Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29 August 2004

SCHEDULING FOR ON-LINE TAXI PROBLEM ON A REAL LINE AND COMPETITIVE ALGORITHMS

CHUN-LIN XIN [1], WEI-MIN MA[2]

[1] School of Management, Xi'an Jiaotong University, Xi'an, ShaanXi 710049, P.R. of China
[2]School of Economics and Management Tsinghua University, Beijing 100084, P.R. of China
E-MAIL: xincl@mailst.xjtu.edu.cn, mawm@em.tsinghua.edu.cn

Abstract:

The problem of *k*-taxi is a generalization of the famous *k*-server problem, where we must choose how *k* mobile servers will serve a sequence of requests occurring in the metric space. In our case we consider how *k* taxies on a real line to serve a sequence of requests must be determined in an on-line manner. Our goal is to minimize the transportation cost of all taxies. With the help of Position Maintaining Strategy (PMS for short), a competitive algorithm that has a ratio of *k*+2 is given. Besides, another deterministic on-line strategy is also illustrated, namely, the Partial Double Coverage Strategy (PDCS for short), a competitive algorithm with competitive ratio of *k* is given. In this paper, a particular case concerning elevator is also analyzed and some results have been obtained. Finally, some open problems for the on-line *k*-taxi problem on a real line are proposed.

Keywords:

On-line problem; *k*-taxi on a real line; competitive algorithms; competitive ratio

1. Introduction

FOUNDATIONS

Most traditional optimization theories are considered from the angle of stander-by. Optimal resolutions are obtained under the condition of mathematic models and qualified provisions. Hence, many algorithms and related theories are gotten; these algorithms make full use of all parameters of the problem. But the parameters of a lot of optimization problems in real life can not be thoroughly known at the very beginning, which are obtained one by one in the process of resolution. We know nothing about the further but only make our decision according to the current event. These problems different from the traditional ones are considered as on-line problems.

An on-line algorithm receives the input incrementally, one piece at a time. In response to each input portion, the algorithm must generate output, not knowing the future input. In a competitive analysis, an on-line algorithm A is compared to an optimal off-line algorithm *OPT*. An optimal offline algorithm knows the entire input sequence in advance and can process it optimally. Given an input sequence σ, let $C_A(\sigma)$ and $C_{OPT}(\sigma)$ denote the costs incurred by A and *OPT* in processing σ respectively. Algorithm A is called α-competitive if there exists a constant α and β, such that

$$C_A(\sigma) \le \alpha \cdot C_{OPT}(\sigma) + \beta \text{ , } \forall \sigma \text{ .}$$

An analogous definition can be given for on-line maximization problems. We note that a competitive algorithm must perform well on all input sequences. When the additive constant β is less than or equal to zero, we may say for emphasis that on-line algorithm is strictly α-competitive.

RELATED WORK

Over the past two decades, *on-line problems* and their *competitive analysis* have received considerable interest. On-line problems had been investigated already in the 1970s and early 1980s but an extensive and systematic study started only when Sleator and Tarjan [1] suggested comparing an on-line algorithm to an optimal off-line algorithm and Karlin et al. [2] coined the term competitive analysis. In the late 1980s and early 1990s, three basic on-line problems were studied extensively, namely paging, the *k*-server problem and metrical task systems. The *k*-server problem, which is introduced by Manasse et al. [3], generalizes paging as well as more general caching problems. The problem consists of scheduling the motion of *k* mobile servers to reside on the points of a metric space *S*. The metrical task system, which is introduced by Borodin et al. [4], can model a wide class of on-line problems. An on-line algorithm deals with events that require an immediate response. Future events are unknown when the current is dealt with. The task system [4], the *k*-srver problem [5], and on-line/off-line games [6] all attempt to model on-line problems and algorithms. During the past few years, apart from the three basic problems, many on-line problems have been investigated in application areas such as data structures, distributed data management, scheduling and load balancing, routing, robotics, financial games, graph theory, and a number of problems arising in computer systems.

OUR CONTRIBUTIONS

We are the first to propose the on-line problem of *k*-taxi on the real line. The objective is to minimize total cost needed to serve the request sequence without the future knowledge. We adopt the Position Maintaining Strategy (PMS for short) and Partial Doubling Coverage Strategy (PDCS for short) respectively, and the results obtained show that PDCS is better than PMS. Finally, a particular case concerning elevator is analyzed and some results have been obtained successfully.

2. Problem Statement

The problem of the *k*-taxi on a real line can be stated as follows: there are many people who flow in a very busy main street of some city every day. The main transportation chosen by them is taxi. Because of the limitation of transportation infrastructure, the management department only permits *k* taxies to serve in this street. If the taxi company wants to serve their customers to the greatest extent and make profit for their company as well, how to schedule taxies for the customers who give special service request one by one is very important. We suppose without loss of generality that there are *k* taxies to serve requests; the service centre through scheduling taxies serves every request. Let's consider following problems first:

(1) *Given a service request sequence, how can we schedule the taxies so as to minimize the cost?*
(2) *If the service request is received one by one in the process of service without any knowledge of the future requests, how to minimize the cost as possible as we can?*

The *k*-taxi problem on a line aims at minimizing the cost of all taxies, namely, the least total distance. Problem (1) is an off-line problem, (2) an on-line problem. The difference between them is whether the known service request sequence is all or nothing. The former can be solved easily with the dynamic programming, but the latter is difficult to be solved. We must serve the request only based on the information of the previous requests; the decision must be made on-line, as we have no information about the future request. In fact, service request sequence has fatal effect on scheduling plan. With different service request, optimal plan will change accordingly.

In this paper, we first establish the mathematic model of the *k*-taxi problem on a line in section 3, and then give the result of this problem with the Position Maintaining Strategy (PMS for short) in section 4. In the section 5, another better result is obtained with the Partial Double Coverage Strategy (PDCS for short). Finally, a special case is solved successfully.

3. The Model of the *k*-taxi Problem on a Real Line

We suppose that *k* taxies serve in a flourishing urban main street of some city. A service request $r_i=(a_i,b_i)$, implies that a customer moves form a_i to b_i by taxi, A service request sequence *R* consists of some service request in turn, namely, $R=(r_1,r_2,\cdots,r_m)$ where $r_i=(a_i,b_i)$. On-line problem of *k*-taxi on a real line is to decide which taxi should be moved when every new service request occurs on the basis that we have no information about future possible requests.

All discussion is based on an essential assumption: when a new service request occurs, *k* taxies are all free.

For a request $r_i=(a_i,b_i)$, the process of scheduling a taxi to a_i is called empty load; And that from a_i to b_i is heavy load ; If a_i=b_i, then $r_i=(a_i,b_i)$ is called the degenerate service request. If there is no limit for the *R*, the on-line *k*-taxi problem is called *P*; If for any $r_i=(a_i,b_i)$, $d(a_i,b_i)>0$ holds, the problem is called P1; If for any $r_i=(a_i',b_i)$, $d(a_i,b_i)=0$ holds, that is, a_i=b_i, the on-line problem is called P2 (in this case, all of the service request are degenerate; Problem P2 is also called *k*-server problem on a real line.

4. Result by the Position Maintaining Strategy (PMS for short)

4.1. Background of relative theory

On-line problem of *k*-taxi on a real line is presented as a special case of on-line *k*-taxi scheduling problem, that is, all of the taxies and service requests fall on the real line. In the paper [13, 14], the PMS was proposed and was used to get several good research results for the *k*-taxi problem, there exists an on-line algorithm for the normal *k*-taxi problem with competitive ratio 2*k*+1, and there exists an on-line algorithm for the *k*-taxi problem on constrained graphs with competitive ratio $1+(n-k)\cdot\lambda$.

The on-line *k*-server problem on a real line is presented as a special case of on-line taxi problem on a real line in 1990 [7]. There is a very good result for this problem, M. Chrobak, H.Karloff, F.Payne, and S.vishwanathan showed that there exists an on-line algorithm for the *k*-server problem on a real line with competitive ratio *k*. Because *k*-server problem on a real line is a special case of the *k*-taxi problem on a real line, the researches concerning the on-line *k*-taxi problem on a real line must discover the relationship between them. Here we firstly give following lemma [7].

LEMMA 1. *There exists an on-line algorithm for the k-server problem on a real line with competitive ratio k.*

4.2. Result by Position Maintaining Strategy on This Problem

In the paper [13, 14], the *PMS* was proposed and was used to get several good research results for the *k*-taxi problem. For our investigation of *k*-taxi problem on a real line, we outline the PMS as follows.

Position Maintaining Strategy is defined as follows. For the present request $r_i=(a_i,b_i)$, after a_i is reached, the taxi reaching a_i must move from a_i to b_i to complete r_i. After the service for r_i is finished, the PMS moves the taxi at b_i back to a_i before the next request arrives.

Main idea of PMS:

To keep the system steady before any service request arrives. It is advantage to compare two problems.

Here, we give two lemmas [13] quoted in this paper.

LEMMA 2. *For an arbitrary known sequence* $R=(r_1,r_2,\cdots,r_m)$, *where* $r_i=(a_i,b_i)$, $\sigma=(a_1,a_2,\cdots,a_m)$, *let* $C_{OPT}(R)$ *denote the optimal total cost after finishing off-line service request R, let* $C_{OPT}(\sigma)$ *denote the optimal total cost after finishing off-line service request* σ, *and* σ *is the service sequence* $R=((a_1,a_1),(a_2,a_2),\cdots,(a_m,am))$, *then the following inequality hold.*

$$C_{OPT}(R)\le C_{OPT}(\sigma)$$

LEMMA 3. *If there is a c-competitive on-line algorithm for k-server problem on a real line, then there is a* $c+2$ *-competitive on-line algorithm for the k-taxi problem on a real line.*

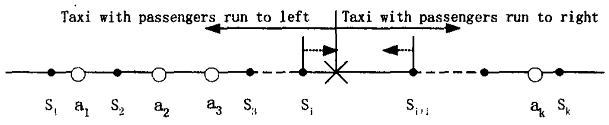
From the lemmas mentioned above, the following corollary holds.

COROLLARY 4. *There exists a* $k+2$ *-competitive on-line algorithm for k-taxi problem on a real line.*

5. Result by Partial Double Coverage Strategy (PDCS for short)

In this section, we firstly give the model of *k*-taxi problem on a real line, and then introduce an algorithm, namely, Partial Double Coverage Strategy. The algorithm is simple, memoryless, and achieves the competitive ratio: *k*.

5.1. The Model of *k*-taxi Problem on a Real Line


Taxi with passengers run to left
Taxi with passengers run to right
S1
a1
S2
a2
a3
S3
Si
Si+1
ak
Sk
Figure 1. request occurs between neighboring taxies

In the figure 1, we labeled our *k* taxies $s_1,s_2,\cdots,s_k$; the adversary's *k* taxies are labeled $a_1,a_2,\cdots,a_k$. If necessary, we may assume that always $s_1\le s_2\cdots\le s_k$ and $a_1\le a_2\cdots\le a_k$.

5.2. Partial Double Coverage Strategy

If the request falls outside of all the taxies, then serve it with the nearest taxi to finish all the service. Otherwise, if the request occurs between two adjacent taxies, and then both taxies run toward the request at equal speed until one reaches it. (If two taxies occupy the same spot, then one is chosen arbitrarily). And then let the taxi that reaches the request spot serve the passenger.

Main idea of the PDCS:

If the request occurs outside of all the taxies, greedy strategy is adopted (the taxi which is the nearest goes to serve the request). Clearly the cost of other taxies for the request will be more than that of the nearest taxi; if the request occurs between two adjacent taxies, greedy strategy can lead to the situation in which one is busy and others are idle. In other words, one of the taxies is continually be scheduled by adversary, others leave unused (in the paper [12]). Optimal strategy is that two taxies serve passenger at the same time.

At first, we want a potential function Φ that has the following properties:

a) $\Phi \geq 0$;

b) When the adversary incurs a cost x, the change in potential, $\Delta\Phi \leq k \cdot x$;

c) When the online player incurs a cost x, the change in potential, $\Delta\Phi \leq -x$;

LEMMA 5. [7] *The existence of such a potential function* Φ *implies that the PDCS is k-competitive.*

With the above algorithm, we have the following theorem.

THEOREM 6. *There exists a k-competitive on-line algorithm for k-taxi problem on a real line.*

Proof. For on-line *k*-taxi problem on a real line, we construct a potential function as follow.

$\Phi = k\psi + \gamma$. Here ψ is the weight of the minimum weight in the bipartite graph where the taxies of the adversary and the online algorithm form the two parts. For the line, it trivial to see that it is simply $\psi = \sum_{i=1}^{k} d(s_i, a_i)$, Further, $\gamma = \sum_{i<j} d(s_i, s_j)$, which is simply the sum of the distances between every pair of on-line taxies. S_i denotes the *i*th taxi by the PDCS, $i = 1, 2, \cdots, k$. a_i denotes a corresponding nearest taxi with the adversary.

1) With this definition, Φ is clearly non-negative, namely, $\Phi \geq 0$.

2) Assume the adversary serves by moving a_i to position a_i', and then carries the passengers from a_i' to a_i'', Hypothesis $x_1 = d(a_i, a_i')$, $x_2 = d(a_i', a_i'')$, the cost of the move is x, $x = x_1 + x_2$, This does not change γ. Namely, $\Delta\gamma = 0$. For ψ,

$$\begin{aligned}\Delta\psi &= d(a_i'', a_i) \\ &= d(a_i'', s_i) - d(a_i, s_i) \\ &\leq x_1 + x_2\end{aligned}$$

Hence, $\Delta\Phi \leq kx$。

We discuss on-line algorithm with two possibilities.

Suppose that the nearest distance of moving a taxi to the request is x_1, the distance from the request spot to destination is x_2.

First possibility: if the request happens on one side of all the taxies, then moves the nearest taxi to satisfy service according to PDCS.

1) The cost is incurred by on-line player is x, namely, $x = x_1 + x_2$;

2) . The potential function is changed, ψ is decreased by $x_1 + x_2$, γ is increased by $(k-1)\cdot(x_1 + x_2)$, and finally,

$$\begin{aligned}\Delta\Phi &\leq -k\cdot(x_1 + x_2) + (k-1)\cdot(x_1 + x_2) \\ &= -(x_1 + x_2) \\ &= -x\end{aligned}$$

Second possibility: the request of scheduling taxi is given between s_i and s_{i+1}, namely at *r*. According to PDCS, we know that at the same time taxi s_i and s_{i+1} run toward *r* at the same speed. At least one of the taxies arrive *r* from its match, and then let it finish the task of transporting the passenger.

1) The cost is incurred by on-line player is x, namely, $x = 2x_1 + x_2$;

2) By analyzing the change of the potential function, we divide the service request into two phases: demanding and transporting.

Firstly, we analyze the change of ψ in the potential function. Two adjacent taxies run toward *r* at the same speed in the first phase, therefore at least one of the taxies reduces its distance from its match, the other potentially may increase its distance from its match by at most the same amount. Suppose that both of the increased costs are $\Delta\psi$, namely, $\Delta\psi' \leq 0$. In the second phase, the taxi that has reached the spot finishes the service of transporting the passenger. Clearly $\Delta\psi'' \leq x_2$, all $\Delta\psi$ is:

$$\begin{aligned}\Delta\psi &= \Delta\psi' + \Delta\psi'' \\ &\leq x_2\end{aligned}$$

Secondly, we analyze the change of γ. Suppose that the taxi s_i arrives at first, the distance between s_i and *r* is x_1, the change of γ is $\Delta r'$ in the first phase, according to DPCS we know, $\Delta r' = -2x_1$; in the second phase, the taxi s_i serve the request of transporting passenger, and the distance from request point to destination is x_2, there are two possible directions to the destination, either on the left of the taxi s_i, or on the right of the taxi s_i, we firstly consider the right side, then the change of γ is $\Delta r''$,

$$\begin{aligned}\Delta\gamma'' &= (i-1)\cdot x_2 - (k-i)\cdot x_2 \\ &= (2i-k-1)\cdot x_2\end{aligned}$$

since $1 \le i \le k$, then $\Delta r'' \le (k-1)\cdot x_2$,

The change of γ is

$$\begin{aligned}\Delta r &= \Delta r' + \Delta r'' \\ &\le (k-1)\cdot x_2 - 2x_1\end{aligned}$$

This time we consider the destination at the left of the taxi s_i, suppose that the change of γ is $\Delta r''$,

$$\begin{aligned}\Delta\gamma'' &= -(i-1)\cdot x_2 + (k-i)\cdot x_2 \\ &= (k-2i+1)\cdot x_2\end{aligned}$$

since $1 \le i \le k$, then $\Delta r'' \le (k-1)\cdot x_2$,

we know again,

$$\begin{aligned}\Delta r &= \Delta r' + \Delta r'' \\ &\le (k-1)\cdot x_2 - 2x_1\end{aligned}$$

In a word, the change of this potential function is

$$\begin{aligned}\Delta\Phi &\le -k\cdot x_2 + (k-1)\cdot x_2 - 2x_1 \\ &= -(x_2 + 2x_1) \\ &= -x\end{aligned}$$

The proof is completed. □

6. A Special Case

In the paper [16], on-line k-elevator problem was proposed. This problem is a special case of on-line k-taxi problem on a real line, namely, suppose that the distance between every two neighboring lays is equal to 1, the competitive rate given by PMS is $k+2$. This problem can be considered as on-line k-taxi problem under restrained condition. The result can be obtained by PDCS as follows.

COROLLARY 7. *There exists a k-competitive on-line algorithm for k-elevator problem on a real line.*

We can compare two algorithms as follow:

In this paper and the paper [16], two competitive algorithms of on-line k-elevator problem are given respectively. The criterion, with which one can judge which on-line algorithm is better than other, is the competitive ratio concerning relevant on-line algorithm. Respectively the competitive ratios of PMS and PDCS are

$$C_A = k+2, \text{and}$$
$$C_B = k.$$

Clearly, the following result holds.

For the problem of on-line k-elevator, B is better than A.

7. Conclusions and Future Work

The k-taxi problem on a real line is a generalization of the k-server problem on a real line. The competitive ratio of two problems is k, this research result shows that the task of transporting passenger does not increase the competitive ratio. Recently, on-line k-server problem [9, 10, 11] and relevant variation had been studied successfully, such as on-line k-server problem on a tree or on a star tree etc [7, 8]. At present, there are also some good research results about on-line k-taxi problem [13, 14, 15, 17, 18]. Some realistic optimal guidelines and problems also exist.

Some cases for the k-taxi problem on a real line are still open. For example, (1)in the condition of request sequence having some probability distribution, how to design a reasonable scheduling strategy. (2) In the condition of minimizing the maximal time that the passengers are waiting, how to design an on-line scheduling strategy. Namely, to every service request $r_i = (a_i, b_i)$, how to schedule a taxi to reach a_i with proper fee firstly? And with these competitive algorithms, can we get some better competitive ratio?

The research of on-line k-taxi problem on the real line and competitive strategy provide us new ideas about many practical on-line problems. Concerning the dynamic complicated problems of social economy and management, we can try to explore in this way.

Acknowledgements

This project was granted financial support from China Postdoctoral Science Foundation (2003034014) and the National Natural Science Foundation of China (70231010).

References

[1] D. D. Sleator and R. E. Tarjan. Amortized efficiency

of list update and paging rules, *Comm. ACM*, Vol: 28, Iss: 2, 1985, P:202-208

[2] A.R.Karlin, M.S.Manasse, L.Rudolph and D.D.Sleator, Competitive snoopy caching, *Algorithmica*, 3 (1) 79-1 19, 1988.

[3] M.S. Manasse, L.A. McGeoch and D.D. Sleator. Competitive algorithms for on-line problems. *In Proc. 20th Annual ACM Symp. On Theory of Computing*, 322-33, 1988.

[4] A.Borodin, N. Linial and M. Sakes. An optimal on-line algorithm for metrical task systems. *Journal of the ACM*,39: 745-763, 1992

[5] M.S. Manasse, L.A. McGeoch, and D.D. Sleator, Competitive algorithms for server problems, *Journal of Algorithms*, 1990(11), 208-230.

[6] S. Ben-David, A. Borodin, R.M. Karp, G. Tardos and A. Wigderson. On the power of randomization in on-line algorithms. *Algorithmica*, 11:2-14,1994

[7] M. Chrobak, H. Karloff, T.Payne, and S. Vishwanathan. New Results on Server Problem. In *Proc. 1st ACM-SIAM Symp. On Discrete Algorithms*, 291-300, January 1990.

[8] M. Chrobak and L.Larmore. An Optimal On-Line *k*-Server Algorithm for Trees. *SIAM Journal of Computing*, 20:144-148,1991

[9] Alon N, Karp R M, peleg D, etal. A graph-theoretic game and its application to the *k*-Server problem. *SIAM J Comput*,1995,24(1):78～100

[10] S. BenDavid and A. Borodin, A new measure for the study of the on-line algorithm *[J]. Algorithmca*, 1994(11), 73-91

[11] E. Koutsoupias, C. Papadimitriou, On the *k*-Server conjecture, *STOC.*,507-511,1994

[12] D.Z.Du, *k*-server problem and competitive algorithm, *Practice and Acquaintanceship of Mathematics* (In Chinese), 1991(4): 36-40

[13] Y.F.Xu, K.L.Wang, On-line *k*-taxi problem and competitive algorithm. *Journal of Xi'an Jiaotong University (In Chinese)*, 1997(1): 56-61

[14] Y.F.Xu, K.L.Wang, J.H.Ding, On-line *k*-taxi scheduling on a constrained graph and its competitive algorithm, *Journal of System Engineering (In Chinese)*, 1999(12): 361-365

[15] W.M.Ma, Q.C.Xu, Off-line scheduling of *k*-taxi problem and its dynamic programming procedure optimal algorithm, *Journal of System Engineering (In Chinese)*, 2001(16): 481-490

[16] B.A.Ying, Y.F.Xu, Y.Zhu, Scheduling for on-line elevator problem and competitive algorithm, *Aeronautical Computer Technique (In Chinese)*, 2001(2): 47-50

[17] W.M.Ma, Y.F. Xu, and K.L. Wang. On-line *k*-truck problem and its competitive algorithm. *Journal of Global Optimization*, 21 (1): 15-25, September 2001.

[18] W. M. Ma, J. You, Y. F. Xu, J. Liu, and K. L. Wang. On the on-line number of snacks problem. *Journal of Global Optimization*, 24 (4): 449-462, December 2002.