Journal of Zhejiang University SCIENCE
ISSN 1009-3095
http://www.zju.edu.cn/jzus
E-mail: jzus@zju.edu.cn

Deterministic and randomized scheduling problems under the l_p norm on two identical machines[*]

LIN Ling (林　凌), TAN Zhi-yi (谈之奕), HE Yong (何　勇)[†]

(*Department of Mathematics, State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, China*)

[†]E-mail: mathhey@zju.edu.cn

Received Feb. 10, 2004;　revision accepted July 3, 2004

Abstract: Parallel machine scheduling problems, which are important discrete optimization problems, may occur in many applications. For example, load balancing in network communication channel assignment, parallel processing in large-size computing, task arrangement in flexible manufacturing systems, etc., are multiprocessor scheduling problem. In the traditional parallel machine scheduling problems, it is assumed that the problems are considered in offline or online environment. But in practice, problems are often not really offline or online but somehow in-between. This means that, with respect to the online problem, some further information about the tasks is available, which allows the improvement of the performance of the best possible algorithms. Problems of this class are called semi-online ones. In this paper, the semi-online problem $P2|\text{decr}|l_p$ (p>1) is considered where jobs come in non-increasing order of their processing times and the objective is to minimize the sum of the l_p norm of every machine's load. It is shown that *LS* algorithm is optimal for any l_p norm, which extends the results known in the literature. Furthermore, randomized lower bounds for the problems $P2|\text{online}|l_p$ and $P2|\text{decr}|l_p$ are presented.

Keywords: Semi-online, Scheduling, Randomization, Competitive ratio
doi:10.1631/jzus.2005.A0020 **Document code:** A **CLC number:** TP393, O223

INTRODUCTION

In this work, we study online and semi-online scheduling problems under the l_p norm (p>1). We are given a sequence $J=\{p_1, p_2, \ldots, p_n\}$ of independent jobs, each with a positive processing time (size) p_i, i=1, …, n, that must be non-preemptively scheduled on one of m identical machines. The load of a machine is the sum of the size of the jobs assigned to it. The load of ith machine related to an algorithm S is denoted by $t_i(J,S)$, i=1, 2, …, m, respectively. The objective is to minimize the sum of the l_p norm of every machine's load, i.e. $L_p(J,S)\equiv[t_1(J,S)^p+t_2(J,S)^p+\ldots+t_m(J,S)^p]^{1/p}$, where p>1 (Note that for p=1, the problem is trivial and any algorithm yields an optimal solution).

A scheduling problem is called online if jobs come one by one and it is required to schedule jobs irrevocably onto machines as soon as they are given, without any knowledge about jobs that follow later on. If some partially additional information about jobs is available in advance, and we cannot rearrange any job which has been assigned to machines, then it is called semi-online. Different partial information results in different semi-online problem. In the semi-online version with non-increasing job sizes (Graham, 1969; Seiden *et al.*, 2000), we know in advance that jobs come in non-increasing order of their sizes, i.e. $p_i\geq p_{i+1}$, i=1, …, n−1. If we have full information on the job data before constructing a scheduling, it is called offline. Algorithms for online and semi-online problems are called online and semi-online algorithms, respectively. Naturally, one wishes to achieve improvement of the performance of the optimal semi-online algorithm with respect to

[*]Project supported by the National Natural Science Foundation of China (Nos. 10271110, 10301028) and the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of MOE, China

the online version by exploiting additional information. Though it is a relatively new area, more and more results of semi-online algorithms for scheduling problems appeared in the last decade. The readers may refer to recent survey papers (He *et al.*, 2003a; 2003b). This paper will also consider such a problem.

We measure the performance of an algorithm for online and semi-online problems by its competitive ratio. For any fixed $p>1$, the competitive ratio of an algorithm S is defined as $C_p(S)=\sup_J L_p(J,S)/L_p(J,OPT)$, where J is an arbitrary job sequence, and OPT is an optimal offline algorithm. For randomized algorithm, we use $E(L_p(J,S))$ instead of $L_p(J,S)$ in the above definition. If no deterministic (randomized) algorithm for an online (semi-online) scheduling problem has a competitive ratio smaller than c, then c is called the deterministic (randomized) lower bound for the corresponding problem. A deterministic or randomized algorithm is called optimal if its competitive ratio matches the corresponding lower bound.

The prime motivation of this study is as follows: So far most researches have been done in various scheduling models for the makespan measure, i.e. the l_∞ norm, but in some applications other norms may be suitable such as the l_2 norm (Azar *et al.*, 2002). Consider for example a case where the processing time of a job corresponds to its machine disk access frequency. Then each job may see a delay that is proportional to the load on the machine it is assigned to. Thus the average delay is proportional to the sum of squares of the machines loads (i.e. l_2 norm of machine loads), whereas the maximum delay is proportional to the maximum load (Chandra and Wong, 1975). Besides the real applications of the scheduling under l_p norm, study for this kind of problems has also theoretical significance. In fact, a major drawback in optimization problems and in particular in scheduling problems is that for every measure there may be a different optimal solution. Usually, different algorithms are used for diverse measures, each providing its own solution. Therefore, one may ask what is the "correct" solution for a given scheduling problem? In many cases there is no right answer to this question. In this paper, we show a case for which we can provide an appropriate answer, especially when the measures are different l_p norms. Here "appropriate answer" means that the algorithm is optimal for every l_p norm.

For the classical parallel machine scheduling problem $Pm|\text{online}|C_{\max}$, which can be viewed as the l_∞ norm, Graham (1966) showed that the competitive ratio of LS algorithm is $2-1/m$. Here LS algorithm always puts the first unassigned job on the machine with minimum current load. Faigle *et al.*(1989) further proved that LS is an optimal algorithm when m=2, 3. For the problem $Pm|\text{online}|l_2$, Avidor *et al.*(2001) showed that the competitive ratio of LS is 4/3 for any number of machines divisible by 3 but strictly less than 4/3 in all other cases; and that LS is optimal. For the general problem $Pm|\text{online}|l_p$, the exact competitive ratio of LS turns out to be $2-\Theta(\ln p/p)$ (Avidor *et al.*, 2001). Specially, the competitive ratio of LS for m=2 is $\sup_{\Delta\geq 0}(((1+\Delta)^p+1)/(\Delta^p+2^p))^{1/p}$. Moreover, for $P2|\text{online}|l_\infty$, an optimal randomized algorithm with competitive ratio 4/3 was proposed by Bartal *et al.*(1995). But to the authors' knowledge, there is few literature considering randomized algorithm in the l_p norm, where $p>1$. In this paper, we will propose a randomized lower bound of the problem $P2|\text{online}|l_p$.

For the semi-online problem with decreasing job size $Pm|\text{decr}|l_\infty$, Graham (1969) proved that the competitive ratio of LS algorithm (i.e. LPT algorithm) is $4/3-1/(3m)$. Seiden *et al.*(2000) proved LS is optimal for m=2. Furthermore, an optimal randomized algorithm with competitive ratio 8/7 was also proposed for m=2 in the same paper. However the general problem $P2|\text{decr}|l_p$ is relatively unexplored. In this paper, we will consider this problem by proving that LS is an optimal algorithm for any l_p norm. The competitive ratio is $\max_{1\leq\Delta<2}(((\Delta+2)^p+(\Delta+1)^p)/((2\Delta)^p+3^p))^{1/p}$. Thus LS algorithm is an all-norm optimal algorithm (Azar *et al.*, 2002) for this semi-online scheduling problem, supplying one solution guaranteeing simultaneously optimal approximation with respect to the optimal solutions for all norms. This result extends the one obtained by Seiden *et al.*(2000). Moreover, we will also give a randomized lower bound of $P2|\text{decr}|l_p$.

A closely related problem is the off-line problem $P||l_p$. Chandra and Wong (1975) proved that the worst-case ratio of LPT lies in the interval [1.03, 1.04] for p=2. In 1995, tighter bounds of this algorithm was given by Leung and Wei (1995), which is a piecewise function depending on m. Alon *et al.*(1998) provided a polynomial time approximation scheme of $P||l_p$, for any p>1.

OPTIMAL SEMI-ONLINE DETERMINISTIC ALGORITHM

In this section, we are devoted to proving LS algorithm is an optimal algorithm of the problem $P2|\text{decr}|l_p$ for any p>1. We assume $p_1 \ge p_2 \ge \ldots \ge p_n$ in this section. We first present several technical lemmas.

Lemma 1 Define $H(r) \equiv r^p + (c-r)^p$ over $0 \le r \le c$, where c is a positive constant. We have: (1) $H(r)$ monotonically increases when $r \ge c/2$; (2) $c^p/2^{p-1} \le H(r) \le c^p$; (3) $H(r)$ monotonically increases in terms of $|c-2r|$.

Proof Clearly $H'(r) = pr^{p-1} - p(c-r)^{p-1}$. The equation $H'(r)=0$ has a unique solution $r=c/2$, and $H'(r) \ge 0$ if and only if $r \in [c/2, c]$ for the considered interval of r. We can conclude that $H(r)$ monotonically increases when $r \in [c/2, c]$, and decreases when $r \in [0, c/2]$. Thus

$$\frac{c^p}{2^{p-1}} = H\left(\frac{r}{2}\right) = \min_{0 \le r \le c} H(r) \le H(r)$$
$$\le \max_{0 \le r \le c} H(r) = H(c) = c^p .$$

Since $|c-2r|$ also monotonically increases when $r \in [c/2, c]$ and decreases when $r \in [0, c/2]$, we conclude that $H(r)$ monotonically increases in terms of $|c-2r|$.

Lemma 2 Algorithm LS always yields an optimal solution for any sequence J of the problem $P2|\text{decr}|l_p$, where $|J| \le 4$.

Proof The cases of $|J| \le 3$ are trivial. Hence we focus on the case $|J|=4$. For the sake of simplicity, we assume that the sum of all job sizes in J is 1. Let B(C) denote the algorithm that assigns two jobs (three jobs) to one machine, and the remaining jobs to another machine. Then OPT must be one of B and C. Then it suffices to show that $L_p(J,LS) \le L_p(J,B)$ and $L_p(J,LS) \le L_p(J,C)$ to get the desired result. Let i, j, $k \in \{2,3,4\}$ with $i \ne j \ne k$. Note that if an algorithm A assigns jobs to one machine with load r, then its objective value is $L_p(J,A) = (r^p + (1-r)^p)^{1/p}$. Since $\max\{r, 1-r\} \ge 1/2$ is true for any $r \in [0,1]$, by Lemma 1(1), $(L_p(J,A))^p$ monotonically increases in terms of $\max\{r, 1-r\}$.

If $p_1 \ge p_2 + p_3$, then $L_p(J,LS) = (p_1^p + (p_2 + p_3 + p_4)^p)^{1/p}$. Since $\max\{p_1+p_i, p_j+p_k\} = p_1+p_i \ge \max\{p_1, p_2+p_3+p_4\}$, we get $L_p(J,LS) \le L_p(J,B)$. Similarly, we can get $L_p(J,LS) \le L_p(J,C)$. If $p_1 < p_2+p_3$, then $L_p(J,LS) = ((p_1+p_4)^p + (p_2+p_3)^p)^{1/p}$, and $L_p(J,B) = ((p_1+p_i)^p + (p_j+p_k)^p)^{1/p}$. Since i, j, $k \in \{2,3,4\}$ with $i \ne j \ne k$, and $p_1 \ge p_2 \ge p_3 \ge p_4$, we have $\max\{p_1+p_i, p_j+p_k\} \ge p_1+p_4$ and $\max\{p_1+p_i, p_j+p_k\} \ge p_2+p_3$. It follows that $L_p(J,LS) \le L_p(J,B)$. Similarly, $L_p(J,LS) \le L_p(J,C)$ holds. Thus LS is optimal for any sequence with 4 jobs.

Lemma 3 $C_p(LS)$ is achieved by the job sequence $J_0 = \{y, y, z, z, z\}$, where $y/2 < z \le y$.

Proof By normalization, we can assume that the sum of all job sizes in J is 1. Furthermore, we assume that we are considering the job sequence with the fewest jobs. In order to get a sequence J such that the ratio $L_p(J,LS)/L_p(J,OPT)$ achieves the supremum, we show that we only need to consider such J satisfying

(i) $p_n \ge |t_1(J,LS) - t_2(J,LS)|$,

(ii) J only has 5 jobs.

To prove (i), we suppose there exists a job sequence J such that $p_n < |t_1(J,LS) - t_2(J,LS)|$. Assume that $t_1(J,LS) > t_2(J,LS)$ (the case $t_2(J,LS) > t_1(J,LS)$ is similar). Then p_n must be assigned to the second machine. In fact, if not, denote by s the starting time of p_n on the first machine. Since we are considering the instance with fewest number of jobs, we have $t_1(J,LS) = s + p_n$. According to LS rule, $s \le t_2(J,LS)$. Then $t_1(J,LS) - t_2(J,LS) \le p_n$, a contradiction.

Let $J' = \{p_1, p_2, \ldots, p_{n-1}\}$, we have $t_1(J',LS) = t_1(J,LS)$, $t_2(J',LS) = t_2(J,LS) - p_n$. W.l.o.g., let p_n be on the second machine of OPT. Let P be the algorithm for J' that agrees with OPT for J. Set $u = t_2(J,LS)$, $v = t_2(J,OPT)$. We claim that

$$\frac{L_p(J',LS)}{L_p(J',OPT)} \ge \frac{L_p(J,LS)}{L_p(J,OPT)} \qquad (1)$$

i.e. $\frac{(1-u)^p+(u-p_n)^p}{(L_p(J',OPT))^p}\geq\frac{(1-u)^p+u^p}{(1-v)^p+v^p}$. To see it, since $L_p(J',OPT)\leq L_p(J',P)=((1-v)^p+(v-p_n)^p)^{1/p}$, we have

$$\frac{(1-u)^p+(u-p_n)^p}{(L_p(J',OPT))^p}\geq\frac{(1-u)^p+(u-p_n)^p}{(1-v)^p+(v-p_n)^p}.$$

Denote $C_1(x)=(1-u)^p+(u-x)^p$, $C_2(x)=(1-v)^p+(v-x)^p$, and $w(x)=C_1(x)/C_2(x)$, then we know

$$w'(x)=\frac{p(C_1(x)(v-x)^{p-1}-C_2(x)(u-x)^{p-1})}{(C_2(x))^2}.$$

Since $v\geq u\geq x$ and $C_1(x)\geq C_2(x)$, we have $w'(x)\geq 0$, and thus $w(p_n)>w(0)$. It leads to $\frac{(1-u)^p+(u-p_n)^p}{(1-v)^p+(v-p_n)^p}\geq\frac{(1-u)^p+u^p}{(1-v)^p+v^p}$. Hence Eq.(1) holds. Since the sum of the jobs in sequence J' is less than 1, we can make the sum equal to 1 by scaling up all sizes by the same number (i.e. by re-normalization). So we claim that for any J violating (i) we can remove the last job from J until (i) is satisfied (Note that for the job sequence with one job, (i) is valid trivially).

Next we prove (ii). Consider the sequence $J=\{1/4,1/4,1/6,1/6,1/6\}$. Note that the gap between two machine loads is $h(A)\equiv|t_1(J,A)-t_2(J,A)|=|1-2t_2(J,A)|$ for any algorithm A. We have $h(LS)=1/6$ and $h(OPT)=0$. By Lemma 1(3), $L_p(J,LS)$ is a monotonically increasing function of $h(LS)$ for any J with five jobs. If $J=\{1/4,1/4,1/6,1/6,1/6\}$ achieves the supremum, we are done since this instance satisfies the result of the lemma. Thus we must select the sequence J satisfying $h(LS)>1/6$ in order to increase $L_p(J,LS)/L_p(J,OPT)$. By the condition (i) we have $p_n>1/6$. However since the total size of all jobs is 1 and $p_1\geq p_2\geq\ldots\geq p_n$, we know $p_n\leq 1/n$, which implies $n\leq 5$. Combining it with Lemma 2, we know that $|J|=5$. Hence we obtain (ii).

Now we return to the proof of the lemma. Given a non-increasing sequence of jobs $J=\{p_1, p_2, \ldots, p_5\}$ with $p_5>1/6$, since $1-p_1-p_5=p_2+p_3+p_4\geq 3p_5$, we have $p_1<1/3$. Obviously the sum of any three jobs is greater than 1/2. It follows that the sum of any three jobs is greater than that of the remaining. The optimal algorithm *OPT* must assign p_1, p_2 to one machine and p_3, p_4, p_5 to anther machine, while *LS* assigns p_1, p_4 to one machine, and p_2, p_3 to anther machine, since $p_1<1/3<p_2+p_3$. Two cases are considered with respect to the assignment of p_5 in *LS* schedule.

Case 1 $p_1+p_4>p_2+p_3$. Then p_2, p_3 and p_5 are on the same machine in *LS* schedule, and $p_2+p_3+p_5>1/2$. We get the desired sequence by implementing the following three steps:

(a) Let $\varepsilon=((p_1+p_4)-(p_2+p_3))/2$. Replacing p_1 by $p_1'=p_1-\varepsilon$, p_2 by $p_2'=p_2+\varepsilon$, results in increasing $h(LS)$ but keeping $h(OPT)$ unchanged. Thus $L_p(J,LS)/L_p(J,OPT)$ becomes larger. And we have $p_1'+p_4=p_2'+p_3$.

(b) Let $\varepsilon'=(p_1'-p_2')/2$. By replacing p_1' by $p_1''=p_1'-\varepsilon'$, p_2' by $p_2''=p_2'+\varepsilon'$, p_3 by $p_3'=p_3-\varepsilon'$, and p_4 by $p_4'=p_4+\varepsilon'$, we have $p_1''=p_2''$, $p_3'=p_4'$, and both $h(LS)$ and $h(OPT)$ remain unchanged.

(c) Finally, let $\varepsilon''=(p_3'-p_5)/3$, $p_3''=p_3'-\varepsilon''$, and $p_4''=p_4'-\varepsilon''$, $p_5'=p_5+2\varepsilon''$, then $p_3''=p_4''=p_5'$. It increases $h(LS)$ and has no effect on $h(OPT)$, and thus yields a larger $L_p(J,LS)/L_p(J,OPT)$. Now we get a sequence $J''=\{p_1'', p_2'', p_3'', p_4'', p_5'\}$ satisfying $p_1''=p_2''$, $p_3''=p_4''=p_5'$, and $p_1''/2<p_3''\leq p_1''$.

Case 2 $p_1+p_4\leq p_2+p_3$. Then p_1, p_4 and p_5 are on the same machine in *LS* schedule. Let $\varepsilon=((p_2+p_3)-(p_1+p_4))/2$, $p_3'=p_3-\varepsilon$, $p_4'=p_4+\varepsilon$. It increases $h(LS)$ and remains $h(OPT)$ unchanged, and thus increases $L_p(J,LS)/L_p(J,OPT)$. Furthermore $p_1+p_4'=p_2+p_3'$, which is similar to the situation in Case 1(a). Therefore the result can be obtained by similar arguments as those in Case 1(b) and 1(c).

Thus $\sup_J\frac{L_p(J,LS)}{L_p(J,OPT)}$ is achieved by the job sequence $J_0=\{y, y, z, z, z\}$, $y/2<z\leq y$.

Define $f_p(\Delta)\equiv\left(\frac{(\Delta+2)^p+(\Delta+1)^p}{(2\Delta)^p+3^p}\right)^{1/p}$.

Lemma 4 For any fixed $p>1$, $C_p(LS)\leq\max_{1\leq\Delta<2}f_p(\Delta)$.

Proof According to Lemma 3, we have $C_p(LS)\leq\sup_{J_0}\frac{L_p(J_0,LS)}{L_p(J_0,OPT)}$. Since $L_p(J_0,LS)=[(y+2z)^p+(y+z)^p]^{1/p}$ and $L_p(J_0,OPT)=[(2y)^p+(3z)^p]^{1/p}$, we have

$$C_p(LS) \le \sup_{z\le y<2z}\left(\frac{(y+2z)^p+(y+z)^p}{(2y)^p+(3z)^p}\right)^{1/p}.$$

Set $\Delta = y/z$, then $1\le\Delta<2$, we have

$$C_p(LS) \le \sup_{1\le\Delta<2}\left(\frac{(\Delta+2)^p+(\Delta+1)^p}{(2\Delta)^p+3^p}\right)^{1/p} = \sup_{1\le\Delta<2} f_p(\Delta).$$

We next show that $\max_{\Delta\ge 0} f_p(\Delta)$ is achieved at a point $\Delta^*\in[1, 2]$ for any fixed $p>1$, which implies $\sup_{1\le\Delta<2} f_p(\Delta) = \max_{1\le\Delta<2} f_p(\Delta)$. In fact, since $C_p(LS)\ge 1$, we have $(\Delta^*+2)^p+(\Delta^*+1)^p \ge (2\Delta^*)^p+3^p$. Combining it with Lemma 1(1), we have $\max\{\Delta^*+2,\Delta^*+1\}=\Delta^*+2\ge \max(2\Delta^*,3)\ge(2\Delta^*+3)/2$. $\Delta^*+2\ge\max\{2\Delta^*,3\}$ implies that $\Delta^*\in[1,2]$. By setting $f_p'(\Delta)=0$, we know $\max_{\Delta\ge 0} f_p(\Delta)$ is achieved at the solution of the following equation in x:

$$[(x+2)^{p-1}+(x+1)^{p-1}][(2x)^p+3^p] = 2(2x)^{p-1}[(x+2)^p+(x+1)^p].$$

Since $p>1$, $x=2$ is not the solution of the above equation. So $\Delta^*\ne 2$ and the desired conclusion follows.

Lemma 5 For any $p>1$, any deterministic algorithm for $P2|\text{decr}|l_p$ has competitive ratio at least $\max_{1\le\Delta<2} f_p(\Delta)$.

Proof We show the result by adversary method. First, two jobs with size Δ^* come, where Δ^* is defined above. If an algorithm A assigns them to the same machine, then no new job comes, and we have

$$L_p(\{\Delta^*,\Delta^*\},A) = [(2\Delta^*)^p]^{1/p} = 2\Delta^*,$$

and

$$L_p(\{\Delta^*,\Delta^*\},OPT) = (2\Delta^{*p})^{1/p} = 2^{1/p}\Delta^*.$$

Hence

$$L_p(\{\Delta^*,\Delta^*\},A)/L_p(\{\Delta^*,\Delta^*\},OPT) = 2^{1-1/p}.$$

By Lemma 1(2), we have

$$((\Delta^*+2)/(2\Delta^*+3))^p + ((\Delta^*+1)/(2\Delta^*+3))^p \le 1$$

and

$$((2\Delta^*)/(2\Delta^*+3))^p + (3/(2\Delta^*+3))^p \ge 1/2^{p-1}.$$

These two inequalities imply that

$$(f_p(\Delta^*))^p = \frac{(\Delta^*+2)^p+(\Delta^*+1)^p}{(2\Delta^*)^p+3^p} \le 2^{p-1},$$

i.e. $$\frac{L_p(\{\Delta^*,\Delta^*\},A)}{L_p(\{\Delta^*,\Delta^*\},OPT)} = 2^{1-1/p} \ge \max_{1\le\Delta<2} f_p(\Delta).$$

If algorithm A assigns the first two jobs to distinct machines, then the last three jobs with size 1 come. Denote $J=\{\Delta^*, \Delta^*, 1, 1, 1\}$. We have

$$\frac{L_p(J,LS)}{L_p(J,OPT)} \ge \left(\frac{(\Delta^*+2)^p+(\Delta^*+1)^p}{(2\Delta^*)^p+3^p}\right)^{1/p} = \max_{1\le\Delta<2} f_p(\Delta).$$

So we conclude that any semi-online algorithm has a competitive ratio at least $\max_{1\le\Delta<2} f_p(\Delta)$.

By Lemmas 4 and 5, we obtain the main result of this section:

Theorem 1 For the problem $P2|\text{decr}|l_p$, where $p>1$, LS is an optimal deterministic semi-online algorithm with competitive ratio $\max_{1\le\Delta<2} f_p(\Delta)$.

RANDOMIZED LOWER BOUNDS

This section discusses randomized lower bounds of $P2|\text{online}|l_p$ and $P2|\text{decr}|l_p$, for every $p>1$. To the authors' best knowledge, no papers ever presented any lower bound for these problems so far.

Define $$g_p(\Delta) \equiv \left(\frac{(1+\Delta)^p+1}{\Delta^p+2^p}\right)^{1/p},$$

$$G_p(\Delta) = \frac{2g_p(\Delta)-2^{1/p}}{2^{1/p}g_p(\Delta)+2(1-2^{1/p})}.$$

Recall that the optimal deterministic algorithm of $P2|\text{online}|l_p$ is LS with competitive ratio $\sup_{\Delta\ge 0} g_p(\Delta)$, and $\sup_{\Delta\ge 0} g_p(\Delta)$ is achieved at the unique solution $\Delta\in[0,\infty]$ of the following equation in x: $x^{p-1}((1+x)^{1-p}+1)=2^p$. Similar to that in the proof of

Lemma 4, it can be shown that $\sup_{\Delta\geq 0} g_p(\Delta)$ is achieved at a point $\Delta^{**}\geq 1$ for every fixed $p>1$, i.e., $\sup_{\Delta\geq 0} g_p(\Delta)=\max_{\Delta\geq 1} g_p(\Delta)$. Furthermore it can also be shown that $\max_{\Delta\geq 1} G_p(\Delta)$ is achieved at the same Δ^{**}.

Theorem 2 For any fixed $p>1$, any randomized algorithm for $P2|\text{online}|l_p$ has a competitive ratio at least $\max_{\Delta\geq 1} G_p(\Delta)$.

Proof Let $q_1=\dfrac{2-2^{1/p}}{2^{1/p}g_p(\Delta^{**})+2(1-2^{1/p})}$. Obviously $0\leq q_1\leq 1$. First, two jobs with size 1 come. If an algorithm A assigns the first two jobs to distinct machines with probability $q<q_1$, then no new job comes. We have

$$E(L_p(\{1,1\},A))=q(1^p+1^p)^{1/p}+(1-q)(2^p+0)^{1/p},$$

and

$$L_p(\{1,1\},OPT)=(1^p+1^p)^{1/p}=2^{1/p}.$$

It follows that

$$\frac{E(L_p(\{1,1\},A))}{L_p(\{1,1\},OPT)}=\frac{q(1^p+1^p)^{1/p}+(1-q)(2^p)^{1/p}}{2^{1/p}}>\frac{2-(2-2^{1/p})q_1}{2^{1/p}}=G_p(\Delta^{**}),$$

and we are done. If algorithm A assigns with probability $q\geq q_1$ to the first two jobs to distinct machines, then the last and third job with size Δ^{**} comes, where Δ^{**} is defined above. The optimal solution assigns the first two jobs together to a machine, and the last job to another machine, having objective value $(\Delta^{**p}+2^p)^{1/p}$, while

$$E(L_p(\{1,1,\Delta^{**}\},A))\geq q((\Delta^{**}+1)^p+1)^{1/p}+(1-q)(\Delta^{**p}+2^p)^{1/p}.$$

We have

$$\frac{E(L_p(\{1,1,\Delta^{**}\},A))}{L_p(\{1,1,\Delta^{**}\},OPT)}\geq\frac{q((\Delta^{**}+1)^p+1^p)^{1/p}+(1-q)(\Delta^{**p}+2^p)^{1/p}}{(\Delta^{**p}+2^p)^{1/p}}$$
$$\geq 1+\frac{(((\Delta^{**}+1)^p+1)^{1/p}-(\Delta^{**p}+2^p)^{1/p})q_1}{(\Delta^{**p}+2^p)^{1/p}}$$
$$=G_p(\Delta^{**}).$$

Hence $\dfrac{E(L_p(\{1,1,\Delta^{**}\},A))}{L_p(\{1,1,\Delta^{**}\},OPT)}\geq\max_{\Delta\geq 1} G_p(\Delta)$, and the proof is complete.

Define $F_p(\Delta)=\dfrac{2f_p(\Delta)-2^{1/p}}{2^{1/p}f_p(\Delta)+2(1-2^{1/p})}$, we then know $\max_{\Delta\geq 1} F_p(\Delta)$ is achieved at $\Delta^*\in[1, 2)$, where Δ^* is defined before.

Theorem 3 For any fixed $p>1$, any randomized algorithm for $P2|\text{decr}|l_p$ has a competitive ratio at least $\max_{1\leq\Delta<2} F_p(\Delta)$.

Proof Let $q_2=\dfrac{2-2^{1/p}}{2^{1/p}f_p(\Delta^*)+2(1-2^{1/p})}$. Obviously $0\leq q_2\leq 1$. If an algorithm A assigns the first two jobs to distinct machines with probability $q<q_2$, then no new job comes. We have,

$$E(L_p(\{\Delta^*,\Delta^*\},A))=q(\Delta^{*p}+\Delta^{*p})^{1/p}+(1-q)((2\Delta^*)^p)^{1/p}$$

and

$$L_p(\{\Delta^*,\Delta^*\},OPT)=(\Delta^{*p}+\Delta^{*p})^{1/p}=(2\Delta^{*p})^{1/p}.$$

It follows that

$$\frac{E(L_p(\{\Delta^*,\Delta^*\},A))}{L_p(\{\Delta^*,\Delta^*\},OPT)}=\frac{q(\Delta^{*p}+\Delta^{*p})^{1/p}+(1-q_2)((2\Delta^*)^p)^{1/p}}{(2\Delta^{*p})^{1/p}}>\frac{q_2(2\Delta^{*p})^{1/p}+(1-q_2)(2\Delta^*)}{(2\Delta^{*p})^{1/p}}=F_p(\Delta^*).$$

If algorithm A assigns the first two jobs to distinct machines with probability $q\geq q_2$, then the last three jobs with same size 1 come. Denote to $J=\{\Delta^*, \Delta^*, 1, 1, 1\}$. The optimal solution assigns the first two jobs together to a machine, and the last three together to another machine with objective value $[(2\Delta^*)^p+3^p]^{1/p}$, while

$$E(L_p(J,A)) \geq q((\Delta^*+2)^p+(\Delta^*+1)^p)^{1/p}+(1-q)((2\Delta^*)^p+3^p)^{1/p}.$$

We have

$$\frac{E(L_p(J,A))}{L_p(J,OPT)} \geq \frac{q((\Delta^*+2)^p+(\Delta^*+1)^p)^{1/p}}{((2\Delta^*)^p+3^p)^{1/p}}+1-q \geq F_p(\Delta^*).$$

Hence $\frac{E(L_p(J,A))}{L_p(J,OPT)} \geq \max_{1\leq\Delta<2} F_p(\Delta)$, and the proof is complete.

Table 1 shows the value of bounds for different $p>1$. We can see from Table 1 that the values of these functions monotonically increase in terms of p, and matches exactly the known results of the makespan problem when $p\to\infty$.

CONCLUSION

In this paper we proved that LS algorithm is an optimal algorithm of $P2|\text{decr}|l_p$ for any $p>1$. We further presented randomized lower bounds of the problems $P2|\text{online}|l_p$ and $P2|\text{decr}|l_p$. It will be interesting to propose optimal randomized algorithms under the l_p norm for the above problems.

Table 1 Upper and lower bounds for different p

l_p	Online version		Semi-online version	
	Deterministic upper bound $\max_{\Delta\geq1} g_p(\Delta)$	Randomized lower bound $\max_{\Delta\geq1} G_p(\Delta)$	Deterministic upper bound $\max_{1\leq\Delta<2} f_p(\Delta)$	Randomized lower bound $\max_{1\leq\Delta<2} F_p(\Delta)$
1.5	1.0817	1.0622	1.0071	1.0070
2	1.1441	1.1069	1.0142	1.0137
3	1.2311	1.1659	1.0277	1.0265
5	1.3248	1.2258	1.0519	1.0485
10	1.4083	1.2775	1.0933	1.0842
20	1.4532	1.3049	1.1276	1.1122
100	1.4905	1.3276	1.1587	1.1367
∞	1.5000	1.3333	1.1667	1.1429

References

Alon, N., Azar, Y., Woeginger, G., Yadid, T., 1998. Approximation schemes for scheduling. *J of Scheduling*, **1**:55-66.

Avidor, A., Azar, Y., Sgall, J., 2001. Ancient and new algorithms for load balancing in the l_p norm. *Algorithmica*, **29**:422-441.

Azar, Y., Epstein, L., Richter, Y., Woegininger, G.J., 2002. All-norm Approximation Algorithms. Lecture Notes in Computer Science 2368, Springer-Verlag, p.288-297.

Bartal, Y., Fiat, A., Karloff, H., Vohra, R., 1995. New algorithms for an ancient scheduling problem. *J of Computer and System Science*, **51**:359-366.

Chandra, A.K., Wong, C.K., 1975. Worst-case analysis of a placement algorithm related to storage allocation. *SIAM J on Computing*, **1**: 249-263.

Faigle, U., Kern, W., Turan, G., 1989. On the performance of online algorithms for partition problems. *Acta Cybernetica*, **9**:107-119.

Graham, R.L., 1966. Bounds for certain multiprocessor anomalies. *Bell Systems Technical J*, **45**:1563-1581.

Graham, R.L., 1969. Bounds on multiprocessing timing anomalies. *SIAM J on Applied Mathematics*, **17**:416-429.

He, Y., Yang, Q.F., Tan, Z.Y., 2003a. Semi online scheduling on parallel machines (I). *Applied Mathematics–A J Chinese Universities*, **18A**:105-114 (in Chinese).

He, Y., Yang, Q.F., Tan, Z.Y., 2003b. Semi online scheduling on parallel machines (II). *Applied Mathematics–A J Chinese Universities*, **18A**:213-222 (in Chinese).

Leung, J.Y.T., Wei, W.D., 1995. Tighter bounds on heuristic for a partition problem. *Information Processing Letters*, **56**:51-57.

Seiden, S., Sgall, J., Woeginger, G.J., 2000. Semi-on-line scheduling with decreasing job sizes. *Operations Research Letters*, **27**:215-221.