Optimal On-line Algorithms for the Uniform Machine Scheduling Problem with Ordinal Data

Zhiyi Tan * Yong He † Leah Epstein ‡

Abstract

In this paper, we consider an ordinal on-line scheduling problem. A sequence of n independent jobs has to be assigned non-preemptively to two uniformly related machines. We study two objectives which are maximizing the minimum machine completion time, and minimizing the l_p norm of the completion times. It is assumed that the values of the processing times of jobs are unknown at the time of assignment. However it is known in advance that the processing times of arriving jobs are sorted in a non-increasing order. We are asked to construct an assignment of all jobs to the machines at time zero, by utilizing only ordinal data rather than actual magnitudes of jobs. For the problem of maximizing the minimum completion time we first present a comprehensive lower bound on the competitive ratio, which is a piecewise function of machine speed ratio s. Then we propose an algorithm which is optimal for any $s \geq 1$. For minimizing the l_p norm we study the case of identical machines ($s = 1$) and present tight bounds as a function of p.

Mathematics Subject Classification. 90B35, 90C27

Keywords: analysis of algorithm, scheduling, semi-online, competitive ratio.

1 Introduction

In this paper, we consider the following scheduling problem. Jobs are to be assigned to uniformly related machines. The objective is either maximizing the minimum machine completion time (also called "machine covering") or minimizing the l_p norm of the completion times (also called "scheduling in the l_p norm"). We are confronted with a sequence of independent jobs $p_1, p_2, \ldots, p_n$ each with a non-negative processing time, which must be scheduled non-preemptively on one of two uniformly related machines M_1 and M_2. We identify the jobs with their processing times. Machine M_1 has speed $s_1 = 1$ and machine M_2 has speed $s_2 = s \geq 1$. If p_i is assigned to machine M_j, then p_i/s_j time units are required to process this job. Machines are available at time zero. In the *on-line* version of the problem, we assume that jobs arrive

*Dept. of Mathematics, State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, P. R. China. tanzy@zju.edu.cn. Research supported by the National Natural Science Foundation of China (10301028).

†Dept. of Mathematics, State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, P. R. China. mathhey@zju.edu.cn. Research supported by the Teaching and Research Award Program for Outstanding Young Teachers in Higher Education Institutions of MOE, China, and National Natural Science Foundation of China (10271110, 60021201).

‡Corresponding author. School of Computer Science, The Interdisciplinary Center, P.O. Box 167, 46150 Herzliya, Israel. lea@idc.ac.il. Research supported in part by Israel Science Foundation (grant no. 250/01).

one by one and must be assigned to a machine immediately upon arrival. The decision cannot be changed later, when subsequent jobs become available. Furthermore we consider the problem under the *ordinal data* scenario: the values of the processing times are unknown but the sorted order of the jobs according to their processing times is known in advance. Accordingly we suppose $p_1 \geq p_2 \geq \cdots \geq p_n$. We are asked to create an assignment of all jobs at time zero by utilizing only ordinal (rank) data rather than the actual magnitudes. We denote this problem as $Q2|ordinal\ on-line|C_{\min}$.

Scheduling, given the goal of maximizing the minimum machine completion time, has applications in the sequencing of maintenance actions for modular gas turbine aircraft engines [10] and was deeply studied for last two decades [5, 4, 22, 2]. But to the best knowledge of the authors, very few papers considered the case of uniformly related machines. The only such paper we are aware of is [2], where semi-online versions with known optimal value and non-increasing job processing times were discussed.

Scheduling in the l_p norm was first presented in [3] where semi-online scheduling on identical machines is studied. The l_2 norm measure has applications in computation of the average delay in disk access of jobs. On-line scheduling on identical machines in the l_p norm was studied in [1]. In that paper (among other results) the tight bound for two identical machines is given for every value of p.

Scheduling problems and algorithms for them, which utilize only ordinal data rather than actual magnitudes, are called *ordinal* [17], and have many real world applications. Though the exact values of processing times of jobs are unknown, the additional knowledge on their relative order is useful to derive algorithms with good approximation performance. This is the reason why we also say that the problem we study is actually semi-online. The notion of semi-online was defined to be a relaxation of some on-line problem [13]. Ordinal algorithms are particularly important in practical applications where it is nearly impossible to know the exact value of a processing time of a job in advance, due to cost, time, or material property. However, the comparison of two jobs in such situations is relatively simple. Another possibility is that the processing times of jobs are flexible or easily disturbed, while the relative order remains unchanged. Under these conditions or some other circumstances, we prefer to use an ordinal algorithm rather than an algorithm which depends on the exact values of processing times, such as LPT. Note that ordinal algorithms are known to be able to achieve more robustness than LPT [18, 12]. Due to their various applications, ordinal problems and algorithms had been studied also in many other classical combinatorial optimization problems, such as matroids [14], bin-packing [16], and packing [15].

Competitive analysis is a type of worst-case analysis where the performance of an on-line (or a semi-online) algorithm is compared to that of the optimal off-line algorithm [19]. For an on-line (semi-online) algorithm A, let $C^A(J)$ (C^A for short) denote the minimum machine completion time of instance J produced by algorithm A, and $C^{OPT}(J)$ (C^{OPT} for short) denote the optimal value in an off-line version. Then the competitive ratio of algorithm A is defined as the smallest number c such that $cC^A \geq C^{OPT}$ for all instances. An on-line (semi-online) algorithm A is called *optimal* if there is no on-line (semi-online)

algorithm for the discussed problem with competitive ratio smaller than that of A. The combination of an on-line (semi-online) algorithm and a negative result showing that the algorithm is optimal, allows us to find the best competitive ratio for the problem. The competitive ratio of an optimal on-line (semi-online) algorithm is called *a tight bound*. Moreover, for scheduling problems on two uniformly related machines, we see both the competitive ratio and the lower bound as functions of speed ratio s. Algorithm A is called *parametrically optimal*, if the two above functions match for any $s \geq 1$. We are interested in finding the tight bound as a function of s.

For the goal of minimizing the l_p norm, given an algorithm A, let C^A denote the l_p norm of the machine completion times, and let C^{OPT} denote that value in an optimal off-line algorithm. Then the competitive ratio of algorithm A is defined as the smallest number c such that $C^A \leq cC^{OPT}$ for all instances.

A strongly related problem is ordinal on-line scheduling on parallel identical machines with the objective of maximizing the minimum machine completion time, denoted by $Pm|ordinal\ on-line|C_{\min}$. He and Tan [12] presented an algorithm with competitive ratio no greater than $\lceil \sum_{i=1}^{m} \frac{1}{i} \rceil + 1$ while the lower bound is $\sum_{i=1}^{m} \frac{1}{i}$ for general m machine case. Both are on the order of $\Theta(\ln m)$. Moreover, for the special case of $m = 2, 3$, optimal algorithms were presented in [11, 12]. The tight bound for two machines is $3/2$ which is a special case of our results. For minimizing the makespan using an ordinal algorithm, [17] showed that the tight bound for two machines is $4/3$.

Another strongly related problem is on-line (semi-online) scheduling problem on two uniformly related machines with the objective of minimizing the makespan, denoted by $Q2||C_{\max}$. Epstein et al. [9] showed LS is a parametric optimal algorithm for the on-line version and presented randomized algorithms with smaller competitive ratios. Tan and He [21] presented an algorithm for the ordinal on-line version. It is optimal for the majority of values of $s \in [1, \infty)$. The total length of the intervals of s where the competitive ratio does not match the lower bound is less than 0.7784 and the biggest gap between them is under 0.0521. Tan and He [20] also considered algorithms for other two semi-online versions where the total processing time of jobs is known in advance, or the largest job processing time is known in advance, respectively. Epstein and Favrholdt [7, 8] considered a semi-online version where jobs arrive in non-increasing order, for both the preemptive and the non-preemptive cases, parametric optimal algorithms are proposed. Recently, Epstein [6] considered a generalization of on-line bin stretching problem, which can also be viewed as a semi-online scheduling problem with known optimal makespan on two uniformly related machines. For the preemptive version, she gave an optimal algorithm with competitive ratio 1, and for the non-preemptive version she gave an algorithm with largest gap between the competitive ratio and lower bound less than 0.073.

In this paper, we propose a parametric optimal algorithm for $Q2|ordinal\ on-line|C_{\min}$. In Section 2,

we will prove that the parametric lower bound $c(s)$ is as follows

$$c(s) = \begin{cases} \frac{2(2k-1)s}{ks+2(k-1)}, & a_{k-1} \leq s < b_k, \ \ k \geq 2, \\ \frac{2k+1}{k+1}, & b_k \leq s < a_k, \ \ k \geq 2, \\ 2, & s \geq 2, \end{cases}$$

where $a_k = \frac{2k}{k+1}$ and $b_k = \frac{2(k-1)(2k+1)}{2k^2+k-2}$. In Section 3, we present an ordinal algorithm $QOrdinal\_Min$ and prove that its competitive ratio matches the lower bound for any $s \geq 1$. Thus $QOrdinal\_Min$ is a parametric optimal algorithm for $Q2|ordinal\ on-line|C_{\min}$. The graph of $c(s)$ is shown in Figure 1. Note that $c(s) \leq 2$ for all values of s.

In Section 4 we consider scheduling in the l_p norm on two identical machines (i.e. $s = 1$). We give a simple algorithm and compute its competitive ratio as a function of p, then we design matching lower bounds.

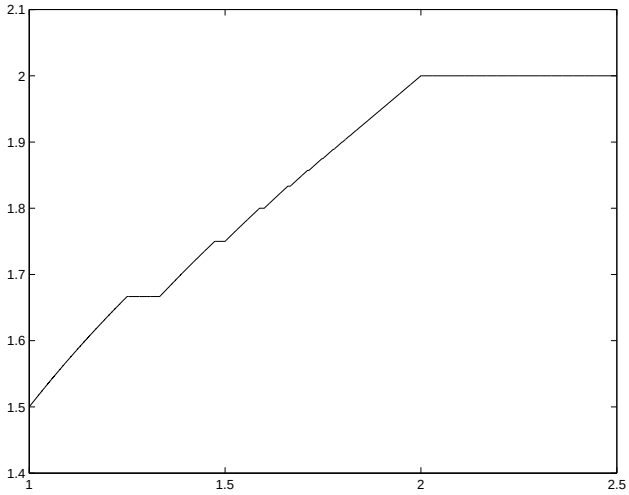

Figure 1: The parametric optimal bound for $Q2|ordinal\ on-line|C_{\min}$

2 Lower bound

In this section, we present the parametric lower bound for $Q2|ordinal\ on-line|C_{min}$, which is stated in the following theorem.

Theorem 2.1 *Any algorithm A has a competitive ratio at least*

$$c(s) = \begin{cases} \min\{\frac{2(2k-1)s}{ks+2(k-1)}, \frac{2k+1}{k+1}\} = \begin{cases} \frac{2(2k-1)s}{ks+2(k-1)} & a_{k-1} \leq s < b_k, \ \ k \geq 2, \\ \frac{2k+1}{k+1} & b_k \leq s < a_k, \ \ k \geq 2, \end{cases} \\ 2, \qquad\qquad s \geq 2, \end{cases}$$

where $a_k = \frac{2k}{k+1}$ is the root of the equation $\frac{2k+1}{k+1} = \frac{2(2k+1)x}{(k+1)x+2k}$, $b_k = \frac{2(k-1)(2k+1)}{2k^2+k-2}$ is the root of the equation $\frac{2(2k-1)x}{kx+2(k-1)} = \frac{2k+1}{k+1}$.

Since $a_1 = 1$, $a_{k-1} < b_k < a_k < b_{k+1}, k \geq 2$ and $a_k \to 2$ $(k \to \infty)$. The function $c(s)$ is well-defined for any $s \in [1, \infty)$.

The proof will be completed by using an adversarial method. All instances used in this section have optimal value 1, and thus $C^{OPT}/C^A = 1/C^A$. For easy reading and understanding, we show Theorem 2.1 by distinguishing several cases according to the value of s. We prove the case $s \in [2, \infty)$ in detail. The remaining cases of $s \in [1, 2)$ can be verified by essentially similar arguments, hence we sketch the proof by listing the schedules of algorithm A, and the adversarial sequences for all possible situations are given in Tables 1-3 case by case afterwards.

Lemma 2.1 *For $s \in [2, \infty)$, any algorithm for $Q2|ordinal\ on-line|C_{min}$ has competitive ratio of at least 2.*

Proof: Obviously, the first two jobs must be assigned to different machines by any algorithm A. Otherwise, consider the instance $p_1 = s$, $p_2 = 1$, we have $C^A = 0$ and thus $1/C^A = \infty$. Next, if algorithm A assigns p_1 to M_1 and p_2 to M_2, the above instance implies $C^A = 1/s$ and thus $1/C^A = s \geq 2$. Finally, if A assigns p_1 to M_2 and p_2 to M_1, consider the assignment of p_3. If A assigns p_3 to M_1, consider the instance $p_1 = p_2 = s/2$, $p_3 = 1$, we have $C^A = 1/2$, $1/C^A = 2$. Otherwise, consider the instance $p_1 = s$, $p_2 = p_3 = 1/2$, we also have $C^A = 1/2$, $1/C^A = 2$. □

We separate the analysis into two cases according to which machine receives the very first job. In Lemma 2.2, we prove that if algorithm A assigns p_1 to M_2, the competitive ratio of A is at least $c(s)$. In Lemma 2.3, we prove that if A assigns p_1 to M_1, the competitive ratio of A is at least $c(s)$. Combining Lemmas 2.2,2.3, we get the desired lower bound for any $s \in [1, 2)$.

Lemma 2.2 *If A assigns p_1 to M_2, the competitive ratio of A is at least $c(s)$ for any s with $1 \leq s < 2$.*

Proof: Table 1 implies that if A assigns p_1 to M_2, the competitive ratio of A is at least $\min\{3s/(s+1), 5/3\}$, which equals $c(s)$ for any s with $1 = a_1 \leq s < a_2 = 4/3$.

To prove the result for s with $a_{k-1} \leq s < a_k$, $k > 2$, we replace the last row of Table 1 with all $2k - 3$ rows of Table 2.

Since $2 \geq \frac{2(2l-1)s}{ls+2(l-1)} \geq \frac{2(2k-1)s}{ks+2(k-1)}$, $2 \leq l < k$, all values in the last column of the new table are greater than or equal to $\min\{\frac{2(2k-1)s}{ks+2(k-1)}, \frac{2k+1}{k+1}\}$ for any s with $a_{k-1} \leq s < a_k$. The lemma is thus proved. □

Lemma 2.3 *If A assigns p_1 to M_1, the competitive ratio of A is at least $c(s)$ for $1 \leq s < 2$.*

Proof: Consider Table 3. Note that row 4 is valid only for $s \leq 3/2$, while rows 5 and 10 are valid for $s > 3/2$. All other rows are valid for the complete interval $[1, 2)$.

Schedule by A		Adversary instance	$\frac{1}{C^A}$
M_1	M_2		
$\emptyset$	$\{p_1, p_2\}$	$\{s, 1\}$	∞
$\{p_2\}$	$\{p_1, p_3\}$	$\{s, \frac{1}{2}, \frac{1}{2}\}$	2
$\{p_2, p_3, p_4\}$	$\{p_1\}$	$\{\frac{s}{2}, \frac{s}{2}, \frac{1}{2}, \frac{1}{2}\}$	2
$\{p_2, p_3\}$	$\{p_1, p_4, p_5\}$	$\{s, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\}$	2
$\{p_2, p_3, p_5, p_6\}$	$\{p_1, p_4\}$	$\{\frac{s}{2}, \frac{s}{2}, \frac{s}{2}, \frac{2-s}{6}, \frac{2-s}{6}, \frac{2-s}{6}\}$	$\frac{3s}{s+1}$
$\{p_2, p_3, p_5\}$	$\{p_1, p_4, p_6\}$	$\{s, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}\}$	$\frac{5}{3}$

Table 1: The case $1 \leq s \leq 4/3$ for Lemma 2.2

$\{p_2, p_3, p_5\}$	$\{p_1, p_4, p_6, p_7\}$	$\{s, \frac{1}{6}, \cdots, \frac{1}{6}\}$	2
$\{p_2, p_3, p_5, p_7, p_8\}$	$\{p_1, p_4, p_6\}$	$\{\frac{s}{2}, \frac{s}{2}, \frac{s}{2}, \frac{2-s}{10}, \cdots, \frac{2-s}{10}\}$	$\frac{10s}{3s+4}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\{p_2, p_3, p_5, \cdots, p_{2l-1}\}$	$\{p_1, p_4, p_6, p_{2l}, p_{2l+1}\}$	$\{s, \frac{1}{2l}, \cdots, \frac{1}{2l}\}$	2
$\{p_2, p_3, p_5, \cdots, p_{2l+1}, p_{2l+2}\}$	$\{p_1, p_4, p_6, \cdots, p_{2l}\}$	$\{\frac{s}{2}, \frac{s}{2}, \frac{s}{2}, \frac{2-s}{2(2l-1)}, \cdots, \frac{2-s}{2(2l-1)}\}$	$\frac{2(2l-1)s}{ls+2(l-1)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\{p_2, p_3, p_5, \cdots, p_{2k-1}\}$	$\{p_1, p_4, p_6, p_{2k}, p_{2k+1}\}$	$\{s, \frac{1}{2k}, \cdots, \frac{1}{2k}\}$	2
$\{p_2, p_3, p_5, \cdots, p_{2k+1}, p_{2k+2}\}$	$\{p_1, p_4, p_6, \cdots, p_{2k}\}$	$\{\frac{s}{2}, \frac{s}{2}, \frac{s}{2}, \frac{2-s}{2(2k-1)}, \cdots, \frac{2-s}{2(2k-1)}\}$	$\frac{2(2k-1)s}{ks+2(k-1)}$
$\{p_2, p_3, p_5, \cdots, p_{2k-1}, p_{2k+1}\}$	$\{p_1, p_4, p_6, \cdots, p_{2k}, p_{2k+2}\}$	$\{s, \frac{1}{2k+1}, \cdots, \frac{1}{2k+1}\}$	$\frac{2k+1}{k+1}$

Table 2: The case $s > 4/3$ for Lemma 2.2

row	Schedule by A		Adversary instance	$\frac{1}{C^A}$
	M_1	M_2		
1	$\{p_1, p_2\}$	$\emptyset$	$\{s, 1\}$	∞
2	$\{p_1, p_3\}$	$\{p_2\}$	$\{s, \frac{1}{2}, \frac{1}{2}\}$	$2s$
3	$\{p_1, p_4\}$	$\{p_2, p_3\}$	$\{s, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\}$	$\frac{3s}{2}$
4	$\{p_1\}$	$\{p_2, p_3, p_4, p_5\}$	$\{\frac{1}{2}, \frac{1}{2}, \frac{s}{3}, \frac{s}{3}, \frac{s}{3}\}$	2
5	$\{p_1\}$	$\{p_2, p_3, p_4, p_5\}$	$\{\frac{s}{3}, \frac{s}{3}, \frac{s}{3}, \frac{1}{2}, \frac{1}{2}\}$	$\frac{3}{s}$
6	$\{p_1, p_5\}$	$\{p_2, p_3, p_4\}$	$\{s, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\}$	$\frac{4s}{3}$
7	$\{p_1, p_5, p_6\}$	$\{p_2, p_3, p_4\}$	$\{s, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}\}$	$\frac{5s}{3}$
8	$\{p_1, p_5, p_7\}$	$\{p_2, p_3, p_4, p_6\}$	$\{s, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}\}$	$\frac{3s}{2}$
9	$\{p_1, p_5\}$	$\{p_2, p_3, p_4, p_6, p_7\}$	$\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{s-1}{3}, \frac{s-1}{3}, \frac{s-1}{3}\}$	$\frac{6}{2s+1}$
10	$\{p_1\}$	$\{p_2, p_3, p_4, p_5, p_6\}$	$\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, s - \frac{3}{2}\}$	2
11	$\{p_1, p_6\}$	$\{p_2, p_3, p_4, p_5\}$	$\{s, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}\}$	$\frac{5s}{4}$

Table 3: Inputs for Lemma 2.3

If A assigns p_1 to M_1, the first four rows in Table 3 together with row 6 show that the competitive ratio of A is at least $4s/3$, and the first four rows together with rows 7-9 imply that the competitive ratio of A is at least $\min\{3s/2, 6/(2s+1)\}$. Thus the competitive ratio for $1 \leq s \leq 3/2$ is at least $q(s) = \max\{4s/3, \min\{3s/2, 6/(2s+1)\}\}$. It is not difficult to show that for $s \in [1, 3/2]$, $c(s) \leq 3s/(s+1)$, whereas in the same interval $q(s) \geq 3s/(s+1)$. This proves the lower bound for $1 \leq s \leq 3/2$.

For $3/2 < s \leq 1.6$ we use rows 1-3 and 5-6. We get a lower bound of $\min\{3/s, 4s/3\}$. In this interval $4s/3 \geq 2$ and $3/s \geq 15/8 \geq c(s)$.

For $s > 1.6$ we use rows 1-3,6,10,11. We get a lower bound of $\min\{5s/4, 2\} \geq 2 \geq c(s)$, therefore Lemma 2.3 is proved. □

By Lemmas 2.1,2.2,2.3, the proof of Theorem 2.1 is completed.

3 A parametric optimal algorithm *QOrdinal_Min*

In this section, we present an algorithm $QOrdinal\_Min$ (QOM for short) and study its competitive ratio. The algorithm consists of an infinite sequence of procedures. For any $s \geq 1$, it chooses exactly one procedure to assign jobs. We first give the definition of procedures.

Procedure(0):

Assign jobs in the subset $\{p_{2i+2} | i \geq 0\}$ to M_1;

Assign jobs in the subset $\{p_{2i+1} | i \geq 0\}$ to M_2.

Procedure(k), $k \geq 1$:

Assign jobs in the subset $\{p_2, p_3\} \cup \{p_{3+(2k+1)j+i} | j \geq 0, i = 2, 4, \cdots, 2k\} \cup \{p_{3+(2k+1)j+2k+1} | j \geq 0\}$ to M_1;

Assign jobs in the subset $\{p_1\} \cup \{p_{3+(2k+1)j+i} | j \geq 0, i = 1, 3 \cdots, 2k-1\}$ to M_2.

Algorithm *QOrdinal_Min***:**

1. If $s \geq 2$, assign all jobs by Procedure(0).

2. If $s \in [a_k, a_{k+1})$, $k \geq 1$, assign all jobs by Procedure(k).

Theorem 3.1 *The parametric competitive ratio of the algorithm QOM is $c(s)$, and it is an optimal algorithm for $Q2|ordinal\ on-line|C_{\min}$.*

Proof: As we have already shown that $c(s)$ is a lower bound for $Q2|ordinal\ on-line|C_{\min}$, we only need to show $C^{OPT}/C^{QOM} \leq c(s)$. Let T be the total processing time of all jobs, L_1 and L_2 be the completion times of M_1 and M_2 after processing all jobs by the algorithm QOM, respectively, then $C^{QOM} = \min\{L_1, L_2\}$. Obviously, $C^{OPT} \leq T/(s+1)$ and $C^{OPT} \leq T - p_1$. We get the claimed competitive ratio by considering two cases according to the value of s.

Lemma 3.1 *For any $s \geq 2$, we have $C^{OPT}/C^{QOM} \leq c(s) = 2$.*

Proof: Note that *QOM* chooses Procedure(0) for $s \geq 2$. We only prove the case of $n = 2l$, the case of $n = 2l-1$ can be proved by adding a dummy job $p_{2l} = 0$. By the definition of the procedure and $p_1 \geq \cdots \geq p_n$, we have

$$L_1 = \sum_{i=1}^{l} p_{2i} \geq \frac{1}{2}\sum_{i=1}^{l-1}(p_{2i} + p_{2i+1}) + \frac{1}{2}p_{2l} = \frac{T - p_1}{2} \geq \frac{1}{2}C^{OPT},$$

$$L_2 = \frac{1}{s}\sum_{i=1}^{l} p_{2i-1} \geq \frac{1}{2s}\sum_{i=1}^{l}(p_{2i-1} + p_{2i}) = \frac{T}{2s} \geq \frac{s+1}{2s}C^{OPT}.$$

Note that $1/2 < (s+1)/(2s)$, we have $C^{OPT}/C^{QOM} \leq 2$ and thus the result is proved for $s \geq 2$. □

Before we prove the result for the case of $1 \leq s < 2$, we give two estimations for C^{OPT}. Denote $P = \sum_{i=4}^{n} p_i$. We assume that the sequence contains at least three jobs, otherwise we add jobs of processing time zero to the sequence.

Lemma 3.2 (1) *If $p_1 \geq (p_2 + p_3)/s$, then $C^{OPT} \geq (p_2 + p_3)/s$.*
(2) *If $p_1 + P \leq (p_2 + p_3)/s$, then $C^{OPT} = p_1 + P$.*

Proof: (1) Consider the following feasible subschedule for the jobs $\{p_1, p_2, p_3\}$: p_1 is assigned to M_1, p_2 and p_3 are assigned to M_2. Since $p_1 > (p_2 + p_3)/s$, the objective value of this subschedule is at least $(p_2 + p_3)/s$, which implies that $C^{OPT} \geq (p_2 + p_3)/s$.

(2) Consider the following feasible schedule for the complete sequence of jobs: p_2 and p_3 are assigned to M_2, and all other jobs are assigned to M_1. Then its objective value is $\min\{p_1+P, (p_2+p_3)/s\} = p_1+P$. So $C^{OPT} \geq p_1 + P$.

On the other hand, if p_1 shares a machine with at least one of p_2, p_3 in a schedule, then the objective value is no greater than the completion time of the machine which is not processing p_1, and thus no greater than $p_2 + P \leq p_1 + P$. Otherwise, both p_2 and p_3 do not share a machine with p_1, then we also have $C^{OPT} \leq p_1 + P$. The lemma is thus proved. □

The next lemma estimates the machine completion times yielded by Procedure(k), $k \geq 1$.

Lemma 3.3 *If QOM chooses Procedure(k), $k \geq 1$, we have*

$$L_1 \geq \frac{k+1}{2k+1}(T - p_1), \quad L_2 \geq \frac{1}{s}\left(p_1 + \frac{k}{2k+1}P\right).$$

Proof: We only prove the case of $n = 3+(2k+1)l$, other cases can be proved by adding at most $2k$ dummy jobs of processing time zero. Since $p_1 \geq \cdots \geq p_n$, we have

$$p_2 + p_3 \geq \frac{2}{3}(p_2 + p_3 + p_4),$$

$$\sum_{i=1}^{k-1} p_{3+(2k+1)j+2i} \geq \frac{1}{2}\sum_{i=1}^{k-1}\left(p_{3+(2k+1)j+2i} + p_{3+(2k+1)j+2i+1}\right) = \frac{1}{2}\sum_{i=2}^{2k-1} p_{3+(2k+1)j+i}.$$

Using a similar approach repeatedly, we have

$$\begin{aligned}
L_1 &= p_2 + p_3 + \sum_{j=0}^{l-1}\left(\sum_{i=1}^{k-1} p_{3+(2k+1)j+2i} + p_{3+(2k+1)j+2k} + p_{3+(2k+1)j+2k+1}\right) \\
&\geq \frac{2(p_2+p_3+p_4)}{3} + \sum_{j=0}^{l-2}\left(\frac{1}{2}\sum_{i=2}^{2k-1} p_{3+(2k+1)j+i} + \frac{2}{3}\sum_{i=2k}^{2k+2} p_{3+(2k+1)j+i}\right) \\
&\quad + \left(\frac{1}{2}\sum_{i=2}^{2k-1} p_{3+(2k+1)(l-1)+i} + \frac{2}{3}\sum_{i=2k}^{2k+1} p_{3+(2k+1)(l-1)+i}\right) \\
&= \frac{T-p_1}{2} + \frac{p_2+p_3+p_4}{6} + \frac{1}{6}\sum_{j=0}^{l-2}\sum_{i=2k}^{2k+2} p_{3+(2k+1)j+i} + \frac{p_{2+(2k+1)l} + p_{3+(2k+1)l}}{6} \\
&\geq \frac{T-p_1}{2} + \frac{1}{6}\cdot\frac{3}{2k+1}\sum_{i=2}^{2k+2} p_i + \frac{1}{6}\cdot\frac{3}{2k+1}\sum_{j=0}^{l-2}\sum_{i=2k}^{4k} p_{3+(2k+1)j+i} \\
&\quad + \frac{p_{2+(2k+1)l} + p_{3+(2k+1)l}}{6} \\
&= \frac{T-p_1}{2} + \frac{T-p_1}{2(2k+1)} = \frac{k+1}{2k+1}(T-p_1),
\end{aligned}$$

and

$$\begin{aligned}
L_2 &= \frac{1}{s}\left(p_1+\sum_{j=0}^{l-1}\sum_{i=1}^{k} p_{3+(2k+1)j+2i-1}\right)\\
&= \frac{1}{s}\left(p_1+\sum_{j=0}^{l-1}\left(\frac{1}{2}\sum_{i=1}^{2k-2} p_{3+(2k+1)j+i}+\frac{1}{3}\sum_{i=2k-1}^{2k+1} p_{3+(2k+1)j+i}\right)\right)\\
&= \frac{1}{s}\left(p_1+\sum_{j=0}^{l-1}\left(\frac{1}{6}\sum_{i=1}^{2k-2} p_{3+(2k+1)j+i}+\frac{1}{3}\sum_{i=1}^{2k-2} p_{3+(2k+1)j+i}+\frac{1}{3}\sum_{i=2k-1}^{2k+1} p_{3+(2k+1)j+i}\right)\right)\\
&= \frac{1}{s}\left(p_1+\frac{1}{6}\sum_{j=0}^{l-1}\sum_{i=1}^{2k-2} p_{3+(2k+1)j+i}+\frac{T-(p_1+p_2+p_3)}{3}\right)\\
&\geq \frac{1}{s}\left(p_1+\frac{1}{6}\cdot\frac{2k-2}{2k+1}\sum_{j=0}^{l-1}\sum_{i=1}^{2k+1} p_{3+(2k+1)j+i}+\frac{T-(p_1+p_2+p_3)}{3}\right)\\
&= \frac{1}{s}\left(p_1+\frac{k-1}{3(2k+1)}\left(T-(p_1+p_2+p_3)\right)+\frac{T-(p_1+p_2+p_3)}{3}\right)\\
&= \frac{1}{s}\left(p_1+\frac{k}{2k+1}P\right). \quad \square
\end{aligned}$$

Lemma 3.4 *For any s with $b_k \leq s < a_k, k \geq 2$, we have $\frac{C^{OPT}}{C^{QOM}} \leq c(s) = \frac{2k+1}{k+1}$.*

Proof: In fact, by Lemma 3.3, we get

$$L_1 \geq \frac{k+1}{2k+1}\left(T-p_1\right) \geq \frac{k+1}{2k+1}C^{OPT}.$$

To prove $L_2 \geq (k+1)C^{OPT}/(2k+1)$, we distinguish three cases according to the values of p_1, p_2, p_3 and P.

Case 1 $p_1 \geq (p_2+p_3)/s$.

By Lemma 3.2(1), $C^{OPT} \geq (p_2+p_3)/s$. By Lemma 3.3 and $C^{OPT} \leq T/(s+1)$, we have

$$\begin{aligned}
L_2 &\geq \frac{1}{s}\left(p_1+\frac{k}{2k+1}P\right) = \left(\frac{1}{s}-\frac{k}{(2k+1)s}\right)p_1-\frac{k}{2k+1}\cdot\frac{p_2+p_3}{s}+\frac{k}{2k+1}\cdot\frac{T}{s}\\
&\geq \frac{k+1}{(2k+1)s}p_1-\frac{k}{2k+1}\cdot\frac{p_2+p_3}{s}+\left(\frac{k}{2k+1}\cdot\frac{s+1}{s}-\frac{k+1}{2k+1}\right)C^{OPT}+\frac{k+1}{2k+1}C^{OPT}\\
&\geq \frac{k+1}{(2k+1)s}\cdot\frac{p_2+p_3}{s}-\frac{k}{2k+1}\cdot\frac{p_2+p_3}{s}+\frac{k-s}{(2k+1)s}\cdot\frac{p_2+p_3}{s}+\frac{k+1}{2k+1}C^{OPT}\\
&= \left(\frac{1}{s}-\frac{k+1}{2k+1}\right)\frac{p_2+p_3}{s}+\frac{k+1}{2k+1}C^{OPT} \geq \frac{k+1}{2k+1}C^{OPT}.
\end{aligned}$$

The last inequality is true for $s \geq (2k+1)/(k+1)$ which is always true for $s \leq a_k$.

Case 2 $(p_2+p_3)/s - p_1 > 0$ and $P \le (p_2+p_3)/s - p_1$.

In this case, we have $P < 2p_1/s - p_1 = (2-s)p_1/s$ (since $p_1 \ge p_2 \ge p_3$). By Lemma 3.3 (2), we get $C^{OPT} = p_1 + P$, and thus

$$\begin{aligned} L_2 &\ge \frac{1}{s}\left(p_1 + \frac{k}{2k+1}P\right) \\ &= \left(\frac{1}{s} - \frac{k+1}{2k+1}\right)p_1 + \left(\frac{k}{(2k+1)s} - \frac{k+1}{2k+1}\right)P + \frac{k+1}{2k+1}(p_1+P) \\ &\ge \left(\frac{1}{s} - \frac{k+1}{2k+1}\right)\cdot\frac{s}{2-s}P + \left(\frac{k}{(2k+1)s} - \frac{k+1}{2k+1}\right)P + \frac{k+1}{2k+1}C^{OPT} \\ &= \frac{2k-(k+1)s}{(2k+1)(2-s)s}P + \frac{k+1}{2k+1}C^{OPT} \ge \frac{k+1}{2k+1}C^{OPT}. \end{aligned}$$

The last inequality is true for $s \ge 2k/(k+1) = a_k$.

Case 3 $(p_2+p_3)/s - p_1 > 0$ and $P > (p_2+p_3)/s - p_1$.

Since $p_1 > (p_2+p_3)/2$ and $p_1 + P > (p_2+p_3)/s$, we have

$$\begin{aligned} L_2 &\ge \frac{1}{s}\left(p_1 + \frac{k}{2k+1}P\right) \\ &= \left(\frac{1}{s} - \frac{k}{(2k+1)s}\right)p_1 + \left(\frac{k}{(2k+1)s} - \frac{k+1}{(2k+1)(s+1)}\right)(p_1+P) \\ &\quad -\frac{k+1}{2k+1}\cdot\frac{p_2+p_3}{s+1} + \frac{k+1}{2k+1}\cdot\frac{p_1+p_2+p_3+P}{s+1} \\ &\ge \frac{k+1}{(2k+1)s}\cdot\frac{p_2+p_3}{2} + \frac{k-s}{s(s+1)(2k+1)}\cdot\frac{p_2+p_3}{s} - \frac{k+1}{2k+1}\cdot\frac{p_2+p_3}{s+1} \\ &\quad +\frac{k+1}{2k+1}\cdot\frac{T}{s+1} \\ &\ge \frac{2k-(k+1)s}{2s^2(2k+1)}\cdot\frac{p_2+p_3}{s} + \frac{k+1}{2k+1}C^{OPT} \ge \frac{k+1}{2k+1}C^{OPT}. \quad \square \end{aligned}$$

Lemma 3.5 *For any s with $a_k \le s < b_{k+1}$, $k \ge 1$, we have $\frac{C^{OPT}}{C^{QOM}} \le c(s) = \frac{2(2k+1)s}{(k+1)s+2k}$.*

Proof: Since $s \ge a_k = 2k/(k+1)$, we obtain

$$L_1 \ge \frac{k+1}{2k+1}(T-p_1) \ge \frac{(k+1)s+2k}{2s(2k+1)}(T-p_1) \ge \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT}.$$

Similarly to the proof of Lemma 3.4, we split the analysis into three cases according to the values of p_1, p_2, p_3 and P. The goal here is to show that $L_2 \ge \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT}$.

Case 1 $p_1 \ge (p_2+p_3)/s$.

$$L_2 \ge \frac{1}{s}\left(p_1 + \frac{k}{2k+1}P\right) = \left(\frac{1}{s} - \frac{k}{(2k+1)s}\right)p_1 - \frac{k}{2k+1}\cdot\frac{p_2+p_3}{s} + \frac{k}{2k+1}\cdot\frac{T}{s}$$

$$
\begin{aligned}
&\geq \frac{k+1}{(2k+1)s}p_1 - \frac{k}{2k+1}\cdot\frac{p_2+p_3}{s} + \left(\frac{k}{2k+1}\cdot\frac{s+1}{s} - \frac{(k+1)s+2k}{2s(2k+1)}\right)C^{OPT} \\
&\quad + \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT} \\
&\geq \frac{k+1}{(2k+1)s}\cdot\frac{p_2+p_3}{s} - \frac{k}{2k+1}\cdot\frac{p_2+p_3}{s} + \frac{k-1}{2(2k+1)}\cdot\frac{p_2+p_3}{s} \\
&\quad + \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT} \\
&= \frac{(k+1)(2-s)}{2s(2k+1)}\cdot\frac{p_2+p_3}{s} + \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT} \geq \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT}.
\end{aligned}
$$

Case 2 $(p_2+p_3)/s - p_1 > 0$ and $P \leq (p_2+p_3)/s - p_1$.

$$
\begin{aligned}
L_2 &\geq \frac{1}{s}\left(p_1 + \frac{k}{2k+1}P\right) \\
&= \left(\frac{1}{s} - \frac{(k+1)s+2k}{2s(2k+1)}\right)p_1 + \left(\frac{k}{(2k+1)s} - \frac{(k+1)s+2k}{2s(2k+1)}\right)P \\
&\quad + \frac{(k+1)s+2k}{2s(2k+1)}(p_1+P) \\
&\geq \left(\frac{1}{s} - \frac{(k+1)s+2k}{2s(2k+1)}\right)\cdot\frac{s}{2-s}P + \left(\frac{k}{(2k+1)s} - \frac{(k+1)s+2k}{2s(2k+1)}\right)P \\
&\quad + \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT} \\
&= \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT}.
\end{aligned}
$$

Case 3 $(p_2+p_3)/s - p_1 > 0$ and $P > (p_2+p_3)/s - p_1$.

$$
\begin{aligned}
L_2 &\geq \frac{1}{s}\left(p_1 + \frac{k}{2k+1}P\right) \\
&= \left(\frac{1}{s} - \frac{k}{(2k+1)s}\right)p_1 + \left(\frac{k}{(2k+1)s} - \frac{(k+1)s+2k}{2s(s+1)(2k+1)}\right)(p_1+P) \\
&\quad - \frac{(k+1)s+2k}{2s(2k+1)}\cdot\frac{p_2+p_3}{s+1} + \frac{(k+1)s+2k}{2s(2k+1)}\cdot\frac{p_1+p_2+p_3+P}{s+1} \\
&\geq \frac{k+1}{(2k+1)s}\cdot\frac{p_2+p_3}{2} + \frac{k-1}{2(s+1)(2k+1)}\cdot\frac{p_2+p_3}{s} \\
&\quad - \frac{(k+1)s+2k}{2s(2k+1)}\cdot\frac{p_2+p_3}{s+1} + \frac{(k+1)s+2k}{2s(2k+1)}\cdot\frac{T}{s+1} \\
&= \frac{(k+1)s+2k}{2s(2k+1)}C^{OPT}. \quad \square
\end{aligned}
$$

Combining Lemmas 3.1, 3.4 and 3.5, the proof of Theorem 3.1 is completed. □

4 Scheduling in the l_p norm

We start with defining the algorithm which is the same for all values of p. We simply use Procedure(1) from Section 3.

P(1): Assign jobs in the subset $\{p_{3i+2}, p_{3i+3} | i \geq 0\}$ to M_1;

Assign jobs in the subset $\{p_{3i+1} | i \geq 0\}$ to M_2.

Note that the same ordinal algorithm was used for identical machines both for minimizing makespan, and maximizing the minimum completion time [11, 17].

Theorem 4.1 *The competitive ratio of any on-line algorithm for scheduling on two identical machines in the l_p norm is at least*

$$\left(\max_{a \geq 1} \frac{(a+1)^p + 2^p}{a^p + 3^p} \right)^{\frac{1}{p}} .$$

The competitive ratio of P(1) is at most this bound, and therefore it is an optimal ordinal algorithm for all norms. Note that for $p = 2$ this value is $\sqrt{7 + \sqrt{13}}/3 \approx 1.0855$.

Proof: We start with the lower bound. Consider a sequence of four jobs. If p_1 is assigned on a machine alone then use the values $1, 1, 1, 1$ for the processing times. Otherwise use $a, 1, 1, 1$ for the value of $a \geq 1$ that maximizes $((a+1)^p + 2^p)/(a^p + 3^p)$. In the first case clearly $C^{OPT} \geq (2 \cdot 2^p)^{1/p}$ and in the second case clearly $C^{OPT} \geq (a^p + 3^p)^{1/p}$. The cost of the ordinal algorithm is $(1 + 3^p)^{1/p}$ in the first case and $((a+1)^p + 2^p)^{1/p}$ in the second case. It is possible to show, using some algebra and calculus, that for every value of a, $(1 + 3^p)/(2 \cdot 2^p) \geq ((a+1)^p + 2^p)/(a^p + 3^p)$. Note that the maximum of the function $((a+1)^p + 2^p)/(a^p + 3^p)$ is achieved in a single point in the interval $[1, \infty)$ which is the solution of the equation $3(3/a)^{p-1} = 1 + 2(2/(a+1))^{p-1}$. For $p = 2$ we get the quadratic equation $a^2 - 4a - 9 = 0$ whose positive root is $a = \sqrt{13} + 2$.

To prove the upper bound we use some additional notations. Let X be the load of M_1 and Y be the load of M_2. By the definition of P(1) we know that $Y \geq X/2$. On the other hand we can see that $Y - p_1 \leq X/2$. We consider two cases.

Case 1: If $p_1 \leq (X+Y)/2$, then $(C^{OPT})^p \geq 2 \cdot ((X+Y)/2)^p$ and $(C^{P(1)})^p = X^p + Y^p$. We can use the two bounds on p_1 to get $X \geq Y/2$. The maximum of the function $(X^p + Y^p)/((X+Y)/2)^p)$ is obtained in the boundary, i.e. for $X = 2Y$ and for $Y = 2X$. The maximum value for $(C^{P(1)})^p/(C^{OPT})^p$ is $(4^p + 2^p)/(2 \cdot 3^p)$. This is exactly the function $((a+1)^p + 2^p)/(a^p + 3^p)$ for $a = 3$, and therefore is it at most $\max_{a \geq 1}((a+1)^p + 2^p)/(a^p + 3^p)$.

Case 2: If $p_1 > (X+Y)/2$, then $(C^{OPT})^p \geq p_1^p + (X + Y - p_1)^p$ and $(C^{P(1)})^p = X^p + Y^p$. Since the problem is scalable, we can assume without loss of generality that $X = 2$. We also substitute $Y = b + 1$. We now need to bound the maximum of the function $((b+1)^p + 2^p)/(p_1^p + (b + 3 - p_1)^p))^{1/p}$. First we

can see that the function is clearly bounded from above by 2 (even if all jobs were scheduled on the same machine by the ordinal algorithm). We search for the maximum of $((b+1)^p + 2^p)/(p_1^p + (b+3-p_1)^p)$. Given the conditions on X, Y, p_1, we can restrict ourselves to $b \leq p_1 \leq b+1$ and $2p_1 \geq b+3$. Taking the partial derivative with respect to p_1 we get that an extremal point must satisfy $p = (b+3)/2$ and therefore it is only left to consider the boundary. The case $p = b+1$ gives the value 1 for the function. The case $p = b$ gives $((b+1)^p + 2^p)/(b^p + 3^p)$. The case $p = (b+3)/2$ gives $((b+1)^p + 2^p)/(2((b+3)/2)^p)$ which is at most $((b+1)^p + 2^p)/(b^p + 3^p)$ for $b \geq 1$. For $b < 1$ the function $((b+1)^p + 2^p)/(b^p + 3^p)$ is smaller than one and thus irrelevant. We are therefore left with the upper bound $\max_{a \geq 1}((a+1)^p + 2^p)/(a^p + 3^p)$ as claimed. □

References

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

[2] Y. Azar, L. Epstein, On-line machine covering, *Algorithms-ESA'97*, Lecture Notes in Computer Science 1284, Springer Verlag, 23-36(1997).

[3] A.K. Chandra and C.K. Wong. Worst-case analysis of a placement algorithm related to storage allocation. *SIAM Journal on Computing*, **4(3)**, 249-263 (1975).

[4] J. Csirik, H. Kellerer, G. Woeginger, The exact LPT-bound for maximizing the minimum machine completion time, *Operations Research Letters*, **11,** 281-287(1992).

[5] B. Deuermeyer, D. Friesen, M. Langston, Scheduling to maximize the minimum processor finish time in a multiprocessor system, *SIAM Journal of Discrete Methods*, **3,** 190-196(1982).

[6] L. Epstein, Bin stretching revisited, *Acta Informatica*, **39**, 98-117(2003).

[7] L. Epstein, L. M. Favrholdt, Optimal preemptive semi-online scheduling to minimize makespan on two related machines, *Operations Research Letters*, **30,** 269-275(2002).

[8] L. Epstein, L. M. Favrholdt, Optimal non-preemptive semi-online scheduling on two related machines, *Proc. of the 27th International Symposium on Mathematical Foundations of Computer Science*, 245-256(2002).

[9] L. Epstein, J. Noga, S. Seiden, J. Sgall, G. Woeginger, Randomized on-line scheduling on two uniform machines, *Journal of Scheduling*, **4,** 71-92(2001).

[10] D. Friesen, B. Deuermeyer, Analysis of greedy solutions for a replacement part sequencing problem, *Mathematics of Operations Research*, **6,** 74-87(1981).

[11] Y. He, Semi-on-line scheduling problems for maximizing the minimum machine completion time, *Acta Mathematica Applicatae Sinica,* **17,** 107-113(2001).

[12] Y. He, Z. Y. Tan, Ordinal on-line scheduling for maximizing the minimum machine completion time, *Journal of Combinatorial Optimization,* **6,** 199-206(2002).

[13] H. Kellerer, V. Kotov, M. Speranza, Z. Tuza, Semi online algorithms for the partition problem, *Operations Research Letters,* **21,** 235-242(1997).

[14] E. L. Lawler, Combinatorial optimization: Networks and matroids, Holt, Rinehart and Winston, Toronto(1976).

[15] W. P. Liu, J. B. Sidney, Ordinal algorithm for packing with target center of gravity, *Order,* **13,** 17-31(1996).

[16] W. P. Liu, J. B. Sidney, Bin packing using semi-ordinal data, *Operations Research Letters,* **19,** 101-104(1996).

[17] W. P. Liu, J. B. Sidney, A. van Vliet, Ordinal algorithms for parallel machine scheduling, *Operations Research Letters,* **18,** 223-232(1996).

[18] N. V. R. Mahadev, A. Pekec, F. S. Roberts, On the meaningfulness of optimal solutions to scheduling problems: Can an optimal solution be nonoptimal, *Operations Research*, **46,** S120-S134(1998).

[19] D. Sleator, R. E. Tarjan, Amortized efficiency of list update and paging rules, *Communications of the ACM,* **28,** 202-208(1985).

[20] Z. Y. Tan, Y. He, Semi on-line scheduling on two uniform machines, *Systems Engineering-Theory and Practice*, **21**, 53-57(2001). (in Chinese)

[21] Z. Y. Tan, Y. He, Semi-on-line scheduling with ordinal data on two uniform machines, *Operations Research Letters*, **28,** 221-231(2001).

[22] G. Woeginger, A polynomial time approximation scheme for maximizing the minimum machine completion time, *Operations Research Letters,* **20,** 149-154(1997).