J Comb Optim (2009) 17: 206–213
DOI 10.1007/s10878-007-9108-y

A best on-line algorithm for the single machine parallel-batch scheduling with restricted delivery times

Jinjiang Yuan · Shisheng Li · Ji Tian · Ruyan Fu

Published online: 7 November 2007

Abstract We consider a single batch machine on-line scheduling problem with delivery times. In this paper *on-line* means that jobs arrive over time and the characteristics of jobs are unknown until their arrival times. Once the processing of a job is completed it is delivered to the destination. The objective is to minimize the time by which all jobs have been delivered. For each job J_j, its processing time and delivery time are denoted by p_j and q_j, respectively. We consider two restricted models: (1) the jobs have small delivery times, i.e., for each job J_j, $q_j \leq p_j$; (2) the jobs have agreeable processing and delivery times, i.e., for any two jobs J_i and J_j, $p_i > p_j$ implies $q_i \geq q_j$. We provide an on-line algorithm with competitive ratio $(\sqrt{5}+1)/2$ for both problems, and the results are the best possible.

Keywords Scheduling · On-line algorithm · Parallel-batching · Delivery time

1 Introduction

In the last decade, on-line scheduling and parallel-batch scheduling have been extensively studied. Here, *on-line* means that jobs arrive over time, and all characteristics of a job is unknown until its arrival time.

The quality of an on-line algorithm is measured by the competitive ratio, which is defined by

$$R = \sup_{\forall L} \{C_{\text{on}}(L)/C_{\text{opt}}(L)\}.$$

Project supported by NSFC (10671183).

J. Yuan (✉) · S. Li · T. Tian · R. Fu
Department of Mathematics, Zhengzhou University, Zhengzhou, Henan 450052, People's Republic of China
e-mail: yuanjj@zzu.edu.cn

Here, $C_{on}(L)$ denotes the objective value of the schedule given by the on-line algorithm for job list L, whereas $C_{opt}(L)$ denotes the objective value in an optimal off-line schedule for L. A new measure called the relative ratio for the quality of on-line algorithms has been introduced by Epstein et al. (2006).

Parallel batch scheduling means that a machine can process several jobs simultaneously as a batch. All jobs in a common batch have the same starting time and completion time. The processing time of a batch is equal to the longest processing time of the jobs in the batch. This framework was first introduced by Lee et al. (1992), which was motivated by burn-in operations in semiconductor manufacturing. There are two categories about parallel batch scheduling. One is the bounded model in which the batch size b is finite, i.e., $b < n$. The other is the unbounded model in which the batch size b is sufficiently large, i.e., $b = \infty$.

For the scheduling problem $1|p\text{-}batch, b < n|C_{\max}$, an optimal schedule can be found by the *FBLPT* rule of Bartholdi (1988) (Brucker et al. 1998; Lee and Uzsoy 1999). For the scheduling problem $1|p\text{-}batch, r_j, b|C_{\max}$, when $b = \infty$, Lee and Uzsoy (1999) presented a dynamic programming algorithm with running time $O(n^2)$; when $b < n$, Brucker et al. (1998) and Liu and Yu (2000) showed that the problem is NP-hard. Under on-line setting, the parallel batch scheduling problem was first studied by Deng et al. (2003) and Zhang et al. (2001). They independently provided on-line algorithms with competitive ratio $(\sqrt{5}+1)/2$ for $1|p\text{-}batch, on\text{-}line, b = \infty|C_{\max}$, and proved that it is the best possible. For problem $1|p\text{-}batch, on\text{-}line, b < n|C_{\max}$, Poon and Yu (2005) presented a class of algorithms called $FBLPT - based$ algorithms. They showed that any $FBLPT - based$ algorithm has competitive ratio at most 2. Especially, for the case $b = 2$, they gave an on-line algorithm with competitive ratio 7/4. However, the best on-line algorithm for this problem is still open.

The problem studied in this paper can be described as follows. We have a batch machine and sufficiently many vehicles. There are n jobs $J_1, J_2, \ldots, J_n$. Each job has a release time, a processing time, and a delivery time. All characteristics of a job are unknown until it arrives. Each job needs to be processed on the machine, and once the job is completed we deliver it to the destination by some vehicle. The objective is to minimize the time by which all jobs have been delivered. Let r_j, p_j, q_j denote the release time, the processing time, and the delivery time of job J_j, respectively. For a job set $B \subseteq \{J_1, J_2, \ldots, J_n\}$, we define $r(B) = \max\{r_j : J_j \in B\}$, $p(B) = \max\{p_j : J_j \in B\}$, and $q(B) = \max\{q_j : J_j \in B\}$. For a given schedule σ of the jobs, we denote by $C_j(\sigma)$ and $L_j(\sigma) = C_j(\sigma) + q_j$, respectively, the completion time of J_j and the time by which J_j is delivered in schedule σ.

There have been some results about on-line scheduling problem in which the objective is to minimize the time by which all jobs have been delivered. For non-batch machine model $1|on\text{-}line, q_j|L_{\max}$, Hoogeveen and Vestjens (2000) provided an on-line algorithm with competitive ratio $(\sqrt{5}+1)/2$, and it is the best possible. For parallel batch machine model $1|p\text{-}batch, on\text{-}line, q_j, b|L_{\max}$, Tian et al. (2007) provided an on-line algorithm with competitive ratio 2 for $b = \infty$ and one with competitive ratio 3 for $b < n$. In the same paper, they also studied a special model $1|p\text{-}batch, on\text{-}line, p_j = p, q_j, b|L_{\max}$. For $b = \infty$ and $b < n$, respectively, they provided the best on-line algorithms with competitive ratio $(\sqrt{5}+1)/2$. Till now, the

best possible on-line algorithm for the general problem $1|p\text{-}batch, on\text{-}line, q_j, b|L_{\max}$ is still open.

In this paper, we assume $b = \infty$ and consider two restricted models: (1) the jobs have small delivery times, i.e., for each job J_j, $q_j \leq p_j$; (2) the jobs have agreeable processing and delivery times, i.e., for any two jobs J_i and J_j, $p_i > p_j$ implies $q_i \geq q_j$. Using the 3-field notation of Graham et al. (1979), the former is denoted by $1|p\text{-}batch, on\text{-}line, p_j \geq q_j, b = \infty|L_{\max}$, and the latter is denoted by $1|p\text{-}batch, on\text{-}line, agreeable(p_j, q_j), b = \infty|L_{\max}$, where $L_{\max} = \max\{L_j : L_j = C_j + q_j, 1 \leq j \leq n\}$. We will provide an on-line algorithm with a competitive ratio $(\sqrt{5}+1)/2$ for both problems. The algorithm is the best possible.

2 A lower bound

Consider first the scheduling model $1|on\text{-}line, r_j, b = \infty|C_{\max}$, which is a special case of the scheduling problems studied in this paper. Hence, a lower bound of it is also a lower bound of the problems we study. For the former, Zhang et al. (2001) presented the following lower bound of competitive ratio for all on-line algorithms.

Lemma 1 (Zhang et al. 2001) *There is no on-line algorithm with competitive ratio less than* $1 + \alpha$ *for the scheduling problem* $1|on\text{-}line, r_j, b = \infty|C_{\max}$, *where* $\alpha = (\sqrt{5} - 1)/2$.

Corollary 2 *There is no on-line algorithm with competitive ratio less than* $1 + \alpha$ *for the following two problems*:

(1) $1|p\text{-}batch, on\text{-}line, p_j \geq q_j, b = \infty|L_{\max}$;
(2) $1|p\text{-}batch, on\text{-}line, agreeable(p_j, q_j), b = \infty|L_{\max}$.

3 An on-line algorithm

For a time instant t, let $U(t)$ be the set of all unscheduled jobs available at time t. Let $\alpha = (\sqrt{5} - 1)/2$. The on-line algorithm runs as follows:

Algorithm H

Step 0: Let t be the minimum time instant such that $U(t) \neq \emptyset$.

Step 1: Find a job $J_k \in U(t)$ such that $p_k = \max\{p_j : J_j \in U(t)\}$. Let $\alpha_k = \alpha p_k$ and set $s = \max\{t, \alpha_k\}$.

Step 2: In the time interval $(t, s]$, whenever a set of new job U' comes in, say, at time t', do the following: If there is some job $J_h \in U'$ such that $p_h \geq p_k$, then reset $k = h$ and reset α_k and s accordingly. Set $t = t'$ and repeat Step 2.

Step 3: At time s, start to process the jobs in $U(s)$ as a single batch. Reset t to be the minimum time in $[s + p_k, \infty)$ such that $U(t) \neq \emptyset$ and go to Step 1.

For a given instance $\mathcal{I}$ of the scheduling problem, we still use H to designate the schedule formed by Algorithm H. For each batch B in H, $s(B)$ and $p(B)$ represent the starting time and processing time of B, respectively. Let J^* be the first longest job in H that assumes the value $L_{\max}(H)$; and let p^* and q^* denote its processing time and delivery time, respectively. The completion time of job J^* in schedule H is denoted by C^*. Moreover, $L_{\text{opt}}(\mathcal{I})$ stands for the objective value of an optimal off-line schedule for $\mathcal{I}$.

Let B_0 be the first batch in H such that the machine has no idle time in $[s(B_0), C^*)$. Let the batches processed in $[s(B_0), C^*)$ be $B_0, B_1, \ldots, B_m$ with $s(B_0) < s(B_1) < \cdots < s(B_m)$. Then $J^* \in B_m$ and q^* is the maximum delivery time of jobs in B_m.

Theorem 3 *If either $p^* \geq q^*$ or $q_j \geq q^*$ for every job J_j with $p_j > p^*$, then $L_{\max}(H) \leq (1+\alpha)L_{\text{opt}}(\mathcal{I})$.*

Proof For each j with $0 \leq j \leq m$, let $J_j \in B_j$ be the job with $p_j = p(B_j)$ and r_j being as large as possible. By the implementation of Algorithm H, we have

$$
\begin{aligned}
&s(B_0) = \max\{r_0, \alpha p_0\},\\
&s(B_j) \geq \alpha p_j, \quad 0 \leq j \leq m,\\
&s(B_{j-1}) + p_{j-1} = s(B_j), \quad 1 \leq j \leq m,\\
&r_j > s(B_{j-1}), \quad 1 \leq j \leq m,\\
&r_0 < r_1 < \cdots < r_m.
\end{aligned}
$$

We further have

$$L_{\max}(H) = s(B_j) + p_j + p_{j+1} + \cdots + p_m + q^* \quad \text{for each } j \text{ with } 0 \leq j \leq m.$$

Without loss of generality, we assume in the sequel that the instance $\mathcal{I}$ just consists of the jobs in $B_0 \cup B_1 \cup \cdots \cup B_m$. We construct a new instance $\mathcal{I}^*$ of the scheduling problem by the following way.

The jobs appeared in $\mathcal{I}^*$ are those in $\mathcal{J}^* = \{J_0, J_1, \ldots, J_{m-1}\} \cup B_m$. Their processing times are defined by

$$p_j^* = p_j \quad \text{for each job } J_j \in \mathcal{J}^*.$$

Their release times are defined by

$$
\begin{aligned}
&r_0^* = r_0 \quad \text{if } r_0 \geq \alpha p_0, \quad \text{and} \quad r_0^* = 0 \quad \text{if } r_0 < \alpha p_0;\\
&r_j^* = s(B_{j-1}) \quad \text{for } 1 \leq j \leq m-1;
\end{aligned}
$$

and

$$r_x^* = s(B_{m-1}) \quad \text{for } J_x \in B_m.$$

The delivery times of jobs are defined by

$$q_x^* = q_x \quad \text{for } J_x \in B_m$$

and, for $0 \le j \le m-1$,

$$q_j^* = \begin{cases} 0, & \text{if } p^* \ge q^* \text{ or } p_j \le p^*, \\ q^*, & \text{otherwise.} \end{cases}$$

Claim 1 $L_{\text{opt}}(\mathcal{I}^*) \le L_{\text{opt}}(\mathcal{I})$.

Proof of Claim 1 Note that, for each job $J_j \in \mathcal{I}^*$, we have $r_j^* \le r_j$ and $p_j^* = p_j$. Furthermore, the delivery times of jobs in B_m are unchanged.

If $p^* \ge q^*$, then $q_j^* = 0 \le q_j$ for $0 \le j \le m-1$. Hence, in this case, we have $(r_j^*, p_j^*, q_j^*) \le (r_j, p_j, q_j)$ for each job $J_j \in \mathcal{I}^*$.

If $p^* < q^*$, then, by the assumption of the theorem, we have $q_j \ge q^*$ for every job J_j with $p_j > p^*$. For each j with $0 \le j \le m-1$, by the definition of q_j^*, we have

$$q_j^* = \begin{cases} 0 \le q_j, & \text{if } p_j \le p^*, \\ q^* \le q_j, & \text{if } p_j > p^*, \end{cases}$$

i.e., $q_j^* \le q_j$. Hence, in this case, we still have $(r_j^*, p_j^*, q_j^*) \le (r_j, p_j, q_j)$ for each job $J_j \in \mathcal{I}^*$.

The above argument asserts that the jobs in $\mathcal{I}^*$ are dominated by the corresponding jobs appeared in $\mathcal{I}$. Hence, Claim 1 follows. □

Claim 2 *There is an optimal schedule for instance $\mathcal{I}^*$ such that all jobs in B_m are contained in the last batch of the schedule.*

Proof of Claim 2 Let π be an optimal schedule for instance $\mathcal{I}^*$ such that the number of batches in π is as small as possible. Suppose that t is the earliest starting time of the jobs in B_m in π. Since t is at least $r^*(B_m)$ which is the maximum release time of jobs in $\mathcal{I}^*$, there are no idle-times in the time interval $[t, C_{\max}(\pi))$ in π. Suppose that the batches processed in the time interval $[t, C_{\max}(\pi))$ are in the order $B_1^*, B_2^*, \ldots, B_k^*$ in π. Then, by the choice of π, we have

$$p(B_1^*) < p(B_2^*) < \cdots < p(B_k^*).$$

(In fact, if $p(B_i^*) \ge p(B_{i+1}^*)$ for a certain i with $1 \le i \le k-1$, then, by combining B_i^* and B_{i+1}^* as a single batch $B_i^* \cup B_{i+1}^*$ starting at time $s(B_i^*)$, we obtain another optimal schedule with smaller number of batches, which contradicts the choice of π.)

We further suppose that job J^* belongs to B_x^* for some x with $1 \le x \le k$. Let σ be a schedule obtained from π by setting $B = B_1^* \cup B_2^* \cup \cdots \cup B_k^*$ as a single batch with starting time t. It can be observed that $L_{\max}(\sigma) = t + p(B) + q^* = t + p(B_k^*) + q^*$. Recall that either $p^* \ge q^*$ or $q_j \ge q^*$ for every job J_j with $p_j > p^*$. If $p^* < q^*$, then $q_j \ge q^*$ for every job J_j with $p_j > p^*$. By the definition of q_j^*, batch B has delivery time q^* in $\mathcal{I}^*$. Hence, we have $L_{\max}(\pi) \ge t + p(B) + q^* = L_{\max}(\sigma)$. This means

that σ is also an optimal schedule for the instance $\mathcal{I}^*$. By the choice of π the only possibility is $k = 1$, and so Claim 2 holds under the condition $p^* < q^*$. □

Now suppose that $p^* \geq q^*$. If $x \leq k-1$, then $p(B_x^*) \geq p^* \geq q^*$, and so

$$L_{\max}(\pi) \geq t + p(B_x^*) + p(B_k^*) \geq t + q^* + p(B_k^*) = L_{\max}(\sigma).$$

It follows that σ is also an optimal schedule for the instance $\mathcal{I}^*$. This contradicts the choice of π.

Hence, we have $x = k$. If $k \geq 2$, then

$$L_{\max}(\pi) > t + p(B_k^*) + q^* = L_{\max}(\sigma).$$

This contradicts the assumption that π is an optimal schedule. Hence, the only possibility is $k = 1$. This completes the proof of Claim 2.

Based on Claim 2, we assume in the sequel that B_m contains exactly one job J_m with $q_m^* = q^*$ in $\mathcal{I}^*$. Then we have $\mathcal{J}^* = \{J_0, J_1, \ldots, J_m\}$.

If there is an optimal schedule for $\mathcal{I}^*$ such that no two jobs in $\{J_0, J_1, \ldots, J_m\}$ belong to a common batch, then

$$L_{\text{opt}}(\mathcal{I}^*) \geq r_0^* + p_0 + p_1 + \cdots + p_m + q^*.$$

By Claim 1 and the expression of $L_{\max}(H)$, we have

$$L_{\max}(H) - L_{\text{opt}}(\mathcal{I}) \leq s(B_0) - r_0^* \leq \alpha p_0 \leq \alpha L_{\text{opt}}(\mathcal{I}).$$

Consequently, $L_{\max}(H)/L_{\text{opt}}(\mathcal{I}) \leq 1 + \alpha$.

Otherwise, in any optimal schedule for $\mathcal{I}^*$, there must be two jobs in $\{J_0, J_1, \ldots, J_m\}$ belonging to a common batch. Let Π be the set of optimal schedules for $\mathcal{I}^*$ with J_m lying in the last batch. By Claim 2, Π is not empty. Let $x \in \{1, 2, \ldots, m\}$ be the minimum index such that there is an optimal schedule $\pi \in \Pi$ for $\mathcal{I}^*$ each of whose jobs in $\{J_{x+1}, \ldots, J_m\}$ establishes a single batch and the batch containing J_x has at least two jobs. Note that $x \geq 1$ and such an x must exist, since in any optimal schedule, there must be two jobs in $\{J_0, J_1, \ldots, J_m\}$ belonging to a common batch under the assumption of this case. From the optimal schedules in Π assuming x, we choose the one, denoted by π, such that the batch B containing J_x is as large as possible. We claim that $J_{x-1} \in B$. Otherwise, let $K = \{J_0, J_1, \ldots, J_{x-2}\} \cap B$ and $p = p(B)$. By the definition of x and the assumption that $J_{x-1} \notin B$, K is not empty. If $p_{x-1} \leq p$, we shift job J_{x-1} to batch B. The resulted schedule σ is also optimal for $\mathcal{I}^*$ since $r_{x-1}^* < r_x^*$. But this contradicts the maximality of B. If $p_{x-1} > p$, we shift the jobs in K to the batch containing J_{x-1}. The resulted schedule σ' is still optimal for $\mathcal{I}^*$ since $r^*(K) < r_{x-1}^*$ and $p(K) < p_{x-1}$. But this contradicts the minimality of x. Hence, J_{x-1} and J_x must belong to a common batch in π. Now

$$\begin{aligned} L_{\text{opt}}(\mathcal{I}^*) &\geq r_x^* + \max\{p_x, p_{x-1}\} + p_{x+1} + \cdots + p_m + q^* \\ &= s(B_{x-1}) + \max\{p_x, p_{x-1}\} + p_{x+1} + \cdots + p_m + q^* \\ &\geq \alpha p_{x-1} + p_{x-1}. \end{aligned}$$

By Claim 1 and the fact $L_{\max}(H) = s(B_x) + p_x + p_{x+1} + \cdots + p_m + q^*$, we have $L_{\max}(H) - L_{\text{opt}}(\mathcal{I}) \leq L_{\max}(H) - L_{\text{opt}}(\mathcal{I}^*) \leq s(B_x) - s(B_{x-1}) = p_{x-1}$. Consequently, we have

$$\frac{L_{\max}(H)}{L_{\text{opt}}(\mathcal{I})} \leq 1 + \frac{p_{x-1}}{\alpha p_{x-1} + p_{x-1}} = 1 + \frac{1}{1+\alpha} = 1 + \alpha.$$

The result follows. □

If $p_j \geq q_j$ for every job J_j, we have $p^* \geq q^*$. By Theorem 3, it follows that $L_{\max}(H) \leq (1+\alpha)L_{\text{opt}}(\mathcal{I})$. Thus we conclude

Theorem 4 *For the problem* $1|p\text{-batch}, \text{on-line}, p_j \geq q_j, b = \infty|L_{\max}$, *the competitive ratio of Algorithm H is at most* $1 + \alpha$.

Corollary 2 and Theorem 4 imply that H is the best possible on-line algorithm for problem $1|p\text{-batch}, \text{on-line}, p_j \geq q_j, b = \infty|L_{\max}$.

If jobs have agreeable processing and delivery times, i.e., for every two jobs J_i and J_j, $p_i > p_j$ implies $q_i \geq q_j$, we obtain $q_j \geq q^*$ for every job J_j with $p_j > p^*$. By Theorem 3, we have $L_{\max}(H) \leq (1+\alpha)L_{\text{opt}}(\mathcal{I})$. Then we obtain

Theorem 5 *For the problem* $1|p\text{-batch}, \text{on-line}, \text{agreeable}(p_j, q_j), b = \infty|L_{\max}$, *the competitive ratio of Algorithm H is at most* $1 + \alpha$.

Corollary 2 and Theorem 5 imply that H is the best possible on-line algorithm for problem $1|p\text{-batch}, \text{on-line}, \text{agreeable}(p_j, q_j), b = \infty|L_{\max}$.

Acknowledgement We are grateful for the referee for many constructive comments and helpful suggestions.

References

Bartholdi JJ (1988) Unpublished result

Brucker P, Gladky A, Hoogeveen H, Kovalyov MY, Potts CN, Tautenhahn T, van de Velde SL (1998) Scheduling a batching machine. J Sched 1:31–54

Deng XT, Poon CK, Zhang YZ (2003) Approximation algorithms in batch processing. J Comb Optim 7:247–257

Epstein L, Favrholdt LM, Kohrt JS (2006) Separating on-line scheduling algorithms with the relative worst order ratio. J Comb Optim 12:337–350

Graham RL, Lawer EL, Lenstra JK, Rinnooy Kan AHG (1979) Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann Discrete Math 5:287–326

Hoogeveen JA, Vestjens APA (2000) A best possible deterministic on-line algorithm for minimizing maximum delivery time on a single machine. SIAM J Discrete Math 13:56–63

Lee CY, Uzsoy R (1999) Minimizing makespan on a single batch processing machine with dynamic job arrivals. Int J Prod Res 37:219–236

Lee CY, Uzsoy R, Martin-Vega LA (1992) Efficient algorithms for scheduling semi-conductor burn-in operations. Oper Res 40:764–775

Liu ZH, Yu WC (2000) Scheduling one batch processor subject to job release dates. Discrete Appl Math 105:129–136

Poon CK, Yu WC (2005) On-line scheduling algorithms for a batch machine with finite capacity. J Comb Optim 9:167–186

Tian J, Fu RY, Yuan JJ (2007) On-line scheduling with delivery time on a single batch machine. Theor Comput Sci 374:49–57

Zhang GC, Cai XQ, Wong CK (2001) On-line algorithms for minimizing makespan on batch processing machines. Nav Res Logist 48:241–258