Impact of Competition on Quality of Service in Demand Responsive Transit

Ferdi Grootenboers, Mathijs de Weerdt, Mahdi Zargayouna

Source Lecture Notes in Computer Science, 2010, vol. 6251

Impact of Competition on Quality of Service in Demand Responsive Transit

Ferdi Grootenboers[1], Mathijs de Weerdt[1], and Mahdi Zargayouna[2]

[1] Delft University of Technology,
PO Box 5031, 2600 GA Delft, The Netherlands
[2] INRETS Institute, Gretia laboratory, Building "Descartes II",
2 rue de la Butte Verte, 93166 Noisy le Grand Cedex, France
fgrootenboers@gmail.com, M.M.deWeerdt@tudelft.nl, zargayouna@inrets.fr

Abstract. Demand responsive transportation has the potential to provide efficient public door-to-door transport with a high quality. In currently implemented systems in the Netherlands, however, we observe a decrease in the quality of service (QoS), expressed in longer travel times for the customers. Currently, generally one transport company is responsible for transporting all customers located in a specified geographic zone. In general it is known that when multiple companies compete on costs, the price for customers decreases. In this paper, we investigate whether a similar result can be achieved when competing on quality instead. To arrive at some first conclusions, we set up a multiagent environment to simulate the assignment of rides to companies through an auction on QoS, and the insertion of allocated rides in the companies' schedules using online optimization. Our results reveal that this set-up improves the quality of the service offered to the customers at moderately higher costs.

Key words: Dial-a-ride, multi-company, quality of service, auction

1 Introduction

Demand-Responsive Transit (DRT) services are a form of transport that is a compromise between public transportation and individual taxis. The principle of these systems is to define the itineraries and schedules of the vehicles based on the requests of the users. Customers are thus provided with relatively cheap door-to-door transportation insofar as they accept to share their ride with others and tolerate a certain detour from their direct trip. The main problem with current DRT services as organized in the Netherlands is that the quality of service (QoS) cannot be guaranteed over longer periods of time. A strong competition for the right to serve for a period of usually three years promises a reasonable quality at a low price, but has the effect that a company that is too optimistic in the contracting phase receives the assignment, but subsequently cannot meet the quality objectives without incurring serious losses. Heavily penalizing such a company for a low QoS will soon lead to bankruptcy, and therefore an even lower QoS until a new company has been found.

QoS is usually not specifically addressed in the allocation of rides. The minimization of company's costs is treated as a primary objective, while imposing a minimal QoS [1]. The idea put forward in this paper is to let companies compete on QoS on a day to day basis given a price per kilometer that is fixed in advance. Given known results that competition can reduce the total costs, the question is can we use it to improve the QoS instead, and at what costs? Here we divert from research on using auctions and other price-based mechanisms for task allocation in that not the company with the lowest price receives the task, but the company that guarantees the highest QoS. Our main hypothesis is that this approach significantly increases the QoS without much additional costs.

To test our hypothesis, we implement the proposed approach in a multiagent environment (see Section 3), simulate series of requests, simulate the bidding and scheduling process of the companies (in Section 4), and compute the resulting costs and QoS in Section 5. We compare these results to a single-company setting where the company optimizes costs with and without a guaranteed QoS level.

2 Background

DRT services are usually modeled as a Dial-a-Ride Problem with Time Windows (DARPTW), an extension of the Vehicle Routing Problem. A DARPTW is defined by a set of customers and a fleet of vehicles. Each customer desires to be transported from an origin location to a destination. Customers can impose a time window which includes the earliest possible time and the latest possible time they can be either picked up or delivered. The dynamic DARPTW (D-DARPTW) is NP-hard, which can be proven by a translation from the Traveling Salesman Problem (TSP) [2]. The problem can be solved exactly by modeling it as a Mixed Integer Program (MIP) [3], or by applying heuristics [4]. The disadvantage of using exact algorithms in a dynamic environment is that these algorithms take too much computation time. The disadvantage of using heuristics is that in some cases the solutions are significantly far from the optimal solution.

In a technique called *on-line optimization* the optimal solution is searched for with exact algorithms, but only taking into account that part of the problem that is relevant for the moment [5]. For instance, when searching for the best departure times of the locations of a request to insert into a current schedule, only that part of the current schedule that can be influenced by inserting the new request needs to be considered in the solution process. This results in smaller problems as input for exact algorithms, which implies less computation time. In our simulations of the multi-company environment we apply this online optimization for the insertion of a ride into the schedule of one of the companies.

3 The Multi-Company DARPTW

In this section, we define a DARPTW where multiple companies compete for requests. The general principle is that the companies announce an offer to the customer, who chooses the company that will serve its request. Conditioned on

some negotiated constraints, the winning company can then insert the request into its schedule.

3.1 Problem Definition

The DARPTW can be represented by a directed graph of locations and rides between these locations, $G = (L, R)$. The set of locations L contains two vehicle depots for each company, that serve as start and end vertex of all vehicles (denoted by 0 and $2n + 1$), n pickup locations $P = \{1, \ldots, n\}$, and n delivery locations $D = \{n + 1, \ldots, 2n\}$. This implies that $L = \{0, 2n + 1\} \cup P \cup D$.

A request is a combination of a pickup location $i \in P$ and a delivery location $n + i \in D$. Time windows are associated with a location i as $[s_i, e_i]$, with s_i the earliest possible time the request can be served at that location (either pickup or delivery), and e_i the latest possible time the request can be served. Each location i has an associated load q_i, which denotes the number of passengers that are to be pickup up or delivered at that location. We define $q_0 = q_{2n+1} = 0$, $q \geq 0$ for $i = 1, \ldots, n$, and $q_i = -q_{i-n}$ for $i = n + 1, \ldots, 2n$. Each company has a set of vehicles denoted by K, and with each vehicle $k \in K$ a maximal capacity Q_k and a maximal route duration T_k are associated. The cost of a ride from location i to j with vehicle k is denoted by c_{ij}^k, and the travel time of a ride between i and j is denoted by t_{ij}. To account for service time at locations (i.e. time to get in and out the vehicle), we associate with every location i a service duration $d_i \geq 0$ and $d_0 = d_{2n+1} = 0$.

3.2 Mechanism Overview

A customer announces its request to the center, and this is forwarded to all known companies. Once a company receives a new request, it checks whether it is possible to insert it into one of its vehicle schedules. If it is not possible to insert the request into one of the company's vehicle schedules, the company will not place a bid in the current auction. Otherwise, a bid value is calculated for this request. When the center has received all bids, it determines the best one (the highest QoS) and sets the conditions that have to be met by the winning company in serving the request. The winning company is informed of the determined conditions, and all other companies are sent a message that they have not won the auction. The winning company then inserts the request into the schedule of one of its vehicles. We assume there is always the possibility to have the request served by a taxi company outside the system at a (usually high) so-called reservation price. This is done when no bid is offered below this reservation price. The following subsections detail the elements of this process, starting with the computation of the QoS calculation.

3.3 Bidding Service Quality

Usually, the additional costs needed to serve a request is used as a bid [6], and to minimize overall costs, the request is assigned to the vehicle that has announced

the bid with the lowest additional costs. In our work, we let the companies compete on the QoS for an incoming request. Therefore, the bid value in our setting contains the QoS that a company promises to provide. We define QoS as the ratio of the actual ride time to the direct ride time. For instance, when the time to travel from A to B directly (i.e. with no detours) is equal to 5 minutes, and the vehicle drives from A to B via C, in 7 minutes, then the QoS is $\frac{5}{7}$. For customers, this measure emphasizes one of their biggest complaints, namely large detours. For companies, this ratio is a measure of how efficiently different rides are combined.

A vehicle's route is modeled by a sequence of locations i with associated service times d_i and departure times t_i. Furthermore let the minimal time needed to drive from location i to location j be denoted by t_{ij}. The QoS for a request with pickup location i and delivery location j can then be calculated as follows:

$$QoS_{ij} = \frac{t_{ij} + d_i + d_j}{t_j - t_i} \tag{1}$$

3.4 Auction on QoS and Pre-determined Payments

The mechanism that we propose is based on a *reversed sealed-bid second-price auction*, using QoS instead of prices. In such an auction, each bid is private to the company that submits it, and the winner of the auction has to meet the details of the second-highest bid value. The auction is *reversed*, because there are multiple sellers (the companies) and a single buyer (the customer). This single buyer announces the details of its request, and then the companies can determine a bid value. The winning company is the company that announces the highest QoS, and if multiple companies announce the same highest bid, one of these companies is arbitrarily selected as winner. The request that has been auctioned is allocated to the winning company, which then has to serve the request with the amount promised by the second-highest bidder.

In our setting, the payment for the service is not defined by the auction, but must be set on forehand. We set the payment equal to a *price per kilometer* C_{km} multiplied by the direct distance between the pickup and the delivery location of the customer's request. The profit of a company is then defined as the total *income* a company receives from serving requests minus the total *costs* needed to serve these requests. Clearly, C_{km} essentially determines the income of the companies.

Let for each request $(i, j) \in R$ the direct distance t_{ij} be given. For this problem, we define solutions for two hypothetical situations with complete knowledge of the requests during the day (in advance). We let $OPT(R)$ denote the transportation costs when all rides are optimally combined (in hindsight), and $OPT^{1.0}(R)$ denote the transportation costs when each request is served with a QoS of 1.0.

Proposition 1. *If an auction on QoS is used for multiple companies, and the (fixed) price per kilometer C_{km} is below $\frac{OPT(R)}{\sum_{(i,j)\in R} t_{i,j}}$ few people will be transported. If C_{km} is above $\frac{OPT^{1.0}(R)}{\sum_{(i,j)\in R} t_{i,j}}$, everyone will be transported separately.*

Proof. The lower bound for C_{km} is the minimal total costs needed to serve all requests divided by the total direct distance traveled by all customers. When C_{km} is set below this value, companies have more costs than income, except when a ride largely overlaps with an existing ride (which is never the case when the schedule is still empty). Companies will thus not bid in the auction. Therefore few people will be transported. On the other hand, when C_{km} is set above the average cost of transporting everyone separately, every company makes a net profit for each customer. Therefore, every company will bid a QoS of 1.0 for every request, because it then has the highest chance to win the auction. Since this holds for all companies, all requests have to be served with QoS 1.0.

4 Computations for the Companies

The computations of the transport companies are based on online optimization for the insertion of rides, and for bid determination use a look-ahead on possible future requests via a Monte Carlo simulation in combination with an insertion heuristic. Both approaches are detailed below.

4.1 Online Optimization

Every company has to solve a DARPTW problem to find the set of routes for their vehicles. We define a planning horizon H, which is the time period for which the routes are planned. A solution for one company is represented as follows. Let u_i^k be the time at which vehicle k starts servicing at a location i, w_i^k the load of vehicle k upon leaving location i, and r_i^k the ride time of a customer that places the request to travel from i to $n+i$. In the model that follows, x_{ij}^k is equal to 1 if and only if a ride from location i to j is allocated to vehicle k.

In Figure 1 the entire model is given, in which the objective function is to minimize total routing costs (see Equation 2). Constraint 3 ensures that all requests are served only once and Constraint 4 ensures that every vehicle will once drive to the start and end depot. Together with 5 and 6 this guarantees that every request is served once by the same vehicle and that each vehicle starts and ends its route at a depot. Constraint 7 states that the arrival time at a location must be higher or equal to the start time of servicing at the starting location, plus the service duration at that location, plus the time of the ride from start to end location. It is also obvious that the load of a vehicle at the end location of a ride is higher than or equals the load of that vehicle at the start location (Constraint 8).

The travel time of a user is higher than or equals the time the vehicle is at his delivery location minus the time he picked up the user and minus the duration of servicing at the pickup location; this is ensured by Constraint 9. Constraint 10 states that the duration of a vehicle to drive from the start depot to the end depot must be less than or equal to the total route duration specified for that vehicle, while Constraint 11 ensures that every location is visited within the specified time horizon. The ride time of a passenger is specified to be at least as

Minimize

$$\sum_{k\in K}\sum_{i\in L}\sum_{j\in L} c_{ij}^k x_{ij}^k \tag{2}$$

subject to

$$\sum_{k\in K}\sum_{j\in L} x_{ij}^k = 1 \qquad (i \in P), \tag{3}$$

$$\sum_{i\in L} x_{0i}^k = \sum_{i\in L} x_{i,2n+1}^k = 1 \qquad (k \in K), \tag{4}$$

$$\sum_{j\in L} x_{ij}^k - \sum_{j\in L} x_{n+i,j}^k = 0 \qquad (i \in P,\ k \in K), \tag{5}$$

$$\sum_{j\in L} x_{ji}^k - \sum_{j\in L} x_{ij}^k = 0 \qquad (i \in P \cup D,\ k \in K), \tag{6}$$

$$u_j^k \geq (u_i^k + d_i + t_{ij})x_{ij}^k \qquad (i,\ j \in L,\ k \in K), \tag{7}$$

$$w_j^k \geq (w_i^k + q_j)x_{ij}^k \qquad (i,\ j \in L,\ k \in K), \tag{8}$$

$$r_i^k \geq u_{n+i}^k - (u_i^k + d_i) \qquad (i \in P,\ k \in K), \tag{9}$$

$$u_{2n+1}^k - u_0^k \leq T_k \qquad (k \in K), \tag{10}$$

$$s_i \leq u_i^k \leq e_i \qquad (i \in L,\ k \in K), \tag{11}$$

$$t_{i,n+i} \leq r_i^k \leq H \qquad (i \in P,\ k \in K), \tag{12}$$

$$\max\{0, q_i\} \leq w_i^k \leq \min\{Q_k, Q_k + q_i\} \qquad (i \in L,\ k \in K), \tag{13}$$

$$x_{ij}^k = 0 \text{ or } 1 \qquad (i,\ j \in L,\ k \in K). \tag{14}$$

Fig. 1. Mixed Integer Program formulation to allocate requests to vehicles within a company. The objective function minimizes costs.

big as the time of a direct ride between its pickup and delivery location and at most as big as the planning horizon (Constraint 12). The load of a vehicle can never be higher than the highest possible load specified. Note that the load of a vehicle increases at a pickup location and decreases at a delivery location, so we can state Constraint 13. The last constraint ensures that we deal with binary variables, so that a variable denotes whether a ride is allocated to a vehicle or not, and we cannot specify that half of that ride is allocated to another vehicle. For the insertion of requests by a company, we use the mathematical model that we have just defined. This formulation evolves over time ignoring (previously inserted and) serviced requests.

4.2 Bid Calculation

To check whether an incoming request can be inserted, we use the insertion heuristics developed by Jaw et al. [4] and Solomon [7]. This is done before a bid value is calculated, and can save expensive computation time. If the request is feasible, the company can propose a bid to the customer.

A company wants to maximize its profit, defined as income minus costs. The company can gain income by serving requests (winning auctions) and it can decrease costs by combining requests. Combining requests often leads to a lower QoS (because there are less direct rides), which can lead to winning less auctions. Promising a higher QoS results in a higher probability to win the auction, but decreases the flexibility to insert future requests.

There are different costs associated with the different QoS values that a company can bid. In general, a company can bid a low QoS for low internal costs, or a high QoS for higher internal costs. For a single-shot second-price auction, it can be shown that it is optimal to bid the highest price possible. However, this is not the case in our (repeated) auction on quality.

Proposition 2. *If each company bids the highest QoS possible, all rides will be transported at QoS of* 1*.*

Proof. When a company bids the highest QoS possible for the first request, this bid will be a QoS of 1, since its vehicles have empty routes so far. When all companies follow this behavior, the second "price" will also be a QoS of 1, so the ride is accepted at a QoS of 1. When a route contains only rides with a QoS of 1, a next ride cannot be combined, so the highest QoS possible will also be 1. With induction, all rides will be transported at a QoS of 1.

To avoid this side effect of using QoS as a bid for the companies, we allow them to incorporate knowledge about future requests in their bid calculation. This way, they reason about the future possible combinations of the current request while bidding the promised QoS, instead of reasoning only about the current request. To incorporate the expected profit of future requests, the companies must have some knowledge about the distribution in time and space of future requests. From this distribution, they can calculate the expected profit for an incoming request, based on future requests that can give the companies possibilities to combine rides and lower costs. To this end, we use a Monte Carlo [8] simulation in combination with an insertion heuristic.

Estimating expected profit The idea is to estimate the *expected profit* that a company would make assuming that the current request is inserted into the schedule. To calculate the expected profit, a distribution has to be known on the arrival location and time of future requests. With the help of these distributions, a set of possible future requests can be generated and inserted into the schedule. Once this is done, the total costs needed to insert these requests, and the total income gained by inserting them can be calculated. This expected profit is calculated by using an insertion heuristic based on the works of Jaw et al. [4] and Solomon [7].

Monte Carlo simulations The specific set of generated future requests can have a big influence on the calculated expected profit. Therefore, a Monte Carlo simulation [8] is performed. The above algorithm is repeated a number of times,

and the final expected profit is taken as the average expected profit of these repetitions. To obtain the highest QoS level for the current request, taking into account future requests, Monte Carlo simulations are performed for different levels of QoS. The level for which the expected profit is closest to zero is taken as the bid value.

5 Experiments and Results

The main hypothesis to be tested is the following.

Hypothesis 1 *When multiple companies compete on QoS, the average QoS is higher than in a situation with a single company which minimizes costs. Transportation costs are also higher.*

But also for a single company we expect that transportation costs are higher when the QoS is higher:

Hypothesis 2 *A higher required QoS is more expensive (for a single company).*

As discussed before, requiring a higher QoS from a single company fails in practice, because in general, a higher QoS is more expensive, and for example penalties are not sufficient to incentivize a company to meet the agreed QoS. Our main thesis is that competition can be used to realize a higher QoS, and we expect that this can be done at approximately the same additional cost as for a single company, leading to a third hypothesis.

Hypothesis 3 *When multiple companies compete on QoS, the costs are not significantly higher than in a situation with a single company which minimizes costs with the same average QoS.*

5.1 Experimental Setup

To test these hypotheses, we run the mechanism and algorithms proposed in the previous section on a set of benchmarks. The size of these benchmark problems is chosen such that each one takes at most about 1 hour computation time to solve. This decision resulted in a set of 100 problem instances, each containing 16 customers, in which the coordinates and the pickup and departure times of the requests are distributed following a uniform distribution. We consider these instance over a planning period of 4 hours. The considered network is a continuous map, defined by a square area of 20 by 20 km with a node on every km. To make the problem instances dynamic, we add to each customer the moment at which it becomes available to the system. This is done by randomly choosing a number in the interval $[e_i - 90, e_i - 60]$ (i.e. between 90 and 60 minutes before the earliest pickup or delivery time). We choose these values because we want the instances as dynamic as possible. This means that customers announce their requests quite late, but such that vehicles are able to respond to schedule changes. The maximum capacity for each vehicle is set to 3 passengers. All

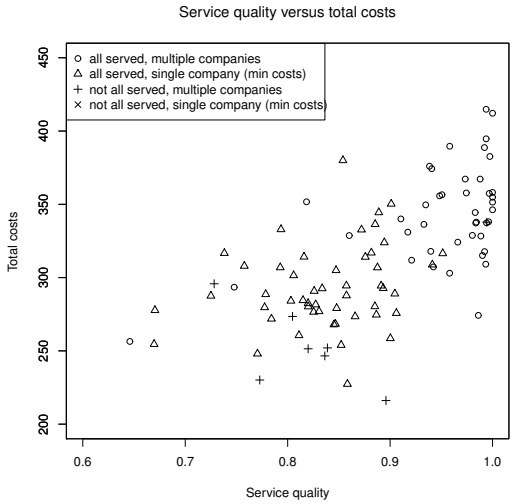

Fig. 2. QoS versus total costs for a multiple company setting and a single company

the experiments have been performed in Java and Java Agent DEvelopment Framework (JADE) [9] on an Intel Xeon E5345 2.33GHz with 16 Gb RAM. Each company uses the MIP-solver SCIP to insert assigned requests. SCIP is one of the fastest non-commercial solvers [10].

To test the hypotheses, we run the system two times for each problem instance. The first time two companies have two vehicles each and compete on QoS. The second time, there is only a single company, having four vehicles and minimizing costs. In a single-company setting, the company does not have any incentive to bid high QoS, because it knows already that it contractually gets assigned all the requests.

5.2 First Experiment

In Figure 2, a plot is given for the comparison of QoS and total costs between a situation with a single and a situation with multiple companies. Two clouds of points can be distinguished. The cloud of points of instances with multiple companies is situated with relative high QoS and high total costs. The other cloud of points has less quality and less total costs and mainly contains instances with a single company. A paired t-test is performed for both average QoS and total costs. The mean difference for QoS is 0.097 with a higher quality in the multi-company setting. The confidence interval is $[0.071, 0.122]$ and the probability that these results are obtained assuming that there is no difference between the two settings is 5.98×10^{-10}. A t-test gives us a mean difference of 37.5 higher total costs for the multi-company setting with a confidence interval of $[25.3, 49.8]$, and a p-value of 1.25×10^{-7}.

In conclusion, we have discovered that total costs are about 13% higher in the multi-company setting than in the single-company setting. The fact that

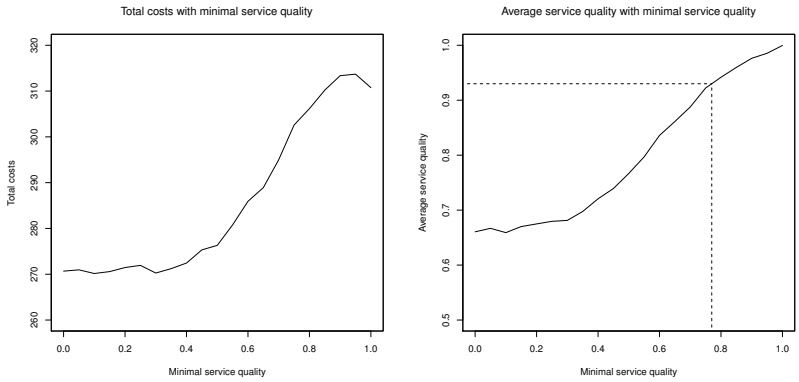

Fig. 3. Average total costs (left) and QoS (right) for instances with a minimal QoS

companies in a multi-company setting have the incentive to bid higher service quality instead of minimizing costs, results in a higher average service quality. This follows from the results of the paired t-test showing that the QoS of the multi-company setting is 12% higher. Both differences are (very) significant, confirming Hypothesis 1.

5.3 Second Experiment

In the previous experiment, the single company did not take any required level of QoS into account. However, to make a fair comparison of costs (to establish the second hypothesis), we would like to have the same average QoS for the single company as the multiple companies obtain by competition. To arrive at a certain average QoS, we ensure that each ride of the single company has a certain minimal QoS. To determine how to set the required QoS to arrive at such a desired *average* QoS, we first run the experiments for a single company for a required QoS of $\{0.00, 0.05, \ldots, 1.00\}$. We then investigate the relation between these required QoS levels and the average QoS, and at the same time the relation between the required QoS levels and the total costs, to test Hypothesis 2.

In Figure 3 (left), it is shown that the average total costs are increasing as from a QoS level of 0.4, with a slight decrease at level 1.0. This last decrease of average total costs can be interpreted by the fact that if the minimal QoS level is 1.0, less requests can be served, which leads to lower total costs. The non-increasing part of the figure (from level 0.0 to 0.4) can be explained by the fact that the levels do not influence the outcome, because even when a single company minimizes costs, it still serves requests with an average QoS of about 0.66. This is also shown in Figure 3 (right), in which we see no increase in QoS at this interval.

Besides these two exceptions, overall there is a strong correlation between total costs and minimal QoS. The correlation coefficient over the complete range 0.93, confirming Hypothesis 2. When we take only the interval from 0.40 to 0.95

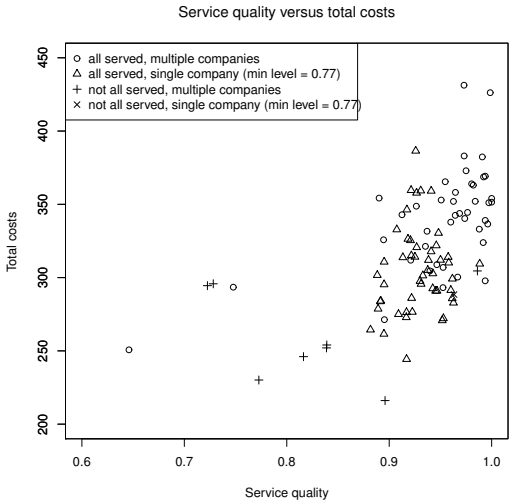

Fig. 4. QoS versus total costs for a multiple company setting and a single company setting in which the single company acts like a company in a multi-company setting.

into account, the correlation coefficient is even 0.99. The minimal QoS level that is needed in order to let the company in the single-company setting serve requests with an equal average QoS as in the multi-company setting is derived by searching in Figure 3 (right) for the corresponding level. The average QoS over all instances in the multi-company setting is 0.93 and when we search for the corresponding minimal QoS level we find a value of 0.77.

Subsequently, we run the first experiment again, but now requiring a QoS of 0.77 for the single company. In Figure 4 a scatter plot is shown in which QoS is plotted against total costs, for the multi-company and single-company setting. From this we observe that the costs are somewhat higher in the multi-company setting, but the points in the plot are too close to each other to give a proper judgment about this measure. The paired t-test gives us a mean difference of 23.5 total costs, a 95%-confidence interval of $[11.9, 35.0]$, and a p-value of 1.71×10^{-4}, with higher total costs in the multi-company setting. Another paired t-test is performed to verify that the difference in average QoS between the two settings is not significant. The results of this test confirm this with a mean difference of 0.0013 (higher in the multi-company setting), and a p-value of 0.92.

We thus conclude that compared to the multi-company setting, the total costs in a single-company setting are less, even if this single company provides equal QoS, rejecting Hypothesis 3.

6 Discussion

We conclude that if the price per kilometer is fixed within a reasonable range (Proposition 1), and expectations about the future are somehow taken into ac-

count (see also Proposition 2), it is indeed possible to obtain a higher QoS in door-to-door transportation by letting multiple companies compete on QoS (Hypothesis 1). However, in our experiments, the costs are about 7% higher than in the idealistic case where a single company always meets a required QoS while minimizing costs.

For future work, we aim to study other definitions of QoS. For instance, we are thinking about taking into account deviations from desired departure/arrival time. We also plan to consider more realistic generation of requests, based on real data. Besides, we plan to define mechanisms where companies compete both on QoS as well as on costs. In addition, we believe better results can be obtained if the mechanism allows for bidding on combinations of requests. Finally, this paper focused mainly on an experimental evaluation of the idea of competing on quality. It would be very interesting to also provide a theoretical analysis. This is particularly challenging, since the mechanism is in fact a sequential auction.

References

1. Cordeau, J.F.: A branch-and-cut algorithm for the dial-a-ride problem. Operations Research 54, 573–586 (2006)
2. Karp, R.M.: Reducibility among combinatorial problems. In Miller, R.E., Thatcher, J.W., eds.: Complexity of Computer Computations. Plenum Press (1972) 85–103
3. Ropke, S., Cordeau, J.F., Laporte, G.: Models and branch-and-cut algorithms for pickup and delivery problems with time windows. Networks 49, 258–272 (2007)
4. Jaw, J.J., Odoni, A.R., Psaraftis, H.N., Wilson, N.H.M.: A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. Transportation Research Part B: Methodological 20, 243–257 (1986)
5. Mahr, T., Srour, J., de Weerdt, M.M., Zuidwijk, R.: Can agents measure up? a comparative study of an agent-based and on-line optimization approach for a drayage problem with uncertainty. Transportation Research: Part C 18, 99–119 (2010)
6. Mes, M.: Sequential Auctions for Full Truckload Allocation. PhD thesis, Universiteit Twente, Enschede, The Netherlands (2008)
7. Solomon, M.: Algorithms for the vehicle routing and scheduling with time window constraints. Operations Research 15, 254–265 (1987)
8. Metropolis, N., Ulam., S.: The monte carlo method. Journal of the American Statistical Association 44, 335–341 (1949)
9. Bellifemine, F., Poggi, A., Rimassa, G.: JADE - a FIPA-compliant agent framework. In: Proceedings of the Practical Applications of Intelligent Agents, (1999)
10. Achterberg, T.: SCIP - a framework to integrate constraint and mixed integer programming. Technical Report 4-19, Zuse Institute Berlin (2004)