European Journal of Operational Research 229 (2013) 59–66

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Competitive strategies for an online generalized assignment problem with a service consecution constraint

Feifeng Zheng [a], Yongxi Cheng [b,*], Yinfeng Xu [b,c], Ming Liu [d]

[a] Glorious Sun School of Business and Management, Donghua University, Shanghai 200051, China
[b] School of Management, Xi'an Jiaotong University, Xi'an 710049, China
[c] State Key Lab for Manufacturing Systems Engineering, Xi'an 710049, China
[d] School of Economics & Management, Tongji University, Shanghai 200092, China

ARTICLE INFO

Article history:
Received 24 June 2012
Accepted 2 February 2013
Available online 13 February 2013

Keywords:
Assignment
Online strategy
Service consecution constraint
Competitive ratio
Lower bound

ABSTRACT

This work studies a variant of the *online generalized assignment problem*, where there are $m \geqslant 2$ heterogeneous servers to process n requests which arrive one by one over time. Each request must either be assigned to one of the servers or be rejected upon its arrival, before knowing any information of future requests. There is a corresponding weight (or revenue) for assigning each request to a server, and the objective is to maximize the total weights obtained from all the requests. We study the above problem with a *service consecution constraint*, such that at any time each server is only allowed to process up to d consecutive requests.

We investigate both deterministic and randomized online strategies for this problem. When the ratio ρ between the largest and smallest possible weights obtained from assigning a request to a server is known in advance, we present an optimal deterministic online strategy with competitive ratio $\rho^{\frac{1}{d}}$. For randomized strategies, we first prove a lower bound on the competitive ratio, then we present a randomized strategy with competitive ratio less than 2, which does not need to know the value of ρ or d. Computational tests show that our proposed strategies have very good practical performance.

1. Introduction

The *assignment problem* (AP) is a well-known optimization problem due to its extensive applications (Pentico, 2007). Roughly speaking, there are n requests (or tasks) to be assigned to m servers (or agents). For each request, there is a subset of servers that are available to process it, and the request can only be assigned to one of these available servers (in certain scenario a request is allowed to be rejected, that is not assigned to any server). Each assignment pair formed by a request and a server processing the request has a specific weight. Depending on different applications, the weights of assignments represent either revenue or cost, and the objective is to maximize or minimize the assignment weight, that is the total weight obtained from all the requests.

In the classical assignment problem, which aims to optimize the total weight, the total number of servers m is equal to the total number of requests n, and each request shall be assigned to some server and each server processes only one request. Since Kuhn (1955) proposed the famous Hungarian method for the classical assignment problem, there have been many variations of the problem proposed in the literature, such as the *bottleneck assignment problem* to minimize the maximum weight obtained from each request (Ravindran and Ramaswami, 1977; Aneja and Punnen, 1999) and the *balanced assignment problem* to minimize the difference between the maximum and minimum weight obtained from each request (Martello et al., 1984). The reader is referred to Pentico (2007) for a survey on more variations of the classical assignment problem.

The most general version of the assignment problem that allows each server to process multiple requests, is the *generalized assignment problem* (GAP). The generalized assignment problem has wide applications including routing (Fisher and Jaikumar, 1981), facility location (Ross and Soland, 1977), loading for flexible manufacturing systems (Mazzola et al., 1989), allocating cross-trained workers to multiple departments (Campbell and Diaby, 2002), etc. Numerous variations of the generalized assignment problem have been studied by, among others, Martello and Toth (1995), Arora and Puri (1998), Chang and Ho (1998), Moccia et al. (2009), etc. The reader is referred to Cattrysse and Van Wassenhove (1992), and Oncan (2007) for surveys on the applications of and algorithms for the generalized assignment problem.

1.1. The online assignment problem and competitive analysis

In the last decades, the *online* version of the classical assignment problem has caught interest due to real applications. In the

* Corresponding author. Tel.: +86 29 82668382.
E-mail addresses: ffzheng@dhu.edu.cn (F. Zheng), chengyx@mail.xjtu.edu.cn (Y. Cheng), yfxu@mail.xjtu.edu.cn (Y. Xu), minyivg@gmail.com (M. Liu).

0377-2217/$ - see front matter
http://dx.doi.org/10.1016/j.ejor.2013.02.004

online assignment problem where the total number of requests n is equal to the total number of servers m, requests arrive one by one over time and each request must be assigned to one of the available servers upon its arrival, that is the decision of assigning the currently arrived request to which server must be made before knowing any information of future requests. At the end each server processes exactly one request. A strategy for solving the above online assignment problem is called an *online assignment strategy*. This online version of the classical assignment problem is also known as the *online bipartite matching problem*, where the request set and the server set are viewed as the two disjoint subsets of vertices of a bipartite graph G, and an assignment pair formed by a request and a server processing the request is viewed as an edge in G.

The performance of an online assignment strategy is measured by the *competitive ratio* (Borodin and El-Yaniv, 1998), which is a widely used measure for the performance of online algorithms. Consider an online assignment problem to maximize the total weight as in this paper (the problem to minimize the total weight can be similarly discussed). For any input request sequence σ, let $|\mathcal{A}(\sigma)|$ and $|O(\sigma)|$ be the total assignment weights obtained by a deterministic online strategy $\mathcal{A}$ and by an optimal offline strategy OPT, respectively. Then, define

$$\gamma = \sup_{\sigma}\frac{|O(\sigma)|}{|\mathcal{A}(\sigma)|}.$$

Clearly, for the maximization problem we have $\gamma \geqslant 1$. If γ is finite, then strategy $\mathcal{A}$ is said to be γ-competitive, and γ is called the competitive ratio of $\mathcal{A}$. By this measurement, an online strategy for a maximization problem with smaller competitive ratio has better performance.

In this paper, for a randomized online strategy $\mathcal{B}$, the competitive ratio of $\mathcal{B}$ is measured with respect to an oblivious adversary (Raghavan and Snir, 1994; Ben-David et al., 1994), which is standard in the analysis of randomized online algorithms. Different from adaptive adversaries, an oblivious adversary must generate a complete request sequence in advance, without knowing the outcome of the random coin tosses made by $\mathcal{B}$ (or the specific actions taken by $\mathcal{B}$ as a result of the coin tosses) on the requests. However, the adversary does know the complete description of the online strategy $\mathcal{B}$, and knows the probability distribution of actions taken by $\mathcal{B}$ for a given input request sequence. For an input request sequence σ, the total assignment weight $|\mathcal{B}(\sigma)|$ obtained by a randomized online strategy $\mathcal{B}$ from σ is a random variable. The competitive ratio of $\mathcal{B}$ is then defined as the ratio between the total weight $|O(\sigma)|$ obtained by an optimal (deterministic) offline strategy OPT, and the expected total weight $E(|\mathcal{B}(\sigma)|)$ obtained by $\mathcal{B}$, on an input request sequence σ in the worst case. That is, define

$$\gamma_r = \sup_{\sigma}\frac{|O(\sigma)|}{E(|\mathcal{B}(\sigma)|)}.$$

If γ_r is finite then the randomized strategy $\mathcal{B}$ is said to be γ_r-competitive, and γ_r is called the competitive ratio of $\mathcal{B}$.

Karp et al. (1990) considered an online unweighted bipartite matching problem, where for each request there is a subset of available servers and the objective is to maximize the total number of satisfied requests. They showed that a simple greedy strategy that, for each request arbitrarily selects an available server for it, if any, is optimally 2-competitive. Moreover, they presented an $\left(\frac{e}{e-1}+o(1)\right)$-competitive randomized strategy against an oblivious adversary, where e is the base of the natural logarithm. They also proved that the competitive ratio $\frac{e}{e-1}+o(1)$ is best possible for any randomized online strategy against an oblivious adversary, up to lower order terms.

For the online weighted bipartite matching problem in non-metric space, neither the maximization nor the minimization problem has deterministic strategies with bounded competitive ratio (Kalyanasundaram and Pruhs, 1993). For the online maximum weighted bipartite matching problem in metric space, Kalyanasundaram and Pruhs (1993) proved that a simple greedy strategy that always selects the available server with the largest weight to process the currently arrived request, reaches the optimal competitive ratio of 3. For the online minimum weighted bipartite matching problem in metric space, both Kalyanasundaram and Pruhs (1993), and Khuller et al. (1994) gave $(2n-1)$-competitive deterministic algorithms, where n is the number of requests (and is also the number of servers), and showed that no better deterministic algorithm is possible even for the star graph. Meyerson et al. (2006) gave an $O(\log^3 n)$-competitive randomized algorithm for the online minimum weighted bipartite matching problem in metric space, which is the first poly-logarithmic competitive online algorithm for this problem. Improved randomized algorithms are proposed later by Csaba and Pluhar (2008) with competitive ratio $O(\log^3 n/\log\log n)$, and by Bansal et al. (2007) with competitive ratio $O(\log^2 n)$. The competitive ratios of all the above mentioned randomized online algorithms are measured under the oblivious adversary model.

1.2. Our contribution

In this paper we investigate a variant of the online generalized assignment problem with a *service consecution constraint*, which is specified by an integer parameter $d \geqslant 1$, such that at any time each server is only allowed to process at most $d \geqslant 1$ consecutive requests. We investigate both deterministic and randomized online strategies for this problem.

The *online generalized assignment problem with service consecution constraint* studied in this paper is formally described as follows. We have $m \geqslant 2$ heterogeneous servers $s_1, s_2, \ldots, s_m$, and n requests that arrive over time one by one in the ordering $r_1, r_2, \ldots, r_n$, where m is known while n is unknown in advance. For $1 \leqslant i \leqslant n$, each request r_i is associated with an m-dimensional positive weight vector $W_i = (w_{i,1}, \ldots, w_{i,m})$, where $w_{i,j} > 0$ is the weight obtained if request r_i is assigned to server s_j, for $1 \leqslant j \leqslant m$. If r_i is rejected, that is not assigned to any server, then no weight is obtained from r_i. The decision of rejecting r_i or assigning r_i to which server must be made upon the arrival of r_i, without knowing any information of future requests. The assignment is required to satisfy the service consecution constraint, that is at any time it is only allowed to assign up to d consecutive requests to any server, where $d \geqslant 1$ is given. The objective is to maximize the total weight obtained from all the requests.

We assume that $w_{i,j} \in [M_1, M_2]$ $(0 < M_1 \leqslant M_2)$ for $1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m$. For convenience, we normalize the weight interval $[M_1, M_2]$ to $[1, \rho]$ where $\rho = M_2/M_1 \geqslant 1$. We use $OGAP|d \geqslant 1$ to denote the above online generalized assignment problem with service consecution constraint specified by parameter $d \geqslant 1$. For the case where d is fixed to be some constant d_0, we denote the problem by $OGAP|d = d_0$.

Problems in operations research with various consecution constraints have been studied in the literature. In parallel machine scheduling where the activity of machine maintenance is a necessary requirement, to ensure that each machine is available during job processing, an upper limit on the maximum consecutive working time between two adjacent maintenance activities for any machine is required (Xu et al., 2008; Sun and Li, 2010). Another example is the traveling tournament problem (TTP) motivated by scheduling Major League Baseball (MLB) in North America (Easton et al., 2001), where each team plays games in a home/away pattern, and the problem is to minimize the total travel distance of the teams, under the constraint that the maximum number of consecutive home games as well as consecutive away games for each team is limited. Because of its fast growing difficulty, the TTP problem has attracted numerous researchers. Some recent results and a comprehensive survey on the problem can be found in Irnich (2010), Gschwind and Irnich (2011), and Rasmussen and Trick (2008).

The online generalized assignment problem with service consecution constraint studied in this paper is motivated from practical applications. For example, in military one well-known ground weapon is the so-called multiple launch rocket system (MLRS), which consists of a couple of rocket launchers that coordinate with each other during a military activity. Each launcher is able to consecutively launch a limited number of shells within a minute, and then it takes several minutes or even longer for shell replenishment. Hence, each launcher behaves as a server with a limited ability of consecutive shooting. The problem is also motivated from some service industries, in which each manager or worker can only handle a small number of tasks within a certain time period. For example, in a law office, lawyers have different specialities and abilities in settling litigation cases. Generally speaking, a lawyer handles a small number of litigation cases during one time period, no matter how excellent he or she is. In this scenario, each lawyer is regarded as a server that can receive a limited number of consecutive cases.

The main results in this paper are the following. For deterministic online strategies solving $OGAP|d \geqslant 1$, we first prove a lower bound of $\rho^{\frac{1}{d}}$ on the competitive ratio for any deterministic strategy. Then, we give an optimal $\rho^{\frac{1}{d}}$-competitive deterministic online strategy, with the value of ρ known in advance. For randomized online strategies solving $OGAP|d \geqslant 1$, we first prove a lower bound of $\left(\frac{d+1}{d} - \frac{d+1}{d^2}\rho^{-\frac{1}{d+1}}\right)$ on the competitive ratio for any randomized strategy, then we present a $(2 - \frac{2}{1+\rho})$-competitive randomized online strategy which does not need to know the value of ρ or d. The competitive ratios of randomized strategies are measured under the oblivious adversary model described above. Computational tests show that our proposed strategies have very good practical performance.

The rest of our paper is organized as follows. In Sections 2 and 3, we present our results on deterministic and randomized online strategies for problem $OGAP|d \geqslant 1$, respectively. Computational tests are performed in Section 4 to evaluate the practical performance of our proposed strategies. We conclude our paper in Section 5. Due to space limit, proofs of some theorems and the detailed description of an optimal offline algorithm OPT for problem $OGAP|d = d_0$ are given as Supplementary material.

2. Deterministic online strategies for problem $OGAP|d \geqslant 1$

For problem $OGAP|d \geqslant 1$, we first give a lower bound on the competitive ratio for any deterministic online strategy when ρ is bounded.

Theorem 1. *For problem $OGAP|d \geqslant 1$, no deterministic online strategy is better than $\rho^{\frac{1}{d}}$-competitive.*

Proof. See Section 1 in Supplementary material. □

Next, we present an optimal deterministic online strategy SG (Semi-Greedy), whose competitive ratio matches the lower bound in Theorem 1.

When $d = 1$, the lower bound on the competitive ratio in Theorem 1 is ρ for problem $OGAP|d = 1$. The simple greedy online strategy, which always assigns each released request to an available server with the largest weight (we will refer to this simple greedy online strategy as strategy G), has competitive ratio ρ and so is optimal. However, when $d > 1$ the worst case ratio of strategy G is generally much larger than the lower bound $\rho^{\frac{1}{d}}$ given in Theorem 1, as demonstrated by the following example.

Consider an input sequence σ' of $d + 1$ requests $r_1, r_2, \ldots, r_{d+1}$, such that $w_{i,1} = 1 + \epsilon$ for $1 \leqslant i \leqslant d$ (where $\epsilon > 0$ is a very small value), $w_{d+1,1} = \rho$, and $w_{i,j} = 1$ for all the remaining (i,j)'s where $1 \leqslant i \leqslant d + 1$ and $2 \leqslant j \leqslant m$. On input sequence σ', strategy G will greedily assign the first d requests $r_1, r_2, \ldots, r_d$ to server s_1, and assign r_{d+1} to a server s_h with $h \neq 1$, thus obtain a total weight $|G(\sigma')| = d(1 + \epsilon) + 1$. On the other hand, the optimal offline strategy OPT can choose to assign r_1 to server s_2, and assign all the remaining d requests $r_2, r_3, \ldots, r_{d+1}$ to server s_1, to obtain a total weight $|O(\sigma')| = 1 + (d-1)(1+\epsilon) + \rho$. Then, the resulting ratio $\frac{|O(\sigma')|}{|G(\sigma')|} = \frac{1+(d-1)(1+\epsilon)+\rho}{d(1+\epsilon)+1}$ can be arbitrarily close to $\frac{d+\rho}{d+1}$ as ϵ approaches zero, which is in general much larger than $\rho^{\frac{1}{d}}$ when $d \geqslant 2$ and ρ/d is large.

We present a deterministic online strategy SG (Semi-Greedy) for the general problem $OGAP|d \geqslant 1$, and show that SG is optimal with competitive ratio $\rho^{\frac{1}{d}}$. For each $i = 1, 2, \ldots$, let k_i be the index such that strategy SG assigns r_i to server s_{k_i}, and let a_i be the index such that server s_{a_i} has the largest assignment weight for r_i among all the m servers, that is $w_{i,a_i} = \max_{1 \leqslant j \leqslant m}\{w_{i,j}\}$. If there are multiple servers with the same largest assignment weight for r_i, we choose a_i to be the smallest such index. The detailed description of strategy SG on an input request sequence $\sigma = \{r_1, r_2, \ldots, r_n\}$ is as follows.

Strategy SG SG assigns the first request r_1 to the server with index a_1, and sets $k_1 = a_1$. For requests r_i $(i = 2, 3, \ldots, n)$, SG works in the following way.

- If $a_i \neq k_{i-1}$, then SG assigns r_i to the server with index a_i, and sets $k_i = a_i$.
- If $a_i = k_{i-1}$, assume that SG has in total assigned $\ell(1 \leqslant \ell \leqslant d)$ preceding consecutive requests $r_{i-\ell}, \ldots, r_{i-1}$ to the server with index k_{i-1} (i.e., $i - \ell = 1$, or $i - \ell > 1$ and SG assigned $r_{i-\ell-1}$ to a server with index different from k_{i-1}). There are the following two possible cases.
    1. One of the following two conditions holds: (1) $\ell = d$ or (2) $\ell < d$ and $w_{i,a_i} < \rho^{\frac{\ell}{d}}$. In this case, SG chooses k_i to be the index from the index set $\{1, 2, \ldots, m\} - \{k_{i-1}\}$, such that s_{k_i} is with the largest assignment weight for r_i among all the m servers except $s_{k_{i-1}}$ (i.e. $k_i \neq k_{i-1}$ and $w_{i,k_i} = \max_{1 \leqslant j \leqslant m,\ j \neq k_{i-1}}\{w_{i,j}\}$), and SG assigns r_i to server s_{k_i}.
    2. $\ell < d$ and $w_{i,a_i} \geqslant \rho^{\frac{\ell}{d}}$. In this case, SG assigns r_i to the server with index a_i (which is server $s_{k_{i-1}}$ since $a_i = k_{i-1}$), and sets $k_i = a_i$ $(=k_{i-1})$.

Theorem 2. *Strategy SG is $\rho^{\frac{1}{d}}$-competitive for problem $OGAP|d \geqslant 1$.*

Proof. See Section 2 in Supplementary material. □

Theorems 1 and 2 imply that SG is an optimal deterministic online strategy for problem $OGAP|d \geqslant 1$. By Theorem 1, for problem $OGAP|d = k$ with any constant $k \geqslant 1$, the lower bound on the competitive ratio of deterministic strategies can be arbitrarily large as ρ goes to infinity.

In the next section, we study online strategies for problem $OGAP|d \geqslant 1$ with the help of randomization. In particular, we present a randomized online strategy for problem $OGAP|d \geqslant 1$ with competitive ratio less than 2, for any $d \geqslant 1$ and $\rho \geqslant 1$.

3. Randomized online strategies for problem $OGAP|d \geqslant 1$

We first give a lower bound on the competitive ratio for any randomized online strategy for the general problem $OGAP|d \geqslant 1$. Here the competitive ratio of a randomized online strategy $\mathcal{B}$ is measured with respect to an oblivious adversary (Raghavan and Snir, 1994; Ben-David et al., 1994), as mentioned in the introduction.

Theorem 3. *For problem OGAP|d ⩾ 1, no randomized online strategies have competitive ratio less than* $\frac{d+1}{d} - \frac{d+1}{d^2}\rho^{-\frac{1}{d+1}}$.

Proof. See Section 3 in Supplementary material. □

Next, we present a randomized online strategy BD (Bi-deterministic strategy) for problem $OGAP|d \geqslant 1$ with competitive ratio less than 2, for any $d \geqslant 1$ and $\rho \geqslant 1$. The main idea is that Strategy BD consists of two deterministic strategies, denoted by D_1 and D_2 respectively, and BD chooses each of them with probability 1/2. Each of D_1 and D_2 assigns any two consecutive requests to two different servers. Thus, they satisfy the service consecution constraint for any $d \geqslant 1$. For any input request sequence, the expected total weight obtained by BD is the average of the total weights obtained by D_1 and D_2.

For each $1 \leqslant i \leqslant n$, let a_i be an index such that w_{i,a_i} is the maximum among all the $w_{i,j}$'s for $1 \leqslant j \leqslant m$, and let b_i be an index such that w_{i,b_i} is the maximum among all the $w_{i,j}$'s for $1 \leqslant j \leqslant m$ and $j \neq a_i$, that is $w_{i,a_i} = \max_{1 \leqslant j \leqslant m}\{w_{i,j}\}$, and $w_{i,b_i} = \max_{1 \leqslant j \leqslant m, j \neq a_i}\{w_{i,j}\}$, where ties are broken by selecting an arbitrary index with the maximum weight. Thus, servers s_{a_i} and s_{b_i} are with the largest and the second largest assignment weight for r_i, respectively. Clearly, from the above definition we have $a_i \neq b_i$, for $1 \leqslant i \leqslant n$. The online strategy BD is formally described as follows.

Strategy BD BD consists of two deterministic online strategies D_1 and D_2, and BD uniformly chooses one of them at the beginning of any execution.

For any request r_i ($1 \leqslant i \leqslant n$), the two servers selected by D_1 and D_2 for r_i are s_{a_i} and s_{b_i} (with a_i and b_i defined as in the above), with the following two possibilities: (1) D_1 assigns r_i to s_{a_i} and D_2 assigns r_i to s_{b_i} and (2) D_1 assigns r_i to s_{b_i} and D_2 assigns r_i to s_{a_i}. The detailed descriptions of D_1 and D_2 are as follows.

- For the first request r_1, D_1 assigns r_1 to s_{a_1}, and D_2 assigns r_1 to s_{b_1}.
- For requests r_i ($2 \leqslant i \leqslant n$), assume that D_1 has assigned r_{i-1} to $s_{a_{i-1}}$ and D_2 has assigned r_{i-1} to $s_{b_{i-1}}$ (for the other case where D_1 has assigned r_{i-1} to $s_{b_{i-1}}$ and D_2 has assigned r_{i-1} to $s_{a_{i-1}}$, it can be similarly discussed).
  1. If $a_i \neq a_{i-1}$ and $b_i \neq b_{i-1}$, then D_1 assigns r_i to s_{a_i}, and D_2 assigns r_i to s_{b_i}.
  2. If $a_i = a_{i-1}$, then $a_i \neq b_{i-1}$ and $b_i \neq a_{i-1}$ (since by definition $a_{i-1} \neq b_{i-1}$ and $a_i \neq b_i$), D_1 assigns r_i to s_{b_i} and D_2 assigns r_i to s_{a_i}.
  3. If $b_i = b_{i-1}$, then similarly we also have $a_i \neq b_{i-1}$ and $b_i \neq a_{i-1}$, and D_1 assigns r_i to s_{b_i} and D_2 assigns r_i to s_{a_i}.

From the above description, Strategy BD does not need to know the values of ρ or d. For each r_i, D_1 and D_2 assign r_i to two different servers such that one of them is of the largest weight and the other one is of the second largest weight.

Theorem 4. *Strategy BD is* $\left(2 - \frac{2}{1+\rho}\right)$*-competitive for problem OGAP|d ⩾ 1.*

Proof. Consider an input request sequence $\sigma = (r_1, r_2, \ldots, r_n)$. For each request r_i ($1 \leqslant i \leqslant n$), let W_{i1} and W_{i2} be the assignment weight obtained from r_i by D_1 and D_2, respectively. Let $|O_i|$ be the assignment weight obtained from r_i by OPT.

Since r_i is assigned to s_{a_i} by either D_1 or D_2, it follows that at least one of the following two inequalities $W_{i1} \geqslant |O_i|$ and $W_{i2} \geqslant |O_i|$ is true. Together with $W_{i1}, W_{i2} \geqslant 1$, we have $W_{i1} + W_{i2} \geqslant |O_i| + 1$. The expected assignment weight obtained by strategy BD from r_i is $(W_{i1} + W_{i2})/2$. Thus, the ratio between the total weight obtained by the offline optimal strategy OPT and the expected total weight obtained by BD from σ is

$$\frac{\sum_{i=1}^n |O_i|}{\sum_{i=1}^n (W_{i1} + W_{i2})/2} \leqslant \frac{\sum_{i=1}^n |O_i|}{\sum_{i=1}^n (|O_i| + 1)/2} \leqslant \frac{2\rho}{\rho + 1} = 2 - \frac{2}{\rho + 1},$$

where the second inequality is due to $|O_i| \leqslant \rho$ for each $1 \leqslant i \leqslant n$. The theorem follows. □

The competitive ratio $\left(2 - \frac{2}{1+\rho}\right)$ on strategy BD given in Theorem 4 is actually tight, which can be seen from the following simple problem instance with $m = 2$ servers and $n = 1$ request, and with $w_{1,1} = 1$ and $w_{1,2} = \rho$. Clearly, for this problem instance the expected weight obtained by BD is $\frac{\rho+1}{2}$, while the optimal offline strategy OPT always obtains a weight of ρ.

From Theorems 3 and 4, for problem $OGAP|d = 1$ strategy BD is asymptotically optimal when ρ goes to infinity, since both the upper and lower bounds on the competitive ratio approach 2.

4. Computational tests

In this section, we present experimental results on strategy SG described in Section 2, and strategy BD described in Section 3, to get an impression of their practical performance. We perform tests on three classes of instances. For the first class, we have two groups of test sets, with $\rho = 10$ for the first group and $\rho = 100$ for the second group. For each group we have four test sets, with the number of servers $m = 2, 3, 5$, and 10 for each test set. In each test set, for each $d_0 \in \{1, 2, \ldots, 10\}$ which is the parameter for the service consecution constraint, we randomly generate 100 instances of problem $OGAP|d = d_0$, and calculate the average competitive ratios of SG and BD on the 100 instances. Each instance generated has $n = 200$ requests, and the weights $w_{i,j}$'s ($1 \leqslant i \leqslant n$, and $1 \leqslant j \leqslant m$) for each instance follow the uniform distribution from $[1,\rho]$, with $\rho = 10$ for the first group and $\rho = 100$ for the second group.

For the second and the third classes of test instances, we set $\rho = 100$ and each class has four test sets, with the number of servers $m = 2, 3, 5$, and 10 for each test set. Similarly as for the first class, in each test set, for each $d_0 \in \{1, 2, \ldots, 10\}$ we randomly generate 100 instances of problem $OGAP|d = d_0$, and calculate the average competitive ratios of SG and BD on the 100 instances. Each instance generated has $n = 200$ requests. For the second class, the weights $w_{i,j}$'s for each instance follow the normal distribution $N(\mu,\sigma^2)$ (where μ and σ^2 specify the mean and the variance of the normal distribution, respectively) with $\mu = \rho/2 = 50$ and $\sigma = \rho/4 = 25$. For the third class of instances, each request r_i has more concentrated weights $w_{i,j}$'s ($j = 1, 2, \ldots, m$). More specifically, for each request r_i, the m weights $w_{i,j}$'s ($j = 1, 2, \ldots, m$) of r_i follow the normal distribution $N(\mu_i, (\mu_i/4)^2)$, where μ_i follows the uniform distribution from $[1,\rho = 100]$ for each $i = 1, 2, \ldots, n$.

For each random weight $w_{i,j}$ generated in the second and the third classes, if $w_{i,j}$ falls outside the interval [1, 100] then we discard the value and regenerate $w_{i,j}$, until it falls within [1, 100], so that all the weights $w_{i,j}$'s of the instances in the second and the third class are in [1, 100]. From preliminary computational tests for all the above three classes of instances, the performance of SG and BD, in terms of the average competitive ratios, is almost independent of n (for the range $n \in [10, 10{,}000]$), the number of requests in each instance. Therefore, we do not intend to exhibit the variations of the average competitive ratios of SG and BD for different values of n (since the average competitive ratios of SG and BD almost stay as constants, when m, d, and ρ are all fixed). Instead, we set $n = 200$ for all instances tested.

For each instance of problem $OGAP|d = d_0$ with n requests and m servers, the optimal solution can be obtained by an offline

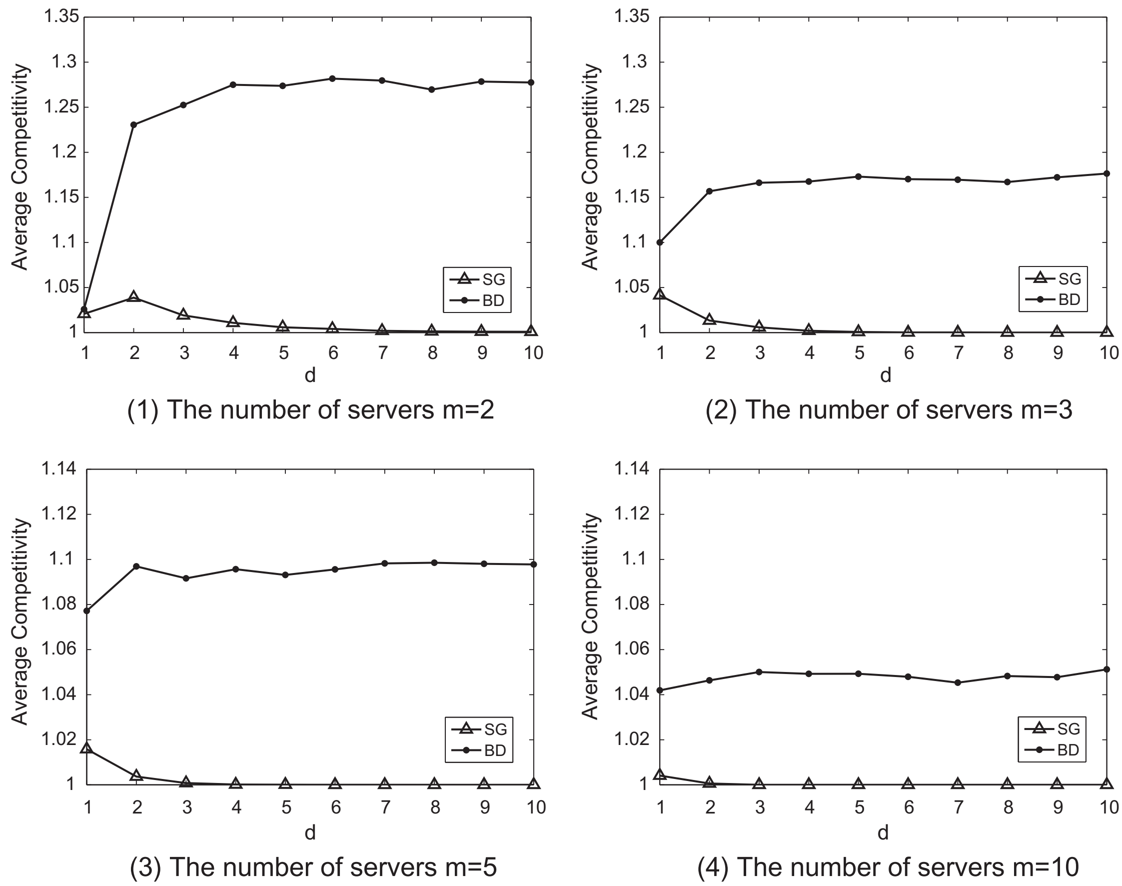

Fig. 1. Experimental competitivities on the instances in the first class, where the weights $w_{i,j}$'s follow the uniform distribution from $[1, \rho = 10]$.

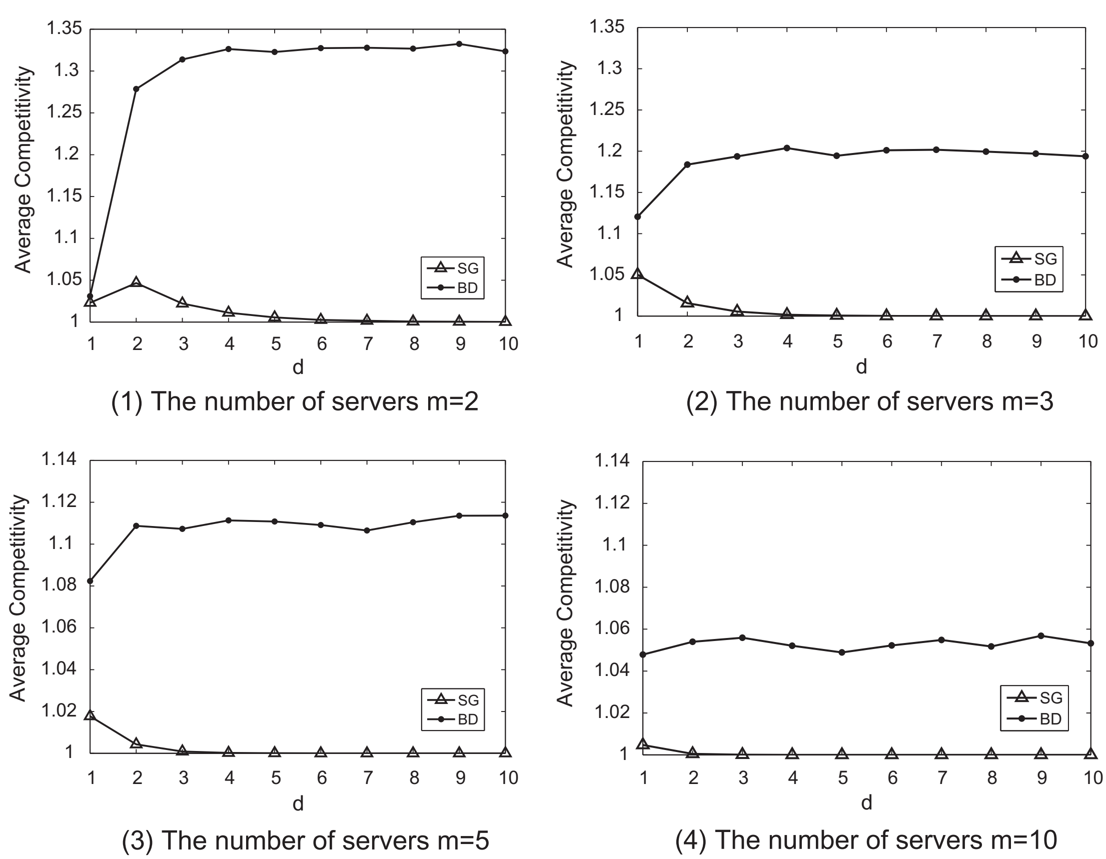

Fig. 2. Experimental competitivities on the instances in the first class, where the weights $w_{i,j}$'s follow the uniform distribution from $[1, \rho = 100]$.

algorithm OPT using the standard dynamic programming technique, in time $poly(m, n, d_0)$ (i.e., polynomial in m, n, and d_0). The detailed description of Algorithm OPT is referred to Section 4 in Supplementary material. The solutions produced by SG and BD are compared with the optimal solutions produced by Strategy OPT. The experimental results on the above three classes of instances indicate that, both SG and BD have better practical performance as the value of m increases, the practical performance of strategy SG generally improves as the value of d increases, and is close to optimal except for a few small values of d; while the practical performance of strategy BD almost stays the same for $d \geqslant 2$ and is generally better for $d = 1$. The experimental results also indicate that both SG and BD have better performance when the weights $w_{i,j}$'s of the requests are more concentrated. Overall, both SG and BD exhibit good practical performance on all the above three classes of problem instances.

4.1. Uniformly distributed weights

For the first group of test instances in the first class, where the weights $w_{i,j}$'s for each instance follow the uniform distribution from [1, 10]. Fig. 1 exhibits, for $m = 2, 3, 5$, and 10, the practical average competitivities of SG and BD over 100 randomly generated instances of problem $OGAP|d = d_0$, for each $d_0 \in \{1, 2, \ldots, 10\}$. From Fig. 1, in general both SG and BD have better practical performance (i.e., smaller average competitivities) as the value of m increases.

For strategy SG, from Fig. 1 its practical performance in general improves as the value of d increases, which is in accordance with the theoretical competitive ratio $\rho^{\frac{1}{d}}$ of SG given in Section 2. Strategy SG produces almost optimal solutions (i.e., with average competitivities close to 1) as long as d is not very small (e.g., when $d \geqslant 4$ for $m = 3$). Overall, for the first group of problem instances in the first class, SG exhibits very good practical performance, with average competitivities less than 1.05 for all parameters m and d tested.

For strategy BD, from Fig. 1 its practical performance almost stays the same for $d \geqslant 2$ and is generally better for $d = 1$. This is in accordance with that strategy BD actually does not make use of the information of value d, and the assignment produced by BD always satisfies the service consecution constraint specified by parameter one, no matter what the actual value of d is. From Fig. 1, for the first group of problem instances in the first class, strategy BD also exhibits good practical performance, with average competitivities less than 1.3 for all parameters m and d tested.

The above trends on the performance of SG and BD on the first group of instances are as well demonstrated in Fig. 2, on the second group of problem instances, where all parameter settings are the same as for the first group except that $\rho = 100$. From Figs. 1 and 2, both SG and BD have better practical performance when the value of ρ is smaller, which, again, is in accordance with their theoretical competitive ratios of $\rho^{\frac{1}{d}}$ and $\left(2 - \frac{2}{1+\rho}\right)$, given in Sections 2 and 3 respectively. This is also in accordance with the intuition, since when the weights $w_{i,j}$'s of the requests are more concentrated, the total weight obtained by SG (or, BD) and the optimal weight obtained by OPT tend to be more close to each other.

As shown in Figs. 1 and 2, the value of ρ has a relatively strong impact on the practical performance of BD, while has a much weaker impact on the practical performance of SG. From Fig. 2, for the second group of problem instances in the first class, SG and BD also exhibit good practical performance, with average

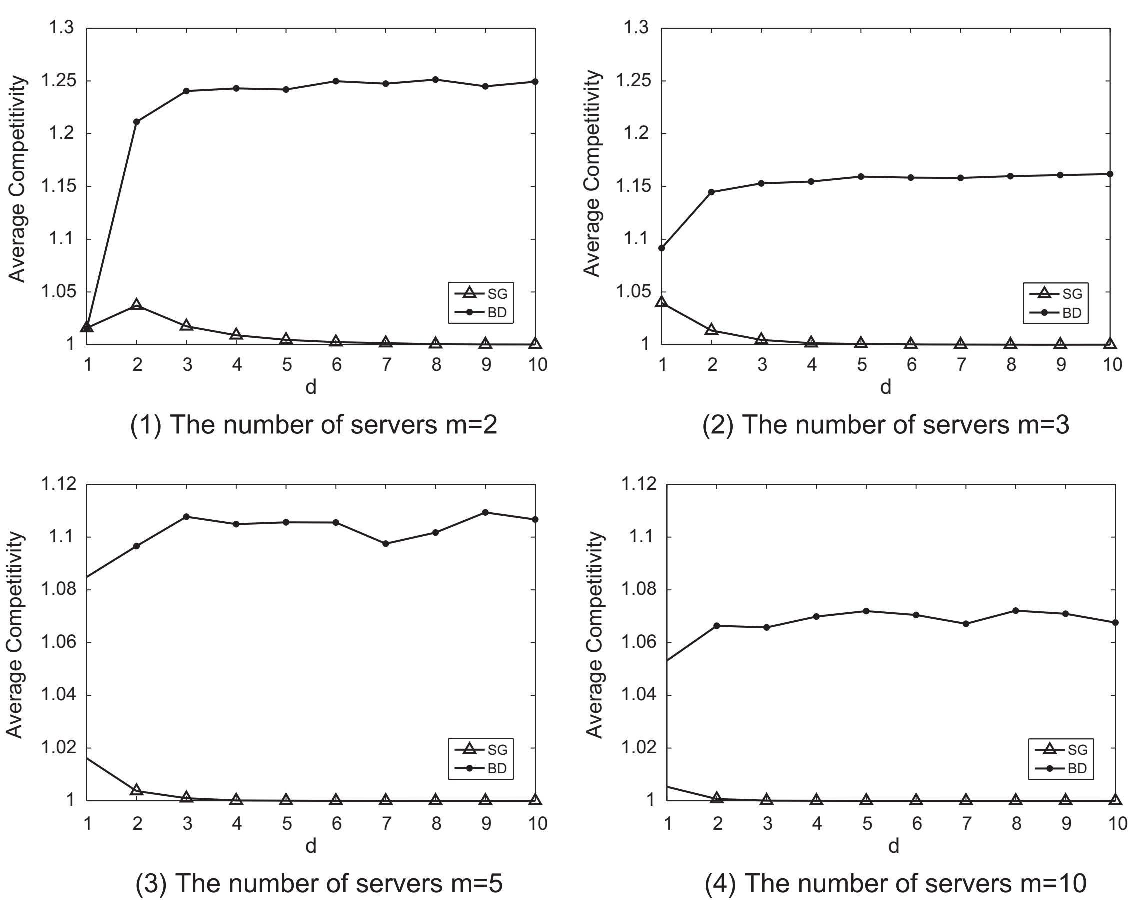

Fig. 3. Experimental competitivities on the instances in the second class, where $\rho = 100$ and the weights $w_{i,j}$'s follow the normal distribution $N(50, 25^2)$.

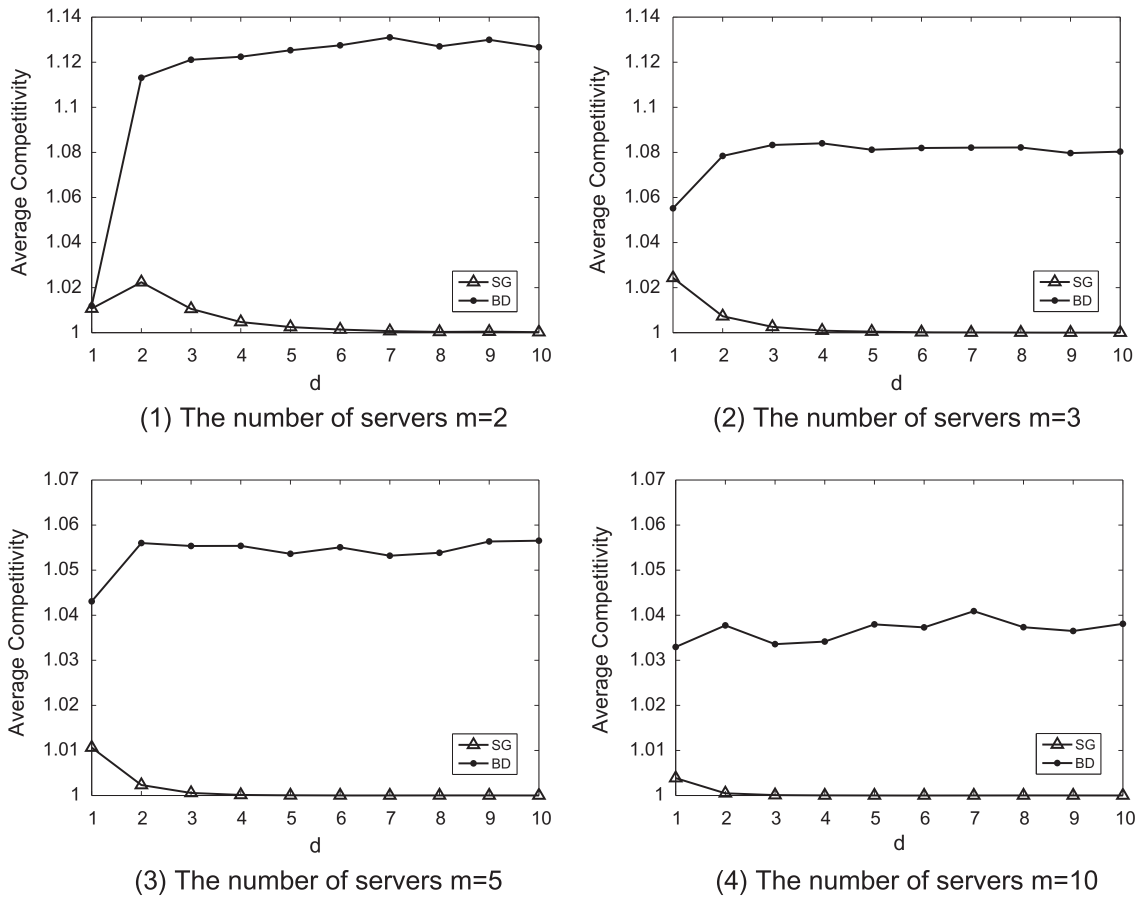

Fig. 4. Experimental competitivities on the instances in the third class, where the weights $w_{i,j}$'s ($j = 1, 2, \ldots, m$) of each request r_i follow the normal distribution $N(\mu_i, (\mu_i/4)^2)$, and μ_i follows the uniform distribution from $[1, \rho = 100]$ for each $i = 1, 2, \ldots, n$.

competitivities less than 1.05 and 1.35 respectively, for all parameters m and d tested.

4.2. Normally distributed weights

Fig. 3 illustrates the average competitivities of SG and BD on the second class of test instances, where $\rho = 100$ and the weights $w_{i,j}$'s follow the normal distribution $N(50, 25^2)$. Since it is well known that for a random variable following normal distribution $N(\mu,\sigma^2)$, about 95.44% of values of the random variable are within two times of its standard deviation from its mean (i.e., are within $[\mu - 2\sigma, \mu + 2\sigma]$), which indicates that it is safe to view the weights $w_{i,j}$'s generated in the second class (which are within $[1, 100]$) as being approximately normally distributed.

By comparing Figs. 3 and 2, the findings summarized from Fig. 2 (where the weights $w_{i,j}$'s follow the uniform distribution from $[1, 100]$) on the trends of the practical performance of SG and BD also apply here. In particular, both SG and BD have better practical performance as the value of m increases, the practical performance of SG generally improves as the value of d increases, while the practical performance of BD almost stays the same for $d \geqslant 2$ and is generally better for $d = 1$. Both SG and BD exhibit good practical performance on the second class of instances, with average competitivities less than 1.05 and 1.3 respectively, for all parameters m and d tested.

Experimental results also indicate that when the weights follow the normal distribution $N(50,\sigma^2)$, in general the practical performance of both SG and BD improves as the value of σ decreases (i.e., when the weights $w_{i,j}$'s are more concentrated). Similarly as the explanation given in Section 4.1 on the first class of test instances, this is intuitively correct since when the weights $w_{i,j}$'s are more concentrated, the difference between the total weight obtained by SG (or, BD) and the optimal weight obtained by OPT tends to be smaller.

4.3. Concentrated weights for each request

For the third class of instances, for each request r_i the weights $w_{i,j}$'s ($j = 1, 2, \ldots, m$) follow the normal distribution $N(\mu_i, (\mu_i/4)^2)$, where μ_i follows the uniform distribution from $[1, \rho = 100]$ for $i = 1, 2, \ldots, n$. Since for any random variable following normal distribution $N(\mu, \sigma^2)$, about 68.26% (95.44%, and 99.72%) of values of the random variable are within $[\mu - \sigma, \mu + \sigma]$ ($[\mu - 2\sigma, \mu + 2\sigma]$, and $[\mu - 3\sigma, \mu + 3\sigma]$), which indicates that the weights $w_{i,j}$'s ($j = 1, 2, \ldots, m$) for any fixed request r_i generated in the third class can be considered to be well concentrated around μ_i.

Fig. 4 illustrates the average competitivities of SG and BD on the third class of test instances. By comparing Fig. 4 with Figs. 2 and 3 ($\rho = 100$ for all the three figures), in general both SG and BD have better average competitivities on the third class of instances than on the first and the second class of instances (with the same value of ρ). Similarly to the previous arguments in Sections 4.1 and 4.2, this is intuitively correct since when the weights $w_{i,j}$'s ($j = 1, 2, \ldots, m$) are more concentrated for each request r_i, the total weight obtained by SG (or, BD) and the optimal weight obtained by OPT tend to be more close to each other.

5. Conclusions and future studies

In this paper we investigated a variant of the online generalized assignment problem, $OGAP|d \geqslant 1$, which has a service consecution constraint such that at any time each server is only allowed to process up to $d \geqslant 1$ consecutive requests. We investigate both deterministic and randomized online strategies for this problem. For deterministic online strategies, we first prove a lower bound of $\rho^{\frac{1}{d}}$ on the competitive ratio for any deterministic strategy. Then, we give an optimal $\rho^{\frac{1}{d}}$-competitive deterministic online strategy, which requires to know the value of ρ in advance. For randomized online strategies, we first prove a lower bound of $\left(\frac{d+1}{d} - \frac{d+1}{d^2}\rho^{-\frac{1}{d+1}}\right)$ on the competitive ratio for any randomized strategy, then we present a $\left(2 - \frac{2}{1+\rho}\right)$-competitive randomized online strategy which does not need to know the value of ρ or d. Computational tests show that our proposed strategies have very good practical performance.

The studies reported in this paper also left several interesting problems to be studied further. Strategy SG proposed in this paper requires to know in advance the value of ρ, i.e. the upper bound on the possible assignment weights obtained from assigning a request to a server. It is interesting to investigate deterministic online strategies for problem $OGAP|d \geqslant 1$ with bounded assignment weights $w_{i,j}$'s, while without knowing the value of ρ in advance. For randomized strategies for problem $OGAP|d \geqslant 1$, it is interesting to investigate online strategies that are able to utilize the information of value d to achieve better competitive ratios, unlike strategy BD proposed in this paper which does not make any use of the value of d.

Acknowledgements

The authors thank the two anonymous reviewers for their helpful comments and suggestions. This work was partially supported by the National Natural Science Foundation of China under Grant Nos. 71172189, 11101326, 71071123, and 71101106, the Program for New Century Excellent Talents in University (NCET-12-0824), and the Program for Changjiang Scholars and Innovative Research Team in University (IRT1173).

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.ejor.2013.02.004.

References

Aneja, Y.P., Punnen, A.P., 1999. Multiple bottleneck assignment problem. European Journal of Operational Research 112 (1), 167–173.
Arora, S., Puri, M.C., 1998. A variant of time minimizing assignment problem. European Journal of Operational Research 110 (2), 314–325.
Bansal, N., Buchbinder, N., Gupta, A., Naor, J., 2007. An O($\log^2 k$)-competitive algorithm for metric bipartite matching. In: Proceedings of the 15th Annual European Symposium on Algorithms, Eilat, Israel, pp. 522–533.
Ben-David, S., Borodin, A., Karp, R., Tardos, G., Wigderson, A., 1994. On the power of randomization in on-line algorithms. Algorithmica 11, 2–14.
Borodin, A., El-Yaniv, R., 1998. Online Computation and Competitive Analysis. Cambridge University Press, Cambridge.
Campbell, G.M., Diaby, M., 2002. Development and evaluation of an assignment heuristic for allocating cross-trained workers. European Journal of Operational Research 138 (1), 9–20.
Cattrysse, D.G., Van Wassenhove, L.N., 1992. A survey of algorithms for the generalized assignment problem. European Journal of Operational Research 60 (3), 260–272.
Chang, G.J., Ho, P.H., 1998. The β-assignment problems. European Journal of Operational Research 104 (3), 593–600.
Csaba, B., Pluhar, A., 2008. A randomized algorithm for the on-line weighted bipartite matching problem. Journal of Scheduling 11 (6), 449–455.
Easton, K., Nemhauser, G., Trick, M., 2001. The traveling tournament problem: description and benchmarks. In: Walsh, T. (Ed.), Principles and Practice of Constraint Programming – CP 2001, Lecture Notes in Computer Science, vol. 2239. Springer, Berlin/Heidelberg, pp. 580–585.
Fisher, M., Jaikumar, R., 1981. A generalized assignment heuristic for vehicle routing. Networks 11 (2), 109–124.
Gschwind, T., Irnich, S., 2011. A note on symmetry reduction for circular traveling tournament problems. European Journal of Operational Research 210 (2), 452–456.
Irnich, S., 2010. A new branch-and-price algorithm for the traveling tournament problem. European Journal of Operational Research 204 (2), 218–228.
Kalyanasundaram, B., Pruhs, K., 1993. Online weighted matching. Journal of Algorithms 4 (3), 478–488.
Karp, R.M., Vazirani, U.V., Vazirani, V.V., 1990. An optimal algorithm for on-line bipartite matching. In: Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, Baltimore, Maryland, pp. 352–358.
Khuller, S., Mitchell, S.G., Vazirani, V.V., 1994. On-line algorithms for weighted bipartite matching and stable marriages. Theoretical Computer Science 127 (2), 255–267.
Kuhn, H.W., 1955. The Hungarian method for the assignment problem. Naval Research Logistics 2, 83–97.
Martello, S., Toth, P., 1995. The bottleneck generalized assignment problem. European Journal of Operational Research 83 (3), 621–638.
Martello, S., Pulleyblank, W.R., Toth, P., Werra, D., 1984. Balanced optimization problems. Operations Research Letters 3 (5), 275–278.
Mazzola, J., Neebe, A., Dunn, C., 1989. Production planning of a flexible manufacturing system in a material requirements planning environment. International Journal of Flexible Manufacturing Systems 1 (2), 115–142.
Meyerson, A., Nanavati, A., Poplawski, L.J., 2006. Randomized online algorithms for minimum metric bipartite matching. In: Proceedings of the 17th Annual ACM–SIAM Symposium on Discrete Algorithm, Miami, Florida, pp. 954–959.
Moccia, L., Cordeau, J.F., Monaco, M.F., Sammarra, M., 2009. A column generation heuristic for a dynamic generalized assignment problem. Computers & Operations Research 36 (9), 2670–2681.
Oncan, T., 2007. A survey of the generalized assignment problem and its applications. INFOR: Information Systems and Operational Research 45 (3), 123–141.
Pentico, D.W., 2007. Assignment problems: a golden anniversary survey. European Journal of Operational Research 176 (2), 774–793.
Raghavan, P., Snir, M., 1994. Memory versus randomization in on-line algorithms. IBM Journal of Research and Development 38 (6), 683–708.
Rasmussen, R.V., Trick, M.A., 2008. Round robin scheduling – a survey. European Journal of Operational Research 188 (3), 617–636.
Ravindran, A., Ramaswami, V., 1977. On the bottleneck assignment problem. Journal of Optimization Theory and Applications 21 (4), 451–458.
Ross, G.T., Soland, R.M., 1977. Modeling facility location problems as generalized assignment problems. Management Science 24 (3), 345–357.
Sun, K., Li, H., 2010. Scheduling problems with multiple maintenance activities and non-preemptive jobs on two identical parallel machines. International Journal of Production Economics 124 (1), 151–158.
Xu, D., Sun, K., Li, H., 2008. Parallel machine scheduling with almost periodic maintenance and non-preemptive jobs to minimize makespan. Computers & Operations Research 35 (4), 1344–1349.