentioned several drawbacks of the proposed look-ahead policies of Chapters 5 and 6. To overcome these difficulties, we propose a direction for further research that consists of using approximate dynamic programming with reinforcement learning techniques. A great advantage of this is that we find the optimal value functions purely from experience without requiring a detailed model of the environment's dynamics, i.e., of its rewards and transition probabilities. This enables us to combine both strategies without the need of modeling the opponents' behavior. In order to speed up the learning process and to reduce computation time, we propose to use value function approximation.

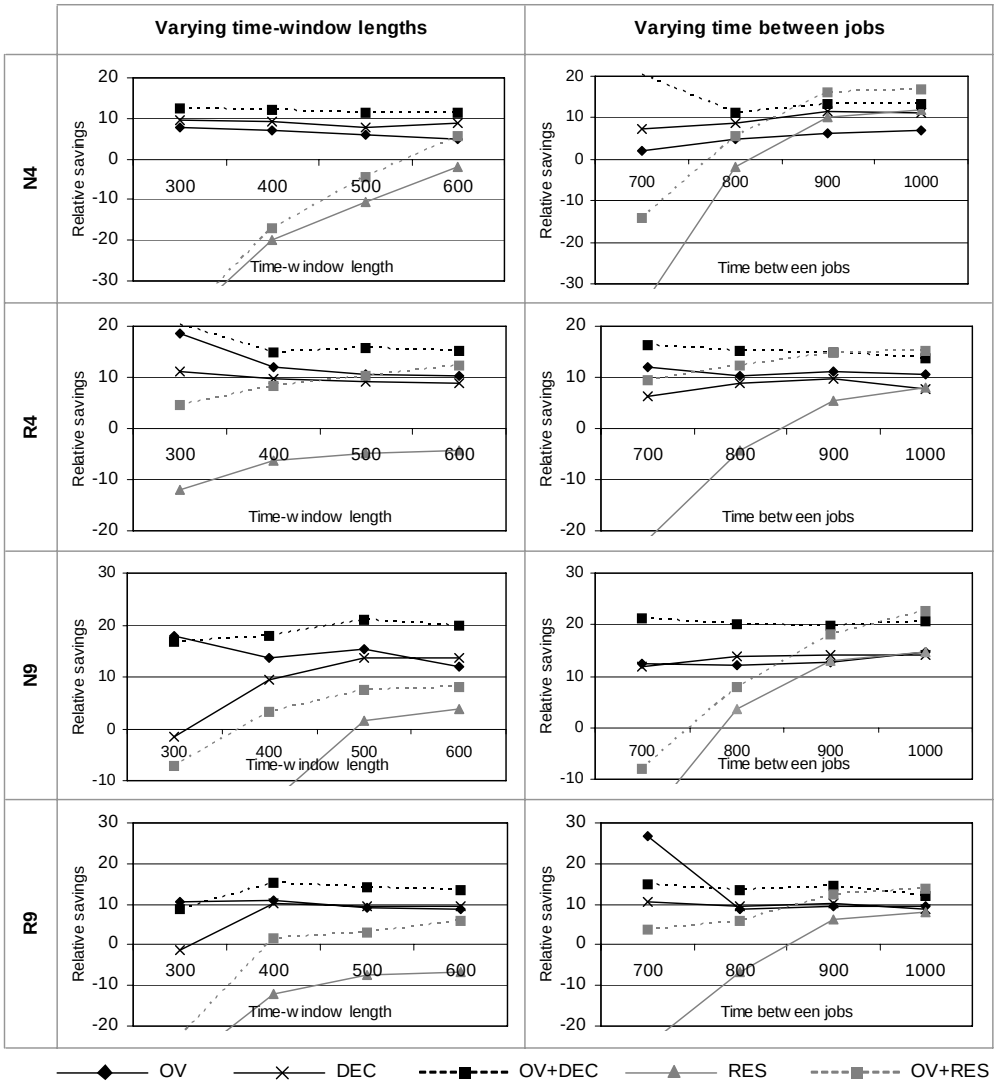

Figure 7.20: Relative savings for different values of the factors network structure, time-window length, and time between jobs

7.9.1 Value function approximation

In Chapters 5 and 6, we derived various value functions. These value functions are represented by tables with one entry for each state. This approach may become problematic when the state space becomes large. First, when using dynamic programming, we have to calculate the value of an increasing number of table entries. Second, when using reinforcement learning, we have to learn the value of many states. This makes the learning problem difficult because it is very unlikely that we experience exactly the same situation as we experienced before.

In order to allow generalization across states and a more compact representation of the value function, we use Value-Function Approximation (VFA). Here the tabular representation is replaced by a function approximator. There are many possible approximation architectures such as neural networks, decision trees, multivariate regression, self-organizing maps, and instance-based methods. A popular approximation in reinforcement learning relies on linear function approximators (Tsitsiklis and Van Roy, 1997). Here the value $V(s)$ of state s, is expressed as a linear weighted combination of k basis functions $\phi_j(s)$ (also called features):

$$\hat{V}(s) = \sum_{j=1}^{k} \phi_j(s) w_j \tag{7.5}$$

We used the notation $\hat{V}$ to indicate that this function is an approximation. The free parameters of this linear approximation - which have to be learned - are the coefficients w_j of the combination (also called weights). The basis functions $\phi_j(s)$ are fixed, but arbitrary functions of s. Therefore, the characterization "linear" refers to the way the free parameters enter into the architecture and not to the approximation ability of the architecture.

Typical linear approximation architectures are polynomials of any degree (each basis function is a polynomial term), radial functions (each basis function is a Gaussian with fixed mean and variance), piece-wise linear functions (each basis function is a linear function), and coarse coding (each basis function is an indicator function). The choice for a particular approximation depends on the characteristics of the value function. It especially depends on the parameters for which we believe they are necessary to explain the value of a particular state. We illustrate this using the end-values $V^e(i, \sigma, t)$, where i is a location in the network, σ the time-to-go until arrival at the schedule destination, and t a time period.

First, consider the approximate end-value $\tilde{V}^e(i, t)$. As shown in Section 7.5.1, the value $\tilde{V}^e(i, t)$ increases linearly in t for a given schedule destination i. To describe this dependency, we may use (1) a set of indicator functions $\phi_j(i)$ which equal one if $j = i$ with schedule destination i and (2) a basis

function t to describe the dependency on the planning period t. We propose the following approximation:

$$\hat{V}^e(i,t) = tw_t + \sum_{j \in \mathcal{N}} \phi_j(i) w_j \tag{7.6}$$

To derive a value function approximation for $V^e(i,\sigma,t)$ we have to include the time-to-go σ. Luckily it is possible to use a separable approximation for the dependency on the time-to-go σ, because the slope of the linear function does not depend on it. Moreover, for given i and t, the value $V^e(i,\sigma,t)$ is concave in σ: a larger time-to-go σ reduces the expected waiting time at location i, however, the added value will decrease with increasing time-to-go. This concave function can be described by a piecewise linear concave function (see Powell et al., 2005). To be precise, we add a set of basis functions in the linear approximation of (7.6) corresponding with each segment of the piecewise linear concave function.

7.9.2 Reinforcement learning

By using VFA, we replace the tabular representation of our value functions by a function approximator. A remaining task is to learn the weights of the function approximator. We propose to do this by using reinforcement learning, more specifically, by using temporal difference learning.

Temporal difference learning, originally proposed by Sutton and Barto (1998), is a method which uses experiences to progressively learn the value function. The basic idea is that we update a value function $V(s)$ based on the difference between the existing value $V(s)$ and the value $V(s')$ of a state s' that is encountered after state s. We denote subsequent states by $(s_u, s_{u+1}, ...)$ with corresponding rewards $(r_u, r_{u+1}, ...)$. The simplest TD method, known as TD(0), is given by:

$$V(s_u) \leftarrow V(s_u) + \alpha \left[R_u^1 - V(s_u)\right] \tag{7.7}$$

where R_u^1 denotes the so-called one-step return which is given by $R_u^1 = r_{u+1} + V(s_{u+1})$. So the existing value of the state s_u is updated directly after the next state $s_{u+1}$with a new observation R_u^1 using an exponential smoothing factor α. The n-step return is given by:

$$R_u^n = r_{u+1} + r_{u+2} + .. + r_{u+n} + V_u(s_{u+n}) \tag{7.8}$$

For more on these and other return functions we refer to (Sutton and Barto, 1998). Now, let us return to the approximate end-value $\tilde{V}^e(i,t)$. Here a state transition takes place each time a new job is won. The TD(0) method is less appropriate for this case because (1) the profit/costs of an individual job is hard

to calculate and (2) there are large fluctuations in bid prices. Therefore, we propose the n-step return for n sufficiently large. After appending a job ending at node i, we store an entry for this state. Next, we keep record of n jobs that we won afterwards. After the n^{th} job, we calculate the profit r_n of all these jobs and the total time t_n that passed between the scheduled delivery time of the n^{th} job and the first job. The approximate end-values can be calculated as follows:

$$\tilde{V}^e(i,t) = \tilde{V}^e(i,t) + \alpha \left[r_n + \tilde{V}^e(j, t - t_n) - \tilde{V}^e(i,T) \right] \tag{7.9}$$

It is natural to update the value function using gradient-descent methods. The gradient of the linear function approximation with respect to the vector of weights w_i simply equals the vector $\phi_i(s)$ of basis functions. For the end-value $\tilde{V}^e(i,t)$ this means that we add an amount $\alpha \left[r_n + \tilde{V}^e(j, T - t_n) - \tilde{V}^e(i,T) \right]$ to (1) the weight w_i of node i and to (2) the weight w_t of the time t. For more details on gradient-descent methods we refer to (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998).

7.9.3 Advantages and disadvantages

In the previous chapters, we calculated the value functions using a dynamic programming recursion. This results in large tables which describe the value of all states. There are two problems with this: (1) it is a time-consuming process and (2) in highly dynamic environments (i.e., opponents also use dynamic policies) we might use outdated or even the wrong value functions. When the value functions do not match with reality (because we have made some assumptions in order to calculate them) we never discover this discrepancy, let alone that we update the value functions accordingly.

The advantage of temporal difference learning is that we find the optimal value functions only from experience, without requiring a detailed model of the environment's dynamics, i.e., of its rewards and transition probabilities. This enables us to combine both strategies without the need of modeling the opponents' behavior. Temporal difference learning with value function approximation further offers the possibility to add additional explanatory variables with relatively little effort. For example, in Chapter 5 we mentioned the difficulties of using the time-to-go σ in the dynamic programming recursions. However, it is rather easy to include it in the linear approximation. Another example is the transportation area. In the linear approximation of (7.6), we used indicator functions for all nodes $i \in \mathcal{N}$. For continuous networks, such an approach would mean that we have to discretize the transportation area. For example consider the region networks. An option here is to express the end-value as a weighted combination of the value functions of the 4 corner points. It is important to note that we also update the value functions according to this weighted combination (instead of storing the value of the undiscretized ob-

servation). Another option is to use a special form of coarse coding called tile coding (CMAC). Tile coding is a method which involves multiple overlapping grid-tilings (discretizations) that are slightly moved. Each particular state is characterized by the tiles of each grid-tiling in which it appears. For each tile we have a basis function which indicates if a state belongs to this tile. So this can be regarded as a discretization in which an update of a certain state not only affects the value of this state, but also of neighboring states. Such an approach provides more flexibility.

Besides the advantages there are also some difficulties. We performed some preliminary experiments of the proposed linear end-value approximations with temporal difference learning. In these experiments we encountered some problems with this approach. First, we encounter the same difficulties as mentioned in this section for the dynamic programming based policies: prices increase in closed environments and we may get fluctuations. Second, observations in a dynamic environment - like we study here - fluctuate enormously and as a consequence the one-step backup results in unstable behavior. Therefore, we have to be careful in our choice of learning periods and smoothing factor. Third, the learned behavior affects our bid prices which in turn affect the future profits. This effect, the future impact of current decisions, is not explicitly taken into account. A possible solution is the use of Q-learning where we learn the value of state-action pairs, i.e., the value of making a certain decision (e.g. bid calculation) in a given state (e.g. location i and period t). Obviously, more research is required into this topic.

7.10 Conclusions

In this chapter we studied a closed transportation market where shippers offer full truckload pickup-and-delivery jobs with due times through sequential auctions. A set of vehicles compete with each other to serve these jobs. For the shipper agent we considered two auction strategies, namely a dynamic threshold policy and a decommitment policy. For the vehicles we considered opportunity valuation policies where not only the direct costs of jobs are taken into account, but also the impact on future opportunities.

We used simulation to evaluate the benefits of the different strategies and to study their interrelation. Our main conclusions are the following:

- The combination of vehicle and shipper strategies always performs better than the individual policies. On average we observe a reduction of 10-20% in the costs for tardiness and repositioning of the vehicles.
- The combination of the opportunity valuation policy and the decommitment policy works best in almost all cases. The combination of the

opportunity valuation policy with the dynamic threshold policy comes in favor in settings with long time-windows or fewer jobs.

- The performance of the individual policies depends a lot on the network structure and job characteristics. In other words, each policy has its own characteristics. The opportunity valuation policies of the vehicles benefit from the imbalance in the network where some regions are more popular than others. These policies are therefore especially suitable for unbalanced networks. The dynamic threshold policy and decommitment policy of the shipper benefit from fluctuations in bid prices due to the possibilities of combining jobs. The decommitment policy is especially suitable for balanced networks. The dynamic threshold policy is especially suitable for settings with long time-windows or fewer jobs.

- In contrast to the results of open environments, now the relatively simple policies work very well compared to the more computationally demanding policies. For the opportunity valuation policy we showed that the approximation based on a zero time-to-go yields approximately the same performance as the more precise policy based on an average time-to-go. For the shipper strategies we have seen that the decommitment policy works best. This is remarkable because this policy is relatively simple in the sense that we only have to estimate the lowest bid instead of using a time consuming recursion (as required for the other policies). In most cases the performance of the decommitment policy is close to the opportunity valuation policy (which in many cases is the best among the individual policies).

We also investigated problems that might occur (1) when all jobs are auctioned under a dynamic threshold policy or decommitment policy and (2) when all vehicles use opportunity costs. The main problem for the opportunity valuation policies appeared to be a continuous increase in bid prices. However, we could easily correct this. Another problem is related to the fluctuations in the parameter settings. However, it appeared that these fluctuations flatten out relatively fast. The main problem for the dynamic threshold policy and decommitment policy is that it takes quite some time to learn the right parameters. As a consequence, these methods would be less applicable to fast changing environments. To overcome these problems, we proposed a new research direction. This approach consists of using approximate dynamic programming in combination with temporal difference learning. We mentioned the advantages of such an approach, but also the difficulties in designing and implementing such a method. We argue that more research is required on this topic.

7.11 Appendix

Measuring the realized profits

We measure the realized profits within the planning horizon T in a given learning period as follows:

1. Each time t a vehicle v adds a new job to the end of its schedule, we create an open entry e_{vt}.
2. Each time t' a vehicle incurs costs or receives a payment for a job that is auctioned at time t'', we add this value to all open entries e_{vt} of this vehicle for which $t < t''$ (so this value is only added to the entries of jobs that are auctioned before this job). For all open entries e_{vt} for which $t < t' - T$ we only add a proportional fraction to the entry e_{vt} and close this entry. We assume that costs are incurred directly after they have been made (after each move, and load/unload action, penalties after delivery, and prices directly after winning).
3. At the end of each learning period, we (1) calculate the realized profits from the mean values of all closed entries and (2) delete all entries.

Simulation results

NA	OV	DEC	RES	OV+DEC	OV+RES
67.8 / 97.9	69.7 / 98.6	69.5 / 99.4	69.4 / 94.9	70.9 / 99.6	70.6 / 96.0
67.8 / 97.9	69.4 / 98.4	69.6 / 99.2	69.5 / 95.6	70.8 / 99.5	70.8 / 96.1
67.8 / 97.9	69.7 / 98.6	69.5 / 99.2	69.3 / 96.2	71.1 / 99.4	70.9 / 96.6
67.8 / 97.9	69.6 / 98.5	69.6 / 99.1	69.2 / 96.1	70.9 / 99.6	70.8 / 96.8
67.8 / 97.9	69.6 / 98.5	69.6 / 99.3	69.1 / 96.1	71.0 / 99.6	71.0 / 96.6
67.8 / 97.9	69.7 / 98.6	69.6 / 99.3	69.2 / 96.3	70.9 / 99.6	70.9 / 96.5
67.8 / 97.9	69.6 / 98.6	69.6 / 99.3	69.2 / 96.3	70.8 / 99.5	70.9 / 96.6
67.8 / 97.9	69.6 / 98.6	69.7 / 99.2	69.3 / 95.8	70.9 / 99.6	70.9 / 96.6
67.8 / 97.9	69.6 / 98.6	69.6 / 99.2	69.4 / 96.0	70.9 / 99.6	70.9 / 96.5

Table 7.3: Simulation results for various policies with respect to the percentage of driving loaded (value before the slash) and the service level (value after the slash)

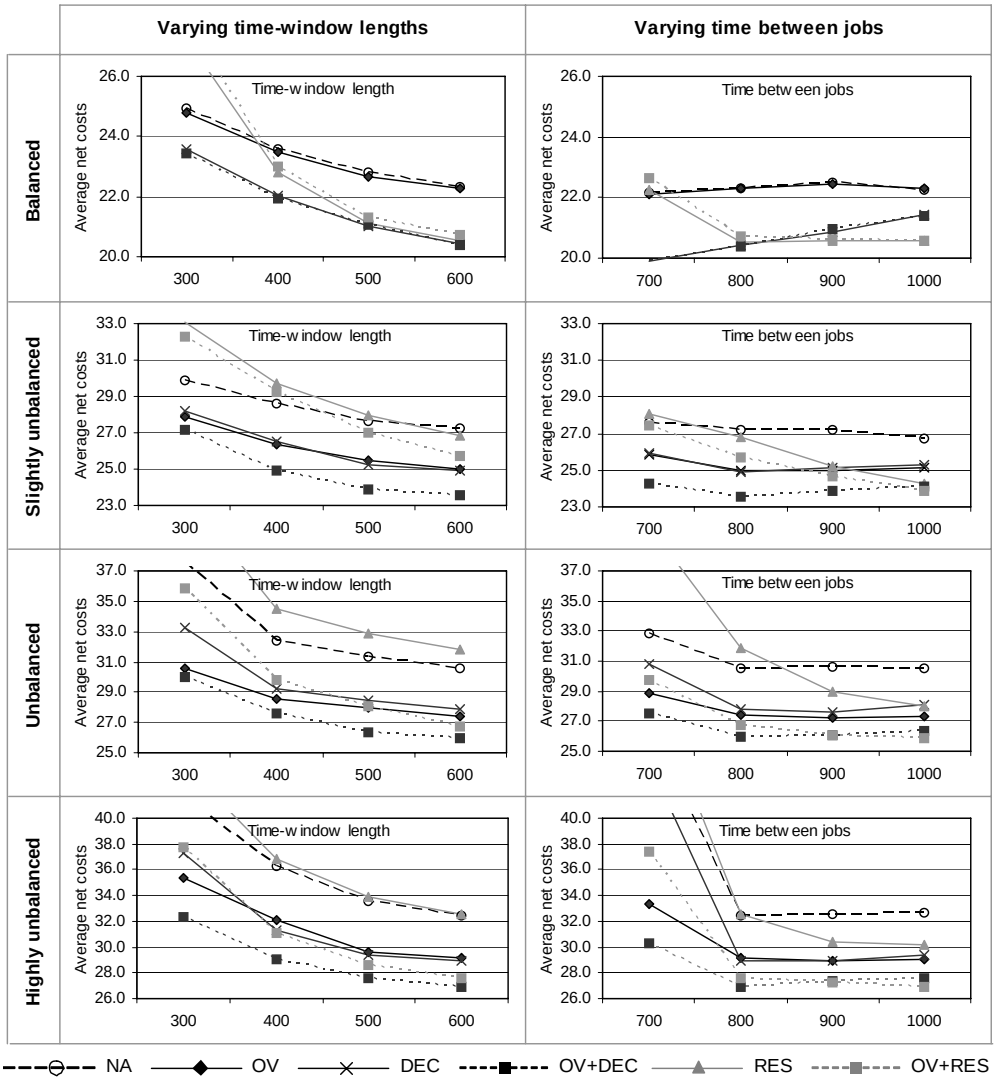

Figure 7.21: Average net costs for different values of the factors degree of balance, time-window length, and time between jobs

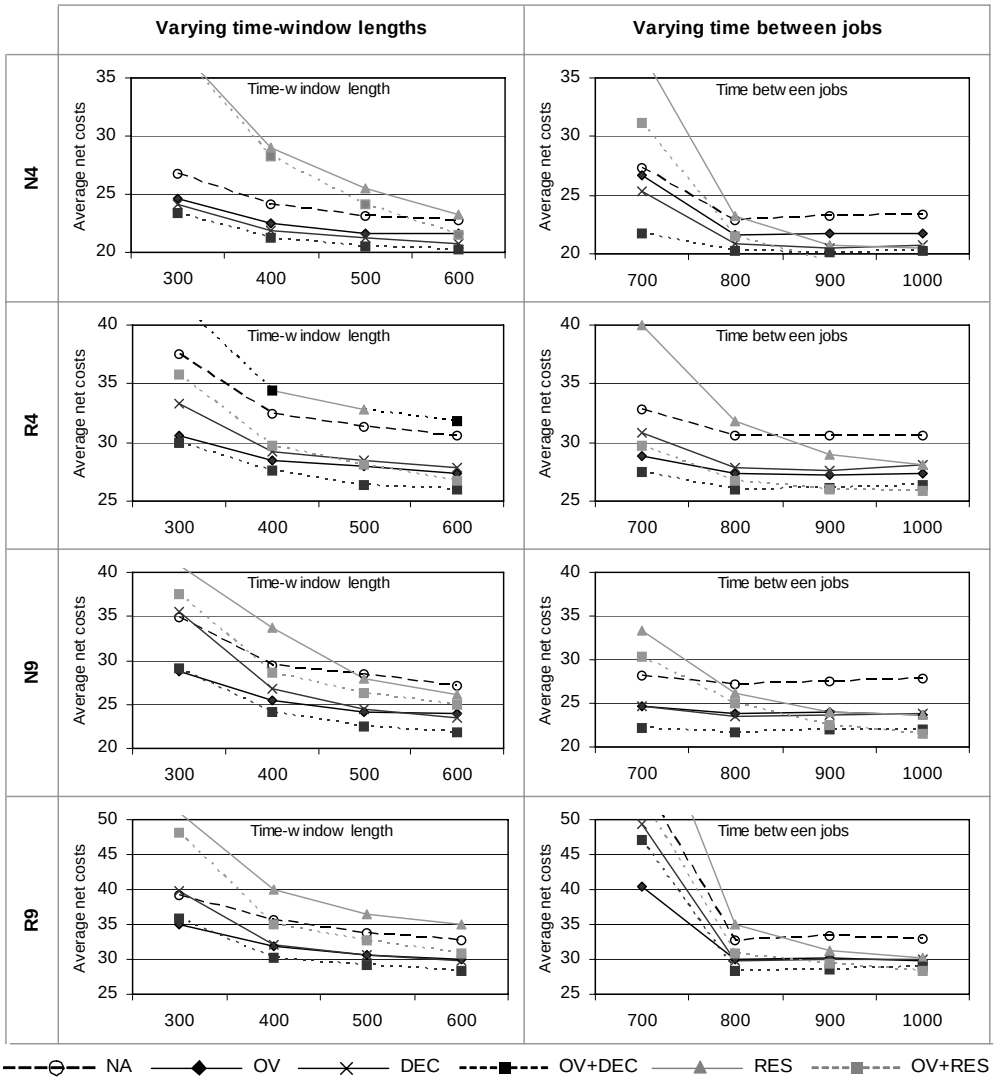

Figure 7.22: Average net costs for different values of the factors network structure, time-window length, and time between jobs

Chapter 8

Conclusions and further research

In this chapter we summarize the results of this thesis taking into account the research questions introduced in Chapter 1. Furthermore, we give directions for further research.

8.1 Conclusions

The initial motivation for this research was to investigate the usefulness of the multi-agent system (MAS) concept for real-time transportation planning and scheduling. In this multi-agent concept, resources and/or tasks are represented by intelligent and autonomous computational entities (agents), which coordinate their capacities in order to achieve certain (local or global) goals. This concept has turned out to be a promising solution for controlling complex networks, providing more flexibility, reliability, adaptability, and reconfigurability. However, it raises three important questions. First, it is unclear whether the system-wide performance of an agent-based approach will be similar or even better than the performance of more centralized and hierarchically organized planning systems. Second, it is unclear how different design choices affect the logistical performance. Third, it is unclear how intelligent agents that represent different interested parties in transport (shippers and carriers) interact and how this influences the system-wide logistical performance.

An important component in multi-agent systems is the auction mechanism that is often used as a coordination mechanism among the agents. This raises a new problem, namely how agents should price their resources and tasks. This is especially hard if agents should anticipate on future tasks and resource

utilization. We decided to study these strategies not only from the perspective of a closed multi-agent system (a *closed environment* where we aim to minimize the system-wide logistical costs), but also from the perspective of a single player participating in an open auction setting (an *open environment* where we aim to maximize the profitability of the individual agent for given behavior of the other players). These problems have led us to the following research objective:

To analyze in which way and to what degree multi-agent systems can be used for real-time operational planning and control of transportation networks. Further, to develop strategies for players in sequential transportation procurement auctions, and to analyze their performance in terms of both the individual benefits for the players and the system-wide logistical costs.

To reach this goal, we posed five research questions in the introduction (Chapter 1), which we answered in the subsequent chapters of this thesis. In this section we present our conclusions concerning these research questions.

8.1.1 Research question 1

How does the performance of a multi-agent system compare to traditional OR-based systems in terms of (1) effectiveness, i.e., the ability to handle jobs according to specified targets, such as delivery time windows; (2) efficiency in terms of the utilization of resources and logistic costs; and (3) robustness against fluctuations in demand in terms of variation in system effectiveness and efficiency?

In Chapter 3 we proposed a distributed agent-based solution to real-time, dynamic transport scheduling problems. We compared this approach with two hierarchical look-ahead heuristics (called *LocalControl* and *SerialScheduling*). These methods were originally developed for a case study on a proposed underground transportation system at Amsterdam Airport Schiphol, the Netherlands. We refer to this application as the OLS case, which is the Dutch abbreviation for underground logistic system. We used the simulation test environment that had been developed for the OLS case, to compare the performance of our agent-based system with the two more traditional transportation planning heuristics. In addition, we simulated and compared the different control methods in a more general transportation network.

From our simulation experiments, we concluded that our agent approach yields a high performance in terms of vehicle utilization and delivery reliability. With respect to vehicle utilization, we have seen that our agent-based scheduling method always performs significantly better than the two hierarchical methods. With respect to delivery reliability, the agent approach performs significantly better in most cases and never significantly worse. In addition, we

have seen that the agent approach is more robust in the sense that its performance is less sensitive to fluctuations in demand or available vehicles compared to the more traditional transport planning heuristics.

We also proposed two extensions of the agent-based approach. First, we proposed an extension called *Trade*, which allows vehicle agents to exchange jobs. Second, we proposed a dynamic threshold policy which allows the shipper agents to reject all bids and to start a new auction later on. These extensions always improved the performance, but appear to be particularly valuable if vehicle schedules contain many jobs on average. Our dynamic threshold policy has been considered again and improved in Chapter 6 (research question 4).

8.1.2 Research question 2

How should a multi-agent system for material sourcing and scheduling of physical distribution be designed in terms of tasks, competences, responsibilities, and goal-directed behavior?

In Chapter 4 we studied the effect of various MAS design choices on the logistical performance. Designs may vary in the roles and responsibilities assigned to the agents, the level of intelligence of the agents (forecasting and learning behavior), and the interaction protocols selected. We concluded that current MAS methodologies lack a mechanism to evaluate such design-choices and provide only limited support to the designer in selecting the preferred design for implementation. Therefore, we proposed to extend current MAS methodologies by multi-agent discrete event simulations.

To demonstrate and test this approach, we applied it to a real life project: the design and development of a multi-agent system for the manufacturing of biscuits at the industrial bakery Merba in the Netherlands. To illustrate the design process, we considered a simplified part of the dough production process at Merba bakeries. By using a stepwise approach, built upon existing MAS development methodologies, we already derived eight alternative designs for this part only. By using qualitative arguments, we were able to reduce this to four alternative designs. In order to select the preferred design for implementation, we used multi-agent discrete event simulation.

This simulation gave us insight into the effect of our MAS design choices on the system performance in terms of delivery punctuality, product quality, robustness, amount of communication, and computation time of the different agents. Our simulation study showed that qualitative arguments are not sufficient because each alternative design has merits of its own. The main conclusion here is (1) to be aware that alternatives exist, moreover, that possibly there is no single best architecture and (2) that these alternative designs should be simulated using different scenarios that might occur in the intended implementation environment.

Although we illustrated our design approach by developing a multi-agent system for the control of AGVs at an industrial bakery, our results are more generally applicable. In Chapter 4 we provided insight into the design choices and improved current MAS development methodologies to offer enhanced support in cases where multiple alternative decision and communication scenarios exist. In a wide range of MAS application areas where different actors collaborate, such method support will be beneficial. Also the proposed multi-agent system itself provides insights that can be generalized to other situations, especially regarding the way agents balance different delivery criteria in the scheduling of jobs.

8.1.3 Research question 3

How can we use information on historic job patterns and auction data to improve the pricing and scheduling of vehicles participating in transportation procurement auctions?

In Chapter 5 we presented a real-time opportunity based bid pricing and scheduling strategy for vehicle agents, where not only the direct costs of a new job insertion are taken into account, but also its impact on future opportunities. The decisions of a vehicle have an impact on its future profitability because some regions within the transportation area are more attractive than others. As an example, suppose a vehicle receives an announcement for a job going to an unattractive location. The probability of receiving another job that leaves from this unattractive location is low and hence it is likely that the vehicle has to wait at this location or has to make an empty move towards another location. This certainly results in a loss of future revenues and the vehicle should include this loss in its bid price. If the vehicle, despite its high bid, still wins the auction for this job, it also has to be careful how to schedule this job because it has an impact on the probabilities for winning other jobs.

To deal with these issues, we include probabilistic knowledge about future job arrivals in the current decisions. Information on job arrivals and on the competitors' bid prices can be acquired by participation in the auctions. We use this information to value the opportunities of future job insertions within a given schedule. Because we consider an insertion scheduling heuristic, future job insertions can only take place between two jobs currently in the schedule or after the last job in the schedule. By valuation of these periods we are able (1) to choose the most appropriate insertion position and pickup time of the new job and (2) to calculate the opportunity costs which are defined as the loss in expected future revenues due to a new job insertion. Including this value in our bid pricing and scheduling decisions, prevents less profitable moves, and increases opportunities by anticipation of future transport demand by better prepositioning of vehicles.

We used simulation to evaluate the proposed approach. From these simulation experiments we concluded that an individual player using opportunity based bid pricing will perform significantly better than other players who use a naive pricing strategy. For example, in an open environment with 10 vehicles, we showed that the profit of the individual vehicle is in some cases higher than the total profit of the 9 other vehicles. Besides the profitability of opportunity based bid pricing and scheduling, we also see an increase in vehicle utilization and service levels. For closed environments (internal use of our approach), we showed that opportunity based bid-pricing and scheduling can reduce the system-wide logistical costs (an average reduction of 10% in the costs for empty moves and penalties on tardiness).

We explain the benefits of the proposed approach from the following behavior. First, the vehicle agents tend to schedule unattractive jobs later in time (at the end of their schedule), thereby increasing the probability that these jobs can be combined with other jobs. Second, if vehicle agents have to agree on the pickup times of jobs in advance, then they tend to schedule idle time before the pickup times of unattractive jobs, thereby increasing the possibilities of (partly) replacing the empty moves by one or more loaded moves. Third, if the pickup times of jobs are not fixed in advance, then the vehicle agents tend to have fewer empty moves in their schedule which results in shorter schedule lengths (difference between the expected delivery time of the last job and the current time). These shorter schedule lengths provide more flexibility for future jobs insertions, i.e., increase the probability that these jobs can be scheduled before their due time. Fourth, in case of open markets, an individual vehicle using opportunity based bid pricing and scheduling tends to obtain the most profitable jobs.

8.1.4 Research question 4

How can we use information on historic auction data to improve the auctioning strategy of shippers to procure their transportation services?

In Chapter 6 we proposed two policies for shippers which may reduce their costs for transportation. Both policies use the potential provided by probabilistic information on the price evolution in time for various job characteristics. The idea of the first policy, called dynamic threshold policy, is that shippers postpone commitments - by setting a time dependent upper bound on the job prices - for which they expect to make a better commitment in the future. So if the shipper has plenty of time to auction a certain job, it will not agree with a relatively high bid. When the time for dispatch gets closer, the price it is willing to accept will rise. The idea of the second policy, called decommitment policy, is that the shipper allows a carrier to decommit from an agreement against a predefined time dependent penalty that is calculated by the shipper itself. These penalties are chosen such, that whenever a carrier decommits a

job, they cover the extra costs of the shipper for finding a new carrier. The dynamic threshold prices and the decommitment penalties are calculated using a dynamic programming recursion.

The benefits of both strategies have been evaluated with simulation. From our simulation experiments, we conclude that in open environments, the costs per job for a shipper using the dynamic threshold policy are significantly lower than for shippers who did not use such a threshold policy (20-30% reduction in the costs for empty moves and penalties on tardiness). However, the decommitment policy - since we derived it without profit margins - results in relatively higher costs per job for the individual shipper, compared to the other shippers. We also considered closed environments where all jobs are auctioned by using one of the two strategies. We found that both policies reduce the total system-wide logistical costs; especially the decommitment policy which yields savings of 13-16% in the costs for empty moves and penalties on tardiness. We also considered the combination of the two policies; we found that this results in a slightly better performance at the expense of a major increase in computation time. Hence, our overall recommendation is to use the dynamic threshold policy for open market settings where only a limited number of other shippers are using such a policy, and to use the decommitment policy for closed environments.

8.1.5 Research question 5

What is the impact of the different pricing and scheduling strategies for carriers and shippers on the system-wide logistical performance?

In Chapter 7 we studied the interrelation between shipper and carrier strategies in closed networks, where we measure the performance in terms of system-wide logistical costs. Our main conclusions are the following:

- The combination of vehicle and shipper strategies always performs better than the individual policies. On average we observe a reduction of 10-20% in the costs for tardiness and repositioning of the vehicles. The combination of the opportunity valuation policy and the decommitment policy works best in almost all cases. The combination of the opportunity valuation policy with the dynamic threshold policy is to be preferred in settings with long time-windows or fewer jobs.
- The performance of the individual policies depends a lot on the network structure and job characteristics. In other words, each policy has its own characteristics. The opportunity valuation policies of the vehicles benefit from the imbalance in the network where some regions are more popular than others. These policies are therefore especially suitable for unbalanced networks. The dynamic threshold policy and decommitment

policy of the shipper benefit from fluctuations in bid prices due to the possibilities of combining jobs. The decommitment policy is especially suitable for balanced networks. The dynamic threshold policy is especially suitable for settings with long time-windows or fewer jobs.

- In contrast to open environments, now the relatively simple policies work very well compared to the more computationally demanding policies. For the opportunity valuation policy, we showed that the approximation based on a zero time-to-go yields approximately the same performance as the more precise policy based on an average time-to-go. For the shipper strategies, we have seen that the decommitment policy works best. This is remarkable because this policy is relatively simple in the sense that we only have to estimate the lowest bid instead of using a time consuming recursion (as required for the other policies). In most cases the performance of the decommitment policy is close to the opportunity valuation policy (which in many cases is the best among the individual policies).

We also investigated problems that might occur (1) when all jobs are auctioned under a dynamic threshold policy or decommitment policy and (2) when all vehicles use opportunity costs. The main problem for the opportunity valuation policies appeared to be a continuous increase in bid prices. However, we could easily correct this. Another problem is related to the fluctuations in the parameter settings. However, it appeared that these fluctuations flatten out relatively fast. The main problem for the dynamic threshold policy and decommitment policy is that it takes quite some time to learn the right parameters. As a consequence, these methods would be less applicable to fast changing environments.

8.2 Further research

We distinguish three areas for further research: model improvements, model extensions, and implementation aspects.

8.2.1 Model improvements

We subsequently present the following model improvements: (1) job exchange by vehicles, (2) architecture adaptability, (3) geographic regulation of the vehicles, and (4) learning in closed environments.

Job exchange by vehicles

In Chapter 3 we proposed an option, denoted by Trade, which allows the vehicles to exchange jobs. Each time a vehicle has to travel empty towards another

location, its agent searches for another vehicle agent that has a job for which it is most beneficial to exchange. This approach has some similarities with the decommitment concept of Chapter 6 where vehicles are allowed to break an agreement with a shipper. However, in a transportation market with multiple carriers each having their own fleet of vehicles, it would be preferable that vehicles within one fleet also can exchange jobs. In that case we have in fact decommitment of vehicles towards an agreement with their carrier. Whenever a vehicle decommits, the carrier starts a new auction where only its own vehicles participate. Of course, this requires some modifications of the current decommitment concept. In particular, the carrier should regulate the exchange of jobs to limit the amount of communication and the computation time. Examples of this regulation include: (1) setting the time-dependent internal decommitment penalties, (2) decide about the time at which vehicles can decommit, and (3) decide which subgroup of vehicles receive an announcement for the decommited job (e.g. neighboring vehicles).

Architecture adaptability

In Chapter 4 we showed that the best multi-agent architecture depends on the network characteristics such as the arrival intensity of jobs, the time-windows of jobs, and the variability in handling times. As a consequence, if the network characteristics change we also have to adjust the multi-agent architecture. Therefore, we plan to investigate the adaptability of multi-agent systems to change their design depending on the system status. For example, in the case study presented in Chapter 4, AGVs may use a different scheduling technique based on the system status (or the architecture itself may even be changed dynamically). To be more concrete, we may use an architecture where all vehicle agents maintain a detailed schedule of jobs (e.g. architecture LC5i) during the night (when it is relatively quiet) and dynamically switch to a dispatching architecture (e.g. architecture AC) whenever we observe increasing congestion (normally during daytime).

Geographic regulation of vehicles

In Chapter 5 we proposed a pricing and scheduling strategy for carriers where not only the direct costs of a new job insertion are taken into account, but also its impact on future opportunities. Our approach decomposes the problem into a multi-agent structure where vehicle agents are responsible for the routing and scheduling decisions, and where the assignment of jobs to vehicles is done by using a second-price auction. All vehicles act selfishly and do not take into account the position of other vehicles within the network. As a consequence, it might be the case that multiple vehicles decide to move to the same region at the same time, thereby increasing their expected waiting times. A particularly interesting model improvement is the regulation of the number of vehicles at

different locations. Now suppose the vehicles have some knowledge about the location of other vehicles at different points in time. For example, they have information n_{it}, which gives the number of vehicles having node i in their schedule, arriving there within a 'certain time period' t having 'some flexibility' to add new jobs after it. Although this definition is still vague, it can be imagined that the information n_{it} can be used to improve the pricing and scheduling decisions. To be precise, this number has an effect on the winning probabilities, transition probabilities, and expected rewards in the dynamic programming recursions that are used to calculate the opportunity costs.

Currently, the opportunity costs for a vehicle that is willing to do a new job from k to l, directly after arrival at its schedule destination i, is given by the decrease in end-values $V^e(i,T) - V^e(l, T-\tau_{ikl})$, where τ_{ikl} is the time required for an empty move from i to k and a loaded move from k to l. To illustrate what happens when we include the number of vehicles n_{it} in the opportunity costs, we introduce the notation t_1 to indicate the expected arrival time of the vehicle at its schedule destination i and t_2 to indicate the expected arrival time at the destination l of the new job. Then the opportunity costs are given by $V^{e-}(i,T,n_{it_1}) - V^{e+}(l, T-\tau_{ikl}, n_{lt_2})$, where $V^{e-}(i,T,n_{it_1})$ is the marginal contribution of one less vehicle at node i at time t_1 ($n_{it_1}-1$ instead of n_{it_1}) and $V^{e+}(l,T-\tau_{ikl},n_{lt_2})$ is the marginal contribution of an additional vehicle at node l at time t_2 ($n_{lt_2}+1$ instead of n_{lt_2}). Within a closed environment, such as a shipper with a private fleet, one can imagine that this information is made available by the shipper to its vehicles. However, it is also possible to acquire probabilistic knowledge on the position of the other vehicles, by using historical observations of the job characteristics (arrival rate of jobs on different routes) and the lowest bid of the competitors.

Learning in closed environments

In Chapters 5 and 6, we proposed profit maximizing strategies for carriers and shippers participating in open environments where we focused on strategies of an individual agent and ignored the impact of the other agents. In Chapter 7 we applied the strategies to a closed environment where we are less interested in the profitability of individual agents, but rather in the reduction of system-wide logistical costs. The look-ahead strategies require the agents to estimate the behavior of the other agents. As a consequence, it is assumed that the behavior of other agents remains the same. Obviously, this no longer holds when the other players also use strategic learning policies. We decided to use the same dynamic programming recursions and we have seen that these methods also perform well in closed environments. However, we have seen that applying the strategies to all players in a closed environment results in some problems. A topic for further research is to specifically redesign the proposed methods of Chapters 5 and 6 for closed environments. There are some extensions possible where players explicitly incorporate the behavior of

others in their decision making. For example, in a closed environment, vehicles can use historical information about job characteristics to estimate the location of other vehicles.

Another way to overcome these problems is to use approximate dynamic programming with reinforcement learning techniques as proposed in Section 7.9. A great advantage of this is that we find the optimal value functions purely from experience without requiring a detailed model of the environment's dynamics. This enables us to combine both strategies without the need of modeling the opponents' behavior. Temporal difference learning with value function approximation further offers the possibility to add additional explanatory variables with relatively little effort, as we illustrated in Section 7.9. However, there are also some difficulties with this approach as mentioned in Section 7.9. Therefore, more research is required into this topic.

8.2.2 Model extensions

We subsequently present the following model extensions: (1) less-than-truckload routing, (2) other auction mechanisms, and (3) timing constraints.

Less-than-truckload routing

In this thesis we considered full truckload (FTL) transportation where one load uses all available space in a tractor trailer. In contrast, less-than-truckload (LTL) transportation involves the transportation of loads that are too small to fill an entire truck. An LTL shipment is delivered with various other shipments and is usually not delivered directly to a destination as full truckloads are. The main advantage of using an LTL is that a shipment may be transported for a fraction of the costs of hiring an entire truck.

The LTL extension mainly affects the carrier and vehicle strategies as presented in this thesis. The main difficulty of LTL is that routes are made up by mixed pickups and deliveries for various jobs. This has an effect (1) on the marginal costs calculation of vehicles as described in Chapters 3 and 4 and (2) on the opportunity costs calculation described in Chapter 5. The marginal costs of an LTL insertion are not straightforward to calculate. For example, if an empty truck receives a long distance job, its fixed costs for doing this job are relatively high, independent on the size of the load. If new jobs are offered to this truck that lie along the route of this long distance job and there is enough space in the truck to transport these jobs, then the marginal costs for doing these jobs are close to zero.

One way to deal with LTL is to use approximate dynamic programming (see Section 8.2.1, learning in closed environments). Here we may learn the value of free capacity as a function of the time and spatial dimension of a vehicle. An-

other way is the use of revenue (or yield) management, as proposed in (Douma et al., 2006). As mentioned in Chapter 1, Revenue Management is an economic technique to increase revenues by accurately matching the available capacity (or product/service availability) with the market prices based on demand forecasting. There is a lot of literature on this topic with well-known applications in air transport (see for an overview McGill and Van Ryzin, 1999; Talluri and Van Ryzin, 2005). Revenue Management is of especially high relevance in cases where (1) the fixed costs are relatively high compared to the variable costs, (2) there is a fixed amount of perishable capacity, (3) demand can be predicted, (4) advance reservation is possible, and (5) different customers are willing to pay a different price for using the same amount of capacity. For example, in the passenger airline case, capacity is regarded fixed because the route is fixed. If the aircraft departs, the unsold seats cannot generate any revenue any more. To maximize profits, the seats are sold at different prices depending on the remaining time until departure and the number of available seats. In vehicle routing we face a similar problem when a vehicle fixates a route by accepting loads. Fixing a route results in fixed costs for the vehicle, because it now has to travel that route for that load. To maximize profits, a vehicle aims to fill its trucks with additional loads. If the vehicle sets the prices too high, then there is a risk to lose customers and profits. However, setting the price too low results in a lot of customers, possibly more than the vehicle can transport. To support the pricing decisions we can not simply apply the airline revenue management ideas because there is an important distinction: the flight schedules are predetermined whereas the vehicle has to decide which route to travel. Obviously, more research is required on this topic.

Other auction mechanisms

Throughout this thesis, we used a simple sealed-bid auction in which the auctioneer (the shipper) announces a job (pickup and delivery route) and a group of bidders (carriers or vehicles) submit their bids in sealed envelopes. The auctioneer then reviews the bids and determines the winner. This process is repeated separately for each job. This approach ignores the interdependencies among jobs. To overcome this we introduced a decommitment policy and dynamic threshold policy in Chapter 6. Another approach is to use a continuous double auction or the combinatorial auction.

In the continuous double auction (CDA), buyers and sellers continuously place offers and a transaction occurs as soon as a buyer's offer is smaller than a seller's offer (in reverse auction). The CDA is one of the most common exchange institutions, and is in fact the primary institution for trading of equities and derivatives. The prevalence of this institution comes from its operational simplicity (any trader may submit or accept an offer or a bid at any time) and its high efficiency. Another advantage of the CDA is that we are able to combine the different architectures as described in Chapter 4. Then the line

agents continuously offer jobs and AGV agents offer their free capacity at certain points in time. This approach also fits nicely with the shipper strategies of delaying and breaking commitments. Then the shipper is not restricted by periodic auctions, but may use a continuous threshold function; we then have in fact carriers' and shippers' take-it-or-leave-it prices.

In a combinatorial auction, bidders can place bids on combinations of items. These auctions are becoming increasingly popular in truckload transportation procurement and have been successfully used in several instances (e.g. see Ledyard et al., 2002; Caplice and Sheffi, 2003). The combinatorial auction also fits naturally in the transportation procurement since carriers have different valuations of different combinations of jobs (pickup and delivery routes) given that jobs can be complementary and substitutable (see Chapter 6). This particularly holds for LTL transportation because then the combination of jobs is even more important compared to FTL transportation. Besides, combinatorial issues already arise when we allow shippers to delay or break commitments; then we may have multiple jobs that are known to the carriers but not allocated yet. In Chapter 6 we ignored this by considering only one job at a time, so carriers do not take into account the current set of open jobs, and the threshold prices and decommitment penalties of the shippers are independent of the current set of open jobs. Obviously, a combinatorial auction can be beneficial here. The main task for the vehicles is to price the opportunities of acquiring attractive job sets and to bid on bundles of jobs. The main task of the shipper is to develop prespecified bundles of jobs and to select the best winner.

Timing constraints

Another aspect is related to practical *timing constraints* in vehicle routing. Throughout this thesis, we assume that vehicles can drive indefinitely without returning to a home base, or having lunch breaks, or other resting times (i.e., legal regulations on driving hours). Partly, we can take these decisions into account by using the gap-values. For example if a vehicle should return to the depot at the end of the day, it will start the working day with a large gap with the home base as end-node. An important aspect here is that the gap-values will gain importance, but also the computation time of these gap-values can become an issue (for large gaps). Another implication of the timing restrictions is that many decisions (e.g. calculation of opportunity costs and threshold prices), are now dependent on the time of the day. As a consequence, players should learn and estimate parameters as a function of the time of the day. In addition, we also study other time-window restrictions such as a hard restriction on the latest pickup time and an earliest pickup time that is not equal to the announcement time.

8.2.3 Implementation aspects

The different models in this thesis contain many simplifications that do not hold in practice. Further research should focus on modification of the proposed methods to make them more suitable for practical applications. Below we subsequently present the following model extensions: (1) fair allocation of costs and profits, (2) scalability, and (3) simulation.

Fair allocation of costs and profits

A first, and from our view the most important point of interest, is concerned with *profits*. For clarity of presentation we often ignored the profits. Especially in open environments (Chapters 5 and 6) this is a major issue. To account for profits in some way, we introduced the second-price auction in Chapter 5. However, this still requires the carriers to bid their true cost estimate. In reality this often does not hold, especially when there are more objectives than only costs (e.g. if carriers aim to increase their market share). Also topics like fixed costs of a carrier may influence the planning and scheduling decisions. For example, fixed costs can be incorporated in the price for an empty move, whereas they probably will be omitted in the costs for a pro-active move.

Also other undesirable system behavior or unfair cost allocation can be expected if we ignore the profits:

- Insertion heuristic: some shippers will pay for others, especially when jobs from one of the shippers can nicely be inserted in one of the vehicle schedules. Obviously, jobs that are located on a frequently traveled route are then the cheapest. However, in reality this may lead to strategic behavior, such as delaying the announcement times (like we did in Chapter 6).
- Decommitment: the decommitment penalties are set such that the shippers expect to play even. In reality we may expect that shippers also price the risk of receiving higher bid prices if decommitment is allowed.
- Opportunity costs: in closed environments the market prices are driven up (although we can correct this). Suppose all nodes are equally attractive to the carriers. Then the end-value is the same for all nodes, but the opportunity costs are always positive. So all vehicles simply increase their bids for all nodes. Because their winning intensities remain the same, all prices will continuously rise. In closed environments we can simply correct this, for example, by scaling down the prices. However, in open environments this is still an open issue.

To support a fair allocation of profits we may use game theory. Game theory is a formal study of the strategic interactions between agents. The

concepts of game theory provide a language to formulate, structure, analyze, and understand strategic scenarios that occur whenever the actions of several agents are interdependent. Economists have long used game theory to analyze a wide array of economic phenomena, including auctions and fair division.

There are two main branches of game theory: cooperative and non-cooperative game theory. Non-cooperative game theory deals largely with how intelligent individuals interact with one another in an effort to achieve their own goals. This branch can be used to describe the strategic interactions between agents in open environments. In cooperative theory, the players are allowed to form binding agreements and so there is strong incentive to work together to receive the largest total payoff. This branch can be used for open environments such as collaborative networks of carriers.

Scalability

A second aspect is related to the *scalability* of our methods to larger problems. For this case it is important to come up with faster solution methods. A first example of scalability is the aggregation of continuous locations to regions. Vehicles should be able to dynamically aggregate past observations into certain regions. A second example is to use approximate dynamic programming where parameterized value functions are updated by using reinforcement learning techniques, as proposed in Chapter 7 (Section 7.9).

Simulation

A final aspect of further research is concerned with the evaluation of the combination of different policies in a *whole range of network settings*. To this end we propose an extensive simulation experiment consisting of various settings, such as network settings (e.g. size and shape of the transportation network), market settings (e.g. open/closed first/second price auctions), company settings (number of shippers, carriers, vehicles), and job settings (time-windows and job arrival intensity). In addition, we may investigate the performance of our proposed strategies in combination with individual objectives, such as specialization of carriers in certain job types or regions. Finally, we may investigate the impact of MAS design choices on aspects regarding flexibility, scalability, adaptability, and extendibility.

The research laid down in this thesis clearly reveals the wealth of opportunities offered by applying a multi-agent approach to complex transport planning problems. Of course, the proof of the pudding will be given by a successful implementation of our methods and techniques in the hectic daily practice of transport logistics. Nonetheless, we are convinced that eventually this pudding will be a treat for the logistic tongue.

Bibliography

Andersson, M. and Sandholm, T. (2001). Leveled commitment contracts with myopic and strategic agents, *Journal of Economic Dynamics and Control* **25**(3-4): 615–640.

Bertsekas, D. and Tsitsiklis, J. (1996). *Neuro-Dynamic Programming*, Athena Scientific, Belmont, MA.

Bertsimas, D. and Ryzin, G. van (1991). A stochastic and dynamic vehicle routing problem in the euclidian plane, *Operations Research* **39**(4): 601–615.

Bertsimas, D. and Ryzin, G. van (1993). Stochastic and dynamic vehicle routing in the euclidean plane with multiple capacitated vehicles, *Operations Research* **41**(1): 60–76.

Bertsimas, D. and Simchi-Levi, D. (1996). A new generation of vehicle routing research: Robust algorithms, addressing uncertainty, *Operations Research* **44**(2): 286–304.

Böcker, J., Lind, J. and Zirkler, B. (2001). Using a multi-agent approach to optimise the train coupling and sharing system, *European Journal of Operational Research* **131**(2): 242–252.

Bose, S., Ozdenoren, E. and Pape, A. (2006). Optimal auctions with ambiguity, *Theoretical Economics* **1**(4): 411–438.

Bosman, P. and La Poutré, J. (2006). Computationally intelligent online dynamic vehicle routing by explicit load prediction in an evolutionary algorithm, *Parallel Problem Solving from Nature (PPSN IX)*, Lecture Notes in Computer Science, Springer-Verlag, pp. 312–321.

Boucke, N., Weyns, D., Holvoet, T. and Mertens, K. (2004). Decentralized allocation of tasks with delayed commencement, *EUMAS'04 Proceedings*, pp. 57–68.

Brandt, F. and Weiß, G. (2002). Antisocial agents and Vickrey auctions, *in* Tambe, J. and Meyer, M. (eds), *Intelligent Agents VIII*, Vol. 2333 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 335–347.

Branke, J., Middendorf, M., Noeth, G. and Dessouky, M. (2005). Waiting strategies for dynamic vehicle routing, *Transportation Science* **39**(3): 298–312.

Broek-Serlé, F. van der (2005). *De Logistieke Kracht van Nederland 2005*, Nederland Distributieland, Zoetermeer, the Netherlands.

Bürckert, H., Fund, P. and Vierke, G. (2000). Holonic transport scheduling with Teletruck, *Applied Artificial Intelligence* **14**(7): 697–725.

Caplice, C. and Sheffi, Y. (2003). Optimization-based procurement for transportation services, *Journal of Business Logistics* **24**(2): 109–128.

Cardon, A., Galinho, T. and Vacher, J. (2000). Genetic algorithms using multiobjectives in a multi-agent system, *Robotics and Autonomous Systems* **33**(2-3): 179–190.

Carvalho, T. and Powell, W. (2000). A multiplier adjustment method for dynamic resource allocation problems, *Transportation Science* **34**(2): 150–164.

Chow, Y., Robbins, H. and Siegmund, D. (1971). *Great Expectations: The Theory of Optimal Stopping*, Houghton Mifflin Company, Boston, MA.

Clearwater, S. (1996). *Market-based control: a paradigm for distributed resource allocation*, World Scientific, Singapore.

Cordeau, J. and Laporte, G. (2007). The dial-a-ride problem: models and algorithms, *Annals of Operations Research* **153**(1): 29–46.

Cordeau, J., Laporte, G., Potvin, J. and Savelsbergh, M. (2007). Transportation on demand, *in* Barnhart, C. and Laporte, G. (eds), *Transportation*, Vol. 14 of *Handbooks in Operations Research and Management Science*, Elsevier, Amsterdam, pp. 429–466.

Cruijssen, F. (2006). *Horizontal Cooperation in Transport and Logistics*, PhD thesis, Faculty of Economics and Business Administration, Tilburg University.

Davidsson, P., Henesey, L., Ramstedt, L., Törnquist, J. and Wernstedt, F. (2005). An analysis of agent-based approaches to transport logistics, *Transportation Research Part C* **13**(4): 255–271.

Desaulniers, G., Desrosiers, J., Erdmann, A., Solomon, M. and Soumis, F. (2001). VRP with pickup and delivery, *The vehicle routing problem*, Society for Industrial and Applied Mathematics, Philadelphia, PA, pp. 225–242.

Desrosiers, J., Dumas, Y., Solomon, M. and Soumis, F. (1995). Time constrained routing and scheduling, *in* Ball, M., Magnanti, T., Monma, C. and Nemhauser, G. (eds), *Network Routing*, Vol. 8 of *Handbooks in Operations Research and Management Science*, Elsevier, Amsterdam, pp. 35–139.

Dewan, P. and Joshi, S. (2000). Dynamic single-machine scheduling under distributed decision-making, *International Journal of Production Research* **38**(16): 3759 – 3777.

Douma, A., Schuur, P. and Heijden, M. van der (2006). Applying revenue management to agent-based transportation planning, *Technical report*, Beta Working Paper Series, WP-169.

Ebben, M., Heijden, M. van der and Harten, A. van (2005). Dynamic transport scheduling under multiple resource constraints, *European Journal of Operational Research* **167**(2): 320–335.

Ergun, O., Gultekin, K. and Savelsbergh, M. (2007). Reducing truckload transportation costs through collaboration, *Transportation Science* **41**(2): 206–221.

Ertogral, K. and Wu, S. (2000). Auction-theoretic coordination of production planning in the supply chain, *IIE Transactions* **32**(10): 931–940.

European Environment Agency (2007). *Transport and Environment: On The Way to a New Common Transport Policy - TERM 2006*, Vol. EEA Report No 1/2007, Office for Official Publications of the European Comumunities.

Ferguson, T. (1989). Who solved the secretary problem?, *Statistical Science* **4**(3): 282–296.

Figliozzi, M. (2004). *Performance and Analysis of Spot Truck-Load Procurement Markets Using Sequential Auctions*, PhD thesis, School of Engineering, University of Maryland.

Figliozzi, M., Mahmassani, H. and Jaillet, P. (2003). Framework for study of carrier strategies in auction-based transportation marketplace, *Transportation Research Record* **1854**: 162–170.

Figliozzi, M., Mahmassani, H. and Jaillet, P. (2004). Competitive performance assessment of dynamic vehicle routing technologies using sequential auctions, *Transportation Research Record* **1882**: 10–18.

Figliozzi, M., Mahmassani, H. and Jaillet, P. (2006). Quantifying opportunity costs in sequential transportation auctions for truckload acquisition, *Transportation Research Record* **1964**: 247–252.

Fischer, K., Muller, J. and Pischel, M. (1996). Cooperative transportation scheduling: an application domain for DAI, *Journal of Applied Artificial Intelligence. Special issue on Intelligent Agents* **10**(1): 1–33.

Fischer, M. (1995). Vehicle routing, *in* Ball, M., Magnanti, T., Monma, C. and Nemhauser, G. (eds), *Network Routing*, Vol. 8 of *Handbooks in Operations Research and Management Science*, Elsevier, Amsterdam, pp. 1–33.

Fisher, R. and Tippett, L. (1928). Limiting forms of the frequency distribution of the largest and smallest member of a sample, *Proceedings of the Cambridge Philosophical Society*, Vol. 24, pp. 180–190.

Frazzoli, E., Pallottino, L., Scordio, V. and Bicchi, A. (2005). Decentralized cooperative conflict resolution for multiple nonholonomic vehicles, *Proceedings of the AIAA Conference on Guidance, Navigation, and Control*, San Francisco, CA.

Gendreau, M., Guertin, F., Potvin, J. and Taillard, É. (1999). Parallel tabu search for real-time vehicle routing and dispatching, *Transportation Science* **33**(4): 381–390.

Gendreau, M., Hertz, A. and Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem, *Management Science* **40**(10): 1276–1290.

Gendreau, M. and Potvin, J. (1998). Dynamic vehicle routing and dispatching, *in* Crainic, T. and Laporte, G. (eds), *Fleet Management and Logistics*, Kluwer Academic Publishers, Boston, MA, pp. 115–126.

Ghiani, G., Guerriero, F., Laporte, G. and Musmanno, R. (2003). Real time vehicle routing: solution concepts, algorithms and parallel computing strategies, *European Journal of Operational Research* **151**(1): 1–11.

Giaglis, G., Minis, I., Tatarakis, A. and Zeimpekis, V. (2004). Minimizing logistics risk through real-time vehicle routing and mobile technologies: Research to date and future trends, *International Journal of Physical Distribution & Logistics Management* **34**(9): 749–764.

Godfrey, G. and Powell, W. (2002). An adaptive dynamic programming algorithm for dynamic fleet management, I: Single period travel times, *Transportation Science* **36**(1): 21–39.

Golob, T. and Regan, A. (2001). Impacts of information technology on personal travel and commercial vehicle operations: Research challenges and opportunities, *Transportation Research Part C* **9**(2): 87–121.

Gumbel, E. (1958). *Statistics of extremes*, Columbia University Press, New York, NY.

Heijden, M. van der, Ebben, M., Gademan, N. and Harten, A. van (2002). Scheduling vehicles in automated transportation systems, *OR Spektrum* **24**(1): 31–58.

Heijden, M. van der, Harten, A. van, Ebben, M., Saanen, Y., Valentin, E. and Verbraeck, A. (2002). Using simulation to design an automated underground system for transporting freight around schiphol airport, *Interfaces* **32**(4): 1–19.

Hemert, J. van and La Poutré, J. (2004). Dynamic routing problems with fruitful regions: Models and evolutionary computation, *Parallel Problem Solving from Nature (PPSN VIII)*, Vol. 3242 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 692–701.

Heragu, S., Graves, R., Kim, B. and St. Onge, A. (2002). Intelligent agent-based framework for manufacturing systems control, *IEEE Transactions on Systems, Man, and Cybernetics* **32**(5): 560–573.

Hevner, A., March, S. and Ram, S. (2004). Design science in information systems research, *MIS Quarterly* **28**(1): 75–105.

Hoen, P. 't and La Poutré, J. (2004). A decommitment strategy in a competitive multi-agent transportation setting, *in* Faratin, P., Parkes, D. and Rodriquez-Aguilar, J. (eds), *Agent Mediated Electronic Commerce V (AMEC-V)*, Vol. 3048 of *Lecture Notes in Artificial Intelligence LNAI*, Springer-Verlag, pp. 56–72.

Hoen, P. 't and La Poutré, J. (2006). Repeated auctions with complementarities, *Agent Mediated Electronic Commerce VII (AMEC-VII*, Vol. 3937 of *Springer Lecture Notes in Artificial Intelligence (LNAI)*, Springer-Verlag, pp. 16–29.

Ichoua, S., Gendreau, M. and Potvin, J. (2006). Exploiting knowledge about future demands for real-time vehicle dispatching, *Transportation Science* **40**(2): 211–225.

Jennings, N., Sycara, K. and Wooldridge, M. (1998). A roadmap of agent research and development, *Autonomous Agents and Multi-Agent Systems* **1**(1): 7–38.

Jiao, W., Debenham, J. and Henderson-Sellers, B. (2005). Organizational models and interaction patterns for use in the analysis and design of multi-agent systems, *Web Intelligence and Agent Systems* **3**(2): 67–83.

Karlin, S. (1962). Stochastic models and optimal policy for selling an asset, *in* Arrow, K., Karlin, S. and Scarf, H. (eds), *Studies in Applied Probability and Management Science*, Stanford University Press, Stanford, CA, pp. 148–158.

Kim, B., Graves, R., Heragu, S. and St. Onge, A. (2002). Intelligent agent modeling of an industrial warehousing problem, *IIE Transactions* **34**(7): 601–612.

Kim, Y., Mahmasanni, H. and Jaillet, P. (2002). Dynamic truckload truck routing and scheduling in oversaturated demand situations, *Transportation Research Record* **1783**: 66–71.

Kozlak, J., Créput, J., Hilaire, V. and Koukam, A. (2004). Multi-agent environment for dynamic transport planning and scheduling, *in* Bubak, M., Albada, G. van, Sloot, P. and Dongarra, J. (eds), *International Conference on Computational Science (ICCS 2004)*, Vol. 3038 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 638–645.

Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms, *European Journal of Operational Research* **59**(3): 345–358.

Larsen, A., Madsen, O. and Solomon, M. (2004). The a priori dynamic traveling salesman problem with time windows, *Transportation Science* **38**(4): 459–472.

Lau, H., Wong, V. and Lee, I. (2003). Immunity-based autonomous guided vehicles control, *Applied Soft Computing* **7**(1): 41–57.

Law, A. and Kelton, W. (2000). *Simulation Modeling and Analysis*, Vol. 3, McGraw-Hill, Singapore.

Le-Anh, T. and Koster, M. de (2006). A review of design and control of automated guided vehicle systems, *European Journal of Operational Research* **171**(1): 1–23.

Ledyard, J., Olson, M., Porter, D., Swanson, J. and Torma, D. (2002). The first use of a combined value auction for transportation services, *Interfaces* **32**(5): 4–12.

Lin, I., Mahmassani, H., Jaillet, P. and Walton, C. (2002). Electronic marketplaces for transportation services: Shipper considerations, *Transportation Research Record* **1790**: 1–9.

Lin, Y. and Solberg, J. (1992). Integrated shop floor control using autonomous agents, *IIE Transactions* **24**(3): 57–71.

Liu, S., Gruver, W. and Kotak, D. (2002). Holonic coordination and control of an automated guided vehicle system, *Integrated Computer-Aided Engineering* **9**(3): 235–250.

Luck, M., McBurney, P. and Preist, C. (2003). Agent technology: Enabling next generation computing-a roadmap for agent based computing, *AgentLink* .

Mahmassani, H., Kim, Y. and Jaillet, P. (2000). Local optimization approaches to solve dynamic commercial fleet management problems, *Transportation Research Record* **1733**: 71–79.

Maione, B. and Naso, D. (2001). Evolutionary adaptation of dispatching agents in heterarchical manufacturing systems, *International Journal of Production Research* **39**(7): 1481–1503.

Maturana, F., Shen, W. and Norrie, D. (1999). Metamorph: An adaptive agent-based architecture for intelligent manufacturing, *International Journal of Production* **37**(10): 2159–2174.

McAfee, R. and McMillan, J. (1987). Auctions and bidding, *Journal of Economic Literature* **25**(2): 699–738.

McAfee, R. and Vincent, D. (1997). Sequentially optimal auctions, *Games and Economic Behavior* **18**(2): 246–276.

McDonnell, P., Smith, G., Joshi, S. and Kumara, S. (1999). A cascading auction protocol as a framework for integrating process planning and heterarchical shop floor control, *International Journal of Flexible Manufacturing Systems* **11**(1): 37–62.

McElroy, J., Stephens, L., Bonnell, R. and Gorman, J. (1989). Communication and cooperation in a distributed automatic guided vehicle system, *Proceedings of the IEEE Southeastcon '89*, Vol. 3, pp. 999–1003.

McGill, J. and Ryzin, G. van (1999). Revenue management: research overview and prospects, *Transportation Science* **33**(2): 233–256.

Mes, M., Heijden, M. van der and Harten, A. van (2007). Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems, *European Journal of Operation Research* **181**(1): 59–75.

Mes, M., Heijden, M. van der and Hillegersberg, J. van (2008). Design choices for agent-based control of AGVS in the dough making process, *Decision Support Systems* **44**(4): 983–999.

Mes, M., Heijden, M. van der and Schuur, P. (2006). Opportunity costs calculation in agent-based vehicle routing and scheduling, *Technical report*, Beta Working Paper Series, WP-168.

Mes, M., Heijden, M. van der and Schuur, P. (2007). Dynamic threshold policy for delaying and breaking commitments in transportation auctions, *Technical report*, Beta Working Paper Series, WP-216.

Mitrović-Minić, S. and Laporte, G. (2004). Waiting strategies for the dynamic pickup and delivery problem with time windows, *Transportation Research Part B* **38**(7): 635–655.

Myerson, R. (1981). Optimal auction design, *Mathematics of Operations Research* **6**(1): 58–73.

Padgham, L. and Winikoff, M. (2004). *Developing Intelligent Agent Systems: A Practical Guide*, John Wiley & Sons, Chichester, UK.

Parunak, H. van dyke (1987). Manufacturing experience with the contract net, *in* Huhns, M. (ed.), *Distributed Artificial Intelligence*, Pitman Publishing: London and Morgan Kaufmann, San Mateo, CA, pp. 285–310.

Parunak, H. van dyke (1999). Industrial and practical applications of DAI, *Multiagent systems: a modern approach to distributed artificial intelligence*, The MIT Press, Cambridge, MA, pp. 377–421.

Parunak, H. van dyke, Baker, A. and Clark, S. (2001). The AARIA agent architecture: From manufacturing requirements to agent-based system design, *Integrated Computer-Aided Engineering* **8**(1): 45–58.

Powell, W. and Carvalho, T. (1998). Dynamic control of logistics queueing networks for large scale fleet management, *Transportation Science* **32**(2): 90–109.

Powell, W., George, A., Bouzaiene-Ayari, B. and Simao, H. (2005). Approximate dynamic programming for high dimensional resource allocation problems, *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, Montreal.

Powell, W., Jaillet, P. and Odoni, A. (1995). Stochastic and dynamic networks and routing, *in* Ball, M., Magnanti, T., Monma, C. and Nemhauser, G. (eds), *Network Routing*, Vol. 8 of *Handbooks in Operations Research and Management Science*, Elsevier, Amsterdam, pp. 141–295.

Powell, W., Sheffi, Y., Nickerson, K. and Atherton, S. (1988). Maximizing profits for north american van lines truckload division a new framework for pricing and operations, *Interfaces* **18**(1): 21–41.

Psaraftis, H. (1988). Dynamic vehicle routing problems, *in* Golden, B. and Assad, A. (eds), *Vehicle Routing: Methods and Studies*, Elsevier, Amsterdam, pp. 223–248.

Psaraftis, H. (1995). Dynamic vehicle routing: Status and prospects, *Annals of Operations Research* **61**(1): 143–164.

Regan, A. and Golob, T. (1999). Freight operators' perceptions of congestion problems and the application of advanced technologies: Results from a 1998 survey of 1200 companies operating in california, *Transportation Journal* **38**(3): 57–67.

Regan, A., Mahmassani, H. and Jaillet, P. (1995). Improving efficiency of commercial vehicle operations using real time information: Potential uses and assignment strategies, *Transportation Research Record* **1493**: 188–198.

Regan, A., Mahmassani, H. and Jaillet, P. (1996). Dynamic decision making for commercial fleet operations using real-time information, *Transportation Research Record* **1537**: 91–97.

Regan, A., Mahmassani, H. and Jaillet, P. (1998). Evaluation of dynamic fleet management systems: Simulation framework, *Transportation Research Record* **1645**: 176–184.

Reiss, R. and Thomas, M. (1997). *Statistical Analysis of Extreme Values*, Birkhauser Verlag, Basel, Switzerland.

Ross, S. (2003). *Introduction To Probability Models*, 8 edn, Academic Press.

Sandholm, T. (1991). A strategy for decreasing the total transportation costs among area-distributed transportation centers, *Nordic Operations Analysis in Cooperation (NOAS-91): OR in Business*, Publications of the Turku School of Economics and Business Administration, Turku School of Economics, Turku, Finland.

Sandholm, T. (1993). An implementation of the contract net protocol based on marginal cost calculations, *Proceedings of the 12th International Workshop on Distributed Artificial Intelligence*, Hidden Valley, PA, pp. 295–308.

Sandholm, T. (1996). Limitations of the Vickrey auction in computational multiagent systems, *Proceedings of the Second International Conference on Multiagent Systems (ICMAS-96)*, Keihanna Plaza, Kyoto, Japan, pp. 299–306.

Sandholm, T. (2000). Issues in computational Vickrey auctions, *International Journal of Electronic Commerce* **4**(3): 107–129.

Sandholm, T. (2002). Algorithm for optimal winner determination in combinatorial auctions, *Artificial Intelligence* **135**(1-2): 1–54.

Sandholm, T. and Lesser, V. (2001). Leveled commitment contracts and strategic breach, *Games and Economic Behavior* **35**(1-2): 212–270.

Sandholm, T., Sikka, S. and Norden, S. (1999). Algorithms for optimizing leveled commitment contracts, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Morgan Kaufmann Publishers, San Francisco, CA, pp. 535–541.

Savelsbergh, M. and Sol, M. (1995). The general pickup and delivery problem, *Transportation Science* **29**(1): 17–29.

Shapiro, J. and Powell, W. (2006). A metastrategy for large-scale resource management based on informational decomposition, *Informs Journal On Computing* **18**(1): 43–60.

Sheffi, Y. (2004). Combinatorial auctions in the procurement of transportation services, *Interfaces* **34**(4): 245–252.

Shen, W. and Norrie, D. (1999). Agent-based systems for intelligent manufacturing: A state-of-the-art survey, *Knowledge and Information Systems, an International Journal* **1**(2): 129–156.

Shen, W. and Norrie, D. (2001). Dynamic manufacturing scheduling using both functional and resource related agents, *Integrated Computer-Aided Engineering* **8**(1): 17–30.

Silver, E., Pyke, D. and Peterson, R. (1998). *Inventory management and production planning and scheduling*, 3 edn, John Wiley & Sons, New York, NY.

Smith, R. (1980). The contract net protocol: High-level communication and control in a distributed problem solver, *IEEE Transactions on Computers* **C-29**(12): 1104–1113.

Song, J. and Regan, A. (2001). Transition or transformation? emerging freight transportation intermediaries, *Transportation Research Record* **1763**: 1–5.

Song, J. and Regan, A. (2002). Combinatorial auctions for transportation service procurement: the carrier perspective, *Transportation Research Record* **1833**: 40–46.

Song, J. and Regan, A. (2003). An auction based collaborative carrier network, *Transportation Research Record* **1763**: 1–5.

Song, J. and Regan, A. (2005). Approximation algorithms for the bid construction problem in combinatorial auctions for the procurement of freight transportation contracts, *Transportation Research Part B* **39**(10): 914–933.

Sutton, R. and Barto, A. (1998). *Reinforcement Learning: An Introduction*, The MIT Press, Cambridge, MA.

Swihart, M. and Papastavrou, J. (1999). A stochastic and dynamic model for the single-vehicle pick-up and delivery problem, *European Journal of Operational Research* **114**(3): 447–464.

Talluri, K. and Ryzin, G. van (2005). *The theory and practice of revenue management*, 1 edn, Springer-Verlag.

Thangiah, S., Shmygelska, O. and Mennell, W. (2001). An agent architecture for vehicle routing problems, *Proceedings of the 2001 ACM symposium on Applied computing (SAC '01)*, ACM, New York, NY.

Thomas, B. and White, C. (2004). Anticipatory route selection, *Transportation Science* **38**(4): 473–487.

Tobin, J. (1958). Estimation of relationships for limited dependent variables, *Econometrica* **26**(1): 24–36.

Topaloglu, H. and Powell, W. (2006). Dynamic-programming approximations for stochastic time-staged integer multicommodity-flow problems, *INFORMS Journal on Computing* **18**(1): 31–42.

Toth, P. and Vigo, D. (2002). *The Vehicle Routing Problem*, SIAM Monographs on Discrete Mathematics and Applications, Society for Industrial and Applied Mathematics, Philadelphia, PA.

Tsitsiklis, J. and Roy, B. van (1997). An analysis of temporal-difference learning with function approximation, *IEEE Transactions on Automatic Control* **42**(5): 674–690.

Vickrey, W. (1961). Counterspeculation, auctions, and competitive sealed tenders, *Journal of Finance* **16**(1): 8–37.

Wallace, A. (2001). Application of AI to AGV control - agent control of AGVS, *International Journal of Production Research* **39**(4): 709–726.

Wellman, M., Walsh, W., Wurman, P. and MacKie-Mason, J. (2001). Auction protocols for decentralized scheduling, *Games and Economic Behavior* **35**(1): 271–303.

Weyns, D., Schelfthout, K., Holvoet, T. and Lefever, T. (2005). Decentralized control of E'GV transportation systems, *in* Pechoucek, M., Steiner, D. and Thompson, S. (eds), *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems (AAMAS '05)*, ACM, New York, NY, pp. 67–74.

Wood, M. and DeLoach, S. (2000). An overview of the multiagent systems engineering methodology, *in* Ciancarini, P. and Wooldridge, M. (eds), *The First International Workshop on Agent-Oriented Software Engineering (AOSE-2000)*, Vol. 1957 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 207–222.

Wooldridge, M. (1999). Intelligent agents, *in* Weiss, G. (ed.), *Multiagent Systems*, The MIT Press, Cambridge, MA, pp. 27–77.

Wooldridge, M. (2002). *An Introduction to MultiAgent Systems*, John Wiley & Sons, Chichester, UK.

Wooldridge, M. and Jennings, N. (1995). Agent theories, architectures, and languages: a survey, *Proceedings of the workshop on agent theories, architectures, and languages on Intelligent agents*, Springer-Verlag, Amsterdam, pp. 1–39.

Wooldridge, M., Jennings, N. and Kinny, D. (2000). The Gaia methodology for agent-oriented analysis and design, *Autonomous Agents and Multi-Agent Systems* **3**(3): 285–312.

Yang, J., Jaillet, P. and Mahmassani, H. (2004). Real-time multivehicle truckload pickup and delivery problems, *Transportation Science* **38**(2): 135–148.

Zhu, K., Ludema, M. and Heijden, R. van der (2000). Air cargo transport by multi-agent based planning, *Proceedings of the Thirty-third Annual Hawaii International Conference on Systems Sciences*, Maui, Hawaii.

Glossary of symbols

Here we present an overview of the symbols that are used in Chapters 3 till 6 of this thesis. To speed up the search process, we present the symbols per chapter, although we tried to be consistent in our notation.

Chapter 3:

Job characteristics:

a	announcement time
i	origin
j	destination
r	earliest pickup time
d	latest delivery time

Times:

τ_{ij}^{e}	empty travel time from i to j
τ_{ij}^{l}	loaded travel time from i to j
R	auction period, time between two successive auction rounds
$\Delta T_{v\varphi}^{n}$	expected additional travel- and handling time required for vehicle v in schedule alternative n to transport the job φ
$\Delta D_{v\varphi}^{n}$	expected additional tardiness required for vehicle v in schedule alternative n to transport φ
$\Delta W_{v\varphi}^{n}$	expected additional waiting time required for vehicle v in schedule alternative n to transport φ

Prices, costs, and revenues:

$c^{r}(t)$	travel costs as a function of the travel- and handling time t

continued on next page

continued from previous page

$c^p(t)$	penalty costs as a function of the tardiness t
$c^w(t)$	waiting costs as a function of the waiting time t
α_n	threshold price of auction round n
$P_{\min}$	a minimum price for the threshold function
$P_{\max}$	a maximum price for the threshold function
$b(v, \varphi)$	bid price of vehicle v for job φ

Other variables:

φ_n	nth job in a schedule
Ψ_v	current schedule of vehicle v
Ψ_v^n	alternative schedule n of vehicle v
$\mathcal{V}$	set of all vehicles v

Chapter 4:

Job characteristics:

a	announcement time
o	origin
d	destination
p	production line
m	mixer
b^p	best delivery time at the rising area
l^p	latest delivery time at the rising area
e_d	earliest delivery time at the production line
b_d	best delivery time at the production line
l_d	latest delivery time at the production line
$r_{\min}$	minimum rising time
r_{best}	best rising time

Times:

τ_{ij}^e	empty travel time from i to j
$h(\varphi)$	handling time of job φ
θ	current time
$z_{v\varphi}$	expected earliest delivery time of job φ by AGV v

Prices, costs, and revenues:

$W_{v\varphi}$	priority value for AGV v doing job φ
$b(v, \varphi)$	bid price of AGV v for job φ
α	penalty factor, costs of 1 time unit tardiness compared to 1 time unit deviation from best staying time
β	value of AGV capacity per time unit

Other variables:

φ_m	m^{th} job in a schedule
ω_m	scheduled pickup time of the m^{th} job
ρ_m	scheduled delivery time of the m^{th} job
Ψ_v	current schedule of vehicle v
Ψ_v^n	alternative schedules n of vehicle v
γ	learning rate
$C(\Psi_v)$	total costs of schedule Ψ_v
$\mathcal{L}^p$	set of all preparation jobs
$\mathcal{L}^d$	set of all delivery jobs

Chapter 5:

Job characteristics:

a	announcement time
o	origin
d	destination
e	latest pickup time
z	time-window length $e - a$

Times:

τ_{ij}^e	empty travel time from i to j
τ_{ij}^f	loaded travel time from i to j
τ_{ikl}	$\tau_{ikl} = \tau_{ik}^e + \tau_{kl}^f$
θ	current time
T	planning horizon
σ	time-to-go
$\overline{\sigma}$	average time-to-go
η	winning time

continued on next page

continued from previous page

t_n	flexibility of the n^{th} gap
s_n	time slack of the n^{th} gap

Prices, costs, and revenues:

$c^r(t)$	travel costs as a function of the travel time t
$c^p(t)$	penalty costs as a function of the tardiness t
$C^d(\varphi, \omega)$	direct costs for a job φ with scheduled pickup time ω
$c_{kl}(\sigma)$	direct costs for a job from k to l with time-to-go σ
OC_{ikl}	opportunity costs for a job from k to l given start-node i and time-to-go σ
$b_{ikl}(\sigma)$	bid price for a vehicle for a job from k to l with start-node i and time-to-go σ
$r_{ikl}(\sigma - \eta)$	expected revenue of a trip from k to l given this job is scheduled after node i a time σ from now on, and the job is won at a time $\eta < \sigma$ from now on.
$OC(\varphi, n, \omega, \Psi)$	opportunity costs of inserting a new job φ at position n with pickup time ω in the current schedule Ψ
$b(\varphi, \Psi)$	bid price for a job φ in schedule Ψ
p_φ	winning price of job φ

Probabilities and distribution functions:

$f_{i\sigma}(\eta)$	probability density function of the winning moment η given location i and time-to-go σ; with $F_{i\sigma}(\eta)$ the CDF.
$q_{i\sigma}(\eta)$	The discretized version of $f_{i\sigma}(\eta)$; with $Q_{i\sigma}(\eta)$ the discretized version of $F_{i\sigma}(\eta)$.
$p_{ikl}(\sigma - \eta)$	the conditional probability that a vehicle ending in location i will have trip from k to l as next job, given that the job is won at the η from now on
$\overline{x}_{ij}$	sample mean of all observations of the winning price for all jobs on route ij
s^2_{ij}	sample variance of all observations of the winning price for all jobs on route ij
$H_i(x)$	distribution of the i^{th} order statistic of the winning prices
$G_i(x)$	Gumbel distribution of the i^{th} order statistic of the winning prices
β_{ij}	location parameter of the Gumbel distribution
α_{ij}	scale parameter of the Gumbel distribution

continued on next page

continued from previous page

$H_{kl}^{\min}(x)$	distribution of the lowest bid x
$\xi_{ikl}(\sigma)$	mean number of winning jobs per time unit from k to l after arrival at node i with time-to-go σ
$p_{ikl}^{win}(\sigma)$	probability of winning a job from k to l with start-node i and time-to-go σ
$u(i)$	probability of not accepting a transition for a job won after node i

Value functions:

$V(\Psi, T)$	expected profits during a period T (from now on) for a certain vehicle given its schedule Ψ
$V^g(i, j, \sigma, t)$	expected profits during a period t after σ from now on given i, j, σ
$V^e(i, \sigma, t)$	expected profits during a period t after σ from now on given i, σ
$V^0(n, \Psi)$	the original value of a schedule Ψ that you lose due to the insertion of a new job at position n
$V^-(\varphi, n, \omega, \Psi)$	value of the new gap before the new job φ that is inserted at position n and pickup time ω in the current schedule Ψ
$V^+(\varphi, n, \omega, \Psi)$	value of the new gap after the new job φ that is inserted at position n and pickup time ω in the current schedule Ψ
$V^p(i, \sigma, \eta, t)$	expected profits during a period t for a vehicle ending at location i given it wins a job at time η during its time-to-go σ
$\widetilde{V}^e(i, t)$	approximate end-value for all uncertain moves after the first uncertain move
$\widetilde{V}^g(i, j, t)$	approximate gap-value for all uncertain moves after the first uncertain move
$\widetilde{V}^p(i, \sigma, \eta, t)$	approximate partial value function
$\Delta\widetilde{V}^e(d, t)$	decrease in end-value if the remaining horizon of the end-gap is decreased by t
$\Delta\widetilde{V}^g(n, t)$	decrease in value for all gaps after the n^{th} gap given the scheduled pickup time of job $n+1$ is postponed a time t
$\widehat{V}^e(i, \sigma, t)$	approximation of $V^e(i, \sigma, t)$
$\widehat{V}^g(i, j, \sigma, t)$	approximation of $V^g(i, j, \sigma, t)$

Other variables:

φ_n	nth job in a schedule

continued on next page

continued from previous page

ω_n	scheduled pickup time of the nth job
ρ_n	scheduled delivery time of the nth job
Ψ	current schedule
$\Psi \cup \varphi$	current schedule combined with the new job
N	number of jobs in a schedule
$\mathcal{V}$	set of all vehicles v
$\mathcal{N}$	set of all nodes
$\mathcal{A}$	set of all directed arcs
$\delta(i)$	decision to move to node $\delta(i)$ after arrival at node i
$\alpha_{ikl}(t)$	fraction of the total travel time τ_{ikl} that falls within period t
δ^a_{kl}	acceptance decision for a job from k to l
λ_{kl}	job arrival intensity for jobs from k to l

Chapter 6:

Job characteristics:

a	announcement time
i	origin
j	destination
l	latest pickup time
σ	time-window length of a job $(l - a)$
d	distance between the origin and destination of a job

Times:

R	auction period, time between two successive auction rounds
t	time-to-go

Prices, costs, and revenues:

$Z(\tau)$	costs for auctioning a job a time τ after the latest pickup time
β	constant costs of auctioning a job after the latest pickup time
b_n	bid in auction round n (with B_n the stochastic counterpart)
$V(t)$	minimum expected price a shipper has to pay eventually given a time-to-go t
$V_n(b_n)$	minimum expected price a shipper has to pay in one of the auction rounds $n..N$ given the current lowest bid b_n
$D_{s,t}$	decommitment penalty for a job committed at time s and decommitted at time t

continued on next page

continued from previous page

$\alpha(t)$	threshold price at time t
$\alpha_n(b_n)$	threshold price in auction round n given a lowest bid b_n
c	penalty costs per time unit tardiness

Probabilities and distribution functions:

λ	rate of the exponential distribution for the time between lowest bid updates
q^u	probability that the lowest bid is updated in the period R
$F_n(b)$	distribution of the lowest bid in auction round n; sometimes n is omitted or replaced by t
$P_n(b)$	discretization of $F_n(b)$
$\mu^w_{td}, \sigma^w_{td}$	mean and stdev of $F_t(b)$, with coefficients $\alpha_w, \beta_w, \gamma_w, \alpha_\sigma, \beta_\sigma, \gamma_\sigma$ for the linear trends
μ^p_{td}	mean penalty costs with coefficients $\alpha_p, \beta_p, \gamma_p$ for the linear trend
q^p_t	probability of having a bid with non-zero penalties given a time-to-go t (or auction round n)

Other variables:

N	maximum number of auction rounds
L	number of bid classes
δ_n	deviation from the expected lowest bid in auction round n
ϕ	coefficient of the linear trend in deviations

Samenvatting

Nieuwe methoden voor flexibele planning en besturing van transportnetwerken zijn vereist om te kunnen inspelen op de huidige trends in de logistieke sector. Een belangrijke trend is de toenemende belangstelling voor online planning en besturing. In het bijzonder de ontwikkelingen in de informatie- en communicatietechnologie (Internet en Global Positioning Systems) bieden vrachtvervoerders de mogelijkheid beter te plannen en online beslissingen te nemen. Ook de opkomst van elektronische marktplaatsen voor het uitwisselen van vracht en laadruimte biedt vervoerders en verladers nieuwe mogelijkheden.

In het licht van bovenstaande ontwikkelingen richten we ons in dit proefschrift op online planning en besturing van transportnetwerken. Beslissingen omvatten het toewijzen van transportopdrachten aan voertuigen, de precieze planning van deze opdrachten (in welke volgorde ze worden uitgevoerd) en keuzes ten aanzien van lege voertuigen (wachten, hoe lang en waar). We richten ons specifiek op het gebruik van online veilingmechanismen voor de toewijzing van FTL ladingen (full truckload) aan voertuigen.

We maken een onderscheid tussen zogenaamde open en gesloten omgevingen. In een open omgeving hebben we te maken met meerdere spelers, verladers en vervoerders. De verladers doen aanbestedingen voor transport via een elektronische veiling en vervoerders bieden op deze aanbestedingen. In een open omgeving zijn we voornamelijk geïnteresseerd in de winst van een individuele speler. We bestuderen de winstgevendheid van verschillende strategieën van een enkel voertuig en vergelijken deze met de gemiddelde winst van de andere spelers. In een gesloten omgeving hebben we te maken met een beperkt aantal spelers die op een bepaalde manier met elkaar verbonden zijn. Voorbeelden van een gesloten omgeving zijn (1) een fabriek die interne transportopdrachten toewijst aan AGVs (automatic guided vehicles), (2) verladers met hun eigen wagenpark en (3) een samenwerkingsverband van verladers. In een gesloten omgeving is het in principe mogelijk alle spelers centraal aan te sturen. In dit proefschrift beargumenteren we echter dat ook een gesloten omgeving baat kan hebben bij een op veilingmechanismen gebaseerde besturing. We zijn dan niet langer primair geïnteresseerd in de opbrengsten van individuele spelers, maar hebben juist het doel te komen tot een efficiënte toewijzing

van orders aan voertuigen, ofwel de minimalisatie van de totale logistieke kosten (zoals het leegrijden) en maximalisatie van de leverbetrouwbaarheid.

Voor de planning en besturing van gesloten omgevingen worden traditioneel wiskundige optimalisatiemethoden gebruikt die centrale planningen opstellen voor de activiteiten van alle spelers in het systeem. Deze methoden zijn echter minder geschikt voor een dynamische en onzekere omgeving waarin de voor de planning benodigde informatie geleidelijk bekend wordt. Ook kunnen centrale methoden gevoelig zijn voor kleine veranderingen: een kleine verandering in informatie kan een grote impact hebben op de planningen van alle voertuigen. Tenslotte, de rekentijden van dergelijke methoden kunnen een adequate reactie bij onverwachte zaken als storingen in de weg staan. Een nieuwe ontwikkeling in de ICT die zeer geschikt lijkt voor dergelijke planningssituaties, is het gebruik van een zogenaamd multi-agent systeem (MAS). Een dergelijk systeem bestaat uit een groep intelligente en autonome softwareprogramma's (de agenten) die met elkaar onderhandelen om individuele en globale doelen te behalen. Vaak worden er veilingmechanismen gebruikt voor de communicatie tussen de agenten. Deze aanpak lijkt een veelbelovende oplossing voor de besturing van complexe netwerken. De kracht ligt hierbij vooral in flexibiliteit, betrouwbaarheid en aanpassingsvermogen. Echter, het is nog niet duidelijk of deze aanpak ook tot lagere logistieke kosten leidt, vooral in vergelijking met meer centrale besturingen.

Een belangrijk element van de transportproblemen die in dit proefschrift aan de orde komen, is de dimensie tijd. Transportopdrachten komen sequentieel binnen terwijl de voertuigen onderweg zijn. Gevolg hiervan is dat beslissingen met betrekking tot de toewijzing en planning van opdrachten gebaseerd worden op onvolledige informatie. Dit, in combinatie met het gebruik van een online veiling, zorgt ervoor dat zowel de verladers als de vervoerders voor moeilijke beslissingen staan met betrekking tot het beprijzen en de planning van transport. Deze beslissingen hebben een direct effect op de winstgevendheid van de spelers en de totale logistieke kosten. Het is dan ook belangrijk om in de beslissingen rekening te houden met toekomstige gebeurtenissen, bijvoorbeeld door gebruik van statistische methoden.

De potentie van multi-agent systemen, in combinatie met de complexe maar veelbelovende mogelijkheden van het nemen van online beslissingen, hebben geleid tot het volgende onderzoeksdoel:

Het analyseren van de mate waarin en de wijze waarop multi-agent systemen gebruikt kunnen worden voor de operationele online planning en besturing van transportnetwerken. Verder, het ontwikkelen van intelligente en anticiperende strategieën voor spelers in sequentiële veilingen voor aanbesteding van transportopdrachten en het analyseren van de prestaties van deze strategieën in termen van individuele opbrengsten als ook de totale logistieke kosten.

Om dit doel te bereiken, hebben we een aantal meer specifieke doelen opgesteld. Elk van deze doelen is uitgewerkt in een apart hoofdstuk. Hieronder geven we een korte samenvatting van elk van deze hoofdstukken weer.

In hoofdstuk 3 maken we een vergelijking tussen een decentrale besturing gebaseerd op een multi-agent systeem en meer traditionele centrale besturingen. We gebruiken hiervoor een proefproject over een ondergronds logistiek systeem (OLS) bij Luchthaven Schiphol. Voor dit project zijn eerder een aantal besturingsmethoden en een simulatieomgeving ontwikkeld. Het gaat hier om hiërarchische methoden die enigszins anticiperen op toekomstige gebeurtenissen. We gebruiken de simulatieomgeving om onze multi-agent aanpak te vergelijken met een tweetal hiërarchische besturingen. We concluderen dat de prestaties van een goed ontworpen multi-agent systeem vergelijkbaar of soms zelfs beter zijn dan die van de hiërarchische methoden. Specifiek leidt de multi-agent benadering tot minder lege kilometers en is deze meer robuust in de zin dat de leverbetrouwbaarheid minder gevoelig is voor fluctuaties in de vraag.

In hoofdstuk 4 geven we inzicht in de ontwerpbeslissingen van een multi-agent systeem voor transportplanning. De belangrijkste beslissingen zijn: (1) het benoemen van de agenten, (2) de taken en verantwoordelijkheden van elk van deze agenten en (3) de manier waarop de agenten met elkaar communiceren. Deze beslissingen worden ondersteund door MAS ontwerpmethodieken. We laten zien dat kwalitatieve richtlijnen voor MAS ontwerp onvoldoende ondersteuning bieden om te komen tot een weloverwogen keuze voor de beste MAS architectuur. We stellen daarom voor om de bestaande MAS ontwerpmethodieken uit te breiden met simulatie. We illustreren deze werkwijze aan de hand van een case study in een industriële bakkerij. Hier bekijken we de besturing van automatisch geleide voertuigen die worden gebruikt voor het transporteren van ingrediënten in het deegbereidingsproces. We evalueren meerdere agent architecturen met behulp van simulatie. We concluderen dat er geen eenduidige beste architectuur bestaat: elke architectuur heeft zijn voors en tegens afhankelijk van het aantal te produceren degen per uur. Een mogelijke aanpak is dan ook om de architectuur dynamisch aan te passen aan veranderingen in de fabriek. We sluiten het hoofdstuk af met een beschouwing van de mogelijk bredere toepassing van deze inzichten in de praktijk.

In hoofdstuk 5 bekijken we een online transportprobleem waarbij transportopdrachten in sequentiële veilingen aan meerdere concurrerende vervoerders worden aangeboden. Doel van dit hoofdstuk is het ontwikkelen van methoden voor online planning en beprijzing van transport, waarbij rekening moet worden gehouden met toekomstige opdrachten. We kiezen voor een decentrale aanpak waarbij voertuigagenten verantwoordelijk zijn voor de planning en besturing van hun voertuig. Bij de beslissingen die voertuigen moeten nemen, wordt niet alleen rekening gehouden met de directe consequenties van het uitvoeren van een nieuwe opdracht, maar ook met de mogelijke impact hiervan op de toekomst. We gebruiken simulatie om de voordelen van een dergelijke

strategie te vergelijken met meer eenvoudige strategieën. We tonen aan dat een dergelijke benadering goed presteert in termen van winst, capaciteitsbenutting en leverbetrouwbaarheid.

In hoofdstuk 6 bekijken we een online transportprobleem waarbij tijdsafhankelijke transportopdrachten door concurrerende verladers worden aangeboden in sequentiële veilingen. Doel van dit hoofdstuk is het ontwikkelen van veilingmethoden voor de verladers. Hierbij dient rekening te worden gehouden met toekomstige gebeurtenissen, zoals aanbiedingen van concurrerende verladers. We bestuderen twee strategieën. Ten eerste bestuderen wij het gebruik van reserveprijzen. Het idee hiervan is dat wanneer alle biedingen hoger zijn dan de reserveprijs, de verlader niet akkoord gaat en later een nieuwe veiling start. Ten tweede bekijken we de situatie waarin een verlader de winnende vervoerder de mogelijkheid biedt af te zien van de gewonnen opdracht tegen bepaalde kosten. In dat geval zal de verlader een nieuwe veiling starten. In beide strategieën gebruikt de verlader stochastische informatie over toekomstige gebeurtenissen. Met behulp van simulatie laten we zien dat de strategieën de winstgevendheid van verladers kunnen vergroten, maar ook dat de totale logistieke kosten hiermee gereduceerd kunnen worden.

In hoofdstuk 5 en 6 bekijken we strategieën voor vervoerders en verladers onafhankelijk van elkaar. Precies gezegd, we bekijken de prestaties van een strategie van een individuele speler waarbij we ervan uitgaan dat alle andere spelers een gegeven eenvoudige strategie toepassen. In hoofdstuk 7 bekijken we de wisselwerking tussen de verschillende strategieën. Doel van dit hoofdstuk is inzicht te geven in de effecten van verschillende strategieën voor zowel de vervoerders als de verladers in sequentiële veilingen. Hierbij richten we ons op gesloten omgevingen. We streven naar minimalisatie van de totale logistieke kosten onder voldoend hoge leverbetrouwbaarheid. Met behulp van simulatie tonen we aan dat de eerder voorgestelde strategieën complementair zijn, dat wil zeggen, door combinatie van deze strategieën kunnen de logistieke kosten verder gereduceerd worden. De combinatie van strategieën stelt ons ook voor nieuwe uitdadingen aangaande het leerproces van de individuele spelers. We besluiten dan ook met een aanzet voor toekomstig onderzoek naar geavanceerde leermethoden.

De in dit proefschrift besproken methoden en technieken vormen een solide basis voor de ontwikkeling van intelligente en flexibele beslissingsondersteunende systemen waarmee in de praktijk van alledag vervoerders en verladers online kunnen plannen en besturen.

About the author

Martijn Mes was born on July 20, 1976, in Dronten, the Netherlands. In 1995 he finished secondary school at the Greijdanus College, Zwolle. In the same year he started his study applied mathematics at the University of Twente, where he specialized in operations research and logistics. In 2002 he graduated after completing his Master's thesis entitled 'Optimal Control of Multi-Echelon Systems'.

In December 2002 he started as a Ph.D. student under the supervision of prof. dr. A. van Harten in the group Operational Methods for Production and Logistics at the School of Management and Governance at the University of Twente. The research project 'Agent-Based Resource Planning and Activities Scheduling for Demand Driven Supply Chains' resulted in this thesis. Since September 2006 he is also a part time university lecturer and upon completion of his Ph.D. project in March 2008, will be employed as an assistant professor at the University of Twente.