Online Dial-A-Ride Problem with Time-Windows Under a Restricted Information Model

Fanglei Yi[1], Yinfeng Xu[1,2], and Chunlin Xin[1]

[1] School of Management, Xi'an Jiaotong University,
Xi'an, ShaanXi 710049, P.R. China
fangleiyi@163.com
[2] The State Key Lab for Manufacturing Systems Engineering,
Xi'an, ShaanXi 710049, P.R. China
yfxu@mail.xjtu.edu.cn

Abstract. In online dial-a-ride problem with time-windows, requests for rides consist of two points in a metric space, a source and a destination. One server with some finite capacity is required to transports a specified amount of goods for requests from the sources to the destinations. Calls for rides come in while the server is travelling. Each request also specifies a deadline. If a request is not be served by its deadline, it will be called off. The server travels at unit speed in the metric space and the goal is to plan the motion of the server in an online way so that the maximum number of requests (or the maximum quantity of goods) is met by the deadlines of the requests. Usually it is assumed that the server knows the complete information on the ride when the requests are presented. We study this problem under a restricted information model. At the release time of one request, only the information on the source is presented. The server does not have the information on the destination until it reaches the source of the request. This models, e.g. the taxi problem, or elevator problem. We study the problem in the *uniform* metric space and *K-constrained* metric space. We perform competitive analysis of two deterministic strategies in the two types of metric spaces. The competitive ratios of the strategies are obtained. We also prove a lower bound on the competitive ratio of any deterministic algorithm of Z for the *uniform* metric space and of KZ for the *K-constrained* metric space, where Z denotes the capacity of the server and K denotes the diameter of the metric space.

1 Introduction

The class of Dial-A-Ride Problem (DARP) has been studied extensively in the area of operations research, management science, and combinatorial optimization because of its usefulness to the logistics and transportation industry. In DARP, there are servers that travel in some metric space to serve requests for rides. Each ride is characterized by two points in the metric space, a *source*, the starting point of the ride, and a *destination*, the ending point of the ride. The problem is to

S.-W. Cheng and C.K. Poon (Eds.): AAIM 2006, LNCS 4041, pp. 22–31, 2006.

design routes for the servers through the metric space, such that all requested rides are made and some optimality criterion is met. To meet real life needs, many new side constraints have been added to the problem. One useful extension is the Dial-A-Ride Problem with Time-Windows (DARPTW). Each request specifies a deadline. If a request is not be served by its deadline, it will be called off. The goal is to plan the motion of servers so that the maximum number of requests is met by their deadlines. In the natural online setting, requests for rides are presented over time while the servers are enroute serving other rides. And the servers do not know any information on the future requests at any time until they are presented. In other words, the time flows while decisions are made and executed in the online setting of the problem.

Traditionally, when a certain request is presented, all the information on the request becomes known. That is, both the source and the destination of the ride are specified completely upon presentation at the release time. However, in many practical situations complete specification of the rides is not realistic. Often only the source of the ride is presented at the release time of the request. The server is not able to get the information on the destination of the ride until it arrives at the source. This models, e.g. the taxi problem, or elevator problem. So this model is called the *restricted information model*.

In this paper we study the Online Dial-A-Ride Problem with Time-Windows under the Restricted Information Model (ODARPTWRIM). There is a server which has some finite capacity, travelling in an unit speed in a metric space to serve a set of requests for ride. Calls for rides come in with some amount of goods while the server is travelling. Each request specifies a deadline. If the server does not arrive at the source of the request by its deadline, the request will be called off. At the release time of one request, only the information on the source is presented. The server does not have the information on the destination until it reaches the source of the request. The goal is to plan the motion of the server in an online way so that the maximum quantity of goods is transported by the deadlines of the requests.

The online DARP and in general vehicle routing and scheduling problems have been widely studied for more than three decades (see [1] for a survey on the subject). Most previous researches on online routing problems focused on the objectives of minimizing the makespan [2, 3, 4], the weighted sum of completion times [2, 5], and the maximum/average flow time [6, 1]. In the paper [7, 8, 9], results on the online k-taxi scheduling problem have been presented, in which a request consists of two points (a *source* and a *destination*) on a graph or in a metric space. Subsequently, a similar problem, online k-truck scheduling problem has been studied in [10]. Both of them (online k-taxi/truck scheduling) assumed that k servers (taxies or trucks) are all free when a new service request occurs, and the goal is to minimize the total distance travelled by servers. [2] studied the online DARP in which calls for rides come in while the server is travelling. The authors also considered two different cases, where the server has infinite capacity and where the server has finite capacity. The first results of the online DARP under a restricted information model have been presented in [11] in which the

objective function is to minimize the time by which the server has executed all the rides and is back in the origin. All of these previous work assumed that the requests could wait for any length of time until the server completed them. That is, they did not consider the constraints of the time window.

Most previous works on the DARP with time window constraints are the off-line point of view. The input is known completely beforehand. For some related works, please refer to [12, 13]. In reference [14], we presented the first results on the online DARP with time windows. We study the problem again in this paper under the restricted information model .

We perform competitive analysis of deterministic algorithm for ODARPTW-RIM in two types of metric space, respectively. In the first metric space, uniform metric space, we present an online FCFS (First-Come-First-Serve) algorithm. We prove that FCFS is $2Z$-competitive, where Z denotes the server's capacity. And in the second metric space, K-constrained metric space, another online algorithm, Greedy Algorithm (abbr.GR), is shown which has a $2KZ$-competitive ratio, where K is the diameter of the metric space. We also prove a lower bound on the competitive ratio of any deterministic algorithm of Z for the uniform metric space and of KZ for the K-constrained metric space.

The rest of this paper is organized as follows. In section 2 we give some definitions and notations. Section 3 contains two online algorithms and the discussions of their performances. In section 4 we present lower bounds for the problem in two types of metric space. Section 5 concludes this paper.

2 Definitions and Notations

Let $\mathcal{M} = (X, d)$ be a metric space with n points which is induced by an undirected unweighted graph $G = (V, E)$ with $V = X$, i.e., for each pair of points from the metric space $\mathcal{M}$ we have $d(x, y)$ that equals the shortest path length in G between vertices x and y. We consider two types of metric space in this paper, the *uniform metric space* and the *K-constrained metric space*. In the uniform metric space, the distance between any two points is unit length. It can be considered as a special metric space that is induced by a complete graph with unit edge weights. And in the K-constrained metric space, $\frac{d_{max}}{d_{min}} = K$, where $d_{max} = \max d(x, y), d_{min} = \min d(x, y), x \neq y, x, y \in V$. Without loss of generality, we assume that d_{min} equals 1 and d_{max} is K long in the K-constrained metric space. We call K the *diameter* of the metric space which can be considered the maximum time required to travel between the two farthest points in the metric space. Note that in the uniform metric space, $K = 1$. An instance of the basic ODARPTW in the metric space $\mathcal{M}$ consists of a sequence $R = (r_1, r_2, \cdots, r_m)$ of *requests*. Each request is a quadruple $r_i = (t_i, z_i, a_i, b_i) \in \mathcal{R} \times \mathbf{N} \times X \times X$ with the following meaning: $t_i \in \mathcal{R}$ is the time that request r_i is released and $z_i \in \mathbf{N}$ is the quantity of goods that needs to be transported by the server; $a_i \in X$ and $b_i \in X$ are the source and destination, respectively, between which the goods corresponding to request r_i is to be transported. The capacity of the server is finite, denoted by constant Z. That is, the upper bound of the goods

loaded by the server is Z units. We assume the goods of the requests is partible. If a given request has overmany goods in the sense of the current capacity of the server, it can be divided into several partitions. The server is allowed to load some partitions of them. And the rest of them can be considered as the goods of new requests. In this paper we assume that $z_i \leq Z, \forall i \in \mathbf{N}$. It is also assumed that the sequence $R = (r_1, r_2, \cdots, r_m)$ of requests is given in order of non-decreasing release times, that is, $0 \leq t_1 \leq t_2 \leq \cdots \leq t_m$. A request is said to be *accepted request* if the corresponding object is picked up by the server at source, and a request is said to be *completed request* if the corresponding object is transported to the destination. We do not allow *preemption*: it is not allowed to drop an accepted request at any other place than its destination. This means, once a request is accepted, it will not be called off.

Definition 1. *[11] Under the restricted information model only the source a_i of r_i is revealed at time t_i. The destination of the ride becomes known only at picking up the ride in the source.*

In this paper, we consider the following assumptions for ODARPTWRIM as we did in [14]: a) The speed of the server is constant 1. This means that, the time it takes to travel from one point to another is exactly the distance between the two points; b) The window sizes for all requests are uniform, denoted by T. To make sure that the problem is feasible, we assume that $T \geq d_{min}$ in the K-constrained metric space and $T \geq 1$ in the uniform metric space.

We evaluate the quality of online algorithms by competitive analysis [1-3], which has become a standard yardstick to measure the performance. In competitive analysis, the performance of an online algorithm is compared to the performance of the optimal off-line algorithm, which knows about all future jobs. An algorithm $\mathcal{A}$ for ODARPTWRIM is called *α-competitive* if for any instance R the number of goods completed (transported) by $\mathcal{A}$ is at least $1/\alpha$ times the number of goods completed by an optimal off-line algorithm OPT.

3 Algorithms for Two Types of Metric Space

In this sections we firstly study the problem in a general metric space. We prove that there is no competitive deterministic online algorithm for a non-constrained metric space if the time windows $T > 0$ is arbitrary. Then we study the problem in two types of metric space, the uniform metric space and the K-constrained metric space, respectively. We propose the FCFS algorithm for the uniform metric space and the Greedy algorithm for the K-constrained metric space. The performance guarantees of the two algorithms for the problem are shown in this section.

Proposition 1. *If the time windows $T > 0$ is arbitrary, there is a metric space in which no deterministic online algorithm can obtain a constant competitive ratio for ODARPTWRIM.*

Proof. Consider a metric space which contains a line. At time 0, two request with one unit of goods is presented on the line. One request requires the server to load the goods at position T and deliver it to position $2T$, where T is the time window of requests. And the other requires the server to transport the goods from position $-T$ to position $-2T$. Note that the requests will be called off at time T if the server does not arrive at the sources by time T. If the online server does not immediately leave for the requests, the sequence stops and the server can not serve any one of the requests. Otherwise, we assume without loss of generality that at time T the online server reaches position $-T$.The off-line server arrives at position T at time T and delivers the goods to the destination $2T$ at time $2T$. Then, from time $t = 2T$ onwards a request is presented with the source position t and the destination position $t + T$ at each time $(2 + i)T$ $(i = 0, 1, 2, \ldots)$. The off-line server can serve all the requests whereas the online server is not able to complete any of them. □

3.1 FCFS Algorithm in the Uniform Metric Space

The FCFS algorithm works as follows. The online server always goes to the source of the request firstly that has been presented for the longest time among all yet unserved requests, getting the information on the destination and picking up the goods, then delivering them to the destination. If there are not unserved requests that can be *accepted* by their deadlines after completing a certain request, the server remains at its current point and waits for new requests to occur.

Theorem 1. *In the uniform metric space, FCFS algorithm is* $2Z$*-competitive for ODARPTWRIM, where* Z *is the capacity of the server.*

Proof. Given any input sequence R, it can always be divided into such m maximal sub-sequences, $R = (R_1, R_2, \ldots, R_m)$ that in each sub-sequence $R_i = (r_{i,1}, r_{i,2}, \ldots)(1 \leq i \leq m)$, the online server works continuously i.e. it serves constantly some requests in R_i one after another.

We will first analyze an arbitrary sub-sequence R_i and then extend the result to the whole sequence R. We will show that in each sub-sequence OPT can not transport (or pick up) more than $2Z$ times as many goods as FCFS does which is followed by the theorem.

Denote by t_1 the released time of the first request $r_{i,1}$ in R_i. Let $r_{i,l} \in R_i$ and $r_{i,l'} \in R_i$ be the last requests that FCFS and OPT served, respectively. Let t_{FCFS} be the time when FCFS reaches the ride destination of $r_{i,l}$, and t_{OPT} denote the time when OPT reaches the ride destination of $r_{i,l'}$. We define t^* which makes sure that $(t_{OPT} - t^*)/2$ is an integral and $t_1 \leq t^* \leq t_1 + 2$. There are two possible cases for t_{FCFS} and t_{OPT}.

Case 1. FCFS completes $r_{i,l}$ no later than time t_{OPT}, i.e., $t_{FCFS} \leq t_{OPT}$. It can be proved that $t_{OPT} - t_{FCFS} < 2$. Since the sub-sequence R_i is maximal and the server works continuously in R_i, no new requests which belongs to R_i can be presented at time $t \geq t_{FCFS}$. And the total time needed to pick up the goods of one request at the source and deliver them to the destination is not

more than 2 units of time in the unit metric space. Thus, in any interval $(t, t+2]$ for $t = t^*, t^* + 2, \ldots, t_{OPT} - 2$, the server of FCFS can pick up at least one unit of goods and/or deliver it to the destination while the OPT's server can not pick up more than $2Z$ units of goods at the sources and/or deliver them to the destinations due to the finite capacity of Z. Also, in the interval $(t_1, t^*]$ the FCFS's server can accept at least one request which has one unit of goods, and the OPT's server can not accept more than 2 requests (with $2Z$ units of goods) because $t^* - t_1 \leq 2$. We notice that the number of the goods which is accepted (picked up) by the server is as many as the one that is transported (delivered) since we do not allow preemption. So we can say that in any interval of $(t_1, t^*]$ and $(t, t+2]$ for $t = t^*, t^* + 2, \ldots, t_{OPT} - 2$, OPT can not transport more than $2Z$ times as many goods as FCFS does.

Case 2. FCFS still works after time t_{OPT}, i.e., $t_{FCFS} > t_{OPT}$. Obviously, in each interval $(t, t+2]$ for $t = t^*, t^*+2, \ldots, t_{OPT}-2$, OPT can transport at most $2Z$ times as many goods as FCFS does. It also holds in the interval $(t_1, t^*]$ with the same reasoning in case 1.

This completes the proof. □

3.2 Greedy Algorithm in the K-Constrained Metric Space

We present the GR as follows. At any time t when the server arrives at one point in the metric space, it finds the source of the request which has the least time path from its current position among all the outstanding requests that it can reach by their deadlines, picking up the goods and getting the information on the destinations, until it is overweighted or there are no outstanding requests. Then it delivers the goods of the accepted requests to their destinations.

Theorem 2. *In the K-constrained metric space, the Greedy algorithm is $2ZK$-competitive for ODARPTWRIM, where Z is the capacity of the server and K denotes the diameter of the metric space.*

Proof. The proof of theorem 1 goes through with a few changes. Given any input sequence R, we can always divide it into such m sub-sequences $R = (R_1, R_2, \ldots, R_m)$ that in each sub-sequence $R_i = (r_{i,1}, r_{i,2}, \ldots)(1 \leq i \leq m)$, the GR's server is empty when it begins to serve the first request $r_{i,1}$ of R_i. According to the algorithm of GR, under the circumstance of that the online server is not overweighted, the server always picks up the goods of requests if there are some outstanding requests existing or delivers the accepted goods to the destinations otherwise. So in each sub-sequence the online server will works continuously. As we did in theorem 1, we need only to show that OPT can transport at most $2ZK$ times as many goods as GR does for each sub-sequence.

Denote by t_1 the released time of the first request $r_{i,1}$ in R_i. Let $r_{i,l} \in R_i$ and $r_{i,l'} \in R_i$ be the last requests that GR and OPT served, respectively. Let t_{GR} be the time when GR reaches the ride destination of $r_{i,l}$, and t_{OPT} denote the time when OPT reaches the ride destination of $r_{i,l'}$. We define t^* such that

$(t_{OPT} - t^*)/2K$ is an integral and $t_1 \leq t^* \leq t_1 + 2K$, where K is the diameter of the metric space. There are two possible cases for t_{GR} and t_{OPT} in the same way. Note that in case 1, $t_{OPT} - t_{GR} < 2K$ since the total time needed to pick up the goods of one request at the source and deliver them to the destination is not more than $2K$ units of time in the K-constrained metric space. The rest of the proof holds unchanged. □

Corollary 1. *GR is $2Z$-competitive for ODARPTWRIM in the uniform metric space.*

Though GR has the same competitive ratio as FCFS does in the uniform metric space, we notice that FCFS is simpler than GR and the FCFS's server may carry less goods than GR's server during the working time.

4 Lower Bounds

In this section we derive lower bounds on the competitive ratio of any deterministic online algorithm for serving the requests in the version of the problem in two types of metric space, respectively. The results are obtained by considering the optimal algorithm as an adversary that specifies the request sequence in a way that the online algorithm performs badly.

Theorem 3. *In the uniform metric space, no deterministic online algorithm can obtain a competitive ratio less than Z for ODARPTWRIM, where Z is the capacity of the server.*

Proof. We consider an arbitrary deterministic algorithm $\mathcal{A}$ and an adversary ($\mathcal{AD}$) which constructs an input request sequence so that $\mathcal{A}$ will not achieve a competitive ratio small than Z. At time 0 both of their servers locate at the origin. Set the size of time window T is 1. $\mathcal{AD}$ will present the requests in steps. In the first step, $\mathcal{AD}$ ensures that the server of itself and the server of $\mathcal{A}$ are not at the same point in the metric space.

Step 1. Two requests with different positions of sources and destinations from each other are presented at time 0, each with one unit goods. That is, $r_1 = (0, 1, a_1, b_1)$ and $r'_1 = (0, 1, a'_1, b'_1)$ where $a_1 \neq a'_1 \neq b_1 \neq b'_1$. If the server of $\mathcal{A}$ does not leave immediately for the requests, then $\mathcal{AD}$ stops the sequence and let its server go to any one of points a_1 and a'_1, completing one request with 1 unit goods. Otherwise, we assume without loss of generality that $\mathcal{A}$'s server goes to a'_1, then the server of $\mathcal{AD}$ goes to a_1 for serving the request r_1. So $\mathcal{A}$ can serve at most 1 unit goods while $\mathcal{AD}$ is able to complete at least 1 unit goods. Go to the step 2.

Step 2. After step 1, there are two possible position for the server of $\mathcal{A}$.

P1. The server of $\mathcal{A}$ is at a vertex point different from the position of $\mathcal{AD}$.

P2. $\mathcal{A}$'s server is on an edge.

If *P1* holds, $\mathcal{AD}$ releases another two requests which have the same characteristic to the ones that are presented in step 1. If $\mathcal{A}$'s server does not immediately

go to one of points of the sources, it does not serve any one of these two requests, whereas $\mathcal{AD}$'s server can complete at least one request. Otherwise, after a short period of time $0 < \Delta t < 1$, $\mathcal{AD}$ releases a request r^* with Z units goods whose source point is not incident to the edge on which $\mathcal{A}$'s server is. Hence, the $\mathcal{A}$'s server can not pick up the goods by the deadline of request r^*, whereas $\mathcal{AD}$'s server can do that by remaining in its position for Δt time and then going to the source of r^*.

If *P2* holds, $\mathcal{A}$'s server is in the interior of and edge. Hence, it can not reach any vertex point which is not incident to this edge in 1 unit time. Now $\mathcal{AD}$ presents a request with Z units goods whose point of source is not incident to the edge on which $\mathcal{A}$'s server is. Thus, $\mathcal{A}$'s server is not able to serve the request, whereas $\mathcal{AD}$ can complete these Z units goods.

So in this step the server of $\mathcal{A}$ can complete at most 1 unit goods, and $\mathcal{AD}$'s server is able to serve at leat Z units goods.

In step 2, the adversary can arrange to make its server stay at a point different from the position of $\mathcal{A}$'s server when presenting the requests.

Step 3. Repeating step 2 for M times.

Denote by $|\mathcal{AD}|$ and $|\mathcal{A}|$ the total number of goods completed by $\mathcal{AD}$ and $\mathcal{A}$ respectively. We get

$$\frac{|\mathcal{AD}|}{|\mathcal{A}|} \geq \frac{ZM+1}{M+1}$$

As M grows, the right hand side gets arbitrarily close to Z. So the theorem holds. □

Theorem 4. *In the K-constrained metric space, no deterministic online algorithm can obtain a competitive ratio less than KZ for ODARPTWRIM, where Z is the capacity of the server and K denotes the diameter of the metric space.*

Proof. The proof follows the proof of theorem 3 closely. Let $T = K$. In step 1 the way in which $\mathcal{AD}$ presents the requests is the same to theorem 3.

In step 2, when *P1* holds, $\mathcal{AD}$ then presents a request r^* with 1 unit goods of which the source point a^* is K units length away from the current position of $\mathcal{A}$'s server. If the server of $\mathcal{A}$ does not go to a^* immediately, $\mathcal{AD}$ stop the sequence and let its server to complete the request r^* while $\mathcal{A}$'s server can not serve the request. Otherwise, $\mathcal{AD}$ releases K requests, $r_i (i = 1, 2, \ldots, K)$, each with Z units goods just after the $\mathcal{A}$'s server leaves. The source points $a_i (i = 1, 2, \ldots, K)$ and the destination points $b_i (i = 1, 2, \ldots, K)$ of the requests have the following characteristic. Each request for ride needs only 1 unit time to complete,.i.e., $d(a_i, b_i) = 1 (i = 1, 2, \ldots, K)$ and $a_i = b_{i-1}$ for $i = 2, 3, \ldots, K$. The source point of the first request is the current position of $\mathcal{AD}$'s server. And none of the source points of K requests is incident to the edge on which $\mathcal{A}$'s server is. Thus, the $\mathcal{A}$'s server can not serve any one of the K requests, whereas $\mathcal{AD}$'s server can complete all of them with KZ units goods. The rest of the proof remains the same. □

5 Conclusions

In this paper we discuss the online dial-a-ride problems with time windows under the restricted information model which is occurring in a wide variety of practical settings. It is an important issue since in practice complete information is often lacking[11]. We present two online algorithm for two types of metric space. For upper bounds, we analyze the performance of FCFS in the uniform metric space and of GR in the K-constrained metric space respectively. We also give the analysis for lower bounds on the competitive ratio of any deterministic algorithm for the problem in different metric spaces.

It is worth the whistle that the uniform metric space is in fact a special case of the K-constrained metric space with $K = 1$. It is shown in corollary 1 that GR has the same competitive ratio as FCFS does in the uniform metric space. However, we feel that FCFS is a simpler algorithm than GR. And more important, the FCFS's server may carry less goods than GR's server during the working time which implies that FCFS may cost less energy for per unit of goods than GR does in practice.

An interesting extension of the problem considered in this paper may take into account the non-uniform time windows. That is, each of the requests has a different size of time window. It would also be interesting to study other particular metric spaces (such as trees, cycles, etc.) to see if better bounds can be obtained. Other possible extension of the problem is to crew scheduling in which more than one server is used to serve the requests. All of these can be further investigated.

Acknowledgements

This work was partly supported by a grant from the National Science Fund of China (No.70471035) and the National Science Fund of China for Distinguished Young Scholars (No.70525004).

References

1. Krumke S.O., Laura L., Lipmann M., and Marchetti-Spaccamela et al. Non-abusiveness helps: An $o(1)$-competitive algorithm for minimizing the maximum flow time in the online traveling salesman problem. *Lecture Notes in Computer Science*, (2002) 200–214.
2. Feuerstein E. and Stougie L. On-line single server dial-a-ride problems. *Theoretical Computer Science*, **268**(1), (2001) 91–105.
3. Ascheuer N., Krumke S.O., and Rambau J. Online dial-a-ride problems: Minimizing the completion time. *Lecture Notes in Computer Science*, (2000) 639–650.
4. Ausiello G., Feuerstein E., Leonardi S., Stougie L., and Talamo M. Algorithms for the on-line traveling salesman. *Algorithmica*, **29**(4), (2001) 560–581.
5. Krumke S.O., de Paepe W.E., Poensgen D., and Stougie L. News from the online traveling repairman. *Theoretical Computer Science*, **295**, (2003) 279–294.

6. Hauptmeier D., Krumke S.O., and Rambau J. The online dial-a-ride problem under reasonable load. *Lecture Notes in Computer Science*, (2000) 125–136.
7. Xu Y.F. and Wang K.L. Scheduling for on-line taxi problem and competitive algorithms. *Journal of Xi'an Jiao Tong University*, **31**(1), (1997) 56–61.
8. Xu Y.F., Wang K.L., and Zhu B. On the k-taxi problem. *Journal of Information*, **2**, (1999) 429–434.
9. Xu Y.F., Wang K.L., and Ding J.H. On-line k-taxi scheduling on a constrained graph and its competitive algorithm. *Journal of System Engineering(P.R. China)*, **4**, (1999).
10. Ma W.M., XU Y.F., and Wang K.L. On-line k-truck problem and its competitive algorithm. *Journal of Global Optimization*, **21**(1), (2001) 15–25.
11. Lipmann M., Lu X., de Paepe W.E., and Sitters R.A. On-line dial-a-ride problems under a restricted information model. *Lecture Notes in Computer Science*, **2461**, (2002) 674–685.
12. Psaraftis H.N. An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows. *Transportation Science*, **17**, (1983) 351–357.
13. Diana M. and Dessouky M.M. A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows. *Trasportation Reserch Part B*, **38**, (2004) 539–557.
14. Yi F.L. and Tian L. On the online dial-a-ride problem with time windows. *Lecture Notes in Computer Science*, **3521**, (2005) 85–94.