Int. J. Mach. Learn. & Cyber. (2017) 8:1185–1195
DOI 10.1007/s13042-015-0489-4

ORIGINAL ARTICLE

Value of foreknowledge in the online *k*-taxi problem

Xin Zheng[1] · Ke Wang[2] · Weimin Ma[1]

Received: 18 October 2015 / Accepted: 24 December 2015 / Published online: 5 January 2016

Abstract This paper investigates the online *k*-taxi problem with a new feature that partial information about future service requests is provided in advance when deciding which taxi should be dispatched to serve the current request, whereas none of this is known in the traditional online *k*-taxi problem. Benefited by the foreknowledge, improved covering strategies are proposed for the problem in different scenarios with respect to the information known in advance. Following that, the value of foreknowledge in the online *k*-taxi problem with this new feature is quantified in the form of improved competitive ratios. It is proved that in some special cases, the competitive ratios are decreased from $1 + \lambda$ to $(3 + \lambda)/2$, where λ is a parameter determined by the metric space in which the problem is discussed. Furthermore, it is also shown that, in all cases, the improved covering strategies would never perform wore than the classical position maintaining strategy. In addition to these theoretical analyses, some numerical examples are presented to illustrate the proposed online strategies and their performances in practice as well.

Keywords *k*-Taxi problem · Online problem · Value of foreknowledge · Competitive analysis · Competitive ratio

✉ Ke Wang
ke@shu.edu.cn

1 School of Economics and Management, Tongji University, Shanghai 200092, China

2 School of Management, Shanghai University, Shanghai 200444, China

1 Introduction

In the real world, many ongoing decision-making activities in various areas, such as foreign currency trading [1, 2], vehicle routing [3, 4], and inventory ordering [5, 6], must be carried out in due course with no secure knowledge of future situations. However, such knowledge (e.g., future exchange rates in the foreign currency trading, service requests in the vehicle routing, and customer demands in the inventory ordering) often has a critical impact on the decision results.

Faced with this lack of knowledge, players of these decision-making games often formulate them as stochastic models based on assumptions about the future distributions of relevant quantities. Unfortunately, they may give some real-time solutions that are far from the relevant optimal solutions in some scenarios. Additionally, for most non-trivial decision making activities, it is difficult to estimate probability distributions accurately. In this case, the approach of probabilistic analysis is no longer suitable for addressing this type of ongoing decision-making problems. Another common way to deal with uncertain information in the decision-making problems is to model it as fuzziness (e.g., [7–9]), which also requires to make some estimations of the future uncertain situations. Unlike these two approaches, competitive analysis proposed by Sleator and Tarjan [10] provides a new appropriate framework to handle these problems, in which estimations of the uncertain situations are not required.

In the competitive analysis theory [11, 12], an algorithm is said to be *online* if its decisions are made in due course with no secure knowledge of future events. In general, due to the absence of information about the future, an online algorithm cannot make the decision in an optimal fashion. The performance of an online algorithm is measured against that of the optimal *offline* algorithm, which knows all the information when making its decision and thus, the decision can be optimized. The supremum of the ratio between the performances of these two algorithms is called *competitive ratio*. If the ratio is a constant α, then we say the online algorithm is an *α-competitive algorithm*. It implies that the cost of the online algorithm will not exceed α times of the optimal offline cost for any inputs. It is clear that this performance measure provides very robust statements about the performance of an online solution, against all possible future scenarios.

The online k-taxi problem [13] is a generalization of the k-server problem introduced by Manasse et al. [14], which has been extensively studied (see, e.g., [15–18]) as one of the famous basic online problems discussed in competitive analysis theory. In the k-server problem, k servers reside and move in a metric space to supply service. When a request is made by a point, one of the servers must be moved to the point to satisfy this request immediately. An algorithm which decides a server to satisfy the request at each step is said to be online if its decisions are made without the information about future requests. The goal is to minimize the total cost or distance of satisfying all requests.

In the online k-taxi problem, different from that in the k-server problem, each request contains two points, one of which is the start point making the request and the other is the destination point. It implies that there is a passenger at the start point to be picked up and taken to the destination point. When a request is made we must move one taxi to serve it immediately without the information about future possible requests. Some important results concerning this problem and its variants have been presented by Xu et al. [13], Xin and Ma [19], and Ma et al. [20–22].

In the traditional online k-taxi problem, it is usually assumed that the taxis are dispatched completely without any other information about future possible requests. In this paper, we consider the problem with a new feature that partial information about prospective possible service requests is provided in advance when deciding which taxi should be dispatched to serve the current request, and investigate the value of foreknowledge in this problem. This feature arises from realistic application background and also can be seen in many other online decision-making problems. In practice, the foreknowledge (i.e., information about future possible requests) may be provided by forecasting or encouraging customers to release their requests in advance (e.g., discounted price available for reservation).

In general, we can use the extra information, available in advance, to make better decisions. As the performance of an online algorithm is measured by competitive ratio in the traditional online k-taxi problem, it is natural to quantify the value of foreknowledge in the k-taxi problem with this new feature in the form of improved competitive ratios (with respect to the ratios of the traditional problem). This idea can also be seen in the online traveling salesman problem and traveling repairman problem with advanced information proposed by Jaillet and Wagner [3].

The rest of this paper is organized as follows. The problem description and formulation are presented in Sect. 2. Improved algorithms for the problem with foreknowledge are proposed in Sect. 3. Then the competitive analysis of the algorithms is given in Sect. 4. The value of foreknowledge as improved competitive ratios is also discussed in this section. Subsequently, Sect. 5 presents some numerical examples to illustrate the proposed algorithms as well as their performances in practice. Finally, conclusions are given in Sect. 6.

2 Problem description and formulation

Let $G(V, E)$ denote a connected and edge weighted graph, where V is the metric space consisting of n ($n \geq 2$) points, and E is the set of all weighted edges. For any points $x, y \in V$, $d(x, y)$ indicates the distance of the shortest path between the points x and y. The weights of edges are symmetric and satisfy triangle inequality, i.e., for any $x, y, z \in V$, $d(x, y) = d(y, x)$ and $d(x, z) + d(z, y) \geq d(x, y)$. For any points $x \neq y \in V$, let $d_{\max} = \max d(x, y)$, $d_{\min} = \min d(x, y)$, and

$$\lambda = \frac{d_{\max}}{d_{\min}}. \tag{1}$$

It is clear that $\lambda \geq 1$.

There are k ($k \geq 1$) taxis that reside and move on the graph G to supply service. A service request $r = (a, b)$ (a, b are different points in the given graph G), implies there is a passenger at point a that must be taken to b by a taxi. A service request sequence R consists of a series of service requests in turn, namely $R = (r_1, r_2, \ldots, r_m)$, where $r_i = (a_i, b_i)$, $1 \leq i \leq m$. It is assumed that $a_i \neq b_i$ in the following discussion, because if $a_i = b_i$, the request r_i does not exist in fact and this case is of no significance to discuss.

2.1 Traditional online k-taxi problem

Let us consider the following two problems:

1) Given a service request sequence $R = (r_1, r_2, \ldots, r_m)$ in advance, how to dispatch the taxis to minimize the total distance of serving all the requests?
2) How can we deploy the taxis to reduce the relevant moving distance when the service requests are received one by one, and each request is to be

served immediately after it is made without any information about future possible requests?

Obviously, problem (1) is offline, whereas (2) is an online problem. The difference between them lies in whether complete information of the service request sequence is known while serving the current request. The latter is also known as the traditional online k-taxi problem. Due to the absence of information about future requests, it is hard to make the decision in an optimal fashion for problem (2).

Following many earlier studies on the classical online service dispatching problems (e.g. [13, 14, 23, 24]), this paper also assumes that when a new service request occurs, all taxis are available. In other words, when a new service request occurs, previous service requests have been completed and all the taxis are idle and ready to serve the new one.

For any request sequence $R = (r_1, r_2, \ldots, r_m)$, let $C_{\mathrm{OPT}}(R)$ denote the optimal (minimum) cost (total moving distance) required to complete the whole sequence with the optimal offline algorithm which knows the whole request sequence before it starts. Then, we can get that

$$C_{\mathrm{OPT}}(R) \geq \sum_{i=1}^{m} d(a_i, b_i), \tag{2}$$

since $\sum_{i=1}^{m} d(a_i, b_i)$ is the total distance travelled by passengers. It is the minimum cost that should be paid for serving the sequence R. In other words, if the sequence R is completed by taxis without the case that moving with on passenger, the cost of serving R is $\sum_{i=1}^{m} d(a_i, b_i)$. However, in order to serve the requests, the taxis may should be dispatched to the start points of these requests first via moving with no passenger, and then the cost of serving R would be more than $\sum_{i=1}^{m} d(a_i, b_i)$.

Let $C_{\mathrm{ON}}(R)$ denote the cost (total moving distance) of completing the whole sequences with an online algorithm which has not any knowledge about the future requests when serving the current request. If there exist constants α ($\alpha \geq 1$) and β satisfying the following inequality for any possible request sequence R,

$$C_{\mathrm{ON}}(R) \leq \alpha \cdot C_{\mathrm{OPT}}(R) + \beta, \tag{3}$$

the on-line algorithm is called an α-competitive algorithm and α is the competitive ratio. Our goal is to design some online algorithms with competitive ratios as small as possible.

2.2 Online k-taxi problem with foreknowledge

Now, let us consider the problem with some foreknowledge about the future possible requests. In other words, the online player is empowered with the ability of limited looking ahead and knows partial information about future requests in advance. More specifically, we consider the following two scenarios.

1) The online player only knows the start point of next request in advance when making the decision for serving the current request. The start point a_1 of the first request is given before the game plays. In the following whole process, when the i-th request r_i occurs, the end point of the i-th request and the start point of the $(i+1)$-th request r_{i+1} are known by the player. In other words, the information of the $(i+1)$-th request is not completely released before the i-th request is served, and only the start point is known in advance. In the following discussion, the online k-taxi problem in this scenario is denoted as P1.
2) The online player knows the next request (both the start point and the destination point) in advance when making the decision to serve the current request. The first request $r_1 = (a_1, b_1)$ is known before the game plays. In the following whole process, when the i-th request r_i occurs, the $(i+1)$-th request r_{i+1} is known by the player in advance (before r_i is served). In other words, the information of the $(i+2)$-th request is completely released after the i-th request and all of its previous requests are served. In this case, it is assumed that the order of serving the adjacent two requests can be exchanged. In the following discussion, the online k-taxi problem in this scenario is denoted as P2.

The problems are to decide which taxi is to be dispatched to serve the request when a new service request occurs on the basis that no more information about future possible requests is known. With the aid of partial information known in advance, it is anticipated that a better decision than that in the traditional online k-taxi problem can be made.

In the following sections, we propose some improved algorithms for this problem with foreknowledge, and quantify the value of foreknowledge in the form of improved competitive ratios with respect to the ratios of the traditional online k-taxi problem.

3 Improved algorithms with foreknowledge

In many classical online service dispatching problems (e.g. [13, 14, 23]), in order to play against the adversary and reduce the costs of serving the requests in worst cases, covering strategy has been extensively applied. In this section, we propose improved covering strategies for the online k-taxi problem with foreknowledge.

To implement the covering strategies, the initial locations of the taxis are assumed such that each point in the graph is occupied by at least one taxi if $k \geq n$, or else (i.e., $k < n$) these k taxis are respectively located at k different points and there is a taxi at a_1 before the first service request comes in (as described in the previous formulation, the player with foreknowledge knows that the first request will occur at point a_1 before the service starts). Otherwise, we can make it happen by finite number of movements, and the moving distance will not exceed a constant $(k-1)d_{\max}$. This constant makes no influence on the discussion of competitive ratio [11]. In other words, we precondition the taxi locations such that these taxis cover distinct points as many as possible and the point a_1 is occupied by at least one taxi.

3.1 Improved covering strategy for problem P1

For the problem P1, in which the online player only knows the start point of next request in advance when making the decision to serve the current request, an online algorithm, called Improved Covering Strategy (denoted as S1), is designed as follows:

Improved Covering Strategy (S1):

For the i-th ($i \geq 1$) service request $r_i = (a_i, b_i)$:

1) If there are taxis at both a_i and b_i, consider the following two cases:

   i) If there are more than one taxis at a_i, choose one taxi arbitrarily and use it to take the passenger from a_i to b_i. The cost of completing this request is $C_{S1}(r_i) = d(a_i, b_i)$. The move is denoted as $a_i \to b_i$.

   ii) If there is only one taxi at a_i, the taxi at a_i takes the passenger from a_i to b_i. Then if a_{i+1} is without a taxi and $d(a_{i+1}, b_i) \leq d(a_i, b_i)$, the taxi moved from a_i to b_i continues to move to a_{i+1}. The cost of completing this request is $C_{S1}(r_i) \leq 2d(a_i, b_i)$, and the moves are denoted as $a_i \to b_i \to a_{i+1}$. Otherwise, the taxi moved from a_i to b_i moves back to a_i. The cost of completing this request is $C_{S1}(r_i) = 2d(a_i, b_i)$, and the moves are denoted as $a_i \to b_i \to a_i$.

2) If there is a taxi at a_i but no taxi at b_i, the taxi at a_i takes the passenger from a_i to b_i. The cost of completing this request is $C_{S1}(r_i) = d(a_i, b_i)$. The move is denoted as $a_i \to b_i$.

3) If there is a taxi at b_i but no taxi at a_i, the taxi at b_i moves to a_i first and then takes the passenger from a_i to b_i. The cost of completing this request is $C_{S1}(r_i) = 2d(a_i, b_i)$. The moves are denoted as $b_i \to a_i \to b_i$.

4) If neither a_i nor b_i has a taxi ($i \geq 2$ holds), the following two cases concerning the relationship between a_i and a_{i-1} need to be considered.

   i) If $a_i = a_{i-1}$, the taxi at b_{i-1} (since the $(i-1)$-th service request is $r_{i-1} = d(a_{i-1}, b_{i-1})$, there must be a taxi at b_{i-1}) moves to a_i first, and then takes the passenger from a_i to b_i. The cost of completing this request is $C_{S1}(r_i) = d(a_{i-1}, b_{i-1}) + d(a_i, b_i)$. The moves are denoted as $b_{i-1} \to a_i \to b_i$.

   ii) If $a_i \neq a_{i-1}$ and $k \geq 2$, schedule the nearest taxi (suppose it locates at c_i, where $c_i \neq a_{i+1}$) to a_i and then use it to take the passenger from a_i to b_i. If $a_i \neq a_{i-1}$ and $k = 1$, schedule the only one taxi (assume that it locates at c_i) to a_i and then use it to take the passenger from a_i to b_i. The cost of completing this request is $C_{S1}(r_i) = d(c_i, a_i) + d(a_i, b_i)$. The moves are denoted as $c_i \to a_i \to b_i$.

The main idea behind the covering strategy is to let these k taxis always cover as many distinct points as possible in the whole game to reduce the total distance of the moves for serving all possible requests. Similar approaches were also proposed to obtain some good results in [13, 20, 22]. In fact, this idea is extensively applied in online algorithms (such as position maintaining strategy and greedy algorithm) for the k-server problem and its variants [14].

With the foreknowledge about the next request, the algorithm S1 considers how to serve the next request while processing the current one [case (1-ii)] and avoids dispatching the taxi that may be used to serve the next request to complete the current request [case (4-ii)]. Benefited by the foreknowledge, we will show that the competitive ratio of the covering strategy can be improved in the following sections.

3.2 Improved covering strategy for problem P2

For the problem P2, in which the online player knows the next request (both the start point and the destination point) in advance, a similar Improved Covering Strategy (denoted as S2) can be designed as follows. For the sake of simplicity, only the moves in each case of this strategy that have similar meaning as those in Strategy S1 are illustrated.

Improved Covering Strategy (S2):

For the i-th ($i \geq 1$) service request $r_i = (a_i, b_i)$:

1) If there are taxis at both a_i and b_i, consider the following two cases:

    i) If there are more than one taxis at a_i, the move is $a_i \rightarrow b_i$.
    ii) If there is only one taxi at a_i, and if a_{i+1} is without a taxi and $d(a_{i+1}, b_i) \leq d(a_i, b_i)$, the moves are $a_i \longrightarrow b_i \longrightarrow a_{i+1}$, or else, the moves are $a_i \rightarrow b_i \rightarrow a_i$.

2) If there is a taxi at a_i but no taxi at b_i, the move is $a_i \rightarrow b_i$.
3) If there is a taxi at b_i but no taxi at a_i, the moves are $b_i \rightarrow a_i \rightarrow b_i$.
4) If neither a_i nor b_i has a taxi ($i \geq 2$ holds), consider the following four cases:

    i) If $a_i = b_{i+1}$ and a_{i+1} has a taxi, move the taxi at a_{i+1} to b_{i+1} to serve the request r_{i+1} first, and then move it from b_{i+1} (it is also a_i) to b_i to serve the request r_i. The moves are denoted as $a_{i+1} \rightarrow b_{i+1}(a_i) \rightarrow b_i$. Consequently, both requests r_i and r_{i+1} are served, and the strategy turns to serve r_{i+2} next.
    ii) If $a_i = b_{i+1}$ and a_{i+1} does not have a taxi, schedule the nearest (away from a_{i+1}) taxi (suppose it locates at c_i) to a_{i+1} first, and then move it from a_{i+1} to b_{i+1} and from b_{i+1} (a_i) to b_i to serve the requests r_{i+1} and r_i successively. The moves are denoted as $c_i \rightarrow a_{i+1} \rightarrow b_{i+1}(a_i) \rightarrow b_i$. After that, both requests r_i and r_{i+1} are served, and the strategy turns to serve r_{i+2}.
    iii) If $a_i = a_{i-1}$, the moves for serving r_i are $b_{i-1} \rightarrow a_i \rightarrow b_i$, and then go on to serve the next request r_{i+1}.
    iv) If $a_i \neq a_{i-1}$, $a_i \neq b_{i+1}$, and $k \geq 3$, schedule the nearest taxi at point c_i, where $c_i \neq a_{i+1}$ and $c_i \neq b_{i+1}$, to a_i to serve r_i. If $a_i \neq a_{i-1}$, $a_i \neq b_{i+1}$, and $1 \leq k \leq 2$, schedule the nearest taxi at anywhere to serve r_i. The moves for serving this request are denoted as $c_i \rightarrow a_i \rightarrow b_i$. Then the strategy turns to serve r_{i+1}.

The difference between strategies S2 and S1 lies in the case (4). Since more foreknowledge is known in advance, strategy S2 considers more sophisticated cases to reduce the relevant moving distance for serving the requests.

4 Value of foreknowledge as improved competitive ratios

In this section, we first prove the competitive ratios of the proposed two covering strategies, and then compare them with those in the traditional online k-taxi problem. The value of foreknowledge is quantified as the improved competitive ratios.

4.1 Competitive ratio of strategy S1

For the competitive ratio of strategy S1, which is proposed for addressing the problem P1, we have the following theorems.

Theorem 1 *For the problem P1 with k taxis, if $k \geq n - 1$, the strategy S1 is an online algorithm with competitive ratio 2, where n ($n \geq 2$) indicates the number of points of graph G.*

Proof As mentioned previously, we preconditioned the taxi locations such that these taxis cover distinct points as many as possible, and the point a_1 has at least one taxi, before the first service request comes. It is easy to verify that case (4) in the strategy S1 would never happen in the whole game if $k \geq n - 1$, because there are enough taxis to cover at least $n - 1$ distinct points in the graph. Particularly, if $k \geq n$, all the points are always occupied by at least one taxi after a request is served by the strategy S1; whereas if $k = n - 1$, there is only one point unoccupied by a taxi. Consequently, the strategy does not make full use of the foreknowledge about the next request, and the problem degenerates to the traditional k-taxi problem.For the cases (1), (2) and (3), we have

$$C_{S1}(r_i) \leq 2d(a_i, b_i). \tag{4}$$

Therefore,

$$\begin{aligned} C_{S1}(R) &= \sum_{i=1}^{m} C_{S1}(r_i) + \beta \\ &\leq \sum_{i=1}^{m} 2d(a_i, b_i) + \beta \\ &\leq 2C_{OPT}(R) + \beta, \end{aligned} \tag{5}$$

where β is a constant denoting the cost of preconditioning. This proves Theorem 1. □

In order to prove the competitive ratio for a special case with $k = n - 2$, we present some lemmas first.

Lemma 1 *For the problem P1 with k ($k \geq 2$) taxis, if $k = n - 2$, at least one of the following inequalities concerning the cost of strategy S1 to serve the requests r_i and r_{i+1} holds,*

$$C_{S1}(r_i) + C_{S1}(r_{i+1}) \le 2d(a_i, b_i) + d(a_{i+1}, b_{i+1}) + d_{\max}, \tag{6}$$

or

$$C_{S1}(r_i) + C_{S1}(r_{i+1}) \le d(a_i, b_i) + 2d(a_{i+1}, b_{i+1}) + d_{\max}. \tag{7}$$

Proof For any request $r_i = (a_i, b_i)$, according to the strategy S1, we have

$$C_{S1}(r_i) \le d(a_i, b_i) + d_{\max}, \tag{8}$$

since for any points $x \neq y \in V$, $d(x, y) \le d_{\max}$ holds.

At the beginning of the game, there must exist a taxi at the point a_1 as described in the preconditioning. According to the cases (1) and (2) of the strategy S1, we can get that

$$C_{S1}(r_1) \le 2d(a_1, b_1). \tag{9}$$

Combining the inequalities (8) and (9), the following inequality holds,

$$C_{S1}(r_1) + C_{S1}(r_2) \le 2d(a_1, b_1) + d(a_2, b_2) + d_{\max}. \tag{10}$$

Therefore, for the case $i = 1$, Lemma 1 holds.

The following proof considers the general case of the i-th ($i \ge 1$) service request. For $r_i = d(a_i, b_i)$:

I) If the strategy S1 serves the request by the moves described in the cases (1), (2) and (3), then $C_{S1}(r_i) \le 2d(a_i, b_i)$ holds.
Combining it with (8), we get

$$C_{S1}(r_i) + C_{S1}(r_{i+1}) \le 2d(a_i, b_i) + d(a_{i+1}, b_{i+1}) + d_{\max}. \tag{11}$$

Lemma 1 holds.

II) If the strategy S1 serves the request by the moves described in the case (4-i), the cost of completing this request is $C_{S1}(r_i) = d(a_{i-1}, b_{i-1}) + d(a_i, b_i)$. By using (8), we get

$$C_{S1}(r_{i-1}) + C_{S1}(r_i) \le 2d(a_{i-1}, b_{i-1}) + d(a_i, b_i) + d_{\max}. \tag{12}$$

Lemma 1 holds.

III) If the strategy S1 serves the request by the moves described in the case (4-ii), it implies that neither a_i nor b_i has a taxi. The moves and status of the related points are illustrated in Fig. 1, in which the circle filled in black indicates that it is occupied by a taxi while the empty circle denotes the point without a taxi. Since there are $n - 2$ taxis in total, after serving the request $r_i = d(a_i, b_i)$, neither a_i nor c_i has a taxi, whereas b_i and all the other points (not shown in Fig. 1) are with a taxi. The cost of completing request r_i satisfies the formula that $C_{S1}(r_i) \le d(a_i, b_i) + d_{\max}$. For the cost of serving next request $r_{i+1} = d(a_{i+1}, b_{i+1})$, we need to consider the following three cases.

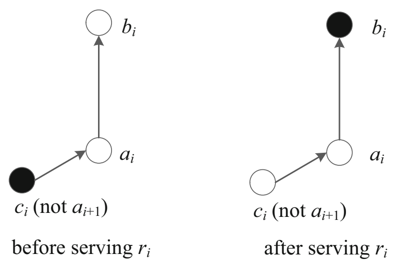

Fig. 1 Illustration of the moves for serving r_i by the strategy S1 in the case (4-ii)

a) $a_{i+1} \neq a_i$ and $a_{i+1} \neq b_i$.
Under this condition there must be a taxi at a_{i+1}, because after serving the last request r_i, neither a_i nor c_i has a taxi, and all the other points are with a taxi, as shown in Fig. 1. Thus, to serve the request r_{i+1}, only the cases (1) or (2) in the strategy S1 could occur and the cost satisfies the formula that $C_{S1}(r_{i+1}) \le 2d(a_{i+1}, b_{i+1})$. Then we have

$$C_{S1}(r_i) + C_{S1}(r_{i+1}) \le d(a_i, b_i) + 2d(a_{i+1}, b_{i+1}) + d_{\max}. \tag{13}$$

Lemma 1 holds.

b) $a_{i+1} = a_i$.

(i) $b_{i+1} = c_i$. As shown in Fig. 1, after serving the last request r_i, neither a_i nor c_i has a taxi. So the strategy S1 moves the taxi at b_i to a_{i+1} to serve the request r_{i+1}. The cost is $C_{S1}(r_{i+1}) = d(a_i, b_i) + d(a_{i+1}, b_{i+1})$. Then Lemma 1 holds.

(ii) $b_{i+1} \neq c_i$. After serving r_i, there must be a taxi at b_{i+1}, since only these two points a_i and c_i are without a taxi. Then we have $C_{S1}(r_{i+1}) = 2d(a_{i+1}, b_{i+1})$. Lemma 1 holds.

c) $a_{i+1} = b_i$.
Under this condition, there must be a taxi at a_{i+1} as shown in Fig. 1. Similarly to the analysis in case (a), it is easy to verify that Lemma 1 holds.

The proof is completed. □

Lemma 2 *For the problem P1 with k (k ≥ 2) taxis, if k = n − 2, the following inequality concerning the cost of strategy S1 to serve the requests r_i and r_{i+1} holds,*

$$C_{S1}(r_i) + C_{S1}(r_{i+1}) \le \frac{3 + \lambda}{2}[d(a_i, b_i) + d(a_{i+1}, b_{i+1})]. \tag{14}$$

Proof For any requests $r_i = (a_i, b_i)$ and $r_{i+1} = d(a_{i+1}, b_{i+1})$, we can prove Lemma 2 in two cases following from Lemma 1.

I) If $C_{S1}(r_i) + C_{S1}(r_{i+1}) \le 2d(a_i, b_i) + d(a_{i+1}, b_{i+1}) + d_{\max}$ holds, we have

$$\begin{aligned}\frac{C_{S1}(r_i) + C_{S1}(r_{i+1})}{d(a_i,b_i) + d(a_{i+1},b_{i+1})} &\le \frac{2d(a_i,b_i) + d(a_{i+1},b_{i+1}) + d_{\max}}{d(a_i,b_i) + d(a_{i+1},b_{i+1})} \\ &= 2 + \frac{d_{\max} - d(a_{i+1},b_{i+1})}{d(a_i,b_i) + d(a_{i+1},b_{i+1})}.\end{aligned} \tag{15}$$

Since $d_{\min} \le d(a_i,b_i), d(a_{i+1},b_{i+1}) \le d_{\max}$, we get that

$$\frac{d_{\max} - d(a_{i+1},b_{i+1})}{d(a_i,b_i) + d(a_{i+1},b_{i+1})} \le \frac{d_{\max} - d_{\min}}{d_{\min} + d_{\min}} = \frac{\lambda - 1}{2}. \tag{16}$$

By taking (16) into (15), (14) is verified.

II) If $C_{S1}(r_i) + C_{S1}(r_{i+1}) \le d(a_i,b_i) + 2d(a_{i+1},b_{i+1}) + d_{\max}$ holds, similarly, we have

$$\begin{aligned}\frac{C_{S1}(r_i) + C_{S1}(r_{i+1})}{d(a_i,b_i) + d(a_{i+1},b_{i+1})} &\le \frac{d(a_i,b_i) + 2d(a_{i+1},b_{i+1}) + d_{\max}}{d(a_i,b_i) + d(a_{i+1},b_{i+1})} \\ &= 2 + \frac{d_{\max} - d(a_i,b_i)}{d(a_i,b_i) + d(a_{i+1},b_{i+1})} \\ &\le 2 + \frac{d_{\max} - d_{\min}}{2d_{\min}} \\ &= \frac{3+\lambda}{2}.\end{aligned} \tag{17}$$

Lemma 2 is proved. □

Following from Lemma 2, we obtain the following theorem for the competitive ratio of strategy S1.

Theorem 2 *For the problem P1 with k (k ≥ 2) taxis, if k = n − 2, the strategy S1 is an online algorithm with competitive ratio (3 + λ)/2, where n (n ≥ 2) indicates the number of points of graph G and* $\lambda = d_{\max}/d_{\min}$.

Proof For any request sequence $R = (r_1, r_2, \ldots, r_m)$, if m is an even number, following from Lemma 2, we can easily get that

$$\begin{aligned}C_{S1}(R) &= \sum_{i=1}^{m}[C_{S1}(r_i)] + \beta \\ &= [C_{S1}(r_1) + C_{S1}(r_2)] + [C_{S1}(r_3) + C_{S1}(r_4)] + \cdots \\ &\quad + [C_{S1}(r_{m-1}) + C_{S1}(r_m)] + \beta \\ &\le \frac{3+\lambda}{2}[d(a_1,b_1) + d(a_2,b_2) + \cdots d(a_m,b_m)] + \beta \\ &= \frac{3+\lambda}{2}\sum_{i=1}^{m} d(a_i,b_i) + \beta \\ &\le \frac{3+\lambda}{2}C_{OPT}(R) + \beta,\end{aligned} \tag{18}$$

where β is a constant denoting the cost of preconditioning.

For the sequence $R = (r_1, r_2, \ldots, r_m)$, if m is an odd number, we have the following similar result.

$$\begin{aligned}C_{S1}(R) &= \sum_{i=1}^{m}[C_{S1}(r_i)] + \beta \\ &= [C_{S1}(r_1) + C_{S1}(r_2)] + \cdots + [C_{S1}(r_{m-2}) + C_{S1}(r_{m-1})] \\ &\quad + C_{S1}(r_m) + \beta \\ &\le \frac{3+\lambda}{2}[d(a_1,b_1) + d(a_2,b_2) + \cdots d(a_{m-1},b_{m-1})] \\ &\quad + C_{S1}(r_m) + \beta \\ &= \frac{3+\lambda}{2}\sum_{i=1}^{m} d(a_i,b_i) - \frac{3+\lambda}{2}d(a_m,b_m) + C_{S1}(r_m) + \beta \\ &\le \frac{3+\lambda}{2}C_{OPT}(R) + \beta',\end{aligned} \tag{19}$$

where β' is a constant such that

$$\begin{aligned}\beta' &= -\frac{3+\lambda}{2}d(a_m,b_m) + C_{S1}(r_m) + \beta \\ &\le -\frac{3+\lambda}{2}d_{\min} + d_{\max} + \beta.\end{aligned} \tag{20}$$

Formula (19) just has a different constant term comparing to (18), which makes no influence to the discussion of competitive ratio while $m \to \infty$. The proof is completed. □

Theorem 3 *For the problem P1 with k taxis, if 1 ≤ k ≤ n − 3, the strategy S1 is an online algorithm with competitive ratio 1+λ, where n (n ≥ 2) indicates the number of points of graph G and* $\lambda = d_{\max}/d_{\min}$.

Proof Since for any request $r_i = (a_i, b_i)$, $C_{S1}(r_i) \le d(a_i,b_i) + d_{\max}$ holds, we have

$$\begin{aligned}C_{S1}(R) &= \sum_{i=1}^{m} C_{S1}(r_i) + \beta \\ &\le \sum_{i=1}^{m}[d(a_i,b_i) + d_{\max}] + \beta \\ &\le (1+\lambda)\sum_{i=1}^{m} d(a_i,b_i) + \beta \\ &\le (1+\lambda)C_{OPT}(R) + \beta.\end{aligned} \tag{21}$$

Theorem 3 is proved. □

In summary, for the problem P1 with k ($k \ge 1$) taxis on a graph with n ($n \ge 2$) points, the competitive ratio of the improved covering strategy S1 is

$$\alpha_{S1} = \begin{cases} 2, & \text{if } k \ge n-1 \\ \frac{3+\lambda}{2}, & \text{if } k \ge 2 \text{ and } k = n-2 \\ 1+\lambda, & \text{if } 1 \le k \le n-3. \end{cases} \tag{22}$$

4.2 Competitive ratio of strategy S2

Similarly, concerning the competitive ratio of strategy S2 for the problem P2, we have the following theorems.

Theorem 4 *For the problem P2 with k taxis, if $k \geq n-1$, the strategy S2 is an online algorithm with competitive ratio 2, where n ($n \geq 2$) indicates the number of points of graph G.*

Proof Following from the proof of Theorem 1, Theorem 4 can be similarly proved. □

Theorem 5 *For the problem P2 with k taxis, if $1 \leq k \leq n-4$, the strategy S2 is an online algorithm with competitive ratio $1+\lambda$, where n ($n \geq 2$) indicates the number of points of graph G and $\lambda = d_{\max}/d_{\min}$.*

Proof Following from the proof of Theorem 3, Theorem 5 can also be similarly proved. □

Theorem 6 *For the problem P2 with k ($k \geq 3$) taxis, if $k = n-2$ or $k = n-3$, the strategy S2 is an online algorithm with competitive ratio $(3+\lambda)/2$, where n ($n \geq 2$) indicates the number of points of graph G and $\lambda = d_{\max}/d_{\min}$.*

Proof Similarly to the proof of Theorem 2, we just need to prove that Lemmas 1 and 2 are also true for the strategy S2 when $k = n-2$ or $k = n-3$, and then Theorem 6 can be easily verified.

Now, we prove that for the problem P2 with k taxis, if $k = n-2$ or $k = n-3$, at least one of the following inequalities concerning the cost of strategy S2 to serve the requests r_i and r_{i+1} holds,

$$C_{S2}(r_i) + C_{S2}(r_{i+1}) \leq 2d(a_i, b_i) + d(a_{i+1}, b_{i+1}) + d_{\max}, \tag{23}$$

or

$$C_{S2}(r_i) + C_{S2}(r_{i+1}) \leq d(a_i, b_i) + 2d(a_{i+1}, b_{i+1}) + d_{\max}. \tag{24}$$

Since the difference between strategies S2 and S1 only lies in the case (4), we just need to verify that the above proposition [namely, at least one of (23) or (24) is true] also holds in this case for $k = n-2$ or $k = n-3$.

What is more, if $k = n-2$, case (4-ii) in the strategy S2 would never happen, since there are only two points that are not covered by a taxi in the whole game. Therefore, in the case (4-ii), we only should consider the situation that $k = n-3$.

For the case (4-i), the cost of serving r_i and r_{i+1} is

$$C_{S2}(r_i) + C_{S2}(r_{i+1}) = d(a_i, b_i) + d(a_{i+1}, b_{i+1}). \tag{25}$$

Both (23) and (24) hold.

For the case (4-ii) and $k = n-3$, the cost of serving r_i and r_{i+1} is

$$C_{S2}(r_i) + C_{S2}(r_{i+1}) = d(c_i, a_{i+1}) + d(a_i, b_i) + d(a_{i+1}, b_{i+1}). \tag{26}$$

(23) and (24) also hold.

Figure 2 shows the moves and status of the related points in the case (4-ii) of strategy S2. The circle filled in black indicates that it is occupied by a taxi, while the empty circle denotes a point without a taxi. Since there are $n-3$ taxis in total, all the other points not shown in Fig. 2 are occupied by a taxi.

For the cases (4-iii) and (4-iv), the above proposition that at least one of (23) or (24) is true can be verified similarly to the proof of Lemma 1.

Consequently, we can get a similar result with Lemma 2 as follows. For the problem P2, if $k = n-2$ or $k = n-3$, the following inequality concerning the cost of strategy S2 to serve the requests r_i and r_{i+1} holds,

$$C_{S2}(r_i) + C_{S2}(r_{i+1}) \leq \frac{3+\lambda}{2}[d(a_i, b_i) + d(a_{i+1}, b_{i+1})]. \tag{27}$$

Following from (27), Theorem 6 holds. □

In summary, for the problem P2 with k ($k \geq 1$) taxis on a graph with n ($n \geq 2$) points, the competitive ratio of the improved covering strategy S2 is

$$\alpha_{S2} = \begin{cases} 2, & \text{if } k \geq n-1 \\ \dfrac{3+\lambda}{2}, & \text{if } k \geq 3 \text{ and } (k = n-2 \text{ or } n-3) \\ 1+\lambda, & \text{if } 1 \leq k \leq n-4. \end{cases} \tag{28}$$

4.3 Improved competitive ratios

The performance of an online algorithm is measured by competitive ratio in the competitive analysis theory. In [13], a competitive algorithm, called position maintaining strategy (PMS), is proposed to handle the traditional online k-taxi problem by Xu et al. It is proved that the competitive ratio of PMS is

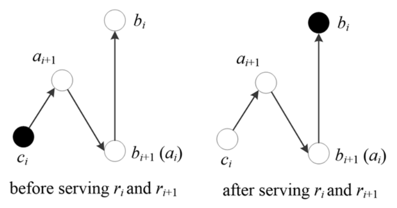

Fig. 2 Illustration of the moves for serving r_i and r_{i+1} by the strategy S2 in the case (4-ii)

$$\alpha_{\mathrm{PMS}} = \begin{cases} 2, & \text{if } k = n \text{ or } n-1 \\ 1+\lambda, & \text{if } 1 \le k \le n-2. \end{cases} \tag{29}$$

Comparing (29) with (22) and (28), it can be seen that with the foreknowledge, the competitive ratios of the strategies S1 and S2 are improved.

To quantify the value of foreknowledge in the form of improved competitive ratios with respect to the ratios of the traditional online k-taxi problem, we can show that the improvements for the case of $k = n-2$ in problem P1 ($k \ge 2$) and the case of $k = n-2$ or $n-3$ in problem P2 ($k \ge 3$) are

$$1+\lambda - \frac{3+\lambda}{2} = \frac{\lambda-1}{2} \ge 0. \tag{30}$$

In other words, the competitive ratios are decreased from $1+\lambda$ to $(3+\lambda)/2$ for some special cases of the online k-taxi problem by introducing the new feature of foreknowledge into it. The improvements, i.e., the value of foreknowledge in the sense of competitive analysis, are $(\lambda-1)/2$, where λ is the parameter determined by the metric space in which the problem is discussed.

Furthermore, it also can be seen that, in all cases, the improved covering strategies would never perform wore, in the sense of competitive analysis, than the classical PMS for the traditional online k-taxi problem.

5 Performances in practice

In the above section, the performances of the proposed improved covering strategies are analyzed in a theoretical way. In this section, we give some numerical examples to illustrate the proposed online algorithms as well as their performances in practice.

Now, consider the graph consisting of 12 points and 17 weighted edges as shown in Fig. 3, which can be seen as a realistic representation of a little town with 6 blocks. All the edge weights are randomly generated following a normal distribution with mean value of 100 and standard deviation of 10.

For this graph, it is easy to get that $d_{\min} = d(G, K) = 84.10$ and $d_{\max} = d(I, D) = 465.69$, and thus $\lambda = 5.54$. Then the competitive ratios of the improved covering strategies and the PMS are compared in Table 1. It is also illustrated that the improved covering strategies S1 and S2 would never perform wore than the classical PMS based on theoretical analysis.

Then, these online algorithms as well as their practical performances are illustrated on the graph shown in Fig. 3, on the basis of some randomly generated service requests. In order to compare the performance of different algorithms in an appropriate way, the precondition cost is not taken into account, and it is assumed that the online

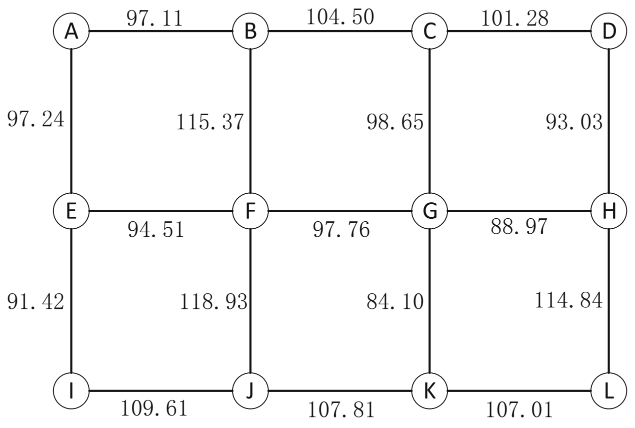

Fig. 3 A graph for numerical examples

Table 1 Competitive ratios of the online algorithms

Online algorithms	Competitive ratios			
	$k \ge 11$	$k = 10$	$k = 9$	$1 \le k \le 8$
PMS	2	6.54	6.54	6.54
S1	2	4.27	6.54	6.54
S2	2	4.27	4.27	6.54

algorithms are launched with the same initial state. In other words, the taxi locations are preconditioned to be same.

In the following, we take a service request sequence $R_1 = ((\mathrm{K, H}), (\mathrm{E, G}), (\mathrm{J, B}), (\mathrm{K, J}), (\mathrm{C, D}), (\mathrm{A, F}))$ for instance, and assume that there are 10 taxis moving on the graph to supply service.

For the optimal offline algorithms, since it knows the sequence before making decision and the precondition cost is not taken into account, it is easy to see that these six requests can be served respectively by one particular taxi being dispatched to the start point of the request in advance. Therefore, the optimal offline cost is $d(\mathrm{K,H}) + d(\mathrm{E,G}) + d(\mathrm{J,B}) + d(\mathrm{K,J}) + d(\mathrm{C,D}) + d(\mathrm{A,F}) = 1000.48$.

For the online algorithms, the initial locations of the taxis can be randomly selected such that these 10 taxis are respectively located at 10 different points and there is a taxi at the start point of the first request. For instance, supposing that the points B and J are not occupied by a taxi, the moves of the three online algorithms to complete these service requests are listed in Table 2. The costs of the PMS and strategies S1 and S2 are 1850.76, 1768.46, and 1658.85, respectively, and their ratios to the optimal offline cost are 1.85, 1.77 and 1.66, respectively.

In this case, the reduction of the costs of S1 and S2, with regard to PMS which does not have any foreknowledge, are respectively 82.3 and 191.91. Meanwhile, as for the ratios of the online solutions to the optimal offline solution, the improvements of S1 and S2 are 0.08 and 0.19, respectively.

Table 2 Moves and performances for serving the sequence R_1

Online algorithms	Moves	Online costs	Ratios
PMS	K → H, H → K; E → G, G → E; K → J → B; G → K → J; C → D, D → C; A → F, F → A	1850.76	1.85
S1	K → H, H → K; E → G, G → E; I → J → B; K → J; C → D, D → C; A → F, F → A	1768.46	1.77
S2	K → H, H → K; E → G, G → E; K → J → B; C → D, D → C; A → F, F → A	1658.85	1.66

* In the initial state, points B and J are without a taxi. Ratio = online cost/optimal offline cost

Table 3 Performances for serving different request sequences

Service request sequence	Optimal offline costs	Online solutions (the ratio to optimal offline cost)		
		PMS	S1	S2
(K, H), (E, G), (J, B), (K, J), (C, D), (A, F)	1000.48	1.85	1.77	1.66
(C, A), (G, L), (D, J), (E, I), (K, F), (G, D)	1221.91	1.55	1.55	1.55
(E, F), (G, J), (K, C), (E, B), (G, A), (H, D)	1046.06	1.63	1.63	1.63
(G, C), (H, L), (E, J), (H, C), (B, E), (K, B)	1083.74	1.46	1.46	1.37

* In the initial state, points B and J are without a taxi

Similarly, more experiments can be made based on some randomly generated service requests. Table 3 illustrates the performances of these online algorithms for serving different request sequences on the graph shown in Fig. 3 with 10 taxis.

From Table 1, we know the competitive ratios of the online algorithms PMS, S1 and S2 for the online k-taxi problem with $k = 10$ are 6.54, 4.27 and 4.27, respectively. Comparing this conclusion with the results shown in Table 3, it can be seen that the practical performances of these online algorithms are much better than the theoretical competitive ratios. In other words, the ratios of the practical online solutions to the optimal offline solutions are much less than the competitive ratios derived by theoretical analysis.

It is easy to understand that these results are consistent with the conclusions previously given in the theoretical analysis, since the approach of competitive analysis is based on worst-case analysis, and consequently, the competitive ratio is actually a worst-case measure, which provides very robust statement about the performance of an online solution.

Moreover, it can also be seen that the proposed improved covering strategies perform better than the traditional PMS in some special cases, not only with regard to theoretical analysis, but also in practice.

6 Conclusions

In this paper, the online k-taxi problem was revisited by introducing a new feature into it, which empowers the online player to have foreknowledge. Two different scenarios concerning how partial information of future possible service requests is released in advanced were discussed, and improved covering strategies with lower competitive ratios, with respect to the traditional online k-taxi problem, were proposed to address them. The competitive ratios are improved due to the utilizing of foreknowledge. Therefore, the improvements are regarded as the value of foreknowledge in this online k-taxi problem within the framework of competitive analysis.

However, concerning the topic that how to effectively dispatch taxis in an online fashion as well as the impact of foreknowledge on the online solutions, although some preliminary results have been presented in this paper, there still are some important issues to be further studied, which will be considered in our future work. For example, firstly, how to design better algorithms for the problem in more general cases? Secondly, it is clear that the more foreknowledge gained, the better decisions may be made. Then how to measure the value of foreknowledge compared to the amount of information known in advance? For this purpose, the notion of entropy [25, 26] may be introduced into this problem. Furthermore, it would also be interesting to consider how to gain more foreknowledge to improve the decisions in the real applications (e.g. learning from big data of the customers' behaviors [27]).

Acknowledgments This work was supported in part by grants from the National Natural Science Foundation of China (No. 71202033), the Ministry of Education Funded Project for Humanities and Social Sciences Research (No. 14YJC630124), the Shanghai Philosophy and Social Science Planning Projects (No. 2013EGL002), and the Fundamental Research Funds for the Central Universities.

References

1. El-Yaniv R, Fiat A, Karp RM, Turpin G (1992) Competitive analysis of financial games. In: Proceedings of 33rd annual symposium on foundations of computer science, pp 327–333
2. El-Yaniv R, Fiat A, Karp RM, Turpin G (2001) Optimal search and one-way trading online algorithms. Algorithmica 30:101–139
3. Jaillet P, Wagner MR (2006) Online routing problems: value of advanced information as improved competitive ratios. Transp Sci 40(2):200–210
4. Jaillet P, Wagner MR (2008) Generalized online routing: new competitive ratios, resource augmentation, and asymptotic analyses. Oper Res 56(3):745–757
5. Ma WM, Wang K (2010) Optimal online risk-tolerant ordering policy for the newsvendor with forecast. J Converg Inf Technol 5(4):54–65
6. Ma WM, Wang K (2010) Robust solutions to the newsvendor problem under the perspective of competitive analysis. In: Proceedings of the 4th international conference on new trends in information science and service science (NISS 2010), pp 424–429
7. Wang XZ, Dong LC, Yan JH (2012) Maximum ambiguity based sample selection in fuzzy decision tree induction. IEEE Trans Knowl Data Eng 24(8):1491–1505
8. Wang XZ, Aamir R, Fu AM (2015) Fuzziness based sample categorization for classifier performance improvement. J Intell Fuzzy Syst 29:1185–1196
9. Wang XZ, Xing HJ, Li Y, Hua Q, Dong CR, Pedrycz W (2015) A study on relationship between generalization abilities and fuzziness of base classifiers in ensemble learning. IEEE Trans Fuzzy Syst 23(5):1638–1654
10. Sleator DD, Tarjan RE (1985) Amortized efficiency of list update and paging rules. Commun ACM 28:202–208
11. Borodin A, El-Yaniv R (1998) Online computation and competitive analysis. Cambridge University Press, Cambridge
12. Albers S (2003) Online algorithms: a survey. Math Program 97(1–2):3–26
13. Xu YF, Wang KL, Zhu B (1999) On the k-taxi problem. Information 2(4):429–434
14. Manasse MS, McGeoch LA, Sleator DD (1990) Competitive algorithms for server problems. J Algorithms 11:208–230
15. Koutsoupias E, Papadimitriou C (1995) On the k-server conjecture. J ACM 42(5):971–983
16. Alon N, Karp RM, Peleg D, West D (1995) A graph-theoretic game and its application to the k-server problem. SIAM J Comput 24(1):78–100
17. Chrobak M, Sgall J (2004) The weighted 2-server problem. Theor Comput Sci 324:289–312
18. Rudec T, Baumgartner A, Manger R (2013) A fast work function algorithm for solving the k-server problem. CEJOR 21(1):187–205
19. Xin CL, Ma WM (2004) Scheduling for on-line taxi problem on a real line and competitive algorithms. In: Proceedings of the third international conference on machine learning and cybernetics (ICMLC 2004), pp 3078–3084
20. Ma WM, Wang K (2006) On-line taxi problem on the benefit-cost graphs. In: Proceeding of 2006 international conference on machine learning and cybernetics (ICMLC 2006), pp 900–905
21. Ma W, Wang K (2007) On the on-line weighted k-taxi problem. In: Proceedings of the first international symposium on combinatorics, algorithms, probabilistic and experimental methodologies (ESCAPE 2007), (lecture notes in computer science, vol 4614), pp 152–162
22. Ma W, Gao T, Wang K (2007) On the on-line k-taxi problem with limited look ahead. In: Proceedings of the first international conference on combinatorial optimization and applications (COCOA 2007), (lecture notes in computer science, vol 4616), pp 72–80
23. Ma W, Xu Y, Wang K (2001) On-line k-truck problem and its competitive algorithms. J Glob Optim 21(1):15–25
24. Xu Y, Li H, He C, Luo L (2015) The online k-server problem with max-distance objective. J Comb Optim 29(4):836–846
25. Wang XZ, Dong CR (2009) Improving generalization of fuzzy if-then rules by maximizing fuzzy entropy. IEEE Trans Fuzzy Syst 17(3):556–567
26. He YL, Liu JNK, Wang XZ, Hu YX (2012) Optimal bandwidth selection for resubstitution entropy estimation. Appl Math Comput 219(8):3425–3460
27. Wang XZ (2015) Learning from big data with uncertainty—editorial. J Intell Fuzzy Syst 28(5):2329–2330