An International Journal
computers & mathematics with applications
PERGAMON Computers and Mathematics with Applications 43 (2002) 1521–1528
www.elsevier.com/locate/camwa

Ordinal Algorithms for Parallel Machine Scheduling with Nonsimultaneous Machine Available Times

ZHIYI TAN AND YONG HE
Department of Mathematics, Zhejiang University
Hangzhou 310027, P.R. China
heyong@math.zju.edu.cn

(Received and accepted July 2001)

Abstract—This paper considers ordinal algorithms for parallel machine scheduling with nonsimultaneous machine available times. Two objects of minimizing the latest job completion time and minimizing the latest machine completion time are studied. For the first objective, we present the optimal algorithms for $m = 2, 3, 4$ machine cases. For $m \geq 5$, we propose an algorithm with competitive ratio $2 - 1/(m-1)$ while the lower bound is $5/3$. For the second objective, the optimal algorithm is also given. Furthermore, for a special case, an algorithm with significantly improved competitive ratio is given. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords—On-line, Parallel machine scheduling, Approximation algorithm, Competitive analysis.

1. INTRODUCTION

In this paper, we consider the ordinal on-line parallel machine scheduling problem with the objective to minimize the makespan (latest job completion time or latest machine completion time). We are given m identical parallel machines $M = \{M_1, M_2, \dots, M_m\}$ and n independent jobs $J = \{p_1, p_2, \dots, p_n\}$. The jobs are all available at time zero, but the machines may not be available at time zero. We assume machine M_j is not available until $r_j \geq 0$. Without loss of generalization, we assume $r_1 \geq r_2 \geq \cdots \geq r_m$. We further assume that jobs arrive one by one and we know nothing about the value of the processing times but the order of the jobs by their processing times. We identify the jobs with their processing times, and hence, without loss of generalization, we assume $p_1 \geq p_2 \geq \cdots \geq p_n$. We are asked to decide the assignment of all the jobs to some machines at time zero by utilizing only ordinal data rather than the actual magnitudes. Using the three-field representation [1], we denote the problem as $P, r_j|\text{ordinal on-line}|C_{\max}$.

This research is supported by the 973 National Fundamental Research Project of China (Grant Number: 1998030401(2)) and National Natural Science Foundation of China (Grant Number: 19701028).
The authors wish to thank an anonymous referee and Prof. Masao Iri for their constructive comments which guided us to make improvements to our presentation. They also thank Prof. J. B. Sidney for making the paper [2] available.

PII: S0898-1221(02)00115-3

Obviously, the parallel machine scheduling problem with nonsimultaneous machine available times (denoted by $P, r_j \parallel C_{\max}$), which was first presented by Lee [3], is a generalization of the classical parallel machine scheduling problem $P \parallel C_{\max}$. This generalized problem arises in practice. For example, machines should be prepared before they can start processing due to preventive maintenance, and the time used for preparation is different among machines. The circumstance also happens in the case where jobs arrive in batches on a periodic basis; we are asked to begin scheduling each batch before the completion of the previous batch. Note that there may exist some machines which do not process any jobs since they are still unavailable before all jobs are completed. It causes that minimizing the latest job completion time may be inconsistent with minimizing the latest machine completion time; this phenomenon is implicitly pointed out in [4]. (Recall that the two objectives are the same in $P \parallel C_{\max}$.) In the above first example, we wish to minimize the latest job completion time because we can ignore the machines which do not process any job in a schedule, while in the latter one, we wish to minimize the latest machine completion time because we must finish processing all the jobs including jobs of the previous batch. Since both objectives have applicants, we will study them in this paper. In the following, we use $C_{\max}(J)$ and $C_{\max}(M)$ to denote the latest job completion time and the latest machine completion time, respectively.

Problems with ordinal data exist in many fields of combinatorial optimization. Algorithms which utilize only ordinal data rather than actual magnitudes are called *ordinal* algorithms. For example, a greedy ordinal algorithm solves the maximum weight sum problem over independent sets in a matroid [5]. Agnetis [6] developed a polynomial asymptotically exact ordinal algorithm for the two machine no wait flow shop problem. Liu and Sidney [2] investigated the ordinal version of the problem of arranging unit length weights on a line segment. Liu and Sidney [7] and Liu *et al.* [8] presented ordinal approximation algorithms for bin-packing problem and $P \parallel C_{\max}$. On the other hand, our discussed problem also belongs to a kind of *semi-on-line* scheduling [9–13], a variant of on-line where we do have some partial knowledge on job set which makes the problem easier to solve than standard on-line scheduling problems. In our problem, we know the order of jobs by their processing times. Due to the above motivation, this paper considers the scheduling problem $P, r_j|\text{ordinal on-line}|C_{\max}$.

Competitive analysis is a type of worst-case analysis where the performance of an on-line algorithm is compared to that of the optimal off-line algorithm [14]. For an on-line algorithm A, let $C^A(M, J)$ (C^A for short) denote the makespan ($C_{\max}(J)$ or $C_{\max}(M)$) of instance (M, J) produced by the algorithm A, and $C^{\text{OPT}}(M, J)$ (C^{OPT} for short) denote the minimal makespan in an off-line version. Then the competitive ratio of the algorithm A is defined as the smallest number c such that $C^A \le cC^{\text{OPT}}$ for all instances. An algorithm with a competitive ratio c is called a *c-competitive algorithm*. An on-line algorithm A is called the *best possible* (or *optimal*) algorithm if there is no on-line algorithm for the discussed problem with a competitive ratio smaller than that of A.

The paper is organized as follows. In Section 2, we show the lower bounds of $P, r_j|\text{ordinal on-line}|C_{\max}(M)$, and present ordinal algorithms with competitive ratio 3/2, 13/8, and $2 - 1/(m-1)$ for $m = 2, 3$ and $m \ge 4$. The algorithms are optimal for $m = 2, 3, 4$. In Section 3, we propose an optimal algorithm for $P, r_j|\text{ordinal on-line}|C_{\max}(J)$ for each m. And for a special case, we modify the algorithm to get a significantly improved competitive ratio. Finally, some conclusions are given in Section 4.

In the rest of the paper, let α_j denote the completion time of M_j in an approximation algorithm and b denote the total processing time, i.e., $b = \sum_{i=1}^{n} p_i$. Let $\lfloor a \rfloor$ denote the largest integer less than or equal to a, and $\lceil a \rceil$ denote the smallest integer greater than or equal to a.

2. $P, r_j|\text{ordinal on-line}|C_{\max}(M)$

In this section, we consider the objective of minimizing the latest machine completion time. We first show its lower bounds by adversary method. Hence, we consider an adversary who presents the on-line algorithm with several different instances. A lower bound obtained in this restricted situation is a lower bound on the competitive ratio in general [8,12,14].

LEMMA 2.1. *No algorithm A can achieve a competitive ratio less than 2 if it does not assign* p_i *to the machine* M_{m-i+1}, $i = 1, 2, \dots, m$.

PROOF. We prove the lemma by induction. First, job p_1 must be assigned to M_m, since otherwise, considering the instance with data $\{r_1 = r_2 = \cdots = r_{m-1} = 1,\ r_m = 0,\ p_1 = 1\}$, we have $C^A = 2$, $C^{\text{OPT}} = 1$, $C^A/C^{\text{OPT}} = 2$. Next, we assume that the result is true for all $i < l$. Then for $i = l$, consider the instance with data $\{r_1 = r_2 = \cdots = r_{m-l} = 1,\ r_{m-l+1} = \cdots = r_m = 0,\ p_1 = \cdots = p_l = 1\}$. It is clear that $C^{\text{OPT}} = 1$. If the algorithm A assigns p_l to any machine in $\{M_{m-l+2}, \dots, M_m\}$ along with another job in $\{p_1, \dots, p_{l-1}\}$, then $C^A = 2$, and hence, $C^A/C^{\text{OPT}} = 2$. If p_l is assigned to one of the machines in $\{M_1, \dots, M_{m-l}\}$, we also have $C^A = 2$, $C^A/C^{\text{OPT}} = 2$. So p_l should be assigned to M_{m-l+1}, and we are done. ∎

As all the lower bounds we will give below are less than 2, we assume that p_i is assigned to M_{m-i+1}, $i = 1, \dots, m$, in this section.

THEOREM 2.2. *For the two machine case, any algorithm A has a competitive ratio at least 3/2.*

PROOF. By Lemma 2.1, p_1 should be assigned to M_2, and p_2 should be assigned to M_1 in order to have a competitive ratio less than 2. By considering the instance with data $\{r_1 = 1,\ r_2 = 0,\ p_1 = p_2 = 1/2\}$, we have $C^A = 3/2$, $C^{\text{OPT}} = 1$, $C^A/C^{\text{OPT}} = 3/2$. ∎

THEOREM 2.3. *For the three machine case, any algorithm A has a competitive ratio at least 13/8.*

PROOF. Consider the assignment of the first seven jobs. If A assigns at least three jobs to M_3, consider the instance with data $\{r_1 = r_2 = r_3 = 0,\ p_1 = 1,\ p_2 = \cdots = p_7 = 1/3\}$. We have $C^A \geq p_1 + p_6 + p_7 = 5/3$, $C^{\text{OPT}} = 1$, and $C^A/C^{\text{OPT}} \geq 5/3 > 13/8$. If A assigns at least four jobs to M_2, consider the instance with data $\{r_1 = r_2 = r_3 = 0,\ p_1 = p_2 = 1,\ p_3 = \cdots = p_7 = 1/2\}$. We have $C^A \geq p_2 + p_5 + p_6 + p_7 = 5/2$, $C^{\text{OPT}} = 3/2$, and $C^A/C^{\text{OPT}} = 5/3 > 13/8$. So there must be at least $7 - 2 - 3 = 2$ jobs of $\{p_1, \dots, p_7\}$ assigned to M_1. At this moment, consider the instance with data $\{r_1 = 1,\ r_2 = r_3 = 0,\ p_1 = p_2 = p_3 = 1/2,\ p_4 = \cdots = p_7 = 1/8\}$, we have $C^A \geq r_1 + p_3 + p_7 = 13/8$, $C^{\text{OPT}} = 1$, and thus, $C^A/C^{\text{OPT}} \geq 13/8$. ∎

THEOREM 2.4. *For the four machine case, any algorithm A has a competitive ratio at least 5/3.*

PROOF. Consider the assignment of the first eight jobs. In order to achieve competitive ratio no greater than 5/3, no job in $\{p_5, \dots, p_8\}$ can be assigned to M_1, since otherwise, M_1 processes at least two jobs, i.e., p_4 and one in $\{p_5, \dots, p_8\}$. By considering the instance with data $\{r_1 = 1,\ r_2 = r_3 = r_4 = 0,\ p_1 = \cdots = p_4 = 1/2,\ p_5 = \cdots = p_8 = 1/4\}$, we have $C^A \geq r_1 + p_4 + p_8 = 7/4$, $C^{\text{OPT}} = 1$, and $C^A/C^{\text{OPT}} \geq 7/4 > 5/3$. Next, if there are at least two more jobs which are assigned to M_3 (or M_4) along with p_2 (or p_1), we consider the instance with data $\{r_1 = \cdots = r_4 = 0,\ p_1 = p_2 = 1,\ p_3 = \cdots = p_8 = 1/3\}$. It deduces that $C^A \geq r_1 + p_2 + p_7 + p_8 = 5/3$, $C^{\text{OPT}} = 1$, and $C^A/C^{\text{OPT}} \geq 5/3$. Hence, we conclude that there must be $8 - 1 - 2 - 2 = 3$ jobs of $\{p_1, \dots, p_8\}$ on M_2. Now we consider the instance with data $\{r_1 = r_2 = 1,\ r_3 = r_4 = 0,\ p_1 = \cdots = p_8 = 1/4\}$, it follows that $C^A \geq r_2 + p_3 + p_7 + p_8 = 7/4$, $C^{\text{OPT}} = 1$, and $C^A/C^{\text{OPT}} \geq 7/4 > 5/3$. ∎

THEOREM 2.5. *For the* $m \geq 5$ *machine case, any algorithm A has a competitive ratio at least 5/3.*

PROOF. Let j_1 denote $\lfloor (m-1)/2 \rfloor$, and j_2 denote $\lfloor m/2 \rfloor$ for simplicity. To show the result, we consider the assignment of job set $\{p_{m+1}, \dots, p_{2m}\}$. If there is at least one job in $\{p_{m+1}, \dots, p_{2m}\}$ which is assigned to machine M_j, $1 \leq j \leq j_1$, we consider the instance with data $\{r_1 = \cdots = r_j = 1,\ r_{j+1} = \cdots = r_m = 0,\ p_1 = \cdots = p_{m-j+1} = 1/2,\ p_{m-j+2} = \cdots = p_{2m} = 1/6\}$. It is not difficult to prove that $C^{\text{OPT}} = 1$ while $C^A \geq 1 + 1/2 + 1/6 = 5/3$. If there are at least two jobs in $\{p_{m+1}, \dots, p_{2m}\}$ which are assigned to machine M_j, $m - j_2 + 1 \leq j \leq m$, we consider the instance with data $\{r_1 = \cdots = r_m = 0,\ p_1 = \cdots = p_{j_2} = 1,\ p_{j_2+1} = \cdots = p_{2m} = 1/3\}$. It follows that $C^A/C^{\mathrm{OPT}} \geq 5/3$. Hence, we claim that machine M_{j_1+1} must process p_{m-j}, and at least $m - j_2$ jobs in $\{p_{m+1}, \dots, p_{2m}\}$. At this moment, we consider the instance with $2m$ jobs where $r_1 = \cdots = r_m = 0$, $p_1 = \cdots = p_{m-j_1} = 1$, and exact values of p_i, $m - j_1 + 1 \leq i \leq 2m$, are chosen arbitrarily such that $\sum_{i=m-j_1+1}^{m+j_2} p_i = j_1 - 2/3$, $\sum_{i=m+j_2+1}^{2m} p_i = 2/3$, and $C^{\mathrm{OPT}} = 1$. It is easy to verify that such an instance exists. Hence, we have $C^A \geq 5/3$ and $C^A/C^{\mathrm{OPT}} \geq 5/3$. ∎

In the following, we present ordinal algorithms to solve the discussed problem. First we give an ordinal algorithm $O2$ for a two machine case with competitive ratio 3/2, so it is optimal.

ALGORITHM $O2$.
Assign jobs in the subset $\{p_{2i-1} \mid i \geq 1\}$ to M_2;
Assign jobs in the subset $\{p_{2i} \mid i \geq 1\}$ to M_1.

THEOREM 2.6. *The competitive ratio of $O2$ is 3/2, and it is the optimal algorithm for* $P2, r_j|\text{ordinal on-line}|C_{\max}(M)$.

PROOF. We show the case of $n = 2l$; the case of $n = 2l + 1$ can be shown similarly. Obviously, $C^{\mathrm{OPT}} \geq r_1$, $C^{\mathrm{OPT}} \geq (b + r_1 + r_2)/2$, $C^{\mathrm{OPT}} \geq r_2 + p_1$, and $C^{O2} = \max\{\alpha_1, \alpha_2\}$. By the definition of the algorithm, we have

$$\alpha_1 = r_1 + \sum_{i=1}^{l} p_{2i} \leq r_1 + \frac{b}{2} \leq \frac{r_1}{2} + \frac{b + r_1}{2} \leq \frac{3}{2} C^{\mathrm{OPT}},$$

$$\alpha_2 = r_2 + \sum_{i=1}^{l} p_{2i-1} \leq r_2 + p_1 + \frac{1}{2} \sum_{i=2}^{l} p_i \leq \frac{p_1 + r_2}{2} + \frac{b + r_2}{2} \leq \frac{3}{2} C^{\mathrm{OPT}}.$$

Hence, the theorem is proved from Theorem 2.2. ∎

Next, we present an ordinal algorithm Om for general $m \geq 3$ machine cases.

ALGORITHM Om.
Assign jobs in the subset $\{p_{m+l(2m-2)} \mid l \geq 0\}$ to M_1;
Assign jobs in the subset $\{p_{m-j+1+l(m-1)} \mid l \geq 0\}$ to M_j, $2 \leq j \leq m - 1$;
Assign jobs in the subset $\{p_{1+l(2m-2)} \mid l \geq 0\}$ to M_m.

THEOREM 2.7. *The competitive ratio of Om for* $Pm, r_j|\text{ordinal on-line}|C_{\max}(M)$ *is*

$$\begin{array}{ll} 13/8, & m = 3, \\ 2 - \dfrac{1}{(m-1)}, & m \geq 4. \end{array}$$

PROOF. We only prove the case of $n = (2m-2)l$; other cases of $n = (2m-2)l + 1, \dots, (2m-2)l + l - 1$ can be proved similarly. Obviously, $C^{\mathrm{OPT}} \geq r_1$, $C^{\mathrm{OPT}} \geq (b + r_1 + \cdots + r_m)/m$, $C^{\mathrm{OPT}} \geq r_m + p_1$, and $C^{Om} = \max\{\alpha_1, \dots, \alpha_m\}$. From $p_{2+(i-1)(2m-2)} \geq \cdots \geq p_{i(2m-2)} \geq p_{1+i(2m-2)}$, we have

$$\begin{aligned} \alpha_m &= r_m + p_1 + \sum_{i=1}^{l-1} p_{1+i(2m-2)} \leq r_m + p_1 + \frac{1}{2m-2} \sum_{i=2}^{n} p_i \\ &= r_m + \frac{2m-3}{2m-2} p_1 + \frac{b}{2m-2} \leq \frac{3}{2} C^{\mathrm{OPT}}. \end{aligned}$$

Similarly, for $1 < j < m$, we have

$$\begin{aligned}
\alpha_j &= r_j + p_{m-j+1} + \sum_{i=1}^{2l-1} p_{m-j+1+i(m-1)} \\
&\le r_j + \frac{j-2}{m-1}p_{m-j+1} + \frac{m-j+1}{m-1}p_{m-j+1} + \frac{1}{m-1}\sum_{i=m-j+2}^{n} p_i \\
&\le \frac{m-j-1}{m-1}r_j + \frac{j-2}{m-1}p_{m-j+1} + \frac{m}{m-1}\frac{b+r_1+\cdots+r_j}{m} \\
&\le \left(\frac{m-1-j}{m-1} + \frac{j-2}{m-1} + \frac{m}{m-1}\right)C^{\text{OPT}} = \left(2 - \frac{1}{m-1}\right)C^{\text{OPT}}
\end{aligned}$$

and

$$\begin{aligned}
\alpha_1 &= r_1 + p_m + \sum_{i=1}^{l-1} p_{m+i(2m-2)} \\
&\le r_1 + \frac{m-2}{2m-2}p_m + \frac{m}{2m-2}p_m + \frac{1}{2m-2}\sum_{i=m+1}^{n} p_i \\
&= \frac{2m-3}{2m-2}r_1 + \frac{m-2}{2m-2}p_m + \frac{m}{2m-2}\frac{b+r_1}{m}.
\end{aligned}$$

For $m = 3$, it is clear from above that $\alpha_2 \le 3C^{\text{OPT}}/2 \le 13C^{\text{OPT}}/8$, $\alpha_3 \le 3C^{\text{OPT}}/2 \le 13C^{\text{OPT}}/8$, and $\alpha_1 = 3r_1/4 + p_3/4 + (b+r_1)/4$. Furthermore, if $p_3 \le C^{\text{OPT}}/2$, then $\alpha_1 \le (3/4 + 1/8 + 3/4)C^{\text{OPT}} = 13C^{\text{OPT}}/8$. If $r_1 \le 5C^{\text{OPT}}/6$, then $\alpha_1 \le (5/8+1/4+3/4)C^{\text{OPT}} = 13C^{\text{OPT}}/8$. Now we are left to consider the case of $p_3 > C^{\text{OPT}}/2$ and $r_1 > 5C^{\text{OPT}}/6$. At this moment, we have $2p_3 > C^{\text{OPT}}$ and $r_1 + p_3 > C^{\text{OPT}}$. But $r_1 + p_3 > C^{\text{OPT}}$ states that p_1, p_2, p_3 must be assigned to M_2, M_3 in optimum, which in turn violates $p_2 + p_3 \ge 2p_3 > C^{\text{OPT}}$. Hence, we have shown $C^{Om}/C^{\text{OPT}} \le 13/8$ for $m = 3$. Theorem 2.3 shows that the ratio of Om could not be less than $3/2$ for $m = 3$.

For $m \ge 4$, we also only need to consider the ratio between α_1 and C^{OPT}. If $p_m \le (m-3)C^{\text{OPT}}/(m-2)$, then

$$\alpha_1 \le \left(\frac{2m-3}{2m-2} + \frac{m-2}{2m-2}\frac{m-3}{m-2} + \frac{m}{2m-2}\right)C^{\text{OPT}} = \left(2 - \frac{1}{m-1}\right)C^{\text{OPT}}.$$

If $r_1 \le (2m-4)C^{\text{OPT}}/(2m-3)$, then

$$\alpha_1 \le \left(\frac{2m-3}{2m-2}\frac{2m-4}{2m-3} + \frac{m-2}{2m-2} + \frac{m}{2m-2}\right)C^{\text{OPT}} = \left(2 - \frac{1}{m-1}\right)C^{\text{OPT}}.$$

Hence, we assume $p_m > (m-3)C^{\text{OPT}}/(m-2)$ and $r_1 > (2m-4)C^{\text{OPT}}/(2m-3)$. We have $2p_m > C^{\text{OPT}}$ and $r_1 + p_m > C^{\text{OPT}}$. We still have no way to assign $\{p_1, \ldots, p_m\}$ in the optimum. This is the desired contradiction.

To prove that the ratio of Om could not be less than $2 - 1/(m-1)$ for $m \ge 4$, consider the instance with data $\{r_1 = r_2 = 1,\ r_3 = \cdots = r_m = 0,\ p_1 = \cdots = p_{(m-2)\cdot(m-1)} = 1/(m-1)\}$. Clearly, we have $C^{\text{OPT}} = 1$, $C^{Om} \ge \alpha_2 = r_2 + (m-2)/(m-1) = 2 - 1/(m-1)$, and $C^{Om}/C^{\text{OPT}} \ge 2 - 1/(m-1)$. ∎

By Theorems 2.3, 2.4, and 2.7, we have the following result.

COROLLARY 2.8. *Algorithm Om is optimal for $m = 3, 4$.* ∎

3. $P, r_j|\text{ordinal on-line}|C_{\max}(J)$

In this section, we consider the objective of minimizing the latest job completion time.

THEOREM 3.1. *Of any ordinal algorithm for* $Pm, r_j|\text{ordinal on-line}|C_{\max}(J)$*, the competitive ratio is at least* m.

PROOF. We prove the result by adversary method, too. If an algorithm A assigns all jobs to M_m, we consider the instance with data $\{r_1 = \cdots = r_{m-1} = 1,\ r_m = 0,\ p_1 = \cdots = p_m = k\}$. We have $C^A = mk$, $C^{\text{OPT}} = k+1$, and $C^A/C^{\text{OPT}} \to m$ $(k \to \infty)$. Otherwise, assume that p_l is the first job to be assigned to one of the first $m-1$ machines. We consider the instance with data $\{r_1 = \cdots = r_{m-1} = 1,\ r_m = 0,\ p_1 = \cdots = p_l = 1/(ml)\}$. We have $C^A = 1 + 1/(ml)$, $C^{\text{OPT}} = 1/m$, and $C^A/C^{\text{OPT}} \geq m$. ∎

It is quite easy to get an optimal algorithm for this objective. In fact, the following Algorithm OJ can reach the goal.

ALGORITHM OJ.
Assign all the jobs to the machine with shortest available time, i.e., M_m.

THEOREM 3.2. *The competitive ratio of Algorithm* OJ *is* m*, and it is the optimal algorithm for* $Pm, r_j|\text{ordinal on-line}|C_{\max}(J)$.

PROOF. It is clear that M_m must be assigned at least one job in the optimum. Hence, $C^{OJ} = \alpha_m = r_m + b \leq m \cdot (r_m + b)/m \leq mC^{\text{OPT}}$. ∎

In the remainder of this section, we discuss a special case of the problem. We assume that machines $M_{m-l+1}, \dots, M_m$, $1 \leq l < m$, have the same available time. This situation may occur in many applications. For example, the available times of all machines have only two possible values, or only partial machines are required to prepare due to preventive maintenance in each periodic basis. Without loss of generalization, we can assume that $r_1 \geq \cdots \geq r_{m-l} > r_{m-l+1} = \cdots = r_m$. We are to propose an algorithm whose competitive ratio is significantly better than m.

THEOREM 3.3. *The competitive ratio of any algorithm* A *for the special case is at least* $\max\{2, m/l\}$.

PROOF. Similar to the proof of Theorem 3.1, A cannot assign any job to the machines $M_1, \dots, M_{m-l}$, in order to achieve a competitive ratio less than m. Next, consider the instance with data $\{r_1 = \cdots = r_{m-l} > 0,\ r_{m-l+1} = \cdots = r_m = 0,\ p_1 = \cdots = p_m = k\}$. Because $l < m$, we have $C^A \geq 2k$, $C^{\text{OPT}} = r_1 + k$, and $C^A/C^{\text{OPT}} \to 2$ $(k \to \infty)$. Moreover, by considering the instance with data $\{r_1 = \cdots = r_{m-l} > 0,\ r_{m-l+1} = \cdots = r_m = 0,\ p_1 = \cdots = p_{ml} = k\}$, we have $C^A \geq mk$, $C^{\text{OPT}} = r_1 + lk$, and $C^A/C^{\text{OPT}} \to m/l$ $(k \to \infty)$. We are done. ∎

The following algorithm MPm (*Modified* Pm) is modified from Algorithm Pm which is proposed to solve the problem $Pm|\text{ordinal on-line}|C_{\max}$ [8]. MPm assigns all the jobs to l machines $\{M_{m-l+1}, \dots, M_m\}$ according to Pm.

ALGORITHM MPm.
Assign jobs in the subset $\{p_j\} \cup \{p_{2l+1-j+k(l+\lceil l/2\rceil)} \mid k \geq 0\}$ to M_{m-l+j}, $1 \leq j \leq \lfloor l/2 \rfloor$;
Assign jobs in the subset $\{p_j\} \cup \{p_{2l+1-j+k(l+\lceil l/2\rceil)} \mid k \geq 0\} \cup \{p_{3l+1-j+k(l+\lceil l/2\rceil)} \mid k \geq 0\}$ to M_{m-l+j}, $\lfloor l/2 \rfloor + 1 \leq j \leq l$.

THEOREM 3.4. *The competitive ratio of* MPm *for the above special case is*

$$\begin{cases} \max\left\{\dfrac{m-l+4}{3}, \dfrac{4m+2l-2}{3l+1}\right\}, & l \text{ is odd}, \\[2ex] \max\left\{\dfrac{m-l+4}{3}, \dfrac{4m+2l-4}{3l}\right\}, & l \text{ is even}. \end{cases}$$

PROOF. We first claim that we can assume

$$r_{m-l+1} = r_{m-l+2} = \cdots = r_m = 0. \tag{$*$}$$

Otherwise, let (M', J) be a new instance modified from (M, J). They have the same job set, and their machine available times satisfy $r'_i = r_i - r_m$, $1 \le i \le m$, where r'_i and r_i denote the machine available times of $M'_i \in M'$ and $M_i \in M$, respectively. If we have already proved that $C^{MPm}(M', J) \le cC^{\text{OPT}}(M', J)$ for all instances satisfying $(*)$, then $C^{MPm}(M, J) = C^{MPm}(M', J) + r_m \le cC^{\text{OPT}}(M', J) + r_m \le c(C^{\text{OPT}}(M', J) + r_m) = cC^{\text{OPT}}(M, J)$. So the ratio is also true for (M, J). In the rest of the proof, we always assume that $(*)$ is valid, and thus, the following lemma is true.

LEMMA 3.5. *(See [8].)*

$$\alpha_{m-l+j} - p_j \le \begin{cases} \dfrac{2}{3(l-j+1)} \sum\limits_{i=j+1}^{n} p_i, & l \text{ is even and } 1 \le j \le l-1 \\ & \text{or } l \text{ is odd and } \dfrac{(l+3)}{2} \le j \le l-1, \\ \dfrac{2}{3(l-j+1)+1} \sum\limits_{i=j+1}^{n} p_i, & l \text{ is odd and } 1 \le j \le \dfrac{(l-1)}{2}. \end{cases}$$

It is trivial that $C^{MPm} = \max_{j=1,\dots,l}\{\alpha_{m-l+j}\}$. To prove the theorem, we distinguish five cases according to the values of l and j.

CASE 1. l is even and $1 \le j \le l-1$ or l is odd and $(l+3)/2 \le j \le l-1$.

By Lemma 3.5,

$$\begin{aligned} \alpha_{m-l+j} \le p_j + \frac{2}{3(l-j+1)} \sum_{i=j+1}^{n} p_i \le p_j + \frac{2(b - jp_j)}{3(l-j+1)} \\ = \frac{3l - 5j + 3}{3(l-j+1)} p_j + \frac{2m}{3(l-j+1)} \frac{b}{m} \\ \le \frac{2m + 3l - 5j + 3}{3(l-j+1)} C^{\text{OPT}}. \end{aligned}$$

Since $\{(2m + 3l - 5j + 3)/(3(l - j + 1))\}_{j=1}^{l-1}$ achieves its maximum at $j = l-1$, we have $\alpha_{m-l+j} \le ((m - l + 4)/3)C^{\text{OPT}}$.

CASE 2. l is odd and $1 \le j \le (l-1)/2$.

Similarly to Case 1, by Lemma 3.5, we have

$$\alpha_{m-l+j} \le \frac{2m + 3l - 5j + 4}{3(l-j+1)+1} C^{\text{OPT}} \le \begin{cases} \dfrac{5l+1}{3l+1} C^{\text{OPT}}, & \text{if } m = l+1, \\ \dfrac{4m + l + 13}{3l + 11} C^{\text{OPT}}, & \text{if } m > l+1. \end{cases}$$

CASE 3. l is even and $j = l$.

We only prove the case of $n = l + 1 + 3lk/2$; other cases of n can be proved similarly.

$$\begin{aligned} \alpha_m &= p_l + \sum_{s=1}^{k-1} p_{l+1+3ls/2} + \sum_{s=1}^{k-1} p_{2l+1+3ls/2} \\ &\le \frac{2}{l+1} \sum_{i=1}^{l+1} p_i + \frac{1}{l} \sum_{s=1}^{k-1} \sum_{i=l+2}^{2l+1} p_{i+3ls/2} + \frac{2}{l} \sum_{s=1}^{k-1} \sum_{i=2l+2}^{(5l)/2+1} p_{i+3ls/2} \\ &\le \frac{b}{l} + \frac{l-1}{l(l+1)} \sum_{i=1}^{l+1} p_i + \frac{1}{l} \sum_{s=1}^{k-1} \sum_{i=2l+2}^{(5l)/2+1} p_{i+3ls/2} \\ &\le \frac{b}{l} + \frac{l-1}{l(l+1)} \sum_{i=1}^{l+1} p_i + \frac{1}{3l} \sum_{i=l+2}^{n} p_i \\ &\le \frac{4b}{3l} + \frac{2l-4}{3l} p_1 \le \frac{4m + 2l - 4}{3l} C^{\text{OPT}}. \end{aligned}$$

CASE 4. l is odd and $j = l$.

Similar to Case 3, we have $\alpha_m \le ((4m + 2l - 2)/3l + 1)C^{\mathrm{OPT}}$.

CASE 5. l is odd and $j = (l + 1)/2$.

Similar to Case 3, we have $\alpha_{m-(l-1)/2} \le ((6m + 2l - 2)/5l + 1)C^{\mathrm{OPT}}$.

By straight arithmetic calculation, one can easily know that the theorem is true. ∎

By Theorems 3.3 and 3.4, we have the following result.

COROLLARY 3.6. *If $m = l+1$, the competitive ratio of MPm is 2 and the algorithm is optimal.* ∎

Though our algorithm may not be optimal when $m > l + 1$, it seems difficult to improve the algorithm unless we get a better algorithm for $P|$ordinal on-line$|C_{\max}$.

4. CONCLUSIONS

In this paper, we discussed the ordinal on-line scheduling problem on m identical machines with nonsimultaneous machine available times. Two different objectives were considered. For minimizing latest machine completion time, we presented lower bounds and ordinal algorithms. The algorithms are optimal for $m = 2, 3, 4$. For minimizing latest job completion time, we gave an optimal ordinal algorithm with competitive ratio m. We further presented a special case which has an algorithm with significantly improved competitive ratio. One of the remarkable phenomena is that approximation algorithms and their competitive ratios for these two objectives are significantly different in $P, r_j|$ordinal on-line$|C_{\max}$, which is not found in other scheduling problems, such as $P \parallel C_{\max}$, no matter what is on-line or off-line. Even for the off-line problem $P, r_j \parallel C_{\max}$, every known algorithm has the same worst-case ratio for these two objectives so far.

REFERENCES

1. R.L. Graham, E.L. Lawler, J.K. Lenstra and A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: A survey, *Annals of Operations Research* **5**, 287–326, (1979).
2. W.P. Liu and J.B. Sidney, Ordinal algorithm for packing with target center of gravity, *Order* **13**, 17–31, (1996).
3. C.Y. Lee, Parallel machine scheduling with non-simultaneous machine available time, *Discrete Applied Mathematics* **30**, 53–61, (1991).
4. C.Y. Lee, Y. He and G. Tang, A note on "Parallel machine scheduling with non-simultaneous machine available time", *Discrete Applied Mathematics* **100**, 133–135, (2000).
5. E.L. Lawler, *Combinatorial Optimization: Networks and Matroids*, Holt, Rinehart and Winston, Toronto, (1976).
6. A. Agnetis, No-wait flow shop scheduling with large lot size, Rap 16.89, Dipartimento di Informatica e Sistemistica, Universita Degli Studi di Roma "La Sapienza", Rome, Italy, (1989).
7. W.P. Liu and J.B. Sidney, Bin packing using semi-ordinal data, *Operations Research Letters* **19**, 101–104, (1996).
8. W.P. Liu, J.B. Sidney and A. van Vliet, Ordinal algorithms for parallel machine scheduling, *Operations Research Letters* **18**, 223–232, (1996).
9. Y. Azar and O. Regev, Online bin stretching, *Proc. of RANDOM '98*, 71–82, (1998).
10. Y. He, The optimal on-line parallel machine scheduling, *Computers Math. Applic.* **39** (7/8), 117–121, (2000).
11. Y. He and G.C. Zhang, Semi on-line scheduling on two identical machines, *Computing* **62**, 179–187, (1999).
12. H. Kellerer, V. Kotov, M.G. Speranza and Z. Tuza, Semi on-line algorithms for the partition problem, *Operations Research Letters* **21**, 235–242, (1997).
13. S. Seiden, J. Sgall and G. Woeginger, Semi-online scheduling with decreasing job sizes, *Operations Research Letters* **27**, 215–227, (2000).
14. D. Sleator and R.E. Tarjan, Amortized efficiency of list update and paging rules, *Communications of the ACM* **28**, 202–208, (1985).