Theoretical Computer Science 412 (2011) 5333–5339

Contents lists available at ScienceDirect

Theoretical Computer Science

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

Online batch scheduling on parallel machines with delivery times

Yang Fang, Xiwen Lu *, Peihai Liu

Department of Mathematics, School of Science, East China University of Science and Technology, Shanghai 200237, China

ARTICLE INFO

Article history:
Received 25 January 2011
Received in revised form 19 May 2011
Accepted 6 June 2011
Communicated by D.-Z. Du

Keywords:
Online algorithm
Batch scheduling
Parallel machines
Delivery time

ABSTRACT

We study the online batch scheduling problem on parallel machines with delivery times. Online algorithms are designed on m parallel batch machines to minimize the time by which all jobs have been delivered. When all jobs have identical processing times, we provide the optimal online algorithms for both bounded and unbounded versions of this problem. For the general case of processing time on unbounded batch machines, an online algorithm with a competitive ratio of 2 is given when the number of machines $m = 2$ or $m = 3$, respectively. When $m \geq 4$, we present an online algorithm with a competitive ratio of $1.5 + o(1)$.

1. Introduction

In this paper, we consider the online scheduling problem on m parallel batch machines to minimize the time by which all jobs have been delivered. For each job J_j, it has a release time r_j, a processing time p_j and a delivery time q_j. There are m parallel batch machines and sufficient number of vehicles. Once a job finishes its processing on a batch machine, it should be delivered to its destination by a vehicle. Let C_j be the completion time on the batch machine of J_j, and L_j the time by which J_j has been delivered, i.e., $L_j = C_j + q_j$. Our goal is to minimize the time $L_{\max}$, by which all jobs have been delivered, i.e., $L_{\max} = \max_j \{L_j : L_j = C_j + q_j\}$. This problem can be described as $Pm|r_j, q_j, B, online|L_{\max}$.

In classic machine scheduling problems, a machine can process at most one job at a time. Lee et al. [3] introduced the batch scheduling model. In our paper, the batch model is a burn-in model. A batch machine can process up to B jobs simultaneously as a batch. The processing time of a batch B_i is the longest processing time of all jobs in the batch, and jobs in B_i have the same beginning time and completion time. Once a batch starts to be processed, we cannot stop it. The batch machine could be bounded $B < +\infty$ if the bound B of each batch size is finite, or unbounded $B = +\infty$ if B is sufficiently large.

In this paper, the online setting is over-time. Each job becomes available until its release time which is not known in advance. Once a job J_j arrives, all its characteristics are known and it could be considered to be processed. Even the number of jobs n is unknown until the last job has been scheduled.

The standard measure of quality of online algorithms is competitive ratio. For minimum optimal problem, an online algorithm is called ρ-competitive if, for any instance, the cost output by the online algorithm is at most ρ times the optimal offline cost. The competitive ratio of an online algorithm is defined as the infimum of all values ρ. Moreover, if there are is not an online algorithm with competitive ratio less than L for some problem, we call the lower bound of this problem as L. If the competitive ratio ρ of an algorithm for this online problem matches the lower bound, i.e. $\rho = L$, we call the algorithm optimal, or best possible.

Hoogeveen and Vestjens [2] first study the online problem $1|r_j, q_j, online|L_{\max}$. They show that the lower bound is $(\sqrt{5} + 1)/2$, and give the best possible algorithm. On identical parallel machines, the lower bound could not be smaller

* Corresponding author. Tel.: +86 21 64253262.
E-mail address: xwlu@ecust.edu.cn (X. Lu).

0304-3975/$ – see front matter
doi:10.1016/j.tcs.2011.06.011

than 1.5 (see Vestjens [8]). Hall and Shmoys show that the competitive ratio of the LS algorithm is 2. Liu [5] gives a 1.618-competitive algorithm for two machines.

For the batch version $1|r_j, q_j, B, online|L_{\max}$, it is easy to see that the lower bound could not be smaller than $(\sqrt{5}+1)/2$ even if $p_j = 1$ and $q_j = 0$ [1,9]. Tian et al. [6] give a 2-competitive algorithm for the unbounded case and a 3-competitive algorithm for the bounded case. When all jobs' processing times are the same, i.e. $p_j = p$, they provide optimal algorithms for both bounded and unbounded cases with competitive ratios of $(\sqrt{5}+1)/2$.

Zhang et al. [9] address the problem $Pm|r_j, B = +\infty, online|C_{\max}$. They give a lower bound $\sqrt[m+1]{2}$, and present an online algorithm with a competitive ratio of $1+\alpha_m$, where $\alpha_m = (1-\alpha_m)^{m-1}$. When all jobs have identical processing times, Zhang et al. [10] show that the lower bound could not be smaller than $1+\beta_m$ for the unbounded case and $(\sqrt{5}+1)/2$ for the bounded case, where $(1+\beta_m)^{m+1} = 2+\beta_m$. They also provide the optimal algorithms for both cases. For the general case on unbounded machines, Liu et al. [4] and Tian et al. [7] prove that the lower bound is $1+(\sqrt{m^2+4}-m)/2$, and give different optimal algorithms independently. Thus the problem $Pm|r_j, B = +\infty, online|C_{\max}$ is settled.

Here are some notations used in the paper. For any job set J, denote by $r(J)$, $p(J)$, and $q(J)$, the minimum release time, the largest processing time, and the largest delivery time of jobs in J, respectively. Let $S(B_i)$ be the beginning time of batch B_i. For any batch B_i, denote by $J_{(i)}(p_{(i)}, q_{(i)})$ and $J^*_{(i)}(p^*_{(i)}, q^*_{(i)})$, the longest job and the job having the largest delivery time of B_i, i.e., $p_{(i)} = p(B_i)$ and $q^*_{(i)} = q(B_i)$. We also use $U(t)$ to denote the set of all unscheduled jobs available at time t. Let $\phi = (\sqrt{5}-1)/2$.

Inspired by [6,10], we study the problems $Pm|r_j, p_j = p, q_j, B < +\infty, online|L_{\max}$ and $Pm|r_j, p_j = p, q_j, B = +\infty, online|L_{\max}$ in Sections 2 and 3, and give the optimal online algorithms respectively. In Section 4, we consider $Pm|r_j, q_j, B = +\infty, online|L_{\max}$. When $m = 2$, a 2-competitive algorithm H_2 is given. Improving from the algorithm H_2, we get an algorithm H_m for the case $m \geq 3$. And the competitive ratio of H_m is $1.5 + o(1)$, which is 1.5 when m intends to infinity.

2. The bounded case with identical processing times

In this section, we assume that all jobs have the same processing times on bounded batch machines. This problem can be expressed as $Pm|r_j, p_j = p, q_j, B < +\infty, online|L_{\max}$. A batch is called *full* if it contains exactly B jobs. Otherwise, it is *non-full*. Without loss of generality, we assume that the first job arrives at time 0.

It is well known that, for offline version of bounded batch scheduling problem on a single machine with identical release times, the FBLPT (Full-Batch Longest Processing Time) rule can be used to achieve the minimal makespan. It is also used to design an online algorithm. Similar to the FBLPT rule, we have the FBLDT (Full-Batch Largest Delivery Times) rule, and provide an algorithm H^{off} based on the FBLDT rule.

H^{off}: Index the jobs such that $q_1 \geq q_2 \geq \cdots \geq q_n$. Group the first B jobs as a batch, the next B jobs as another batch and so on(The last batch may be non-full). Schedule the batches one by one whenever there is an idle machine.

Theorem 1. *For the problem $Pm|p_j = p, q_j, B < +\infty|L_{\max}$, the algorithm H^{off} is optimal.*

Proof. Since all jobs have the same processing times, the job having larger delivery time should be processed earlier. It is easy to get an optimal schedule matching H^{off} by pairwise interchange argument. □

By Theorem 1, it is better to schedule the jobs with larger delivery times. Now we give an online algorithm based on the FBLDT rule. By this algorithm, the schedule begins at $t = \phi p$.

Algorithm H^B

Set $t = \phi p$. Repeat:

Apply FBLDT to $U(t)$. If there are more than m batches, then process the first m batches in $[t, t+p]$ among all available batches. Otherwise, process all the available batches. Let $t = t + p$.

Theorem 2. *For the problem $Pm|r_j, p_j = p, q_j, B < +\infty, online|L_{\max}$, Algorithm H^B is optimal.*

Proof. Let σ and π be the schedule obtained by Algorithm H^B and an optimal schedule. According to Algorithm H^B, we know that any batch in σ starts at $(\phi+i)p$, where $i \geq 0$ and is an integer. Let J_l be the first job such that $L_l(\sigma) = L_{\max}(\sigma)$, which starts at $(\phi+k)p$. Without changing the value obtained by Algorithm H^B and increasing the optimal value, we can remove all jobs processed at or after $(\phi+k)p$ except J_l. Suppose that $r_l < (\phi+k)p$. Otherwise, we have $L_{\max}(\sigma) = (\phi+k+1)p+q_l$ and $L_{\max}(\pi) \geq (\phi+k+1)p+q_l$, which implies that σ is an optimal schedule.

For convenience, we denote by I_i the interval $[(\phi+i)p, (\phi+i+1)p)$ $(0 \leq i \leq k)$.

Let I_x be the last interval before J_l such that during I_x in σ at least one of the following conditions holds:

1. there is at least one idle machine,
2. there is at least one non-full batch,
3. there is at least one job with a delivery time smaller than q_l.

If I_x exists, let $G(l)$ denote the set which consists of job J_l and all jobs between I_x and J_l in σ. Otherwise, let $G(l)$ denote the set of job J_l and all jobs scheduled before J_l in σ. Due to the definition of I_x, all jobs in $G(l)$ have delivery time no less than q_l. We shall refer to I_x as the interference interval for the schedule σ as it may delay the start times of the jobs in $G(l)$.

If I_x exists, then by the definition of $G(l)$, all jobs in $G(l)$ are released after $(\phi + x)p$. We have that $L_{max}(\pi) \geq (\phi + x)p + (k - x)p + q_l = (\phi + k)p + q_l$ and $L_{max}(\sigma) = (\phi + k + 1)p + q_l$, where $k \geq 1$. Hence, we have

$$\frac{L_{\max}(\sigma)}{L_{\max}(\pi)} \leq \frac{(\phi + k + 1)p + q_l}{(\phi + k)p + q_l} \leq 1 + \frac{p}{(\phi + 1)p} = 1 + \phi.$$

Which means that $L_{max}(\sigma) \leq (1 + \phi)L_{max}(\pi)$.

If I_x does not exist, then due to the definition of $G(l)$, we have that $L_{max}(\pi) \geq (k + 1)p + q_l$ and $L_{max}(\sigma) = (\phi + k + 1)p + q_l$. Thus $L_{max}(\sigma) - L_{max}(\pi) \leq \phi p \leq \phi L_{max}(\pi)$ which means that $L_{max}(\sigma) \leq (1 + \phi)L_{max}(\pi)$.

Since the lower bound is $1 + \phi$, Algorithm H^B is optimal. □

3. The unbounded case with identical processing times

In this section, we deal with the problem $Pm|r_j, p_j = p, q_j, B = +\infty, online|L_{\max}$. According to [10], the lower bound cannot be smaller than $1 + \beta_m$, where $(1 + \beta_m)^{m+1} = 2 + \beta_m$. Since B is infinite, we just need to decide when to start unscheduled jobs. Without loss of generality, we assume that the first job arrives at time 0.

Similar to H^B, we consider to start a batch at time $[(1 + \beta_m)^k - 1]p$. Since $(1 + \beta_m)^{m+1} = 2 + \beta_m$ and $k \geq 1$, we have

$$(1 + \beta_m)^{k+m} - 1 = (1 + \beta_m)^{k-1}(2 + \beta_m) - 1 = (1 + \beta_m)^k + (1 + \beta_m)^{k-1} - 1 \geq (1 + \beta_m)^k.$$

Thus, the batch, which starts at $[(1 + \beta_m)^k - 1]p$, will complete not after $[(1 + \beta_m)^{k+m} - 1]p$. The algorithm and its competitive analysis are as follows.

Algorithm H^∞

Step 1. Set $k = 1$, $t = \beta_m p$.

Step 2. If $|U(t)| > 0$, put all jobs of $U(t)$ in batch B_k and start B_k at time t on machine k (mod m). Otherwise, $B_k = \emptyset$.

Step 3. Set $k = k + 1$, and $t = [(1 + \beta_m)^k - 1]p$. Return to Step 2.

According to Algorithm H^∞, batch $B_{(i-1)m+j}$ should be assigned on machine M_j. We have proved that the batch which starts at $[(1 + \beta_m)^k - 1]p$ will complete before or at the time $[(1 + \beta_m)^{k+m} - 1]p$. Therefore, the batch B_{k+m} can start its processing on the same machine at $[(1 + \beta_m)^{k+m} - 1]p$. The beginning time of any batch B_i is $[(1 + \beta_m)^i - 1]p$.

Theorem 3. *The competitive ratio of algorithm H^∞ is $1 + \beta_m$.*

Proof. Let σ and π be the schedule produced by H^∞ and an optimal schedule. Choose any job J_j, and denote the batch containing it by B_k. We have $r_j > S(B_{k-1})$ (set $S(B_0) = 0$). Note that $L_j(\sigma) = (1 + \beta_m)^k p + q_j$ and $L_j(\pi) \geq (1 + \beta_m)^{k-1}p + q_j$. Thus,

$$L_j(\sigma)/L_j(\pi) \leq 1 + \beta_m, \forall J_j$$

Therefore, $L_{\max}(\sigma)/L_{\max}(\pi) \leq 1 + \beta_m$.

Since the lower bound cannot be smaller than $1 + \beta_m$, Algorithm H^∞ is optimal. Its competitive ratio is $1 + \beta_m$. □

4. The general case on unbounded machines

In this section, we first give a 2-competitive online algorithm for $P2|r_j, q_j, B = +\infty, online|L_{\max}$. Then, for the case $m \geq 3$, we provide an algorithm with a competitive ratio of $1.5 + o(1)$, which is not greater than 2.

We partition $U(t)$ into two sets $A(t)$ and $B(t)$.

$$A(t) = \{J_j | q_j \geq \alpha p_j, J_j \in U(t)\},$$
$$B(t) = \{J_j | q_j < \alpha p_j, J_j \in U(t)\},$$

where α depends on m. When $m = 2$, set $\alpha = \phi$.

Now we give an algorithm for $P2|r_j, q_j, B = +\infty, online|L_{\max}$. Denote two machines by M_1 and M_2.

Algorithm H_2

(1) When M_1 is idle and $A(t)$ is not empty, start all jobs in $A(t)$ at t on M_1 if $t \geq (1 + \phi)r(A(t)) + \phi p(A(t))$. Otherwise, wait.

(2) When M_2 is idle and $B(t)$ is not empty, start all jobs in $B(t)$ at t on M_2 if $t \geq (1 + \phi)r(B(t)) + \phi p(B(t))$. Otherwise, wait.

By Algorithm H_2, the jobs of $A(t)$ are assigned to be processed on M_1 and jobs of $B(t)$ on M_2. For convenience, let A_i and B_i be the batch processed on M_1 and M_2, respectively. By the algorithm, jobs of B_i arrive after $S(B_{i-1})$. If $S(B_i) = (1+\phi)r(B_i) + \phi p(B_i)$, we call B_i is a regular batch. Otherwise, it is called a non-regular batch. For a non-regular batch B_i, $S(B_i) = S(B_{i-1}) + p(B_{i-1})$ and $S(B_i) > (1+\phi)r(B_i) + \phi p(B_i)$. There are similar definitions and properties for batch A_i on M_1.

Denote by σ and π, the schedule produced by H_2 and an optimal schedule. We have an important property about the schedule σ according to the following lemma given in [10]:

Lemma 1 ([10]). *If B_i is a regular batch, then B_{i+1} or B_{i+2} is also a regular batch.*

This lemma also holds for batches on M_1. According to Lemma 1, the consecutive batches B_i and B_{i+1} both cannot be non-regular batches. Note that $\phi + \phi^2 = 1$. If B_i is regular and B_{i+1} is not, we have $S(B_i) + p_{(i)} = S(B_{i+1}) > (1+\phi)S(B_i) + \phi p_{(i+1)}$. Then, $p_{(i)} > \phi S(B_i) + \phi p_{(i+1)} \geq \phi^2 p_{(i)} + \phi p_{(i+1)}$. Therefore, $(1-\phi^2)p_{(i)} > \phi p_{(i+1)}$, i.e. $p_{(i)} > p_{(i+1)}$.

Let J_v be the first job such that $L_v(\sigma) = L_{\max}(\sigma)$. Now we check all possible cases.

Lemma 2. *If J_v is processed on M_1, $L_{\max}(\sigma)/L_{\max}(\pi) \leq 2$.*

Proof. Assume that J_v belongs to batch A_l, then $J_v = J^*_{(l)}$.

If $J_{(l)} = J^*_{(l)}$ and A_l is a regular batch, we have $L_{max}(\sigma) = (1+\phi)(r(A_l) + p_{(l)}) + q_{(l)}$ and $L_{\max}(\pi) \geq r(A_l) + p_{(l)} + q_{(l)}$. Therefore, $L_{\max}(\sigma)/L_{\max}(\pi) \leq 1+\phi$.

If $J_{(l)} = J^*_{(l)}$ and A_l is a non-regular batch, we have $L_{max}(\sigma) = S(A_{l-1}) + p_{(l-1)} + p_{(l)} + q_{(l)}$ and $L_{\max}(\pi) \geq S(A_{l-1}) + p_{(l)} + q_{(l)}$. Therefore, $L_{\max}(\sigma) - L_{\max}(\pi) \leq p_{(l-1)} \leq L_{\max}(\pi)$.

Thus, if $J_{(l)} = J^*_{(l)}$ we can get $L_{\max}(\sigma)/L_{\max}(\pi) \leq 2$. We continue our proof under the assumption $J_{(l)}$ is not $J^*_{(l)}$.

Case 1. If A_l is regular, then $L_{\max}(\sigma) = S(A_l) + p_{(l)} + q^*_{(l)} = (1+\phi)(r(A_l) + p_{(l)}) + q^*_{(l)}$. For the optimal value, we have $L_{\max}(\pi) \geq r(A_l) + p_{(l)} + q_{(l)} \geq r(A_l) + (1+\phi)p_{(l)}$ and $L_{\max}(\pi) \geq r(A_l) + q^*_{(l)}$. Thus, $2L_{\max}(\pi) \geq 2r(A_l) + (1+\phi)p_{(l)} + q^*_{(l)} \geq L_{\max}(\sigma)$.

Case 2. If A_l is not regular, then $L_{\max}(\sigma) = S(A_{l-1}) + p_{(l-1)} + p_{(l)} + q^*_{(l)}$. Note that $q_{(l)} \geq \phi p_{(l)}$ and $q_{(l-1)} \geq \phi p_{(l-1)}$. For the optimal value, we have (1) $L_{\max}(\pi) \geq S(A_{l-1}) + p_{(l)} + q_{(l)} \geq \phi p_{(l-1)} + (1+\phi)p_{(l)}$; (2) $L_{\max}(\pi) \geq r(A_{l-1}) + p_{(l-1)} + q_{(l-1)} \geq (1+\phi)p_{(l-1)}$; (3) $L_{\max}(\pi) \geq S(A_{l-1}) + q^*_{(l)}$.

By (1) $\times\ \phi$ + (2) $\times\ \phi^2$ + (3), we have

$$\begin{aligned} 2L_{\max}(\pi) &\geq S(A_{l-1}) + \phi^2 p_{(l-1)} + p_{(l)} + \phi p_{(l-1)} + q^*_{(l)} \\ &= S(A_{l-1}) + p_{(l-1)} + p_{(l)} + q^*_{(l)} = L_{\max}(\sigma). \end{aligned}$$

The proof is complete. □

Lemma 3. *If J_v is processed on M_2, $L_{\max}(\sigma)/L_{\max}(\pi) \leq 2$*

Proof. Assume that J_v belongs to batch B_l; then J_v=$J^*_{(l)}$. Obviously, when $J_{(l)} = J^*_{(l)}$, we can get $L_{\max}(\sigma)/L_{\max}(\pi) \leq 2$ by making similar analysis as Lemma 2. The following proof is under the assumption that $J_{(l)}$ is not $J^*_{(l)}$.

Case 1. B_l is regular. $L_{\max}(\sigma) = (1+\phi)(r(B_l) + p_{(l)}) + q^*_{(l)}$. Note that $p^*_{(l)} \geq (1+\phi)q^*_{(l)}$, we have (1) $L_{\max}(\pi) \geq r(B_l) + p^*_{(l)} + q^*_{(l)} \geq r(B_l) + (2+\phi)q^*_{(l)}$ and (2) $L_{\max}(\pi) \geq r(B_l) + p_{(l)}$.

By (1) $\times\ \phi^2$ + (2) $\times\ (1+\phi)$, we have

$$2L_{\max}(\pi) \geq 2r(B_l) + (1+\phi)p_{(l)} + q^*_{(l)} \geq L_{\max}(\sigma).$$

Case 2. B_l is not regular. $L_{\max}(\sigma) = S(B_{l-1}) + p_{(l-1)} + p_{(l)} + q^*_{(l)}$. Without changing the value obtained by the algorithm and increasing the optimal value, we assume that there is only one job $J_{(l-1)}$ $(r(B_{l-1}), p_{(l-1)}, 0)$ in B_{l-1}. Note that $p^*_{(l)} \geq (1+\phi)q^*_{(l)}$ and $p_{(l-1)} > p_{(l)}$.

1. $J_{(l)}$ and $J^*_{(l)}$ are scheduled on the same machine in π. If they are scheduled in the same batch, then $L_{\max}(\pi) \geq S(B_{l-1}) + p_{(l)} + q^*_{(l)}$. Otherwise, $L_{\max}(\pi) \geq S(B_{l-1}) + p_{(l)} + p^*_{(l)} \geq S(B_{l-1}) + p_{(l)} + q^*_{(l)}$. Therefore, $L_{\max}(\sigma) - L_{\max}(\pi) \leq p_{(l-1)} \leq L_{\max}(\pi)$.
2. $J_{(l-1)}$ and $J^*_{(l)}$ are scheduled on the same machine in π. If they are scheduled in the same batch, then $L_{\max}(\pi) \geq p_{(l-1)} + q^*_{(l)}$. Otherwise, $L_{\max}(\pi) \geq p_{(l-1)} + p^*_{(l)} \geq p_{(l-1)} + q^*_{(l)}$. Therefore, $L_{\max}(\sigma) - L_{\max}(\pi) \leq S(B_{l-1}) + p_{(l)} \leq L_{\max}(\pi)$.
3. $J_{(l-1)}$ and $J_{(l)}$ are scheduled on the same machine in π. If they are scheduled in the same batch, then $L_{\max}(\pi) \geq p_{(l-1)} + S(B_{l-1}) \geq p_{(l-1)} + \phi p_{(l-1)} \geq p_{(l-1)} + \phi p(l)$. Otherwise, $L_{\max}(\pi) \geq p_{(l-1)} + p_{(l)} \geq p_{(l-1)} + \phi p_{(l)}$. Since $p_{(l)} \geq p^*_{(l)} \geq (1+\phi)q^*_{(l)}$, we can derive that $L_{\max}(\pi) \geq p_{(l-1)} + q^*_{(l)}$. Therefore, $L_{\max}(\sigma) - L_{\max}(\pi) \leq S(B_{l-1}) + p_{(l)} \leq L_{\max}(\pi)$.

Hence we get this lemma. □

Theorem 4. *The competitive ratio of Algorithm H_2 is 2.*

Proof. By Lemmas 2 and 3, we know that the competitive ratio of Algorithm H_2 is at most 2. Now we give an instance to show that the bound is tight. At time 0, two jobs $J_1(1, \phi)$ and $J_2(0, 1+\phi)$ arrive. They are put in the same batch and started at time ϕ. In the optimal schedule, J_2 is processed before J_1 at time 0. $L_{\max}(\sigma) = 2(1+\phi)$, and $L_{\max}(\pi) = 1+\phi$. Therefore, the competitive ratio of Algorithm H_2 is 2. □

Similar to the idea of Algorithm H_2, we get the following algorithm. If m is odd, let $m = 2k+1$. Otherwise, let $m = 2k+2$, where $k \geq 1$ and is an integer. Note that $\lceil m/2 \rceil = k+1$. When $m = 2k+1$, $\alpha = \frac{k^2+k-1}{k^2+2k+1}$. When $m = 2k+2$, $\alpha = \frac{k+1}{k+2}$.

Algorithm H_m:

(1) When machine M_j of $M_1, M_2, \ldots, M_{\lceil m/2 \rceil}$ is idle and $A(t) \neq \emptyset$, make decision as follows. If $t \geq (1+\delta)r(A(t)) + \delta p(A(t))$, process all jobs in $A(t)$ at t on M_j. Otherwise, wait.

(2) When machine M_j of $M_{\lceil m/2 \rceil+1}, \ldots, M_m$ is idle and $B(t) \neq \emptyset$, make decision as follows. If $t \geq (1+\hat{\delta})r(B(t)) + \hat{\delta}p(B(t))$, process all jobs in $B(t)$ at t on M_j. Otherwise, wait.

Where $\delta = 1/\lceil m/2 \rceil$ and $\hat{\delta} = 1/\lfloor m/2 \rfloor$.

Denote by σ and π, the schedule produced by H_m and an optimal schedule. According to Algorithm H_m, we can get this property about σ.

Lemma 4. *In σ, any batch B_i scheduled on machine M_j, $j = 1, \ldots, \lceil m/2 \rceil$, is a regular batch, i.e. $S(B_i) = (1+\delta)r(B_i) + \delta p(B_i)$.*

Proof. Note that $\lceil m/2 \rceil = k+1$ whether m is an odd number $2k+1$ or an even number $2k+2$. Suppose to the contrary that there exists at least one non-regular batch on machine M_j, $j = 1, \ldots, k+1$. Let B_v be the first one, which starts at $t = S(B_v) > (1+\delta)r(B_v) + \delta p(B_v)$. Thus on every machine M_j, $j = 1, \ldots, k+1$, there is one batch processed in $[r(B_v), t]$. These $k+1$ batches start before $r(B_v)$ and complete not before t. For ease of description, we denote these batches by $B'_1, \ldots, B'_{k+1}$ such that $S(B'_1) < S(B'_2) < \cdots < S(B'_{k+1})$. Since B_v is the first non-regular batch, the batches $B'_1, \cdots, B'_{k+1}$ are regular.

Since B_v is not regular, we have $\min_{1 \leq i \leq k+1} \{S(B'_i) + p'_{(i)}\} \geq S(B_v) > (1+\delta)r(B_v) + \delta p(B_v)$. By $r(B_v) > S(B'_{k+1})$, we derive that $\min_{1 \leq i \leq k+1} \{S(B'_i) + p'_{(i)}\} > (1+\delta)S(B'_{k+1})$.

We will prove that $S(B'_{k+1}) > \frac{1}{1-j\delta} S(B'_{k+1-j})$, $j = 1, \ldots, k$. (*)

In fact, when $j = 1$, we have $S(B'_{k+1}) + p'_{(k+1)} > (1+\delta)S(B'_{k+1})$, then

$$p'_{(k+1)} > \delta S(B'_{k+1}) > (1+\delta) \cdot \delta S(B'_k) + \delta^2 p'_{(k+1)}.$$

We get $p'_{(k+1)} > \frac{\delta}{1-\delta} S(B'_k)$. Therefore,

$$S(B'_{k+1}) > (1+\delta)S(B'_k) + \delta p'_{(k+1)} > \frac{1}{1-\delta} S(B'_k).$$

Then the inequality (*) is right for $j = 1$.

Now let $j \geq 2$ and the inequality (*) is true for $1, \ldots, j-1$, then $S(B'_{k+1}) > \frac{1}{1-(j-1)\delta} S(B'_{k+2-j})$. Thus,

$$S(B'_{k+2-j}) + p'_{(k+2-j)} > (1+\delta)S(B'_{k+1}) > \frac{1+\delta}{1-(j-1)\delta} S(B'_{k+2-j}).$$

We have

$$p'_{(k+2-j)} > \frac{j\delta}{1-(j-1)\delta} S(B'_{k+2-j}) > \frac{j\delta}{1-(j-1)\delta} [(1+\delta)S(B'_{k+1-j}) + \delta p'_{(k+2-j)}].$$

This implies that

$$\frac{(1+\delta)(1-j\delta)}{1-(j-1)\delta} p'_{(k+2-j)} > \frac{j\delta(1+\delta)}{1-(j-1)\delta} S(B'_{k+1-j}).$$

Then, $p'_{(k+2-j)} > \frac{j\delta}{1-j\delta} S(B'_{k+1-j})$.

Therefore,

$$S(B'_{k+2-j}) > (1+\delta)S(B'_{k+1-j}) + \delta p'_{(k+2-j)} > \frac{1-(j-1)\delta}{1-j\delta} S(B'_{k+1-j}).$$

Reminding of $S(B'_{k+1}) > \frac{1}{1-(j-1)\delta} S(B'_{k+2-j})$, we can get $S(B'_{k+1}) > \frac{1}{1-j\delta} S(B'_{k+1-j})$. Hence the inequality (*) is true.

According to the inequality (*) and $\delta = 1/(k+1)$, we have

$$(1+\delta)S(B'_{k+1}) - S(B'_1) - p'_{(1)} > \frac{1+\delta}{1-k\delta} S(B'_1) - S(B'_1) - p'_{(1)}$$

$$= (k+1)S(B'_1) - p'_{(1)}$$
$$\geq (k+1)\delta p'_{(1)} - p'_{(1)} = 0.$$

Therefore, we get $(1+\delta)S(B'_{k+1}) > S(B'_1) + p'_{(1)}$, which contradicts

$$\min_{1\leq i\leq k+1}\{S(B'_i) + p'_{(i)}\} > (1+\delta)S(B'_{k+1}).$$

This means that B_v is regular. Therefore, any batch B_i scheduled on machine M_j is a regular batch, for $j = 1, \ldots, k+1$. □

Note that the proof of Lemma 4 has no connection with q_j. Hence, by making similar analysis, we can derive that any batch on $M_j, j = \lceil m/2 \rceil + 1, \ldots, m$, is regular. Therefore, we have the following lemma:

Lemma 5. *In σ which is produced by Algorithm H_m, any batch is regular.*

Let J_v be the first job such that $L_v(\sigma) = L_{\max}(\sigma)$. Assuming that J_v belongs to batch B_l, then $J_v = J^*_{(l)}$. We make competitive analysis by checking all possible cases where B_l is scheduled.

Lemma 6. *If B_l is scheduled on $M_j, j = 1, \ldots, \lceil m/2 \rceil$, then $L_{\max}(\sigma) \leq (1+R_1)L_{\max}(\pi)$, where $R_1 = (1+\delta)/(1+\alpha)$.*

Proof. Since B_l is regular, we have $L_{\max}(\sigma) = (1+\delta)(r(B_l) + p_{(l)}) + q^*_{(l)}$. Since $\alpha \leq 1$ and $\delta \leq 1$, we have $\alpha\delta + \delta \leq 1 + \delta$. Then $\delta \leq (1+\delta)/(1+\alpha) = R_1$.

For the optimal value, we have (1) $L_{\max}(\pi) \geq r(B_l) + p_{(l)} + q_{(l)} \geq r(B_l) + (1+\alpha)p_{(l)}$ and (2) $L_{\max}(\pi) \geq r(B_l) + q^*_{(l)}$. Then by (1) $\times R_1 +$ (2), we get

$$\begin{aligned}(1+R_1)L_{\max}(\pi) &\geq (1+R_1)r(B_l) + (1+\delta)p_{(l)} + q^*_{(l)} \\ &\geq (1+\delta)r(B_l) + (1+\delta)p_{(l)} + q^*_{(l)} = L_{\max}(\sigma).\end{aligned}$$

The result holds. □

Lemma 7. *If B_l is scheduled on $M_j, j = \lceil m/2 \rceil + 1, \ldots, m$, then $L_{\max}(\sigma) \leq (1+R_2)L_{\max}(\pi)$, where $R_2 = \hat{\delta} + \alpha/(1+\alpha)$.*

Proof. Since B_l is regular, we have $L_{\max}(\sigma) = (1+\hat{\delta})(r(B_l) + p_{(l)}) + q^*_{(l)}$. For the optimal value, we have (1) $L_{\max}(\pi) \geq r(B_l) + p_{(l)}$ and (2) $L_{\max}(\pi) \geq r(B_l) + p^*_{(l)} + q^*_{(l)} \geq \frac{1+\alpha}{\alpha} q^*_{(l)}$. Then by (1) $\times (1+\hat{\delta})$ + (2) $\times \alpha/(1+\alpha)$, we can derive that

$$(1+R_2)L_{\max}(\pi) \geq (1+\hat{\delta})(r(B_l) + p_{(l)}) + q^*_{(l)} = L_{\max}(\sigma).$$

The result follows. □

Theorem 5. *The competitive ratio of Algorithm H_m is*

$$\rho = \begin{cases} \rho_1 = 1 + \frac{(k+1)(k+2)}{k(2k+3)}, & \text{if } m = 2k+1 \\ \rho_2 = 1 + \frac{(k+2)^2}{(k+1)(2k+3)}, & \text{if } m = 2k+2, \end{cases}$$

where $k \geq 1$ and is an integer.

Proof. When $m = 2k+1$, $\delta = 1/(k+1)$, $\hat{\delta} = 1/k$, the competitive ratio of Algorithm H_m is at most

$$\rho = 1 + \max\{R_1, R_2\} = 1 + \max\left\{\frac{1}{1+\alpha} \times \frac{k+2}{k+1}, \frac{1}{k} + \frac{\alpha}{(1+\alpha)}\right\}.$$

When $\alpha = \frac{k^2+k-1}{k^2+2k+1}$, ρ achieves its minimal value

$$\rho_1 = 1 + \frac{(k+1)(k+2)}{k(2k+3)}.$$

When $m \to +\infty$, $\rho_1 \to 1.5$.

When $m = 2k+2$, $\delta = 1/(k+1)$, $\hat{\delta} = 1/(k+1)$, the competitive ratio of Algorithm H_m is at most

$$\rho = 1 + \max\{R_1, R_2\} = 1 + \max\left\{\frac{1}{1+\alpha} \times \frac{k+2}{k+1}, \frac{1}{k+1} + \frac{\alpha}{(1+\alpha)}\right\}.$$

When $\alpha = \frac{k+1}{k+2}$, ρ achieves its minimal value

$$\rho_2 = 1 + \frac{(k+2)^2}{(k+1)(2k+3)}.$$

When $m \to +\infty$, $\rho_2 \to 1.5$.

Similar to the one in Theorem 4, we can present an instance to show that the bound ρ is tight. □

It is easy to see that $\rho < 2$ when $m \geq 4$. When $m = 3$, $\rho = 2.2 > 2$. We can easily make a small modification for Algorithm H_m for $m = 3$: (1) set $\alpha = 1$; (2) process jobs in $A(t)$ on M_1 if $t \geq (1+\delta)r(A(t)) + p(A(t))$; (3) process jobs in $B(t)$ on M_2 or M_3 if $t \geq (1+\hat{\delta})r(B(t)) + \hat{\delta}(B(t))$, where $\delta = 1$ and $\hat{\delta} = 1/2$. Then, we can get its competitive ratio as 2 by making a similar proof.

Our results in this section can be described as follows:

1. When $m = 2$, upper bound is 2.
2. When $m = 3$, upper bound is 2.
3. When $m = 2k+1$, $k \geq 2$, upper bound is $\rho_1 = 1 + \frac{(k+1)(k+2)}{k(2k+3)}$.
4. When $m = 2k+2$, $k \geq 1$, upper bound is $\rho_2 = 1 + \frac{(k+2)^2}{(k+1)(2k+3)}$.

When m tends to $+\infty$, both ρ_1 and ρ_2 tend to 1.5.

Acknowledgements

This research was supported by the grant 09ZR1407200 of the Science Foundation of Shanghai and the National Natural Science Foundation of China (No. 11071072).

References

[1] X. Deng, C.K. Poon, Y. Zhang, Approximation algorithms in batch processing, Jounal of Combinatorial Optimizations 7 (2003) 247–257.
[2] J.A. Hoogeveen, A.P.A. Vestjean, A best possible deterministic online algorithm for minimizing maximum delivery times on a single machine, SIAM Journal on Discrete Mathematics 13 (2000) 56–63.
[3] C.Y. Lee, R. Uzsoy, L.A. MartinVega, Efficient algorithms for scheduling semiconductor burn-in operations, Operations Research 40 (1992) 764–775.
[4] P. Liu, X. Lu, Y. Fang, A best possible deterministic online algorithm for minimizing makespan on parallel batch machines. Journal of Scheduling, doi:10.1007/s10951-009-0154-4.
[5] P. Liu, Personal Communication.
[6] J. Tian, R. Fu, J. Yuan, Online scheduling with delivery time on a single batch machine, Theoretical Computer Science 374 (2007) 49–57.
[7] J. Tian, T.C.E. Cheng, C.T. Ng, J. Yuan, Online scheduling on unbounded parallel-batch machines to minimize the makespan, Information Processing Letters 109 (2009) 1211–1215.
[8] A.P.A. vestjens, online machine scheduling. Ph.D. Thesis, Department of mathematics and Computing Science, Eindhoven University of Techology, Eindhoven, The Netherlands, 1997.
[9] G. Zhang, X. Cai, C.K. Wong, online algorithms for minimizing makespan on batch processing machines, Naval Research Logistics 48 (2001) 241–258.
[10] G. Zhang, X. Cai, C.K. Wong, Optimal online algorithms for scheduling on parallel batch processing machines, IIE Transactions 35 (2003) 175–181.