# An improved on-line algorithm for scheduling on two unrestrictive parallel batch processing machines

**Q.Q. Nong[1], T.C.E. Cheng[2], C.T. Ng[2*]**
[1]Department of Mathematics, Ocean University of China, Qingdao, Shandong 266071, People's Republic of China
[2]Department of Logistics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, People's Republic of China

**Abstract.** We consider the problem of on-line scheduling a set of $n$ jobs on two parallel batch processing machines. Each machine can handle an infinite number of jobs as a batch simultaneously. The processing time of a batch is the time required for processing the longest job in the batch. Each job becomes available at its release date, which is not known in advance, and its processing time only becomes known at its arrival. We deal with the problem of minimizing the makespan. We provide an algorithm for the problem that is better than one given in the literature, improving the competitive ratio from $\frac{3}{2}$ to $\sqrt{2}$.

**Keywords:** on-line scheduling; parallel machines; batch; worst-case analysis; competitive ratio

## 1 Introduction

We consider the problem of on-line scheduling on two parallel batch processing machines. A parallel batch processing machine is modeled as a system that can handle up to $b$ jobs simultaneously as a batch. The processing time of a batch is the time required for processing the longest job in the batch, and all the jobs in a batch start and complete at the same time. We are given a set of $n$ independent jobs. Each job $J_j$ $(1 \le j \le n)$ becomes available at its release date $r_j$, which is not known in advance, and its processing time $p_j$ only becomes known at its arrival. The problem involves assigning all the jobs to batches and machines and determining the starting times of the resulting batches in such a way that the makespan, i.e., $\max_{1\le j\le n} C_j$, is minimized, where $C_j$ is the completion of job $J_j$. According to the scheduling notation introduced by Graham et al. [4], this model is expressed as

$$P2|b = \infty, r_j, \textit{on-line}|C_{\max}.$$

Scheduling of batch processing machines has been extensively studied in the last decade [1, 2, 3, 6, 7, 9, 10, 11, 12]. According to the limit on the size of each batch, there are two distinct models. One is the *restrictive* model in which the bound $b$ on each batch size is effective, i.e., $b < n$. Problems of this model is motivated by the burn-in operations in semiconductor manufacturing, in which a batch of integrated circuits are placed in an oven and then exposed to a high temperature. Each circuit has a prespecified minimum burn-in time and the burn-in

[*]Corresponding author. E-mail address:lgtctng@polyu.edu.hk (Daniel Ng).

oven has a limited capacity. The other is the *unrestrictive* model in which there is effectively no limit on the sizes of batches, i.e., $b = \infty$. Scheduling problems of this model arise in situations where compositions need to be hardened in kilns and the kiln is sufficiently large that it does not restrict batch sizes. According to the characteristics of the information known before scheduling, scheduling research can be divided into two categories. The first category is *off-line*, in which there is a basic assumption that the scheduler has full information on the problem instance, such as the total number of jobs to be scheduled, their release dates and their processing times, before solution algorithms are applied. The second category is *on-line*, in which, contrary to the off-line case, at any point in time, the scheduler only knows the jobs that have already arrived and has no information at all on whether any more jobs will come. In this situation, the scheduler has to schedule jobs irrevocably.

For many on-line scheduling problems, because of a lack of information, it is normally not possible to have on-line algorithms that guarantee to deliver optimal solutions. Researchers therefore turn to studying approximation on-line algorithms for this kind of problems. The quality of an on-line algorithm is typically assessed by its *competitive ratio*: the nearer the ratio is to 1, the better the algorithm is. We say that an algorithm has a competitive ratio $\rho$ (or is a $\rho$-competitive algorithm) if for any input instance, it always returns a feasible solution with an objective value not greater than $\rho$ times of the optimal (off-line) solution.

Let us survey the previous related results. Lee and Uzsoy [5] provided a number of heuristics for the off-line scheduling problem $1|r_j, b < n|C_{\max}$. Liu and Yu [8] proved that the problem is $NP$-hard even if there are only two release dates and derived a pseudo-polynomial time algorithm for the case where the number of release dates is fixed. Zhang et al. [13] considered the on-line version of the problem. They dealt with both the unrestrictive and restrictive models. For the unrestrictive model, they derived an optimal on-line algorithm with a competitive ratio of $\frac{\sqrt{5}+1}{2}$. For the restrictive model, they provided a 2-competitive algorithm and a lower bound of $\frac{\sqrt{5}+1}{2}$ on the competitive ratio of any on-line algorithm. In the same paper, they considered the problem $Pm|b = \infty, r_j, \textit{on-line}|C_{\max}$ and developed an $1 + \beta_m$-competitive on-line algorithm, where $m$ is the number of machines, and $0 < \beta_m < 1$ is a solution of the equation $\beta_m = (1 - \beta_m)^{m-1}$, which yields $\beta_2 = \frac{1}{2}$. In a recent paper, Zhang et al. [14] addressed the problems $P|b < n, r_j, p_j = p, \textit{on-line}|C_{\max}$ and $P|b = \infty, r_j, p_j = p, \textit{on-line}|C_{\max}$, in both of which there is an assumption that the processing times of the jobs to be scheduled are identical. They first proved that there is no on-line algorithm with a competitive ratio smaller than $\frac{\sqrt{5}+1}{2}$ ($1 + \gamma_m$ resp.) for $P|b < n, r_j, p_j = p, \textit{on-line}|C_{\max}$ (for $Pm|b = \infty, r_j, p_j = p, \textit{on-line}|C_{\max}$, where $0 < \gamma_m < 1$ is a solution of the equation $(1 + \gamma_m)^{m+1} = \gamma_m + 2$). They then provided on-line algorithms with competitive ratios matching the lower bounds.

In this paper we study the problem $P2|b = \infty, r_j, \textit{on-line}|C_{\max}$. According to the results of [14], there does not exist any on-line algorithm for the problem with a competitive ratio smaller than $1 + \gamma_2$, where $\gamma_2 \approx 0.325$ is a solution of the equation $(1 + \gamma_2)^3 = \gamma_2 + 2$. We provide an on-line algorithm for the problem that is better than the one given in [13]. In fact, our algorithm improves the competitive ratio from $\frac{3}{2}$ to $\sqrt{2}$.

## 2 Main results

Before designing an algorithm, let us consider a simple instance $I$ in which each job has a processing time of 1. Let $\mathcal{A}$ be an arbitrary on-line algorithm. The first job $J_1$ in $I$ arrives at time 0. If $J_1$ is processed at or after time $\sqrt{2}-1$, no more job comes in $I$. Then it results in a schedule with a makespan of at least $\sqrt{2}$, while the optimal makespan is 1. If $\mathcal{A}$ starts $J_1$ before time $\gamma_2 \approx 0.325$, say at time $t_1$, the second job $J_2$ is released at time $t_1+\epsilon$, where $\epsilon$ is a sufficiently small positive number. Assume that $\mathcal{A}$ starts $J_2$ at time $t_2$. If $\frac{1+t_2}{1+t_1} \geq 1+\gamma_2$, no more job comes in $I$. Then $\mathcal{A}$ returns a schedule with a makespan of $1+t_2$ and the optimal makespan is $1+t_1+\epsilon$, which means that, when $\epsilon$ tend to 0, the competitive ratio of $\mathcal{A}$ is at least

$$\frac{1+t_2}{1+t_1+\epsilon} \rightarrow \frac{1+t_2}{1+t_1} \geq 1+\gamma_2.$$

If $\frac{1+t_2}{1+t_1} < 1+\gamma_2$, the third job $J_3$ arrives at time $t_2+\epsilon$. Then $\mathcal{A}$ returns a schedule with a makespan of at least $2+t_1$ and the optimal makespan is $1+t_2+\epsilon$. Thus, when $\epsilon$ tend to 0, the competitive ratio of $\mathcal{A}$ is at least

$$\frac{2+t_1}{1+t_2+\epsilon} \rightarrow \frac{2+t_1}{1+t_2} > \frac{2+t_1}{(1+\gamma_2)(1+t_1)} > \frac{2+\gamma_2}{(1+\gamma_2)^2} = 1+\gamma_2,$$

where the last inequality holds from the fact that the function $\frac{2+t_1}{(1+\gamma_2)(1+t_1)}$ is a decreasing function and $t_1 < \gamma_2$, and the equation follows from the fact that $(1+\gamma_2)^3 = \gamma_2+2$. From this instance we realize that, in order to get a better competitive ratio, jobs in an instance should wait for some reasonable time before they are processed.

Let $J(t)$ be the job with the largest processing time among the jobs that are available but not yet scheduled at time $t$; if there are two or more candidates, define $J(t)$ to be the one with the largest release date. Denote by $p(t)$ and $r(t)$ the processing time and the release date of job $J(t)$, respectively. Let $\alpha = \sqrt{2}-1$, which is a solution of the equation $\alpha^2+2\alpha = 1$. Our algorithm runs as follows.

**Algorithm** $A_2(\alpha)$

At time $t$, if a machine is idle and there are jobs available but not yet scheduled, and

$$t \geq (1+\alpha)r(t) + \alpha p(t),$$

then start all the available jobs as a single batch on the machine with the minimum completion time so far; otherwise, do nothing but wait.

Note that the basic idea of Algorithm $A_2(\alpha)$ is to apply the *delay* and *greedy* tactic, and all the batch starting times in the schedule produced are different. Given an instance, denote by $\sigma$ the schedule produced by Algorithm $A_2(\alpha)$ and by $\pi$ the optimal schedule. For a schedule $x$, let $C_{\max}(x)$ denote its makespan and let $s_j$ denote the starting time of job $J_j$. For any two jobs $J_i$ and $J_j$ in two different batches in $\sigma$, if $s_i > s_j$, then, by the description of Algorithm $A_2(\alpha)$, all the jobs in the batch to which job $J_i$ is assigned are released after $s_j$, implying that (1) $s_i > (1+\alpha)s_j + \alpha p_i$ and thus $s_i - s_j > \alpha p_i$; and (2) $C_{\max}(\pi) > s_j + p_i$. Consider a batch starting at time $s$. We say that the batch is *regular* if it starts at $(1+\alpha)r(s)+\alpha p(s)$. Clearly, if the batch is not regular, then $s > (1+\alpha)r(s)+\alpha p(s)$, which means that both machines are processing jobs at time $(1+\alpha)r(s)+\alpha p(s)$.

Consider the structures of $\pi$ and $\sigma$. The following three lemmas provide some useful insights into $\pi$ and $\sigma$.

**Lemma 1** Without increasing the makespan of $\pi$, we can assume that the jobs in $\pi$ are scheduled in decreasing order of their processing times.

**Proof.** Consider an arbitrary pair of jobs $J_i$ and $J_j$ in two different batches. If $p_i \leq p_j$ and job $J_i$ starts before job $J_j$ in $\pi$, then by moving job $J_i$ to the batch to which job $J_j$ is assigned, we get a new schedule with a makespan not greater than $C_{\max}(\pi)$. This procedure can be repeated until a schedule satisfying the property stated in the lemma is obtained. $\Box$

Lemma 1 indicates that for any two jobs $J_i$ and $J_j$ in $\pi$, if job $J_i$ starts before job $J_j$, then $p_i > p_j$. The lemma below will simplify the structure of $\sigma$ significantly.

**Lemma 2** Without decreasing the ratio of $C_{\max}(\sigma)/C_{\max}(\pi)$, we can assume that there is only one job in each batch of $\sigma$.

**Proof.** From each batch in $\sigma$, pick the job with the largest processing time; if there are two or more candidates, pick the one with the largest release date. Let $I'$ be the instance that consists of the jobs being picked only. Apply $A_2(\alpha)$ to $I'$. Then we obtain a schedule that is identical to $\sigma$ in the sense that the processing times and the starting times of the batches in the schedule are the same as those in $\sigma$. Thus, the makespan of the resulting schedule is not smaller than $C_{\max}(\sigma)$. On the other hand, it is evident that the optimal makespan of $I'$ is not greater than $C_{\max}(\pi)$. Thus, we can assume that there is only one job in each batch of $\sigma$ and this assumption does not decrease the ratio of $C_{\max}(\sigma)/C_{\max}(\pi)$. $\Box$

We thus assume in the sequel that there is only one job in each batch of $\sigma$ and we can refer to a batch by only referring to the job in it. For convenience, we index the jobs in $\sigma$ in the order so that $C_1 \leq C_2 \leq \cdots \leq C_n$.

**Lemma 3** Without decreasing the ratio of $C_{\max}(\sigma)/C_{\max}(\pi)$, we can assume that, in $\sigma$, except $J_n$ there is no job starting at or after $s_n$.

**Proof.** Let $I'$ be the instance consisting of $J_n$ and the jobs that start before $s_n$ in $\sigma$ only. Apply $A_2(\alpha)$ to $I'$. Then it results in a schedule with a makespan equal to $C_{\max}(\sigma)$. Clearly, the optimal makespan of $I'$ is not greater than $C_{\max}(\pi)$. Thus the lemma follows. $\Box$

We will eventually show that

$$C_{\max}(\sigma)/C_{\max}(\pi) \leq 1 + \alpha, \tag{1}$$

which yields the following theorem.

**Theorem 4** Algorithm $A_2(\alpha)$ has a competitive ratio of $\sqrt{2}$.

Clearly, if job $J_n$ is regular, then

$$C_{\max}(\sigma) = s_n + p_n = (1+\alpha)(r_n + p_n) \leq (1+\alpha)C_{\max}(\pi),$$

and hence inequality (1) holds. Thus, in the rest of this section, we assume that job $J_n$ is not regular. Consider the last three jobs, $J_{n-2}$, $J_{n-1}$ and $J_n$ in $\sigma$. From the fact that $C_{n-2} \leq C_{n-1} \leq C_n$ and Lemma 3, one can easily see that job $J_{n-2}$ and job $J_n$ are on the same machine and job $J_{n-1}$ is on the other. Since job $J_n$ is not regular, there is no idle time between $s_n$ and

$C_{n-2}$, i.e., $s_n = C_{n-2}$. To show Theorem 4, we distinguish two cases depending on whether or not $s_{n-1} > s_{n-2}$. We will show that inequality (1) holds in each case. We first consider the case where $s_{n-1} > s_{n-2}$ and start with establishing upper bounds on $C_{\max}(\sigma) - C_{\max}(\pi)$.

**Observation 5** If $s_{n-1} > s_{n-2}$, then $C_{\max}(\sigma) - C_{\max}(\pi) < s_n - s_{n-1} \leq (1-\alpha)p_{n-2}$.

**Proof.** Since $s_n = C_{n-2} \leq C_{n-1} = s_{n-1} + p_{n-1}$, we have $s_n - s_{n-1} \leq p_{n-1}$. Furthermore, since $C_{\max}(\pi) > s_{n-1} + p_n$ and $C_{\max}(\sigma) = s_n + p_n$, we have

$$C_{\max}(\sigma) - C_{\max}(\pi) < s_n - s_{n-1} \leq p_{n-1}. \tag{2}$$

If $p_{n-1} \leq (1-\alpha)p_{n-2}$, then the result follows. If $p_{n-1} > (1-\alpha)p_{n-2}$, then by inequality (2) and the property of Algorithm $A_2(\alpha)$, we have

$$\begin{aligned}
C_{\max}(\sigma) - C_{\max}(\pi) &< s_n - s_{n-1} \\
&= s_{n-2} + p_{n-2} - s_{n-1} \\
&< s_{n-2} + p_{n-2} - [(1+\alpha)s_{n-2} + \alpha p_{n-1}] \\
&= p_{n-2} - \alpha s_{n-2} - \alpha p_{n-1} \\
&\leq p_{n-2} - \alpha \cdot \alpha p_{n-2} - \alpha p_{n-1} \\
&< (1-\alpha)p_{n-2},
\end{aligned}$$

as required. □

**Lemma 6** If $s_{n-1} > s_{n-2}$, then $C_{\max}(\sigma)/C_{\max}(\pi) \leq 1 + \alpha$.

**Proof.** Consider the optimal schedule $\pi$. If job $J_{n-1}$ and job $J_n$ are in the same batch, then $C_{\max}(\pi) \geq r_n + p_{n-1} > s_{n-1} + p_{n-1} \geq C_{n-2}$; if job $J_{n-2}$ and job $J_n$ are in the same batch, then $C_{\max}(\pi) \geq r_n + p_{n-2} > s_{n-2} + p_{n-2} = C_{n-2}$; if job $J_{n-2}$ and job $J_{n-1}$ are in the same batch, then $C_{\max}(\pi) \geq r_{n-1} + p_{n-2} > s_{n-2} + p_{n-2} = C_{n-2}$. Together with Observation 5 and the fact that $C_{n-2} \geq (1+\alpha)p_{n-2}$, this implies that in all the cases

$$\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{(1-\alpha)p_{n-2}}{(1+\alpha)p_{n-2}} = 1 + \alpha.$$

We thus assume that in $\pi$ each of the jobs $J_{n-2}$, $J_{n-1}$ and $J_n$ is in a different batch. We can further assume that, in $\pi$, jobs $J_{n-2}$, $J_{n-1}$ and $J_n$ are processed in the order so that $s_{n-2}(\pi) < s_{n-1}(\pi) < s_n(\pi)$, where $s_j(\pi)$ is the starting time of $J_j$ in $\pi$. This is because if the order is violated, then from the fact that $r_{n-1} > s_{n-2}$, $r_n > s_{n-1}$ and $C_{n-1} \geq C_{n-2}$, one can deduce that $C_{\max}(\pi) \geq C_{n-2} \geq (1+\alpha)p_{n-2}$ and thus $C_{\max}(\sigma)/C_{\max}(\pi) \leq 1+\alpha$. By Lemma 1, a consequence of this assumption is $p_{n-2} > p_{n-1} > p_n$. Clearly, there is one machine that processes at least two of the jobs $J_{n-2}$, $J_{n-1}$ and $J_n$. We distinguish two cases in the following analysis.

**Case 1.** In $\pi$, either jobs $J_{n-2}$ and $J_{n-1}$, or jobs $J_{n-2}$ and $J_n$ are processed on the same machine. Then

$$C_{\max}(\pi) \geq \max\{r_{n-2} + p_{n-2} + p_n, s_{n-2} + p_{n-1}\},$$

where the second term of the inequality is from the fact that $r_{n-1} > s_{n-2}$. Consider $\sigma$, the schedule produced by $A_2(\alpha)$. If job $J_{n-2}$ is regular in $\sigma$, then $s_{n-2} = (1+\alpha)r_{n-2} + \alpha p_{n-2}$ and thus

$$\begin{aligned}
C_{\max}(\sigma) &= s_{n-2} + p_{n-2} + p_n \\
&= (1+\alpha)r_{n-2} + \alpha p_{n-2} + p_{n-2} + p_n \\
&\leq (1+\alpha)(r_{n-2} + p_{n-2} + p_n) \\
&\leq (1+\alpha)C_{\max}(\pi).
\end{aligned}$$

Otherwise, on each machine there is one job being processed exactly before $s_{n-2}$. For convenience, we denote the two jobs by $J_b$ and $J_{b'}$, respectively, where job $J_b$ is the one on the machine to which job $J_{n-2}$ is assigned and job $J_{b'}$ is the other.

- If $s_{b'} < s_b$, then

$$\begin{aligned}
s_b &> (1+\alpha)s_{b'} + \alpha p_b \\
&\geq (1+\alpha)\cdot \alpha p_{b'} + \alpha p_b \\
&> (1+\alpha)\cdot \alpha(s_{n-2} - s_b) + \alpha(s_{n-2} - s_b) \\
&= \alpha(2+\alpha)(s_{n-2} - s_b) \\
&\geq \alpha^2(2+\alpha)p_{n-2} \\
&= \alpha p_{n-2},
\end{aligned}$$

where the third inequality holds from the fact that $p_b = s_{n-2} - s_b$ and $p_{b'} \geq s_{n-2} - s_{b'} > s_{n-2} - s_b$ in this case. Furthermore, noting that $r_{n-2} > s_b$, we have the following:

$$C_{\max}(\pi) > s_b + p_{n-2} + p_n > (1+\alpha)p_{n-2}.$$

Thus,

$$\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{(1-\alpha)p_{n-2}}{(1+\alpha)p_{n-2}} = 1 + \alpha.$$

- Consider the subcase with $s_{b'} > s_b$. If $p_b \leq \frac{\alpha(p_{n-2}+p_n)}{1-\alpha^2}$, then

$$\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < \frac{s_b + p_b + p_{n-2} + p_n}{s_b + p_{n-2} + p_n} = 1 + \frac{p_b}{s_b + p_{n-2} + p_n} \leq 1 + \frac{p_b}{\alpha p_b + p_{n-2} + p_n} \leq 1 + \alpha.$$

If $p_b > \frac{\alpha(p_{n-2}+p_n)}{1-\alpha^2}$, then

$$s_{n-2} = s_b + p_b \geq (1+\alpha)p_b > (1+\alpha)\frac{\alpha(p_{n-2}+p_n)}{1-\alpha^2} = \frac{\alpha}{1-\alpha}(p_{n-2}+p_n) \geq \frac{\alpha}{1-\alpha}p_{n-2}.$$

Together with inequalities (2), this implies that

$$\begin{aligned}
C_{\max}(\sigma) - C_{\max}(\pi) &< s_n - s_{n-1} \\
&< (s_{n-2} + p_{n-2}) - [(1+\alpha)s_{n-2} + \alpha p_{n-1}] \\
&= p_{n-2} - \alpha s_{n-2} - \alpha p_{n-1} \qquad (3) \\
&< p_{n-2} - \alpha \cdot \tfrac{\alpha}{1-\alpha}p_{n-2} - \alpha p_{n-1} \\
&= \tfrac{\alpha}{1-\alpha}p_{n-2} - \alpha p_{n-1},
\end{aligned}$$

and

$$\begin{aligned} C_{\max}(\pi) &> s_{n-1} + p_n \\ &> (1+\alpha)s_{n-2} + \alpha p_{n-1} + p_n \\ &> (1+\alpha)\cdot \tfrac{\alpha}{1-\alpha} p_{n-2} + \alpha p_{n-1} + p_n, \\ &\geq p_{n-2} + \alpha p_{n-1}. \end{aligned} \tag{4}$$

If $p_{n-1} \geq \frac{p_{n-2}}{2}$ then, by (3) and (4),

$$\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{\frac{\alpha}{1-\alpha} p_{n-2} - \alpha p_{n-1}}{p_{n-2} + \alpha p_{n-1}} \leq 1 + \frac{\frac{\alpha}{1-\alpha} p_{n-2} - \alpha \cdot \frac{p_{n-2}}{2}}{p_{n-2} + \alpha \cdot \frac{p_{n-2}}{2}} = 1 + \alpha.$$

Recall that $C_{\max}(\sigma) - C_{\max}(\pi) < s_n - s_{n-1} \leq p_{n-1}$. If $p_{n-1} < \frac{p_{n-2}}{2}$, then

$$\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_{n-1}}{p_{n-2} + \alpha p_{n-1}} \leq 1 + \frac{\frac{p_{n-2}}{2}}{p_{n-2} + \alpha \cdot \frac{p_{n-2}}{2}} = 1 + \alpha.$$

**Case 2.** In $\pi$, jobs $J_{n-1}$ and $J_n$ are processed on the same machine. Then, $C_{\max}(\pi) \geq r_{n-1} + p_{n-1} + p_n > s_{n-2} + p_{n-1} + p_n$. Thus, $C_{\max}(\sigma) - C_{\max}(\pi) < (s_{n-2} + p_{n-2} + p_n) - (s_{n-2} + p_{n-1} + p_n) = p_{n-2} - p_{n-1}$. If $p_{n-1} \geq (1-\alpha)p_{n-2}$, then $C_{\max}(\sigma) - C_{\max}(\pi) < p_{n-2} - p_{n-1} \leq \alpha p_{n-2}$, and thus $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{\alpha p_{n-2}}{p_{n-2}} = 1 + \alpha$. In the following, we discuss the case where $p_{n-1} < (1-\alpha)p_{n-2}$. Note that if job $J_{n-1}$ is regular in $\sigma$, then the result can be easily observed. We assume that job $J_{n-1}$ is not regular in $\sigma$, and, therefore, there is one job, denoted by $J_b$, being processed exactly before job $J_{n-1}$.

• If $s_b > s_{n-2}$, then $C_{\max}(\pi) > s_b + p_{n-1} + p_n$ and $C_{\max}(\sigma) \leq C_{n-1} + p_n = s_b + p_b + p_{n-1} + p_n$. If $p_b \leq \frac{\alpha}{1-\alpha^2}[(1+\alpha)s_{n-2} + p_{n-1} + p_n]$, then

$$\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < \frac{s_b + p_b + p_{n-1} + p_n}{s_b + p_{n-1} + p_n} = 1 + \frac{p_b}{s_b + p_{n-1} + p_n} < 1 + \frac{p_b}{(1+\alpha)s_{n-2} + \alpha p_b + p_{n-1} + p_n} \leq 1 + \alpha.$$

Otherwise, by (2) and the fact that $p_{n-1} < (1-\alpha)p_{n-2}$, we have

$$\begin{aligned} C_{\max}(\pi) &> s_b + p_{n-1} + p_n \\ &> (1+\alpha)s_{n-2} + \alpha p_b + p_{n-1} + p_n \\ &> (1+\alpha)s_{n-2} + \alpha \cdot \tfrac{\alpha}{1-\alpha^2}[(1+\alpha)s_{n-2} + p_{n-1} + p_n] + p_{n-1} + p_n \\ &= \tfrac{1}{1-\alpha} s_{n-2} + \tfrac{1}{1-\alpha^2}(p_{n-1} + p_n) \\ &\geq \tfrac{1}{1-\alpha} \cdot \alpha p_{n-2} + \tfrac{1}{1-\alpha^2}(p_{n-1} + p_n) \\ &> \tfrac{1}{1-\alpha} \cdot \alpha \cdot \tfrac{1}{1-\alpha} p_{n-1} + \tfrac{1}{1-\alpha^2} p_{n-1} \\ &= (2+\alpha)p_{n-1}, \end{aligned}$$

implying that $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_{n-1}}{(2+\alpha)p_{n-1}} = 1 + \alpha$.

- Consider the subcase with $s_b < s_{n-2}$. We claim that $p_b > \frac{1}{\alpha}p_{n-1} - \frac{1}{1+\alpha}p_{n-2}$. Otherwise,

$$\begin{aligned} s_{n-2} - s_b &= (s_{n-2} + p_{n-2}) - (s_b + p_{n-2}) \\ &\leq C_{n-1} - (s_b + p_{n-2}) \\ &= (s_b + p_b + p_{n-1}) - (s_b + p_{n-2}) \\ &= p_b + p_{n-1} - p_{n-2} \\ &\leq \tfrac{1}{\alpha}p_{n-1} - \tfrac{1}{1+\alpha}p_{n-2} + p_{n-1} - p_{n-2} \\ &< \tfrac{1}{\alpha}(1-\alpha)p_{n-2} - \tfrac{1}{1+\alpha}p_{n-2} + (1-\alpha)p_{n-2} - p_{n-2} \\ &= \tfrac{\alpha}{1+\alpha}p_{n-2} \\ &< \alpha p_{n-2}, \end{aligned}$$

contradicting the fact that $s_{n-2} - s_b > \alpha p_{n-2}$. Thus, the claim holds and we can deduce that

$$\begin{aligned} C_{\max}(\pi) &> s_{n-2} + p_{n-1} \\ &> (1+\alpha)s_b + \alpha p_{n-2} + p_{n-1} \\ &\geq (1+\alpha) \cdot \alpha p_b + \alpha p_{n-2} + p_{n-1} \\ &> (1+\alpha)\alpha[\tfrac{1}{\alpha}p_{n-1} - \tfrac{1}{1+\alpha}p_{n-2}] + \alpha p_{n-2} + p_{n-1} \\ &= (2+\alpha)p_{n-1}. \end{aligned}$$

Thus $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} \leq 1 + \frac{p_{n-1}}{(2+\alpha)p_{n-1}} = 1 + \alpha$. This completes the proof the lemma. □

We now consider the case with $s_{n-1} < s_{n-2}$. As before, we first establish some upper bounds on $C_{\max}(\sigma) - C_{\max}(\pi)$.

**Observation 7** If $s_{n-1} < s_{n-2}$, then $C_{\max}(\sigma) - C_{\max}(\pi) < p_{n-2} < (1-\alpha)p_{n-1}$.

**Proof.** Since $r_n > s_{n-2}$ and $s_n = C_{n-2} = s_{n-2} + p_{n-2}$, we have

$$C_{\max}(\sigma) - C_{\max}(\pi) < p_{n-2}. \tag{5}$$

We claim that $p_{n-2} < (1-\alpha)p_{n-1}$. In fact, $(1+\alpha)s_{n-1} + \alpha p_{n-2} + p_{n-2} < s_{n-2} + p_{n-2} \leq s_{n-1} + p_{n-1}$. Therefore, $(1+\alpha)p_{n-2} < p_{n-1} - \alpha s_{n-1} \leq p_{n-1} - \alpha \cdot \alpha p_{n-1} = (1-\alpha^2)p_{n-1}$, implying that $p_{n-2} < (1-\alpha)p_{n-1}$. This implies the claim and hence the observation holds. □

A consequence of the fact that $p_{n-2} < (1-\alpha)p_{n-1}$ is the following lower bound on $s_{n-2}$.

$$\begin{aligned} s_{n-2} &> (1+\alpha)s_{n-1} + \alpha p_{n-2} \\ &\geq (1+\alpha) \cdot \alpha p_{n-1} + \alpha p_{n-2} \\ &> (1+\alpha)\alpha\tfrac{p_{n-2}}{1-\alpha} + \alpha p_{n-2} \\ &= (1+\alpha)p_{n-2}. \end{aligned} \tag{6}$$

**Lemma 8** If $s_{n-1} < s_{n-2}$, then $C_{\max}(\sigma)/C_{\max}(\pi) \leq 1 + \alpha$.

**Proof.** Note that $r_n > s_{n-2} > s_{n-1}$, $r_{n-2} > s_{n-1}$ and $C_{n-1} \geq C_{n-2}$. We can deduce that if two of the jobs $J_{n-2}$, $J_{n-1}$ and $J_n$ are in the same batch in $\pi$, then $C_{\max}(\pi) \geq C_{n-2} = s_{n-2} + p_{n-2} > (2+\alpha)p_{n-2}$, where the last inequality holds from (6). This indicates that $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_{n-2}}{(2+\alpha)p_{n-2}} = 1+\alpha$. We thus assume in the sequel that, in $\pi$, each of the jobs $J_{n-2}$, $J_{n-1}$ and $J_n$ is in a different batch. It is evident that at least two of the jobs are on the same machine. Recall that $p_{n-2} < (1-\alpha)p_{n-1}$. If jobs $J_{n-2}$ and $J_{n-1}$ are on the same machine, then $C_{\max}(\pi) \geq p_{n-1} + p_{n-2} > \frac{2-\alpha}{1-\alpha}p_{n-2}$ and thus

$$\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_{n-2}}{\frac{2-\alpha}{1-\alpha}p_{n-2}} < 1+\alpha.$$

**Case 1.** Jobs $J_{n-2}$ and $J_n$ are on the same machine in $\pi$. It is easy to see that if job $J_{n-2}$ is regular in $\sigma$, then $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} \leq 1+\alpha$. Thus, we assume that there is one job being processed exactly before $J_{n-2}$. As before, we denote the job by $J_b$. Then, $r_{n-2} > s_b$ and $C_{\max}(\sigma) - C_{\max}(\pi) < p_b$.

- If $s_b > s_{n-1}$, then

$$\begin{aligned} C_{\max}(\pi) &> s_b + p_{n-2} + p_n \\ &> (1+\alpha)s_{n-1} + \alpha p_b + p_{n-2} + p_n \\ &\geq (1+\alpha)\cdot\alpha p_{n-1} + \alpha p_b + p_{n-2} + p_n \qquad (7)\\ &> (1+\alpha)\alpha\tfrac{p_{n-2}}{1-\alpha} + \alpha p_b + p_{n-2} + p_n \\ &= 2p_{n-2} + \alpha p_b + p_n. \end{aligned}$$

If $p_b \leq \frac{\alpha}{1-\alpha^2}(2p_{n-2}+p_n)$, then $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_b}{2p_{n-2}+\alpha p_b+p_n} \leq 1+\alpha$. Otherwise, by (7), $C_{\max}(\pi)$ is bounded below by $2p_{n-2} + \alpha\cdot\frac{\alpha}{1-\alpha^2}(2p_{n-2}+p_n) + p_n \geq \frac{2}{1-\alpha^2}p_{n-2}$, and thus $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_{n-2}}{\frac{2}{1-\alpha^2}p_{n-2}} = 1+\alpha$.

- If $s_b < s_{n-1}$, then

$$\begin{aligned} s_b &\geq \alpha p_b \\ &= \alpha(s_{n-2} - s_b) \\ &= \alpha[(s_{n-2} - s_{n-1}) + (s_{n-1} - s_b)] \\ &> \alpha\{[(1+\alpha)s_{n-1} + \alpha p_{n-2} - s_{n-1}] + [(1+\alpha)s_b + \alpha p_{n-1} - s_b]\} \\ &\geq \alpha^2(s_{n-1} + p_{n-2} + p_{n-1}) \\ &\geq \alpha^2[(1+\alpha)p_{n-1} + p_{n-2}], \end{aligned}$$

where the last inequality follows from the fact that $s_{n-1} \geq \alpha p_{n-1}$. Thus,

$$\begin{aligned} C_{\max}(\pi) &> s_{n-1} + p_{n-2} + p_n \\ &> (1+\alpha)s_b + \alpha p_{n-1} + p_{n-2} + p_n \\ &> (1+\alpha)\cdot\alpha^2[(1+\alpha)p_{n-1} + p_{n-2}] + \alpha p_{n-1} + p_{n-2} + p_n \\ &> (1+\alpha)\cdot\alpha^2[(1+\alpha)\tfrac{p_{n-2}}{1-\alpha} + p_{n-2}] + \alpha\tfrac{p_{n-2}}{1-\alpha} + p_{n-2} + p_n \\ &> \tfrac{1+\alpha}{1-\alpha}p_{n-2}, \end{aligned}$$

and hence $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_{n-2}}{\frac{1+\alpha}{1-\alpha}p_{n-2}} = 1 + \alpha$.

**Case 2.** Jobs $J_{n-1}$ and $J_n$ are on the same machine in $\pi$. Similarly, if job $J_{n-1}$ is regular in $\sigma$, then $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} \leq 1 + \alpha$. Thus, we assume that on each machine there is exactly one job being processed before $s_{n-1}$. We denote the two jobs by $J_b$ and $J_{b'}$, respectively, where job $J_b$ is the one on the machine to which job $J_{n-2}$ is assigned and job $J_{b'}$ is the other. Clearly, $C_{\max}(\pi) > \max\{s_b, s_{b'}\} + p_{n-1} + p_n$ and $C_{\max}(\sigma) - C_{\max}(\pi) < \min\{p_{n-2}, p_{b'}\}$. If $p_{b'} \leq \frac{\alpha}{1-\alpha^2}(p_{n-1} + p_n)$, then

$$\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_b}{s_{b'} + p_{n-1} + p_n} \leq 1 + \frac{p_{b'}}{\alpha p_{b'} + p_{n-1} + p_n} \leq 1 + \alpha.$$

Otherwise, $p_{b'} > \frac{\alpha}{1-\alpha^2}(p_{n-1} + p_n)$ and we consider the following two cases.

- If $s_{b'} < s_b$, then

$$s_b > (1+\alpha)s_{b'} + \alpha p_b \geq (1+\alpha) \cdot \alpha p_{b'} > (1+\alpha)\alpha \cdot \frac{\alpha}{1-\alpha^2}(p_{n-1} + p_n) \geq \frac{\alpha^2}{1-\alpha}p_{n-1},$$

and hence

$$\begin{aligned} C_{\max}(\pi) &> s_{n-1} + p_{n-2} \\ &> (1+\alpha)s_b + \alpha p_{n-1} + p_{n-2} \\ &> (1+\alpha) \cdot \tfrac{\alpha^2}{1-\alpha}p_{n-1} + \alpha p_{n-1} + p_{n-2} \\ &= 2\alpha p_{n-1} + p_{n-2} \\ &> \tfrac{1+\alpha}{1-\alpha}p_{n-2}, \end{aligned}$$

implying that $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_{n-2}}{\frac{1+\alpha}{1-\alpha}p_{n-2}} = 1 + \alpha$.

- If $s_{b'} > s_b$, then $p_b \geq s_{n-1} - s_b > s_{n-1} - s_{b'} = p_{b'}$. Recall that $p_{b'} > \frac{\alpha}{1-\alpha^2}(p_{n-1} + p_n)$. We have

$$\begin{aligned} s_{n-1} &= C_{b'} \\ &= s_{b'} + p_{b'} \\ &> (1+\alpha)s_b + \alpha p_{b'} + p_{b'} \\ &\geq (1+\alpha) \cdot \alpha p_b + (1+\alpha)p_{b'} \\ &> (1+\alpha) \cdot \alpha p_{b'} + (1+\alpha)p_{b'} \\ &> (1+\alpha)^2 \cdot \tfrac{\alpha}{1-\alpha^2}(p_{n-1} + p_n) \\ &\geq p_{n-1}. \end{aligned}$$

Thus, $\frac{C_{\max}(\sigma)}{C_{\max}(\pi)} < 1 + \frac{p_{n-2}}{s_{n-1}+p_{n-2}} < 1 + \frac{p_{n-2}}{p_{n-1}+p_{n-2}} < 1 + \frac{p_{n-2}}{\frac{1}{1-\alpha}p_{n-2}+p_{n-2}} < 1 + \alpha$, and we complete the proof of the lemma. □

Combining Lemma 6 and Lemma 8, we can obtain inequality (1), and Theorem 4 follows.

## 3 Concluding Remarks

In this paper we designed a $\sqrt{2}$-competitive algorithm for the problem $P2|b = \infty, r_j, \textit{on-line}|C_{\max}$. The algorithm performs better than the one given in [13], which has a competitive ratio of $\frac{3}{2}$. It would be interesting to find out whether for the problem under study there is an on-line algorithm with a competitive ratio matching the lower bound $1 + \gamma_2 \approx 1.325$ provided in [14], or whether the lower bound can be improved. It will be interesting to extend the problem to a general set of $m$ machines and to the variant where the capacity on each batch is restrictive. Further research is required, and it seems that improvement is possible.

**Acknowledgements**

We are grateful for an anonymous referee for his/her helpful comments on an earlier version of the paper. This research was supported in part by The Hong Kong Polytechnic University under grant number *G-U060* and by the *National Natural Science Foundation of China* under grant number 10771200.

## References

[1] P. Brucker, A. Gladky, H. Hoogeveen, M.Y. Kovalyvov, C.N. Potts, T. Tautehahn and S.L. van de Velde, Scheduling a batching machine. *Journal of Scheduling*, **1**(1998), 31-54.

[2] B. Chen, X. Deng and W. Zang , On-line scheduling a batch processing system to minimize total weighted job completion time, *Lecture Notes in Computer Science*, **2223**(2001), 380-389.

[3] T.C.E. Cheng, C.T. Ng, J.J. Yuan and Z.H. Liu, Single machine parallel batch scheduling subject to precedence constraints, *Naval Research Logistics*, **51**(2004), 949-958.

[4] 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 Discrete Mathematics*, **5**(1979), 287-326.

[5] C.Y. Lee and R. Uzsoy, Minimizing makespan on a single batch processing machine with dynamic job arrivals, *International Journal of Production Research*, **37**(1999), 219-236.

[6] C.Y. Lee, R. Uzsoy and L.A. Martin Vega, Efficient algorithms for scheduling semiconductor burn-in operations, *Operations Research*, **40**(1992), 764-775.

[7] S. Li, G. Li, X. Wang and Q. Liu, Minimizing makespan on a single batching machine with release times and non-identical job sizes, *Operations Research Letters*, **33**(2005), 157-164.

[8] Z. Liu and W. Yu, Scheduling one batch processor subject to job release dates, *Discrete Applied Mathematics*, **105**(2000), 129-136.

[9] C.T. Ng, T.C.E. Cheng and J.J. Yuan, A note on the single machine serial batching scheduling problem to minimize maximum lateness with precedence constraints, *Operations Research Letters*, **30**(2002), 66-68.

[10] C.T. Ng, T.C.E. Cheng and J.J. Yuan, The single machine batching problem with family setup times to minimize maximum lateness is strongly $NP$-hard, *Journal of scheduling*, **6**(2003), 483-490.

[11] R. Uzsoy, A single batch processing machine with non-identical job sizes, *International Journal of Production Research*, **32**(1994), 1615-1635.

[12] J.J. Yuan, Z.H. Liu, C.T. Ng and T.C.E. Cheng, The unbounded single machine parallel batch scheduling problem with family jobs and release dates to minnimize makespan, *Theoretical Computer Science*, **320**(2004), 199-212.

[13] G. Zhang, X. Cai and C.K. Wong, On-line algorithms for minimizing makespan on batch processing machines, *Naval Research Logistics*, **48**(2001), 241-258.

[14] G. Zhang, X. Cai and C.K. Wong, Optimal on-line algorithms for scheduling on parallel batch processing machines, *IIE Transactions*, **35**(2003), 175-181.