Interaction between intelligent agent strategies for real-time transportation planning

Martijn Mes*, Matthieu van der Heijden, Peter Schuur

Department of Operational Methods for Production and Logistics

School of Management and Governance, University of Twente

P.O. Box 217, 7500 AE Enschede, The Netherlands

Abstract

In this paper we study the real-time scheduling of time-sensitive full truckload pickup-and-delivery jobs. The problem involves the allocation of jobs to a fixed set of vehicles which might belong to different collaborating transportation agencies. A recently proposed solution methodology for this problem is the use of a multi-agent system where shipper agents offer jobs through sequential auctions and vehicle agents bid on these jobs. In this paper we consider such a multi-agent system where *both* the vehicle agents and the shipper agents are using profit maximizing look-ahead strategies. Our main contribution is that we study the interrelation of these strategies and their impact on the system-wide logistical costs. From our simulation results, we conclude that the system-wide logistical costs (i) are always reduced by using the look-ahead policies instead of a myopic policy (10-20%) and (ii) the joint effect of two look-ahead policies is larger than the effect of an individual policy. To provide an indication of the savings that might be realized with a central solution methodology, we benchmark our results against an integer programming approach.

Keywords: Distributed decision making; Multi-agent systems; Auctions/bidding; Transportation; Vehicle routing

*Corresponding author. Tel.: +31 534894062; fax: +31 534892159; e-mail: m.r.k.mes@utwente.nl.

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 spot markets, a large number of shippers and carriers exchange loads and vehicle capacity. In private fleets, a shipper has exclusive and direct control over a fleet of vehicles. 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 [1].

In this paper we focus on private fleets and the contractual agreement structures where all jobs have to be transported by a fixed set of vehicles and where we aim to reduce the system-wide logistical costs. In the remainder we denote such an environment by *closed environment*. Traditionally, the allocation and scheduling decisions in closed environments are supported by operations research (OR) based optimization methods. Recently, several authors, see [2] for an overview, proposed to use a multi-agent system (MAS) to address the dynamic and real-time nature of the problem. 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 [3]. Typically, shipper agents are responsible for finding transport capacity and vehicle agents for the routing and scheduling decisions. The main decisions here are (i) the allocation of transportation jobs to vehicles and (ii) the timing of these jobs. The allocation of jobs is typically done using a sequential auction procedure where a shipper agent starts an auction for each incoming job and vehicle agents bid on these jobs. Note that, depending on the application area, the vehicle agent might represent a truck, taxi cab, automated guided vehicle, etc.

The idea of an auction-based allocation mechanism raises a problem: since jobs arrive in real-time, an optimal allocation 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 overcome this, the individual agents have to take into account future job arrivals in their current decision making. In the literature, several look-ahead policies have been proposed for shippers and vehicles. However, the interaction of intelligent behavior of vehicle agents and shipper agents has never been studied. This is the focus of the present paper.

For the look-ahead policies we use of the results of two earlier papers. In [4] two auction strategies for shipper agents are proposed, namely the use of reserve prices and decommitment penalties. In [5] pricing and scheduling strategies for vehicle agents are proposed where not only the direct costs of jobs are taken into account, but also the impact on future opportunities. The policies in both papers have been designed for individual players in spot markets whereas we now consider closed environments. The change of application area causes two difficulties. First, our objective differs: in the spot markets we focus on the revenues of a single player compared to those of its competitors whereas in closed environments we aim to minimize the system-wide costs. It might be the case that the individual policies are conflicting in the sense that the system-wide performance decreases. Second, learning might become more difficult: in the spot markets we consider one player which tries to learn the more or less constant behavior of all other players. In closed environments all players might adapt to each other which might not converge to a stable behavior. In this paper we study these difficulties. The goal of this paper is to study the interrelation of the different individual strategies and to benchmark their performance to a centralized solution approach.

The remainder of this paper is structured as follows. In Section 2 we give a brief overview of the relevant literature and state our contribution. In Section 3 we present our model of the transportation market. We present the various look-ahead policies in Section 4. We present the experimental settings and numerical results in Sections 5 and 6 respectively. We close with conclusions in Section 7.

2 Literature

In recent years, many papers on multi-agent systems for transportation problems have appeared. An early contribution is [6], where a multi-agent architecture and decision structure for quite generic transport planning systems is presented and tested on the traditional vehicle routing problem with time-windows. In [7] a multi-agent system is presented for real-time vehicle routing problems with consolidation in a multi-company setting where cargo is assigned to vehicles using a Vickrey auction. A framework for the study of carriers' strategies in an auction marketplace for dynamic full truckload vehicle routing problems with time-windows can be found in [8]. A similar problem is considered in [9] where a comparison is made between the agent-based approach and centralized optimization methods. For a literature survey on multi-agent approaches in the area of transportation (and traffic) we refer to [2].

Besides the tremendous increase in research on agent-based transportation scheduling, there is also an increasing number of software firms active in this area. For example, Magenta Technology provides a software tool called i-Scheduler that assists the human schedulers in scheduling cargo fleets, and Whitestein Technologies provides agent-based software solutions for logistics and production with their product named LS/ATN (Living System/Adaptive Transportation Network). For more information on these applications we refer to [10].

A quite common approach in agent-based transportation planning is to represent the resources and/or tasks by goal-directed agents and to enable cooperation between the agents using an auction protocol. Here job agents may focus on on-time delivery against the lowest possible costs, and a resource agent may strive for utilization and/or profit maximization.

For the decision making capabilities of the resource/vehicle agents we may rely on solutions for the dynamic vehicle routing problem (DVRP). Here a number of vehicles have to satisfy transportation requests that arrive dynamically over time. This requires real-time replanning in order to include the new jobs in the vehicle schedules. Although many papers have been devoted to the dynamic vehicle routing problem, there are still

some issues that have not been addressed yet [11]. Especially regarding look-ahead policies that incorporate the future consequences of certain decisions. Here we distinguish between two types of look-ahead policies: waiting strategies (i.e., where to wait and for how long) and scheduling strategies in anticipation of future job arrivals. Recent examples of waiting strategies can be found in [12, 13, 14]. Recent examples of look-ahead scheduling strategies can be found in [15, 16, 17]. In this paper we use a combination of look-ahead waiting and scheduling strategies as introduced in [5].

For the decision making capabilities of the task/shipper agents our focus is on auction strategies. A commonly used auction protocol in multi-agent systems is the sequential Vickrey auction where jobs are allocated one-by-one. The difficulty with such a system is that subsequent jobs are dependent: serving one job is greatly affected by the opportunity to serve another job. To cope with the interdependencies among jobs, we may use reserve prices and/or decommitment penalties. As shown in [18], the reserve price increases the expected revenue of the seller by preventing the object from being sold at a low price. For an extensive literature survey on this topic we refer to [19]. Decommitment penalties are introduced in [20]. 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 the option to decommit increases the Pareto efficiency of contracts and can make contracts more beneficial for both parties. In [7] the decommitment concept is applied 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. In this paper we use a combined policy for the use of reserve prices and decommitment penalties as introduced in [4].

The main contribution of this paper is to study the *interaction* between carriers and shippers, each using look-ahead profit maximizing strategies, which has not been studied before. 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 fixed set of vehicles. Further, we evaluate the interrelation of the different policies; 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, or they counteract with each other. In addition, we provide a benchmark of the agent-based approach with a mixed-integer programming approach where the multi-vehicle pickup and delivery problem is solved to optimality (with respect to the system-wide logistical costs) at each new job arrival.

3 Model of the transportation market

Jobs to transport unit loads (full truckload) arrive one-by-one. These jobs are characterized by 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 $a \leq e$. To introduce unbalanced transportation networks, i.e., a network in which some areas are more popular than others, we divide the network into disjoint regions a priori. We denote the set of regions by $\mathcal{N}$.

Within the network, all jobs have to be transported by a fixed set of vehicles that might possibly belong to different collaborating transportation agencies. The overall goal is to minimize the system-wide costs. 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 its 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.

We consider two cost components, namely the driving costs $c^d(t)$ as function of the travel time t and the penalty costs $c^p(t)$ in case of tardiness t with respect to the latest pickup time e. The time to transport a load from node i to node j is given by τ_{ij}^f (driving full). This includes travel time, and the handling time to load- and unload the job. The time to drive empty from location i to location j is given by τ_{ij}^e.

To model the transportation market we use a multi-agent system. We represent each player by an agent that acts as a decision maker. Here we restrict ourselves to vehicle agents and shipper agents. The shipper agents submit transportation jobs to the market

according to some stochastic process. Vehicle agents bid on these jobs and maintain a schedule of the jobs they have won.

To match transportation jobs with vehicle capacity we use auctions. In this paper we choose for sequential reverse Vickrey auctions, i.e, for each job we use a single auction round in which the lowest bidder wins the auction at the price of the second-lowest bid. The Vickrey auction has been widely used for multi-agent systems because (i) it requires a single bidding round and (ii) it forces bidders—under some mild conditions, see [21]—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.

To deal with this, we may 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 in [22], 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, (i) it may be unrealistic to bid on a bundle of jobs which belong to different shippers and (ii) these procedures are not directly applicable in situations where jobs arrive at different points in time.

For clarity of exposition, we make the following assumptions:

- There is only one shipper agent that receives and auctions all jobs.
- All jobs have to be transported eventually.
- 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.

- The handling times and travel times are deterministic.

4 Auction and bidding strategies

A job is allocated to a vehicle whenever the shipper decides which vehicle agent will win the auction, if any. After the arrival of new jobs, it may appear that the job assignment is not optimal anymore, i.e., we have a misallocation. 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 of jobs, we take the sequential transportation procurement auction as given, and focus on strategies for the participants. We consider the following options:

1. Delaying commitments: the shipper agent may delay commitments by refusing the current lowest bid based on a reserve price and to start a new auction for the same job later on.

2. Breaking commitments: the vehicle agents are allowed to reject an accepted job in favor of another job. The shipper reconsiders the decommitted job by starting a new auction for this job.

3. Valuation of opportunities: the vehicle agents include opportunity costs in their bids.

For *delaying commitments*, a shipper uses reserve prices in the auctions. When all bids are higher than the reserve price, the shipper rejects them and starts a new auction later on. This way, shippers avoid misallocations by postponing commitments 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 comes 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. This way, potential misallocations can be corrected. After a vehicle has decommitted a job, the shipper re-auctions 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 closed environments (private fleets or collaborative networks), because shippers operating in spot markets certainly would add a risk premium to the decommitment penalties.

In the third option, vehicle agents try to avoid misallocations by not only taking into account the direct impact of doing a certain job, but also its impact on the expected future revenue. This impact on future revenues is captured using the concept of *opportunity costs*. The opportunity costs are affected by job characteristics, such as the destination of the new job, but also by the order and timing of jobs in a schedule. These opportunity costs play a role in the bid pricing decisions of vehicles, but also in their scheduling decisions.

We implement the market-based multi-agent system as follows. When a job arrives at the shipper, it starts an auction by sending an announcement with job requirements to all vehicles. In return, each vehicle calculates a bid considering the marginal costs of doing this job and its impact on future opportunities (Section 4.1). Next, the shipper has to decide whether to accept the lowest bid (Section 4.2). 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 penalty for the new job and sends this to the winning vehicle (Section 4.3). The winning vehicle implements the schedule change. 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.

After each auction, both the shipper and the vehicles store information of the lowest bid together with the job characteristics. They use this information to periodically update their beliefs about other players (see Section 4.1 till 4.3). A general impression of the situation is given in Figure 1.

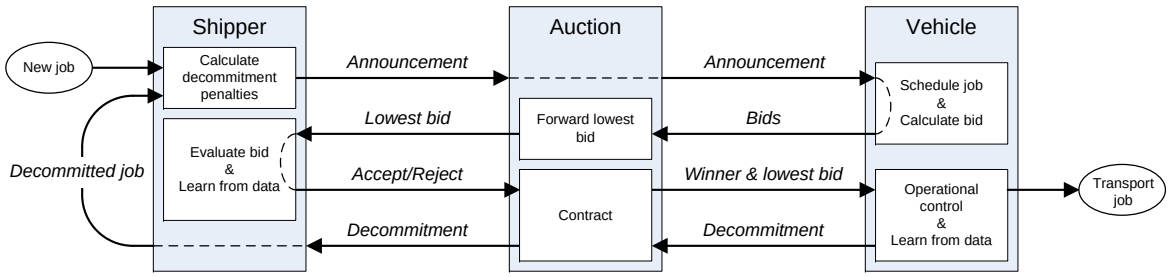

Figure 1: Transportation procurement market

In the next sections (Section 4.1 till 4.3) we describe the three policies in more detail and present some small modifications to adapt these policies to the closed environments.

4.1 Opportunity valuation policy

In this section we briefly describe the opportunity valuation policy as introduced in [5]. Also, we present a minor modification to this policy to apply it to closed environments.

To support job sequencing decisions and bid pricing decisions, vehicles maintain a job schedule. Vehicles are not restricted by the scheduled pickup times, but can simply decide to insert new jobs or to wait at some location after delivery of a job. 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.

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 (i) the number of jobs in a schedule by M, (ii) the destination region of the last job in the schedule Ψ by *schedule destination* $d(\Psi)$, and (iii) the time until the expected arrival time at the schedule destination by *length of a schedule* $l(\Psi)$.

To capture the impact of a schedule on future opportunities, we use an end-value $V(i,t)$ which provides an indication of the attractiveness of a schedule destination i. Specifically, $V(i,t)$ gives the expected profit during a period t after arrival at schedule destination i. The end-values are calculated using Stochastic Dynamic Programming. The information required consists of the job arrival patterns and the distribution of the lowest bid for various job characteristics. This information can be collected from the auctions. For more details on this we refer to [5].

The end-values are used by the vehicle agents (i) to calculate a bid price for a new job, (ii) to choose an appropriate insertion position for a new job, and (iii) to support so-called *pro-active move decisions*, i.e, moving empty in anticipation of future job requests. Below we elaborate on these decisions.

Consider vehicle v with M jobs in its current schedule Ψ_v. If $M = 0$, there is only one way to schedule a new job. If $M > 0$, a new job can be scheduled in M possible ways, since the first job cannot be interrupted. We write $\Psi_{v\varphi}^m$ for schedule alternative m, where the new job φ is inserted after job m. The direct costs for vehicle v for inserting a new job φ after the m^{th} job in its current schedule are given by (i) the costs for the expected additional travel time $\Delta T_{v\varphi}^m$ and (ii) the expected additional tardiness $\Delta D_{v\varphi}^m$. Besides these direct costs, a vehicle also faces *opportunity costs*. The opportunity costs of schedule alternative $\Psi_{v\varphi}^m$ of vehicle v within a given planning horizon T are given by the difference in end-value of the schedule alternative $\Psi_{v\varphi}^m$ compared to the current schedule Ψ_v. These opportunity costs are given by

$$OC\left(\Psi_{v\varphi}^m\right) = V\left(d\left(\Psi_v\right), T - l\left(\Psi_v\right)\right) - V\left(d\left(\Psi_{v\varphi}^m\right), T - l\left(\Psi_{v\varphi}^m\right)\right).$$

The bid price of vehicle v, for inserting a new job φ in its current schedule Ψ_v, is given by the direct costs of the cheapest insertion plus the opportunity costs

$$b(v,\varphi) = \min_{m=1..M} \left\{ c^d\left(\Delta T_{v\varphi}^m\right) + c^p\left(\Delta D_{v\varphi}^m\right) + OC\left(\Psi_{v\varphi}^m\right) \right\}.$$

We denote the schedule $\Psi_{v\varphi}^m$ with the lowest costs by $\Psi_{v\varphi}^*$. A vehicle agent updates

its schedule when (i) an auction for a new job φ is won and (ii) the first loaded move in a schedule has been completed. In the first case, the vehicle agent replaces its current schedule Ψ_v with $\Psi^*_{v\varphi}$. 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 agent has to decide whether to stay or to move pro-actively to another node in anticipation of future demand. For a given node i, the decision to move to node δ will result in an empty move with travel time $\tau^e_{i\delta}$ and costs $c^d(\tau^e_{i\delta})$. The pro-active move decision is then 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

$$\delta(i) = \arg\max_{\delta \in \mathcal{N}} \left\{ -c^d(\tau^e_{i\delta}) + V(\delta, T - \tau^e_{i\delta}) \right\}.$$

Note that more complicated decisions are involved when vehicles not always start the next job as early as possible, see [5].

The opportunity valuation policy has originally been designed for spot markets where we are dealing with a large number of vehicles each applying their own policy. In closed environments, all vehicles include opportunity costs in their bid pricing and scheduling decisions. As a consequence, the performance of each individual player is influenced by (i) other vehicle agents charging opportunity costs and (ii) the shipper agent that employs reserve prices or allows decommitment of jobs. When all players use exactly the same end-values the system might become unstable with ever increasing prices. To illustrate this, suppose all players update their end-values at the same time periodically. As mentioned earlier, the end-values describe the expected profit of a vehicle within a certain period depending on its schedule destination. The profit of a vehicle is given by the prices of the jobs it won minus the transportation and penalty costs for serving these jobs. Given the Vickrey auction, the price of a job is given by the second lowest bid which includes opportunity costs. Since the opportunity costs will typically (and at least on average) be greater then zero, the expected profits of the vehicles increases. Because profits increase, the end-values in the next period will also increase. Hence the opportunity costs vehicle

charge in their bid prices also increases. As a result, the prices for jobs increase with each periodic update of the end-values.

To prevent the increase in bid prices, we slightly modify the opportunity valuation policy for the use in closed environments. The expected rewards are calculated similarly as before by taking the difference between the lowest and second lowest bid. However, because both bids include opportunity costs, we subtract the opportunity costs from the expected rewards.

4.2 Dynamic threshold policy

In this section we briefly describe the dynamic threshold policy as introduced in [4]. Also, we present a minor modification to this policy to apply it to closed environments.

By using the dynamic threshold policy, a shipper has the opportunity to auction a job multiple times. We assume that the time between subsequent auction rounds is fixed and equal to R. After each auction, the shipper agent has to decide whether to accept the lowest bid. This decision can be supported by a threshold value $\alpha(n, d, b)$ which is given by the expected price a shipper has to pay in the auction rounds $n+1, \ldots, N$, given that it rejects the current lowest bid b for a job with distance d. We added the current bid b in the state space, because sequential bids for the same jobs are correlated. For R relatively small, the vehicle schedules at the next auction round will not be that different and the same probably holds for the lowest bid.

The optimal policy is to accept the current bid b in auction round n for a job with distance d, only when this value b is below a threshold value $\alpha(n, d, b)$. To calculate the threshold values we introduce a probability density function $P_{n,d}(b)$ of the lowest bid b at auction round n for a job with distance d. Here we discretize the possible bid prices in K classes. We further introduce B_n as the stochastic variable for the lowest bid at auction round n, q^u being the probability that the lowest bid is updated between two auction rounds, and ϕ being the slope of the linear regression between pairs of lowest bids in subsequent auction rounds. We use this slope to include correlations in price deviations (difference between the expected lowest bid and realized lowest bid) in

subsequent auctions rounds.

We calculate the threshold values backwards, starting from the last auction round N having a threshold value $\alpha_N = \infty$, i.e., in the last auction round we accept the lowest bid. As in [4], the recursive relation for the threshold values is given by

$$\begin{aligned}\alpha(n,d,b) &= (1-q^u)\min\{b,\alpha(n+1,d,b)\} + \\ &\quad q^u\sum_{k=0}^{K} P_{n+1,d}(k)\min\{k+\phi[b-E[B_n]],\alpha(n+1,d,k+\phi[b-E[B_n]])\}.\end{aligned}$$

To calculate the threshold values, the shipper has to learn the values of $P_{n,d}(b)$, q^u, and ϕ. Learning is based on historical observations of the lowest bid, see [4] for details. In closed environments, learning might become difficult because all players learn about each other. As a result, the system might not converge to a stable situation just like we saw with the opportunity valuation policy (see Section 4.1). To see whether the behavior of all players converges to some stable level, and if so, how long this takes, we introduce *learning periods*. In each learning period, players observe the behavior of all other players (through the auction outcomes), and update their policies at the end of each period.

Besides using multiple learning periods, we make one additional modification. Because we consider unbalanced networks where some regions are more popular than others, we have to include the origin region i and destination region j in the threshold values $\alpha_{ij}(n,d,b)$. To calculate the threshold values, the shipper estimates the probability density function $P_{n,d}(b)$ using multiple linear regression. Hence, we also have to include the origin and destination region in the regression functions. We simply do this, by adding $|\mathcal{N}|-1$ indicator functions for both the origin and destination region.

4.3 Decommitment policy

In this section we briefly describe the decommitment policy as introduced in [4]. At the end of this section we present a minor modification to this policy to apply it to closed environments.

By using the decommitment policy, the shipper agent allows vehicles to decommit from

an agreement (a job) against a predetermined time-dependent penalty. This penalty for a given job, as a function of the remaining time until the latest pickup time, is calculated by the shipper directly at the start of an auction for this job and is announced to the vehicles together with the other job characteristics. Whenever a vehicle decommits, (i) it will not receive the agreed price for the decommitted job, (ii) it has to pay the shipper the time-dependent decommitment penalty, and (iii) the shipper immediately starts a new auction for this job.

The decommitment penalty equals the expected extra costs for a shipper to find a new carrier (so we assume risk neutral shippers). The decommitment penalty is given by the expected lowest bid at the decommitment time t minus the expected lowest bid at the initial commitment time $s4$, $D_{s,t} = \mathbb{E}[B_t] - \mathbb{E}[B_s]$. However, when the shipper uses the decommitment policy in combination with the dynamic threshold policy, the decommitment penalties $D_{s,t}$ are given by the difference in threshold prices between the initial commitment time s and the decommitment time t. We modify threshold values by letting them depend on the remaining time t instead of on the auction round n. This is a minor modification which can be done rather easily, see [4]. The decommitment penalties are then given by: $D_{s,t} = \alpha(t, d, b) - \alpha(s, d, b)$.

To adjust the decommitment policy to closed environments we perform the same modifications as with the dynamic threshold policy, i.e., we include the origin and destination region in the threshold values and we use multiple learning periods.

5 Experimental settings

The goal of this simulation study is to evaluate the impact of combinations of shipper's and vehicles' look-ahead strategies on the system-wide logistical costs. To use the local look-ahead strategies, the agents have to learn the behavior of others. In this study, we want to distinguish the effects of learning from the interrelation of the policies themselves. To do this, each simulation run consist of a learning phase where agents learn from their environment and a steady state phase where agents use the information gathered from

the learning phase. During the learning phase, learning takes place periodically (i) by estimation of all required parameters using observations from the past period and (ii) by updating the policies in accordance with this. We set the length of a learning period to 10 days, which is sufficient to allow a reasonable amount of observations for various job characteristics. To study the interrelation of the policies, we only consider data from the steady state phase and regard the learning phase as a warm-up period.

We consider a transportation area where locations are distributed within a 100x100 km square area with Euclidian distances. To distinguish between more and less attractive locations, we divide the area into four equal-sized square regions. The regions are numbered consecutively per row, starting in the upper left corner and ending in the lower right corner. To adjust the transportation flow, we set for each region an origin probability, which is the probability that this region becomes the origin of a new job. For a given job, we first draw an origin region using the given origin probabilities and next draw a destination region randomly from the remaining regions. Within a given origin/destination region, we draw an (x,y) coordinate randomly from the square area. The different origin probabilities are shown in Table 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	$\frac{1}{10}(1 + (i-1) * 1.0)$

Table 1: Origin probabilities

We use 10 vehicles, each having a travel speed of 50 km/hour. The travel costs and penalty costs are 1 and 10 per minute respectively. 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.

For the vehicle agents we consider the following policies:

MY Myopic insertion strategy: the vehicle agents bid the costs of their cheapest insertion.

OV Opportunity valuation policy: the vehicle agents use the end-values in their bid pricing, scheduling, and waiting decisions.

For the shipper agent we consider the following policies:

MY Myopic policy: the shipper agent always accepts the lowest bid and does not allow decommitment.

DEC Decommitment policy: the shipper agent allows decommitment of jobs.

RES Dynamic threshold policy: the shipper agent uses reserve prices.

For the shipper agent, we decided not to consider the combination of DEC and RES given it results in a relative minor improvement at the expense of a major increase in computation time, see [4]. Given the policies mentioned above, we end up with $2 \times 3 = 6$ possible agent-based control structures (combination of individual policies). We denote a control structure by A/B where A refers to the policy used by all the vehicle agents and B to the policy used by the shipper agent.

A problem with online planning is that we generally compare different heuristics without having any benchmark for the effectiveness in terms of total relevant costs. To have an indication of the quality of our multi-agent approach, we would like to have a reasonable lower bound for the minimum costs. One option is to perform central optimization afterwards when all jobs are known. This is the optimal solution, but we use more information than is available during online execution. Therefore, this lower bound is usually far off the performance of online heuristics; so this is not a realistic bound. The option we consider here is central re-optimization of the problem each time new information arrives. Although it is not guaranteed that we find the minimum costs in this way, it gives a reasonable estimate of the performance that could be achieved based on the information we actually have under central planning. Specifically, we consider a reoptimization policy where the offline multi-vehicle pickup and delivery problem is solved at each new job arrival. Obviously, this policy is not practical for (i) real-time planning purposes of problems of realistic size and (ii) situations in which we are dealing

with multiple collaborative transportation agencies that want to maintain a certain level of autonomy. As a benchmark, we use a slightly modified version of the mixed-integer programming formulation given in [16]. In this formulation, the problem is modeled as an assignment problem with timing constraints. The assignment problem consists of finding a least-cost set of cycles describing the order in which each truck should serve the jobs. We slightly modified the formulation in the sense that all jobs have to be carried out and have to be accepted immediately once they are known. We denote the benchmarking policy by BENCH.

The experimental factors are shown in Table 2. A full factorial experiment with respect to these factors would require $7 \times 3 \times 4 \times 4 = 336$ experiments. For clarity of exposition, and to reduce computation time, we consider (i) all combinations of the factors Control, Degree of balance, and Time-window length; with as fixed settings a time between jobs of 800 seconds and (ii) all combinations of the factors Policies, Degree of balance, and Time between jobs; with as fixed settings a time-window length of 600 minutes. As a consequence, we consider $2 \times 7 \times 3 \times 4 = 168$ experiments. In the learning phase we only consider the unbalanced network with a time-window length of 600 minutes and a time between jobs of 800 seconds. In the remainder we refer to this setting of time-windows and time between jobs as default configuration.

Factor	**Values**
Control (vehicle/shipper)	MY/MY, MY/DEC, MY/RES OV/MY, OV/DEC, OV/RES, BENCH
Degree of balance	balanced, slightly unbalanced, unbalanced
Time-window length (min)	300, 400, 500, 600
Time between jobs (seconds)	700, 800, 900, 1000

Table 2: Experimental factors

As primary performance indicator we consider the average 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 costs of this policy compared to the average costs of the myopic policy. In mathematical form

this would be

$$\text{relative savings} = 100 \times \left(\frac{\text{average costs of myopic policy} - \text{average costs of policy}}{\text{average costs of myopic policy}} \right).$$

For our simulations, we use a replication / deletion approach, see [23], where each experiment consists of a number of replications (each with different seeds) and a warm-up period. The warm-up period consists of a 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. For all experiments, we use 5 replications, which appear to be sufficient for a confidence level of 95% with a relative error of 5% with respect to the average costs per job.

6 Numerical results

First we present the results from the learning period (Section 6.1) after which we present the steady state performance of the various policies (Section 6.2).

6.1 Learning behavior

Here we evaluate the impact of the number of learning periods (1 till 9) on the average costs per job, see Figure 2. Obviously, the policies MY/MY and BENCH do not require learning. The individual policies OV and DEC (policies MY/DEC, OV/MY, and OV/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. For the individual policy RES (policies MY/RES and OV/RES) we see that it takes some time to come up with reasonable relative savings; with one learning period we even see that the average costs per job increase compared to the myopic policy MY/MY. For the remainder of this section we use a warm-up period consisting of 5 learning periods. From Figure 2 we see that this number is sufficient for most policies to converge to a relatively stable performance.

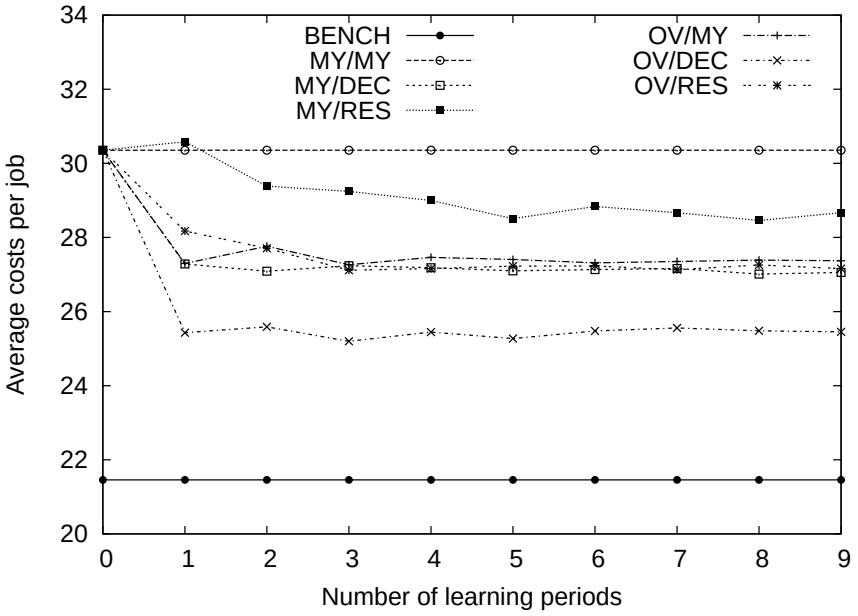

Figure 2: Average costs per job as a function of the number of learning periods

6.2 Steady state comparison of policies

Here we evaluate the interrelation of the shipper's and vehicles' look-ahead strategies. We use the experimental factors as shown in Table 2. All figures in this section display the costs of the agent-based policies relative to the performance of the myopic policy given in percentages. For the unbalanced network we also show performance data with respect to the absolute costs and some additional performance indicators. These data can be found in the Appendix.

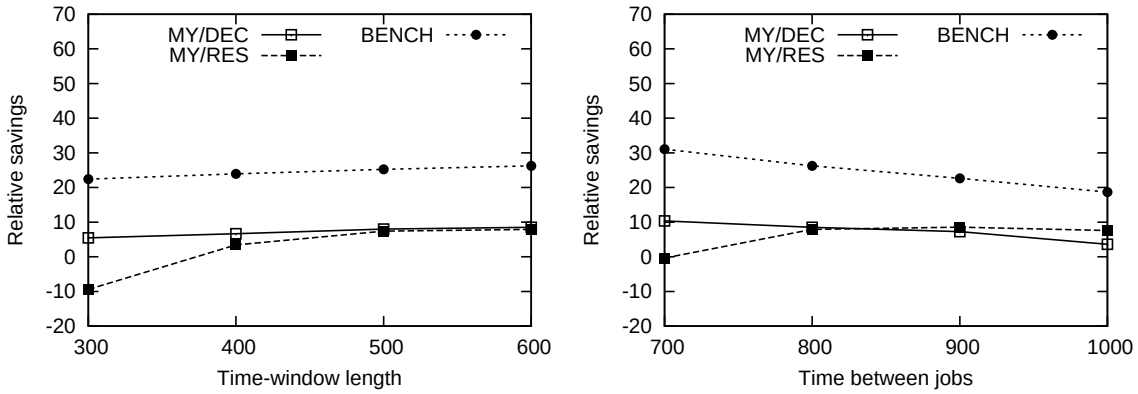

Figure 3: Simulation results for balanced networks

First we consider the balanced network. Given that the opportunity valuation policy only benefits from imbalance in a network, we omit the policies OV/MY, OV/DEC, and

OV/RES. The results for the remaining policies can be found in Figure 3.

From this figure we draw the following conclusions. First, the relative savings of all policies increase with increasing time-window length. This means that with increasing time-window length, the differences between the myopic policy and the other policies are getting larger. The shippers' policies RES and DEC benefit from increasing time-windows because there is simply more time to delay (RES) or break (DEC) commitments. The benchmarking policy also takes advantage of increasing time-windows since there will be an increasing probability that the policy will find a better set of vehicle schedules compared with the myopic policy (see Appendix, Table 4 for the unbalanced network).

We further see that the relative savings of MY/DEC and BENCH decrease with increasing time between jobs (decreasing number of jobs). The reason for this is the following. The advantage of MY/DEC and BENCH over the myopic policy is that they allow exchange of jobs between vehicle schedules (by means of swapping jobs or completely reassigning all jobs). However, with increasing time between jobs, the average schedule length of the vehicle will become shorter. So, there will be less to gain by exchanging jobs. The relative savings of MY/RES increase with increasing time between jobs since the probability of late delivery decreases, which is really an issue with this policy (see Appendix, Table 5 for the unbalanced network). Also, with increasing time between jobs, the probability of finding better vehicle schedules in future auction rounds (the principle behind MY/RES) will increase.

A final observation here is that the gap between the agent-based policies and our benchmark remains relatively large. We come back to this issue at the end of this section.

Next we consider the case of slightly unbalanced networks, see Figure 4. The travel distances (empty as well as loaded) are getting longer with increasing imbalance. As a consequence, it will be harder to deliver all jobs on time. This has a similar effect as the decreasing time-between jobs in the balanced case. This explains why within the slightly unbalanced networks the relative savings of the MY/DEC and BENCH are higher most of the time whereas the savings for the policy MY/RES are lower in most cases. We further see that performances of OV/MY and MY/DEC are close to each other. Finally,

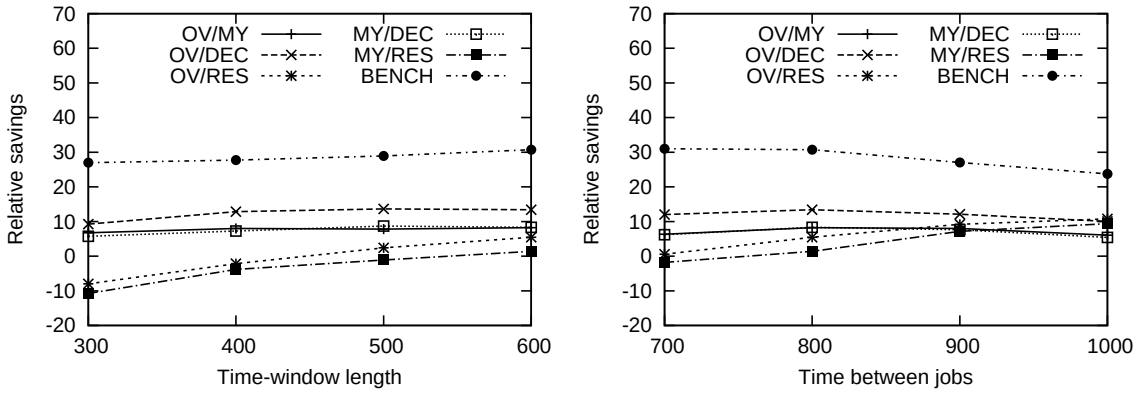

Figure 4: Simulation results for slightly unbalanced networks

the combination of shipper's and vehicles' strategies (OV/RES and OV/DEC) always increases the performance. The best combination of local policies here is OV/DEC.

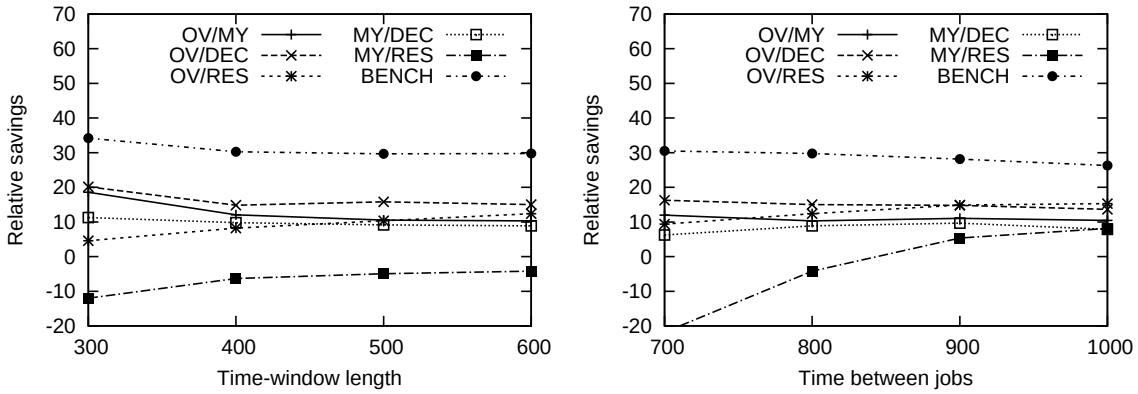

Figure 5: Simulation results for unbalanced networks

Next we consider the case of unbalanced networks. The results can be found in Figure 5. By introducing more imbalance, it becomes even harder to deliver all jobs on time. Using a similar argumentation as given above, the relative savings of MY/DEC and BENCH in the unbalanced network become slightly higher and the relative savings of MY/RES slightly lower. However, the differences between the unbalanced and the slightly unbalanced case are smaller than the differences between the slightly unbalanced and balanced case. The reason for this is that within the 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.

We further see that the gap between the best agent-based policy and our benchmark becomes smaller; especially with increasing time between jobs. Again we see that from

the local policies, the combination OV/DEC performs best in most cases. Only in the situation with relative low job arrival intensity (time between jobs of 1000), the policy OV/RES performs best.

Finally, we observe two peculiarities. First, the relative savings of our benchmark decrease with increasing time-window length. This makes sense since tight time-windows require smart planning. Second, the relative savings of the policy MY/RES increases with decreasing number of jobs (increasing time between jobs). The reason for these peculiarities is that the myopic policy also performs better with increasing time-window length or increasing time between jobs. Because the benchmarking policy always results in the highest service levels, and MY/RES always in lower service levels (see Appendix, Table 4 and 5), this benchmarking policy benefits the least and the policy MY/RES the most from an increase in time-window length or time between jobs. In addition, increasing time-window lengths also gives the opportunity to use more auction rounds for a job (the principle behind RES).

To summarize the results, we have seen that combinations of vehicle and shipper strategies always improve the performance compared to one of the individual policies. In almost all cases, the combination of the opportunity valuation policy and the decommitment policy (OV/DEC) 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%.

The results of the local policies seem promising, but there is still a gap with our benchmark. For example, in the unbalanced networks, the policy OV/DEC (the best combination of local policies) is only able to achieve, on average, 53% of the savings from our benchmark. There are two extenuating circumstances here. First, the problem under consideration is relative simple and clean in the sense that only one type of decision is involved (assigning jobs to certain positions in truck schedules) and only one type of uncertainty is involved (the job arrival process). In earlier work, see [9], where we considered a problem involving many more decision types and also uncertainty in handling

and travel times, we drew an opposite conclusion in favor of the local policies. Second, the benchmarking policy requires considerably more computation time compared to the local policies. For this reason we only considered relatively small problem instances. With more realistic problem sizes, the multi-agent approach would still be able to perform real-time whereas the central approach would require approximations or be replaced by heuristic procedures.

For our experiments we used the simulation software Plant Simulation 8.2 and an Intel Pentium 4 processor at 3.4 GHz. To speed up the simulations, we programmed the dynamic threshold policy in Delphi 7 as a dynamic link library which we included in our simulation environment. We solved the mixed-integer programming formulation using CPLEX 11 with a time limit of 15 minutes. However, it appears that none of the instances reached this time limit. In Table 3 we show the average computation time per job during one simulation run, excluding the warm-up period.

Policy	Time (sec)
MY/MY	0.019
MY/DEC	0.186
MY/RES	3.013
OV/MY	0.026
OV/DEC	0.249
OV/RES	3.974
BENCH	22.048

Table 3: Average computation times per job under the default configuration in the unbalanced network

We see that the computation time for OV/DEC, the best agent-based policy, is relatively short. The computation times for the individual policies OV and DEC are short because some values are calculated offline during the warm-up period. For the policy OV, the end-values are calculated offline and for the policy DEC the function of the expected lowest bid is calculated offline. The policy RES requires more computation time because the shipper has to calculate the dynamic threshold recursion at each auction. An interesting result is that the required computation time of the policy OV/DEC increases almost linearly with the average number of times a job is decommitted. In all cases, this will be far less than the required computation time for the benchmarking policy. Keeping

this in mind, an achievement of 53% of the savings from the benchmark can be regarded as impressive.

7 Conclusions

In this paper we studied the interaction between vehicle agents and shipper agents in a market-based multi-agent system for full truckload transportation. Shipper agents offer the transport jobs through sequential auctions. A set of vehicle agents 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 vehicle agents 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 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 and requires relatively limited computation time. 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. 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.

- There is still a gap between the agent-based policies and our benchmarking policy which reoptimizes the multi-vehicle pickup and delivery problem at each new job arrival. For example, in the unbalanced network, the control OV/DEC achieves on average 53% of the savings from our benchmark. However, the benchmarking policy might simply not always be applicable due to its computational complexity and because it ignores the autonomy of the different actors. To use the benchmarking policy, we only considered small problem instances in this paper; larger instances certainly would require approximations or other solution methodologies. Furthermore, the agent-based approach might come in favor with increasing uncertainty as shown in [9].

The gap between the agent-based policies and our benchmark gives rise to further research. Specifically we focus on two issues. First, the improvement of the local policies by using approximate dynamic programming where we try to learn the value functions without using a detailed model of the environment's dynamics. Second, the integration of the concepts opportunity costs, threshold values, and decommitment penalties, in a mathematical programming approach on fleet level.

References

[1] J. Song, A. Regan, An auction based collaborative carrier network, Transportation Research Record 1763 (2003) 1–5.

[2] P. Davidsson, L. Henesey, L. Ramstedt, J. Törnquist, F. Wernstedt, An analysis of agent-based approaches to transport logistics, Transportation Research Part C 13 (4) (2005) 255–271.

[3] M. Wooldridge, Intelligent agents, in: G. Weiss (Ed.), Multiagent Systems, The MIT Press, Cambridge, MA, 1999, pp. 27–77.

[4] M. Mes, M. Van der Heijden, P. Schuur, Dynamic threshold policy for delaying and breaking commitments in transportation auctions, Transportation Research Part C 17 (2) (2008) 208–223.

[5] M. Mes, M. Van der Heijden, P. Schuur, Look-ahead strategies for dynamic pickup and delivery problems, OR Spectrum Forthcoming - doi:10.1007/s00291-008-0146-3.

[6] K. Fischer, J. Muller, M. Pischel, Cooperative transportation scheduling: an application domain for DAI, Journal of Applied Artificial Intelligence. Special issue on Intelligent Agents 10 (1) (1996) 1–33.

[7] P. 't Hoen, J. La Poutré, A decommitment strategy in a competitive multi-agent transportation setting, in: P. Faratin, D. Parkes, J. Rodriquez-Aguilar (Eds.), Agent Mediated Electronic Commerce V (AMEC-V), Vol. 3048 of Lecture Notes in Artificial Intelligence LNAI, Springer-Verlag, 2004, pp. 56–72.

[8] M. Figliozzi, H. Mahmassani, P. Jaillet, Framework for study of carrier strategies in auction-based transportation marketplace, Transportation Research Record 1854 (2003) 162–170.

[9] M. Mes, M. Van der Heijden, A. Van Harten, Comparison of agent-based scheduling to look-ahead heuristics for real-time transportation problems, European Journal of Operation Research 181 (1) (2007) 59–75.

[10] S. Munroe, T. Miller, R. Belecheanu, M. Pechoucek, P. McBurney, M. Luck, Crossing the agent technology chasm: Lessons, experiences and challenges in commercial applications of agents, The Knowledge Engineering Review 21 (4) (2006) 345–392.

[11] G. Ghiani, F. Guerriero, G. Laporte, R. Musmanno, Real time vehicle routing: solution concepts, algorithms and parallel computing strategies, European Journal of Operational Research 151 (1) (2003) 1–11.

[12] A. Larsen, O. Madsen, M. Solomon, The a priori dynamic traveling salesman problem with time windows, Transportation Science 38 (4) (2004) 459–472.

[13] S. Ichoua, M. Gendreau, J. Potvin, Exploiting knowledge about future demands for real-time vehicle dispatching, Transportation Science 40 (2) (2006) 211–225.

[14] B. Thomas, Waiting strategies for anticipating service requests from known customer locations, Transportation Science 41 (3) (2007) 319–331.

[15] S. Mitrović-Minić, G. Laporte, Waiting strategies for the dynamic pickup and delivery problem with time windows, Transportation Research Part B 38 (7) (2004) 635–655.

[16] J. Yang, P. Jaillet, H. Mahmassani, Real-time multivehicle truckload pickup and delivery problems, Transportation Science 38 (2) (2004) 135–148.

[17] J. Branke, M. Middendorf, G. Noeth, M. Dessouky, Waiting strategies for dynamic vehicle routing, Transportation Science 39 (3) (2005) 298–312.

[18] R. Myerson, Optimal auction design, Mathematics of Operations Research 6 (1) (1981) 58–73.

[19] R. McAfee, J. McMillan, Auctions and bidding, Journal of Economic Literature 25 (2) (1987) 699–738.

[20] T. Sandholm, V. Lesser, Leveled commitment contracts and strategic breach, Games and Economic Behavior 35 (1-2) (2001) 212–270.

[21] W. Vickrey, Counterspeculation, auctions, and competitive sealed tenders, Journal of Finance 16 (1) (1961) 8–37.

[22] J. Song, A. Regan, Approximation algorithms for the bid construction problem in combinatorial auctions for the procurement of freight transportation contracts, Transportation Research Part B 39 (10) (2005) 914–933.

[23] A. M. Law, Simulation Modeling and Analysis, 4th Edition, McGraw-Hill Education, New York, 2007.

Appendix

In this section we show additional performance data with respect to the unbalanced network. As performance indicators we consider the average costs per job (Costs), the percentage of the total driving distance that is driven loaded (DL), and the service level (SL) defined by the percentage of jobs that are delivered on time. The results for varying time-window length (TW) can be found in Table 4 and the results for varying time between jobs (TBJ) in Table 5.

TW	300			400			500			600		
Policy	Costs	DL	SL	Costs	DL	SL	Costs	DL	SL	Costs	DL	SL
MY/MY	37.5	66.5	95.8	32.4	67.0	97.4	31.3	67.5	97.3	30.6	67.8	97.9
MY/DEC	33.3	67.2	97.9	29.2	68.2	98.8	28.4	68.7	99.0	27.8	69.1	99.2
MY/RES	42.0	67.0	93.2	34.5	68.0	95.9	32.9	68.7	94.7	31.8	69.1	94.7
OV/MY	30.6	68.1	97.5	28.5	68.9	98.3	28.0	69.3	98.7	27.4	69.8	98.7
OV/DEC	30.0	68.5	98.5	27.6	69.5	99.1	26.4	70.3	99.4	26.0	70.6	99.4
OV/RES	35.8	68.5	94.7	29.8	69.7	96.3	28.1	70.4	96.4	26.8	70.7	96.7
BENCH	24.7	71.8	99.6	22.6	73.2	99.7	22.0	74.0	99.8	21.5	74.5	99.9

Table 4: Simulation results for varying time-window length (TW) for unbalanced networks

TBJ	700			800			900			1000		
Policy	Costs	DL	SL	Costs	DL	SL	Costs	DL	SL	Costs	DL	SL
MY/MY	32.8	67.7	96.5	30.6	67.8	97.9	30.6	67.7	98.4	30.5	67.7	98.0
MY/DEC	30.8	68.4	97.5	27.8	69.1	99.2	27.6	69.2	99.3	28.1	68.7	99.4
MY/RES	40.0	68.1	93.7	31.8	69.1	94.7	28.9	69.3	95.9	28.0	69.2	96.7
OV/MY	28.9	69.2	97.5	27.4	69.8	98.7	27.2	69.9	98.9	27.3	69.8	99.2
OV/DEC	27.5	70.1	98.3	26.0	70.6	99.4	26.1	70.4	99.6	26.4	70.1	99.7
OV/RES	29.7	70.2	96.8	26.8	70.7	96.7	26.0	70.6	97.2	25.9	70.5	97.9
BENCH	22.8	73.3	99.7	21.5	74.5	99.9	22.0	74.1	99.9	22.5	73.5	99.9

Table 5: Simulation results for varying time between jobs (TBJ) for unbalanced networks