Optimization Letters (2021) 15:1155–1173
https://doi.org/10.1007/s11590-020-01618-8

ORIGINAL PAPER

A best possible online algorithm for parallel batch scheduling with delivery times and limited restart

Hailing Liu[1,2] · **Xiwen Lu**[1] · **Wenjie Li**[3]

Received: 8 November 2019 / Accepted: 7 July 2020 / Published online: 13 July 2020

Abstract
We consider the online scheduling on an unbounded (drop-line) parallel batch machine to minimize the time by which all jobs have been delivered. In this paper, all jobs arrive over time and the running batches are allowed limited restart. Here limited restart means that a running batch which contains restarted jobs cannot be restarted again. A drop-line parallel batch machine can process several jobs simultaneously as a batch, and all jobs in a batch start at the same time, and the completion time of a job equals the sum of its starting time and its processing time. Here we consider the restricted model: all jobs have agreeable processing times and delivery times. We provide a best possible online algorithm H with a competitive ratio of $\frac{3}{2}$ for the problem on an unbounded parallel batch machine and the corresponding problem on an unbounded drop-line parallel batch machine, respectively.

Keywords Online scheduling · Parallel batch · Drop-line · Limited restart · Delivery time

1 Introduction

In the last decade, online scheduling and parallel batch scheduling have been extensively studied. In this paper, online means that jobs arrive over time and the characteristics of a job become known until its arrival time. The quality of an online algorithm is usually assessed by its competitive ratio. An online algorithm is said to be

✉ Xiwen Lu
xwlu@ecust.edu.cn

1 Department of Mathematics, East China University of Science and Technology, Shanghai 200237, People's Republic of China

2 College of Science, Henan University of Engineering, Zhengzhou 451191, Henan, People's Republic of China

3 School of Mathematical Sciences, Luoyang Normal University, Luoyang 471934, Henan, People's Republic of China

ρ-competitive if for any instance, it always returns a schedule whose objective value is not worse than ρ times the objective value of an optimal off-line schedule.

Parallel batch scheduling is largely motivated by burn-in operations in semiconductor manufacturing [7,13,14]. A parallel batch machine is a machine that can process up to b jobs simultaneously as a batch. All jobs in a batch start at the same time and complete at the same time. The processing time of a batch equals the longest processing time of jobs in the batch. There are two models depending on the characteristic of the batch capacity b. One is an unbounded model that means the batch capacity is sufficiently large, i.e., $b = \infty$. The other is a bounded model that means the batch capacity is finite, i.e., $b < \infty$. Here we study the unbounded model. For the online parallel batch scheduling problems to minimize the time by which all jobs have been delivered, there have been some results. Let p_j and q_j denote the processing time and the delivery time of the job J_j, respectively. Yuan et al. [17] studied two restricted models on an unbounded parallel batch machine. One is the model with small delivery times which means that the delivery time of each job is no more than its processing time, i.e., $q_j \leq p_j$ holds for all jobs. The other is the agreeable model which means that for any two jobs J_i and J_j, if $p_i > p_j$, then $q_i \geq q_j$. They provided a best possible online algorithm with a competitive ratio of $(\sqrt{5}+1)/2$ for the two restricted models. Tian et al. [10] considered the general case on an unbounded parallel batch machine and gave an online algorithm whose competitive ratio is $2\sqrt{2}-1$. For the online scheduling problem on m unbounded batch machines, Liu and Lu [8] gave a $(1+\alpha_m)$-competitive best possible online algorithm for the agreeable model, where α_m is the positive solution of the equation $\alpha_m^2 + m\alpha_m - 1 = 0$. A survey of recent researches on parallel batch scheduling can see Tian et al. [11].

Restart (see Hoogeveen et al. [5]) means that we can interrupt a running task and losing all the work done on it. The jobs in the interrupted task, which are called restarted jobs, are then released and become independently unscheduled jobs that need to be scheduled anew later. Allowing restarts means that we have a chance to change our mind and make better decisions according to information on newly arrived jobs. So we can obtain better online algorithms by using restarts. For the online scheduling problem of minimizing the time by which all jobs have been delivered on a single machine, Hoogeveen and Vestjens [6] presented a best possible algorithm whose competitive ratio is $\frac{1+\sqrt{5}}{2}$ without restart and Akker et al. [1] presented a best possible algorithm whose competitive ratio is $\frac{3}{2}$ when restarts are allowed. We can find more online scheduling researches with restarts in Epstein and Van Stee [2], Van Stee and Poutré [15], and Yuan et al. [18]. Fu et al. [3] first introduced limited restarts. Limited restarts mean that a running batch containing restarted jobs cannot be interrupted again. Limited restarts imply that a job can be restarted at most once. Limited restarts are of practice value as too many restarts of a job may cause a waste of cost and increase the possibility of spoiling a product.

For the online scheduling problem of minimizing makespan on an unbounded parallel batch machine with restarts, Yuan et al. [18] gave a best possible $(5-\sqrt{5})/2$-competitive online algorithm. Fu et al. [3] studied the online scheduling of minimizing makespan on an unbounded parallel batch machine with limited restarts. They gave a lower bound $3/2$ and a best possible $3/2$-competitive online algorithm. For the corre-

sponding problem on two unbounded parallel batch machines with limited restarts, Fu et al. [4] gave a best possible online algorithm whose competitive ratio is $(\sqrt{3}+1)/2$ under the second-restart assumption. For minimizing makespan of equal length jobs on m unbounded parallel batch machines with limited restarts, Liu et al. [9] provided a best possible online algorithm.

However, in some practical applications, the above assumption of a parallel batch machine is not suitable, especially the condition of all jobs in a batch having the same completion time. In fact, when the batches are open for finished jobs, a job with a shorter processing time should be completed earlier than a job with a longer processing time in a common batch. Wei [16] first introduced the drop-line parallel batch machine. A drop-line parallel batch machine is a machine that can process several jobs simultaneously as a batch. Jobs in a batch start at the same time and the completion time of a job equals the sum of its starting time and its processing time. A drop-line parallel batch machine means that in a batch, jobs with shorter processing times will be completed earlier than jobs with longer processing times in the batch. Tian et al. [12] studied the online scheduling problem on m unbounded drop-line parallel batch machines to minimize the time by which all jobs have been delivered and gave a best possible $(1+\alpha_m)$-competitive online algorithm, where α_m is the positive solution of the equation $a_m^2 + m\alpha_m - 1 = 0$. In this paper, they also gave two examples of drop-line parallel batch scheduling on commodity delivery problems and optimization of network information. In the delivery problem, a truck is used to deliver several products to different destinations. At each destination, the corresponding products are unloaded from the truck. Here a truck is a drop-line parallel batch machine and a product arriving at its destination earlier will be unloaded earlier.

The problem studied in this paper can be described as follows. There is an unbounded parallel batch machine(or an unbounded drop-line parallel batch machine) and sufficiently many vehicles. Jobs arrive over time and all characteristics of a job are unknown until it arrives. Let r_j, p_j and q_j denote the release time, the processing time and the delivery time of the job J_j, respectively. Here we consider the restricted model that all jobs have agreeable processing times and delivery times, which means that for any two jobs J_i and J_j, if $p_i > p_j$, then $q_i \geq q_j$. Jobs will be first processed on the batch machine. Once a job is completed, it is immediately delivered to its destination by a vehicle. The running batches on the batch machine are allowed limited restarts, i.e., a running batch containing restarted jobs cannot be interrupted again. The objective is to minimize the time by which all jobs have been delivered. Let C_j and L_j denote the completion time of J_j on the batch machine and the time by which J_j has been delivered, respectively. Then $L_j = C_j + q_j$ and the objective function is $L_{\max} = \max_j\{L_j\}$. The corresponding problem on an unbounded parallel batch machine can be denoted by $1|\text{online}, r_j, \text{agreeable}, \text{p-batch}, b = \infty, \text{L-restart}|L_{\max}$. The corresponding problem on an unbounded drop-line parallel batch machine can be denoted by $1|\text{online}, r_j, \text{agreeable}, \text{drop-line p-batch}, b = \infty, \text{L-restart}|L_{\max}$. In this paper, a restricted batch means that the batch contains restarted jobs when the batch starts processing. A free batch means that the batch contains no restarted jobs when the batch starts processing.

The problems considered in this paper are the integration of production and delivery, which are motivated by practical problems. Now we show an example of the drop-line parallel batch scheduling in commodity delivery problems. Suppose that a number of containers are delivered from port O(origin) to different ports (transit points) by a vessel. At each transit point, the containers that are unloaded are individually sent to their destinations by a huge quantity of vehicles. At port O, containers from various merchants arrive and the vessel accommodates them until its capacity is reached. Here each container arrives over time and its characteristics become known until it arrives. The objective of the vessel's company is to determine the container delivery schedule in order to minimize the time by which all containers have been delivered to their destinations. This is just a drop-line parallel batch scheduling problem with jobs arriving online, where a vessel is a drop-line parallel batch processing machine and a container is a job.

This paper is organized as follows. In Sect. 2, we prove that there is no online algorithm with a competitive ratio less than $\frac{3}{2}$ for the problem $1|\text{online}, r_j, \text{agreeable}, \text{p-batch}, b = \infty, \text{L-restart}|L_{\max}$ and give a best possible online algorithm H with a competitive ratio of $\frac{3}{2}$. In Sect. 3, we prove that there is no online algorithm with a competitive ratio less than $\frac{3}{2}$ for the problem $1|\text{online}, r_j, \text{agreeable}, \text{drop-line p-batch}, b = \infty, \text{L-restart}|L_{\max}$ and prove that Algorithm H is also a best possible online algorithm with a competitive ratio of $\frac{3}{2}$.

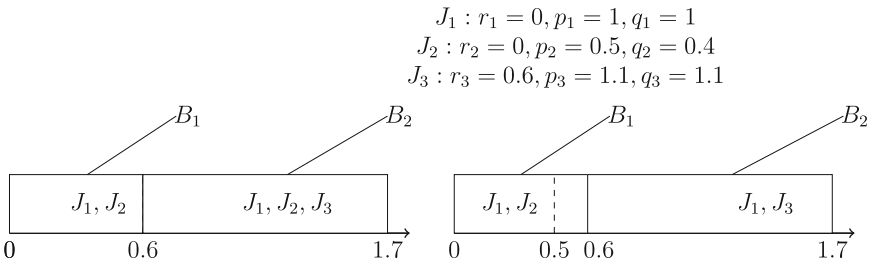
2 The problem on an unbounded parallel batch machine

In this section, we study the problem on an unbounded parallel batch machine, i.e., $1|\text{online}, r_j, \text{agreeable}, \text{p-batch}, b = \infty, \text{L-restart}|L_{\max}$.

Theorem 2.1 *For the problem* $1|\text{online}, r_j, \text{agreeable}, \text{p-batch}, b = \infty, \text{L-restart}|L_{\max}$, *there is no online algorithm with a competitive ratio less than* $\frac{3}{2}$.

Proof For the problem $1|\text{online}, r_j, \text{p-batch}, b = \infty, \text{L-restart}|C_{\max}$, Theorem 2.1 in Fu et al. [3] shows that there does not exist any online algorithm with a competitive ratio less than $\frac{3}{2}$. Note that the problem $1|\text{online}, r_j, \text{p-batch}, b = \infty, \text{L-restart}|C_{\max}$ is a special case of the problem $1|\text{online}, r_j, \text{agreeable}, \text{p-batch}, b = \infty, \text{L-restart}|L_{\max}$ (by setting $q_j = 0$ for all jobs). So any online algorithm has a competitive ratio at least $\frac{3}{2}$ for the problem $1|\text{online}, r_j, \text{agreeable}, \text{p-batch}, b = \infty, \text{L-restart}|L_{\max}$. The result follows. □

Now we give some notations used in the following algorithm. Let $p(B)$ be the longest processing time of jobs in a job set B. Let B_i denote the i-th starting batch in the schedule generated by the algorithm and S_i denote the starting time of B_i. Let $r(B_i)$ be the earliest arrival time of jobs in B_i. Let $p(B_i)$ and $q(B_i)$ denote the longest processing time of jobs in the batch B_i and the largest delivery time of jobs in the batch B_i, respectively. Among the jobs with a processing time $p(B_i)$ in B_i, select one which arrives latest as the job J_i^p. Let q_i^p be the delivery time of the job J_i^p and r_i^p be the arrival time of the job J_i^p. Among the jobs with a delivery time $q(B_i)$ in B_i,



(a) on a parallel batch machine **(b)** on a drop-line parallel batch machine

Fig. 1 restarts on a parallel batch machine and on a drop-line parallel batch machine

select one which arrives latest as the job J_i^q. Let p_i^q be the processing time of the job J_i^q. We use $U(t)$ to denote the set of unscheduled jobs available at time t.

We can use the following sub-procedure Restart(k, t) when we restart the batch B_k at the time t.

Restart(k, t): Interrupt B_k at time t and schedule all jobs (or all uncompleted jobs for the drop-line version) in B_k and all jobs in $U(t)$ as a batch B_{k+1} starting at time t. Set $k = k + 1$. □

Now we show examples of restarts on an unbounded parallel batch machine and on an unbounded drop-line parallel batch machine. See Fig. 1. $B_1 = \{J_1, J_2\}$ is a free batch with a starting time 0. B_1 has a processing time $p(B_1) = \max\{p_1, p_2\} = 1$. At the time 0.6, a new job J_3 arrives with a processing time $p_3 = 1.1$ and a delivery time $q_3 = 1.1$. Assume that we interrupt B_1 and start a restricted batch B_2 at the time 0.6. First we see restarts on an unbounded parallel batch machine (see Fig. 1a). Since all jobs in B_1 will complete at the same time, all jobs in B_1 are released when B_1 is interrupted and the restricted batch B_2 consists of all jobs in B_1 and the newly arrived jobs J_3. Thus $B_2 = \{J_1, J_2, J_3\}$. B_2 has a processing time $p(B_2) = \max\{p_1, p_2, p_3\} = 1.1$. Now we see restarts on an unbounded drop-line parallel batch machine (see Fig. 1b). Since the completion time of a job is equal to its starting time plus its processing time, the job J_2 has been completed at the time 0.5. Only the job J_1 is released when B_1 is interrupted at the time 0.6 and the restricted batch B_2 consists of all uncompleted jobs in B_1 at the time 0.6 and the newly arrived jobs J_3. Thus $B_2 = \{J_1, J_3\}$. B_2 has a processing time $p(B_2) = \max\{p_1, p_3\} = 1.1$.

In the following algorithm, we use $\rho = 0$ to show that the current batch is a free batch and use $\rho = 1$ to show that the current batch is a restricted batch. The symbol "s'' means the starting time of the current batch.

Algorithm H

Step 0: Set $t = 0, s = 0, \rho = 0, k = 0$.

Step 1: If $U(t) = \emptyset$, wait until new jobs arrive or stop without new jobs arriving. Otherwise determine $p(U(t))$.

Step 2: If $t \geq \frac{1}{2}p(U(t))$, then schedule all jobs in $U(t)$ as a single batch B_{k+1} starting at time t, and set $k = k + 1$ and $s = t$. If $t < \frac{1}{2}p(U(t))$, wait until $\frac{1}{2}p(U(t))$ or the next arrival time and go to Step 1.

Step 3: If no new jobs arrive in the time interval $(s, s + p(B_k))$ or $\rho = 1$, then complete the batch B_k, and set $t = s + p(B_k)$, $\rho = 0$, and go to Step 1. If $\rho = 0$ and some new jobs(or job) arrive at sometime t' in the time interval $(s, s + p(B_k))$, then find the job J_h whose processing time is the longest among jobs arriving at time t'.

Step 4: If $r_h \geq \frac{5}{4}p(B_k)$, then complete the batch B_k, set $t = s + p(B_k)$ and go to Step 1. Otherwise, do the following:

Step 4.1: If $p_h \geq \frac{3}{2}p(B_k)$, then complete the batch B_k, set $t = s + p(B_k)$ and go to Step 1.

Step 4.2: If $p(B_k) < p_h < \frac{3}{2}p(B_k)$, do the following:

Step 4.2.1: If $r_h \geq p_h$, then set $t = r_h$, do Restart(k, t), set $s = t$, $\rho = 1$ and go to Step 3.

Step 4.2.2: If $r_h < p_h$, continue to process B_k in the time interval $(r_h, p_h]$ unless a new job $J_{h'}$ with $p_{h'} \geq p_h$ arrives.

If at sometime t' in the time interval $(r_h, p_h]$, a job set which contains some new job $J_{h'}$ with $p_{h'} \geq p_h$ arrives, then update J_h to be the job whose processing time is the longest among jobs arriving at time t' and go to Step 4.

Otherwise, set $t = p_h$, do Restart(k, t), set $s = t$, $\rho = 1$ and go to Step 3.

Step 4.3: If $\frac{3}{4}p(B_k) < p_h \leq p(B_k)$, do the following:

Step 4.3.1: If $r_h < 2p(B_k) - p_h$, continue to process B_k in the time interval $(r_h, 2p(B_k) - p_h]$ unless a new job $J_{h'}$ with $p_{h'} \geq p_h$ arrives.

If at sometime t' in the time interval $(r_h, 2p(B_k) - p_h]$, a job set which contains some new job $J_{h'}$ with $p_{h'} \geq p_h$ arrives, then update J_h to be the job whose processing time is the longest among jobs arriving at time t' and go to Step 4.

Otherwise, (a) if $p_h \geq \frac{3}{4}p(B_k) + \frac{1}{4}q(B_k)$, then set $t = 2p(B_k) - p_h$, do Restart(k, t), set $s = t$, $\rho = 1$ and go to Step 3; (b) if $p_h < \frac{3}{4}p(B_k) + \frac{1}{4}q(B_k)$, go to Step 4.5.

Step 4.3.2: If $r_h \geq 2p(B_k) - p_h$ and $p_h \geq \frac{3}{4}p(B_k) + \frac{1}{4}q(B_k)$, then set $t = r_h$, do Restart(k, t), set $s = t$, $\rho = 1$ and go to Step 3.

Step 4.3.3: If $r_h \geq 2p(B_k) - p_h$ and $p_h < \frac{3}{4}p(B_k) + \frac{1}{4}q(B_k)$, go to Step 4.5.

Step 4.4: If $p_h \leq \frac{3}{4}p(B_k)$, go to Step 4.5.

Step 4.5: Continue to process B_k unless some new job $J_{h'}$ with $p_{h'} \geq p_h$ arrives before the time $\frac{5}{4}p(B_k)$.

If at sometime t' before time $\frac{5}{4}p(B_k)$, a job set which contains some new job $J_{h'}$ with $p_{h'} \geq p_h$ arrives, then update J_h to be the job whose processing time is the longest among jobs arriving at time t' and go to Step 4.1.

If there is no new job $J_{h'}$ with $p_{h'} \geq p_h$ arriving before the time $\frac{5}{4}p(B_k)$, then complete the batch B_k, set $t = s + p(B_k)$ and go to Step 1. □

Let I be an instance. Let σ and π be the schedule produced by Algorithm H for the instance I and the off-line optimal schedule of the instance I, respectively. Let $L_{\text{on}}(I)$ be the objective value of the schedule σ and $L_{\text{opt}}(I)$ be the objective value of

the schedule π. For simplicity, we write $L_{\text{on}}(I)$ as L_{on} and write $L_{\text{opt}}(I)$ as L_{opt}. Let B_n be the first batch in σ satisfying $L_{\text{on}} = S_n + p(B_n) + q(B_n)$. For the schedule σ generated by Algorithm H, we have the following properties.

Observation 2.1 For a free batch B_i, we have the following conclusions.

(1) $S_i \geq \max\{r(B_i), \frac{1}{2}p(B_i)\}$;
(2) $S_1 = \max\{r(B_1), \frac{1}{2}p(B_1)\}$;
(3) If $S_i > S_{i-1} + p(B_{i-1})$ $(i \geq 2)$, then $S_i = \max\{r(B_i), \frac{1}{2}p(B_i)\}$. □

If a restricted batch B_i is got by Step 4.2 of Algorithm H, noting that $p(B_{i-1}) < p_h < \frac{3}{2}p(B_{i-1})$, then we have $p(B_i) = \max\{p_h, p(B_{i-1})\} = p_h$ and $S_i \geq p_h = p(B_i)$. If a restricted batch B_i is got by Step 4.3 of Algorithm H, noting that $\frac{3}{4}p(B_{i-1}) < p_h \leq p(B_{i-1})$, we have $p(B_i) = \max\{p_h, p(B_{i-1})\} = p(B_{i-1})$ and $S_i \geq 2p(B_{i-1}) - p_h \geq p(B_{i-1}) = p(B_i)$. Then the following property holds.

Observation 2.2 For a restricted batch B_i, we have $S_i \geq p(B_i)$. □

From Observations 2.1 and 2.2, we can easily get the following property.

Observation 2.3 $S_i \geq \frac{1}{2}p(B_i)$ $(1 \leq i \leq n)$. □

Lemma 2.1 *Let jobs J' and J'' have processing times p', p'' and delivery times q', q'' with $p' \geq p''$ and $q'' \geq q'$. We can obtain the following conclusions.*

(1) One of J' and J'' has the processing time p' and the delivery time q''.
(2) If both J' and J'' start not earlier than time s in the optimal off-line schedule π, then $L_{opt} \geq s + p' + q''$.

Proof (1) Note that $p' \geq p''$. If $p' = p''$, then the processing time of J'' is p'. Now the processing time of J'' is p' and the delivery time of J'' is q''. If $p' > p''$, considering that processing times are agreeable with delivery times, we have $q' \geq q''$. Also considering that $q'' \geq q'$, then $q' = q''$. Now the delivery time of J' is q'' and the processing time of J' is p'.
(2) Considering that both J' and J'' start not earlier than time s in the optimal off-line schedule π and (1), then $L_{\text{opt}} \geq s + p' + q''$. The result follows. □

Lemma 2.2 *(1) For each batch B_i $(1 \leq i \leq n)$, at least one of the jobs J_i^p and J_i^q has the processing time of $p(B_i)$ and the delivery time of $q(B_i)$.*
(2) $L_{opt} \geq r(B_i) + p(B_i) + q(B_i)$ $(1 \leq i \leq n)$.

Proof (1) By the definitions of J_i^p and J_i^q, J_i^p and J_i^q satisfy the condition of Lemma 2.1. Note that the processing time of J_i^p is $p(B_i)$ and the delivery time of J_i^q is $q(B_i)$. So by Lemma 2.1(1), one of J_i^p and J_i^q has the processing time $p(B_i)$ and the delivery time $q(B_i)$.
(2) By (1), we can easily get that $L_{\text{opt}} \geq r(B_i) + p(B_i) + q(B_i)$ $(1 \leq i \leq n)$. The result follows. □

Lemma 2.3 *(1) If $n = 1$, then $L_{on}/L_{opt} \leq \frac{3}{2}$.*
(2) If $S_n > S_{n-1} + p(B_{n-1})$ $(n \geq 2)$, then $L_{on}/L_{opt} \leq \frac{3}{2}$.

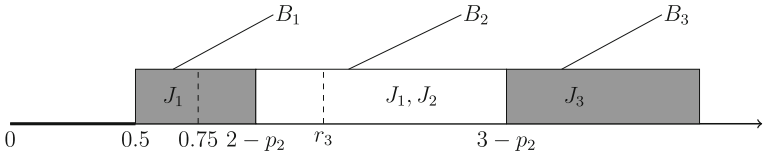


Fig. 2 A case with restarts

Proof By Observation 2.1, $S_n = \max\{\frac{1}{2}p(B_n), r(B_n)\}$ if $n = 1$ or $S_n > S_{n-1} + p(B_{n-1})$ $(n \geq 2)$. Then $L_{\text{on}} = S_n + p(B_n) + q(B_n) = \max\{\frac{1}{2}p(B_n), r(B_n)\} + p(B_n) + q(B_n)$. By Lemma 2.2, we have $L_{\text{opt}} \geq r(B_n) + p(B_n) + q(B_n)$. Then $L_{\text{on}}/L_{\text{opt}} \leq \frac{\max\{\frac{1}{2}p(B_n), r(B_n)\} + p(B_n) + q(B_n)}{r(B_n) + p(B_n) + q(B_n)} \leq 3/2$. The result follows. □

If $n = 1$ or $S_n > S_{n-1} + p(B_{n-1})$ $(n \geq 2)$, by Lemma 2.3, then $L_{\text{on}}/L_{\text{opt}} \leq \frac{3}{2}$. So we only consider that $n \geq 2$ and $S_n \leq S_{n-1} + p(B_{n-1})$. Now we distinguish three cases and use three Lemmas to prove that $L_{\text{on}}/L_{\text{opt}} \leq \frac{3}{2}$. To improve the readability and reduce the length of the paper, considering that the proofs of the following three Lemmas are foundational, we move the detailed proofs of the following three Lemmas to Appendix.

Lemma 2.4 *If $S_n < S_{n-1} + p(B_{n-1})$, then $L_{on}/L_{opt} \leq \frac{3}{2}$.* □

Lemma 2.5 *If $S_n = S_{n-1} + p(B_{n-1})$ and B_{n-1} is a restricted batch, then $L_{on}/L_{opt} \leq \frac{3}{2}$.* □

Lemma 2.6 *If $S_n = S_{n-1} + p(B_{n-1})$ and B_{n-1} is a free batch, then $L_{on}/L_{opt} \leq \frac{3}{2}$.* □

From Theorem 2.1 and Lemma 2.3–2.6, we conclude one of main results in this paper as follows.

Theorem 2.2 *Algorithm H is a best possible online algorithm for the problem $1|online, r_j, agreeable, p\text{-}batch, b = \infty, L\text{-}restart|L_{\max}$.* □

Now we show a case with restarts for better understanding of the considered problem and Algorithm H. See Fig. 2.

At the time 0, a job J_1 arrives with a processing time $p_1 = 1$ and a delivery time q_1 $(q_1 < 1)$. By Step 2 of Algorithm H, we start a free batch $B_1 = \{J_1\}$ on the batch machine with a starting time $S_1 = \frac{1}{2}p_1 = \frac{1}{2}$. Now $p(B_1) = p_1 = 1$ and $q(B_1) = q_1$. Later a new job J_2 with a processing time p_2 $(p_1 > p_2 > \frac{3}{4} + \frac{1}{4}q_1)$ and a delivery time q_2 $(q_2 \leq q_1)$ arrives at the time $r_2 = \frac{3}{4}$. Thus $r_2 < 2p(B_1) - p_2$ as $p_1 > p_2$. Since $\frac{3}{4}p(B_1) + \frac{1}{4}q(B_1) < p_2 < p(B_1)$, $r_2 < \frac{5}{4}p(B_1)$ and $r_2 < 2p(B_1) - p_2$, by Step 4.3.1 of Algorithm H, we interrupt the free batch B_1 and start a restricted batch $B_2 = \{J_1, J_2\}$ with a starting time $S_2 = 2p(B_1) - p_2 = 2 - p_2$. Now $p(B_2) = \max\{p_1, p_2\} = 1$ and $q(B_2) = \max\{q_1, q_2\} = q_1$. Later, a new job J_3 with a processing time p_3 and a delivery time q_3 arrives at the time r_3, where $S_2 < r_3 < S_2 + p(B_2)$ and $q_1 < p_3 + q_3 \leq 4$. No new jobs arrive later. Since B_2

is a restricted batch and cannot be restarted, by Step 2 of Algorithm H, we start a free batch $B_3 = \{J_3\}$ at the time $S_3 = \max\{S_2 + p(B_2), \frac{1}{2}p_3\}$. Note that $\frac{1}{2}p_3 \leq 2$ as $p_3 + q_3 \leq 4$. Also considering that $S_2 + p(B_2) = 3 - p_2 > 2$, we have $S_3 = \max\{S_2 + p(B_2), \frac{1}{2}p_3\} = S_2 + p(B_2) = 3 - p_2$. Considering that $q_1 < p_3 + q_3$ and $q(B_2) = \max\{q_1, q_2\} = q_1$, we have $L_{\text{on}} = S_2 + p(B_2) + \max\{q(B_2), p_3 + q_3\} = 3 - p_2 + p_3 + q_3$. If J_1 or J_2 starts not earlier than the time S_2 in the off-line optimal schedule, we have $L_{\text{opt}} \geq S_2 + \min\{p_1, p_2\} = 2 - p_2 + p_2 = 2$. Also considering that $L_{\text{opt}} \geq r_3 + p_3 + q_3 > S_2 + p_3 + q_3 = 2 - p_2 + p_3 + q_3$ and $L_{\text{opt}} \geq 2$, we have $L_{\text{on}} - L_{\text{opt}} \leq 1$ and $\frac{L_{\text{on}} - L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{1}{2}$. If both J_1 and J_2 start earlier than the time S_2 in the off-line optimal schedule, considering that J_3 arrives after the time S_2, then both J_1 and J_2 start before J_3 in the off-line optimal schedule. Note that the later completion time of J_1 and J_2 is not earlier than the time $\min\{p_1 + p_2, S_1 + p_1\} = \frac{3}{2}$ whether J_1 and J_2 are in a common batch or not in the off-line optimal schedule. Thus J_3 starts not earlier than the time $\frac{3}{2}$ in the off-line optimal schedule and $L_{\text{opt}} \geq \frac{3}{2} + p_3 + q_3$. Then $\frac{L_{\text{on}}}{L_{\text{opt}}} \leq \frac{3 - p_2 + p_3 + q_3}{\frac{3}{2} + p_3 + q_3} \leq \frac{\frac{9}{4} - \frac{1}{4}q_1 + p_3 + q_3}{\frac{3}{2} + p_3 + q_3} \leq \frac{3}{2}$. Here the last but one inequality holds as $p_2 > \frac{3}{4} + \frac{1}{4}q_1$. Thus we have $\frac{L_{\text{on}}}{L_{\text{opt}}} \leq \frac{3}{2}$ for the considered case.

3 The problem on an unbounded drop-line parallel batch machine

In this section, we study the problem on an unbounded drop-line parallel batch machine, i.e., 1|online, r_j, agreeable, drop-line p-batch, $b = \infty$, L-restart| $L_{\max}$.

Theorem 3.1 *For the problem* 1|*online*, r_j, *agreeable*, *drop-line p-batch*, $b = \infty$, *L-restart*| $L_{\max}$, *there is no online algorithm with a competitive ratio less than* $\frac{3}{2}$.

Proof Similar to the method of proof in Fu et al. [3], we can prove that there is no online algorithm with a competitive ratio less than $\frac{3}{2}$ for the problem 1|online, r_j, drop-line p-batch, $b = \infty$, L-restart|$C_{\max}$. The problem 1|online, r_j, drop-line p-batch, $b = \infty$, L-restart|$C_{\max}$ is a special case of the problem 1|online, r_j, agreeable, drop-line p-batch, $b = \infty$, L-restart|$L_{\max}$ (by setting $q_j = 0$ for all jobs). Therefore there is no online algorithm with a competitive ratio less than $\frac{3}{2}$ for the problem 1|online, r_j, agreeable, drop-line p-batch, $b = \infty$, L-restart|$L_{\max}$. The result follows. □

Now we will prove that Algorithm H is also a best possible online algorithm for the problem 1|online, r_j, agreeable, drop-line p-batch, $b = \infty$, L-restart| $L_{\max}$. For simplicity, we call the problem 1|online, r_j, agreeable, p-batch, $b = \infty$, L-restart|$L_{\max}$ as the problem P1 and call the problem 1|online, r_j, agreeable, drop-line p-batch, $b = \infty$, L-restart| $L_{\max}$ as the problem P2.

Let σ_1 and σ_2 be the schedules of an instance I generated by Algorithm H for the problems P1 and P2, respectively. Let σ_1^* and σ_2^* be the optimal off-line schedules of the instance I for the problems P1 and P2, respectively. Note that we may assume that there exist no interrupted batches in σ_1^* and σ_2^* since restarted jobs in σ_1^* and σ_2^* can be removed from the corresponding free batches.

For the schedules $\sigma_i (i = 1, 2)$, we use the following notations.

- n_i $(i = 1, 2)$ is the total number of batches in the schedule σ_i.
- $B_{k,i}$ $(i = 1, 2; 1 \leq k \leq n_i)$ is the k-th starting batch in the schedule σ_i.
- $S_{k,i}$ $(i = 1, 2; 1 \leq k \leq n_i)$ is the starting time of $B_{k,i}$.
- $p(B_{k,i})(i = 1, 2; 1 \leq k \leq n_i)$ is the longest processing time of jobs in the batch $B_{k,i}$.
- $q(B_{k,i})$ $(i = 1, 2; 1 \leq k \leq n_i)$ is the largest delivery time of jobs in the batch $B_{k,i}$.
- $J^p_{k,i}$ $(i = 1, 2; 1 \leq k \leq n_i)$ is got by selecting the latest arriving one from jobs with the processing time $p(B_{k,i})$ in $B_{k,i}$.
- $J^q_{k,i}$ $(i = 1, 2; 1 \leq k \leq n_i)$ is got by selecting the latest arriving one from jobs with the delivery time $q(B_{k,i})$ in $B_{k,i}$.

Theorem 3.2 *Algorithm H is a best possible online algorithm for the problem $1|online,\ r_j,\ agreeable,\ drop\text{-}line\ p\text{-}batch,\ b = \infty,\ L\text{-}restart|\ L_{\max}$.*

Proof By Theorem 3.1, we just need to prove that for an instance I, the objective value of the schedule σ_2 is no more than $\frac{3}{2}$ times the objective value of the schedule σ_2^*.

Similar to the proof of Lemma 2.2, we have the following Claim.

Claim 1 For each batch $B_{k,i}$ $(i = 1, 2; 1 \leq k \leq n_i)$, at least one job of $J^p_{k,i}$ and $J^q_{k,i}$ has the processing time of $p(B_{k,i})$ and the delivery time of $q(B_{k,i})$.

For the number of batches and characteristics of batches in the schedules $\sigma_i (i = 1, 2)$, we have the following property.

Claim 2 (1) $n_1 = n_2$;
(2) For $1 \leq i \leq n_1$, one of the following two conclusions holds:

(a) $B_{i,2}$ and $B_{i,1}$ are both free batches. They have the same jobs and the same starting time.
(b) $B_{i,2}$ and $B_{i,1}$ are both restricted batches. They have the same starting time, the same longest processing time, the same largest delivery time and the same unrestarted jobs.

Clearly, $B_{1,1}$ and $B_{1,2}$ are both free batches, and they have the same starting time and the same jobs.

Suppose that $B_{k,1}$ and $B_{k,2}$ are both free batches, and have the same jobs and the same starting time. By Algorithm H, a new batch $B_{k+1,2}$(or $B_{k+1,1}$) starts depending on the information on newly arrived jobs, the longest processing time of $B_{k,2}$(or $B_{k,1}$) and the largest delivery time of $B_{k,2}$(or $B_{k,1}$). So either $B_{k+1,2}$ and $B_{k+1,1}$ are both free batches, and have the same starting time and the same jobs, or $B_{k+1,2}$ and $B_{k+1,1}$ are both restricted batches with the same starting time. If $B_{k+1,2}$ and $B_{k+1,1}$ are both restricted batches, then $B_{k+1,1}$ consists of all jobs in $B_{k,1}$ and newly arrived jobs, and $B_{k+1,2}$ consists of the jobs in $B_{k,2}$ which do not complete at time $S_{k+1,2}$ and newly arrived jobs. As $B_{k+1,2}$ and $B_{k+1,1}$ have the same starting time, then they have the same unrestarted jobs. Note that the jobs in $B_{k,2}$ which do not complete at time $S_{k+1,2}$ contain one of the jobs $J^p_{k,2}$ and $J^q_{k,2}$ which has the processing time of $p(B_{k,2})$ and the

delivery time of $q(B_{k,2})$. Note that $p(B_{k,2}) = p(B_{k,1})$ and $q(B_{k,2}) = q(B_{k,1})$. So the longest processing time of restarted jobs in $B_{k+1,1}$ is equal to the longest processing time of restarted jobs in $B_{k+1,2}$, and the largest delivery time of restarted jobs in $B_{k+1,1}$ is equal to the largest delivery time of restarted jobs in $B_{k+1,2}$. Also considering that $B_{k+1,2}$ and $B_{k+1,1}$ have the same unrestarted jobs, then the longest processing time of jobs in $B_{k+1,1}$ is equal to the longest processing time of jobs in $B_{k+1,2}$, and the largest delivery time of jobs in $B_{k+1,1}$ is equal to the largest delivery time of jobs in $B_{k+1,2}$. So if $B_{k+1,2}$ and $B_{k+1,1}$ are both restricted batches, then $B_{k+1,2}$ and $B_{k+1,1}$ have the same starting time, the same longest processing time, the same largest delivery time and the same unrestarted jobs.

The above proof also shows that if $B_{m,2}$ and $B_{m,1}$ are the first restricted batches, then $B_{m,2}$ and $B_{m,1}$ have the same starting time, the same longest processing time, the same largest delivery time and the same unrestarted jobs.

Suppose that $B_{k,2}$ and $B_{k,1}$ are both restricted batches, and they have the same starting time, the same longest processing time, the same largest delivery time and the same unrestarted jobs. Then $B_{k,2}$ and $B_{k,1}$ have the same completion time by Algorithm H. Thus $B_{k,2}$ and $B_{k,1}$ have the same starting time and the same completion time. By Algorithm H, $B_{k+1,2}$ and $B_{k+1,1}$ are both free batches and $B_{k+1,2}$(or $B_{k+1,1}$) starts depending on the information on newly arrived jobs. So $B_{k+1,2}$ and $B_{k+1,1}$ are both free batches, and they have the same starting time and the same jobs.

The above proof shows that $n_1 = n_2$ and for $1 \leq i \leq n_1$, either $B_{i,2}$ and $B_{i,1}$ are both free batches and have the same jobs and the same starting time, or $B_{i,2}$ and $B_{i,1}$ are both restricted batches and have the same starting time, the same longest processing time, the same largest delivery time and the same unrestarted jobs. Claim 2 follows.

By Claim 2, we may assume that $n = n_1 = n_2$.

Claim 3 The objective value of the schedule σ_2 for the problem P2 is no more than the objective value of the schedule σ_1 for the problem P1.

For any job J_j in the instance I, we distinguish the following cases.

If in the schedule σ_2, J_j is completed in a batch $B_{k,2}$ which is a free batch uninterrupted or is a restricted batch, by Claim 2, then J_j is completed in the batch $B_{k,1}$ in the schedule σ_1. For the problem P1, the time by which J_j has been delivered in the schedule σ_1 is $S_{k,1} + p(B_{k,1}) + q_j$. For the problem P2, the time by which J_j has been delivered in the schedule σ_2 is $S_{k,2} + p_j + q_j$. Note that $p(B_{k,1}) \geq p_j$ and $S_{k,2} = S_{k,1}$ by Claim 2, then $S_{k,1} + p(B_{k,1}) + q_j \geq S_{k,2} + p_j + q_j$. So if in the schedule σ_2, J_j is completed in a batch $B_{k,2}$ which is a free batch uninterrupted or is a restricted batch, then the time by which J_j has been delivered in the schedule σ_2 for the problem P2 is not later than the time by which J_j has been delivered in the schedule σ_1 for the problem P1.

If in the schedule σ_2, J_j is completed in a free batch $B_{k,2}$ which is interrupted, then J_j is completed in the restricted batch $B_{k+1,1}$ in the schedule σ_1. For the problem P1, the time by which J_j has been delivered in the schedule σ_1 is $S_{k+1,1} + p(B_{k+1,1}) + q_j$. For the problem P2, the time by which J_j has been delivered in the schedule σ_2 is $S_{k,2} + p_j + q_j$. Note that $p(B_{k+1,1}) \geq p_j$ and $S_{k,2} = S_{k,1}$ by Claim 2. Then $S_{k+1,1} + p(B_{k+1,1}) + q_j > S_{k,2} + p_j + q_j$. So if in the schedule σ_2, J_j is completed

in a free batch $B_{k,2}$ which is interrupted, then the time by which J_j has been delivered in the schedule σ_2 for the problem P2 is earlier than the time by which J_j has been delivered in the schedule σ_1 for the problem P1.

Then for any job J_j in the instance I, the time by which J_j has been delivered in the schedule σ_2 for the problem P2 is not later than the time by which J_j has been delivered in the schedule σ_1 for the problem P1. Then the objective value of σ_2 for the problem P2 is no more than the objective value of σ_1 for the problem P1. Claim 3 follows.

Claim 4 The objective value of the schedule σ_2^* for the problem P2 is equal to the objective value of the schedule σ_1^* for the problem P1.

In fact, considering that σ_1^* is the optimal off-line schedule of the instance I for the problem P1 and σ_2^* is the optimal off-line schedule of the instance I for the problem P2, obviously the objective value of the schedule σ_2^* for the problem P2 is no more than the objective value of the schedule σ_1^* for the problem P1. Now we just need to prove that the objective value of the schedule σ_1^* for the problem P1 is no more than the objective value of the schedule σ_2^* for the problem P2.

Similar to the proof of Lemma 2.2, for each batch B_k in the schedule σ_2^*, there exists one job J_l with the longest processing time $p(B_k)$ and the largest delivery time $q(B_k)$.

If we consider the schedule σ_2^* for the problem P2, then for any job J_j in the batch B_k, the time by which J_j has been delivered is $S_k + p_j + q_j \leq S_k + p(B_k) + q(B_k)$ and the equality holds when $j = l$. If we consider the schedule σ_2^* for the problem P1, the time by which all jobs in the batch B_k have been delivered is $S_k + p(B_k) + q(B_k)$. So $S_k + p(B_k) + q(B_k)$ is the time by which all jobs in B_k have been delivered for the schedule σ_2^* for the problem P1 and for the optimal off-line schedule σ_2^* for the problem P2. Thus the objective value of the schedule σ_2^* for the problem P2 is equal to the objective value of the schedule σ_2^* for the problem P1. Because σ_1^* is the optimal off-line schedule of the instance I for the problem P1, the objective value of the schedule σ_1^* for the problem P1 is no more than the objective value of the schedule σ_2^* for the problem P1. So the objective value of the schedule σ_1^* for the problem P1 is no more than the objective value of the schedule σ_2^* for the problem P2. Claim 4 follows.

By Claim 3 and 4, for the problem P2, the ratio of the objective value of σ_2 and the objective value of σ_2^* is no more than the ratio of the objective value of σ_1 for the problem P1 and the objective value of σ_1^* for the problem P1. By Theorem 2.1–2.2, for the problem P1, the ratio of the objective value of σ_1 and the objective value of σ_1^* is no more than $\frac{3}{2}$. Thus for the problem P2, the ratio of the objective value of σ_2 and the objective value of σ_2^* is no more than $\frac{3}{2}$.

By Theorem 3.1, Algorithm H is a best possible online algorithm for the problem $1|\text{online}, r_j, \text{agreeable}, \text{drop-line p-batch}, b = \infty, \text{L-restart}| L_{\max}$. The result follows. □

Conclusions

In this paper, we study the online scheduling problem on an unbounded parallel batch machine (or on an unbounded drop-line parallel batch machine) to minimize the time by which all jobs have been delivered with limited restarts. Here all jobs have agreeable processing times and delivery times. We give an online algorithm H and prove that the online algorithm H is a best possible online algorithm with a competitive ratio of $\frac{3}{2}$ for the two problems considered in this paper, respectively. Note that for the corresponding online scheduling problem on an unbounded parallel batch machine (or on an unbounded drop-line parallel batch machine) without restarts, we can prove that there exists no online algorithm with a competitive ratio less than $\frac{1+\sqrt{5}}{2} \approx 1.618$ by using the proving method of Theorem 1 in Zhang et al. (2001). Thus by using limited restarts, Algorithm H has better performance than the best possible online algorithms of the corresponding problems without restarts.

For online scheduling problems, since we don't know future arrival jobs' information beforehand, we usually use a strategy of moderate wait instead of processing a job immediately when the machine is idle. For example, the condition of $t \geq \frac{1}{2}p(U(t))$ in Step 2 of Algorithm H and the restart time $t = 2p(B_k) - p_h$ in Step 4.3.1 of Algorithm H mean that the machine will have moderate wait before starting a free batch or a restricted batch. Since our problems are online scheduling problems and the design of our algorithm makes full use of the problems' structure and properties, we design a best possible online algorithm H. But it isn't easy that popular deterministic optimization approaches such as relaxation algorithms of integer programming get best possible online algorithms. Since jobs' information are deterministic when they arrive, our problems are different from stochastic optimization problems. Note that stochastic optimization approaches' ideas may be used to solve online scheduling problems. In the future we may design a corresponding randomized algorithm which may have better performance than best possible deterministic online algorithms.

Acknowledgements This work was supported by NSFC (11701148, 11871213, 11571321,11501279), Henan University of Engineering (D2016017) and the Young Backbone Teachers of Henan Colleges (2019GGJS202).

Appendix

Proof of Lemma 2.4 Because $S_n < S_{n-1} + p(B_{n-1})$, B_n is a restricted batch. Let p' be the longest processing time of jobs arriving in the time interval $(S_{n-1}, S_n]$ and q'' be the largest delivery time of jobs arriving in the time interval $(S_{n-1}, S_n]$. Among the jobs which arrive in the time interval $(S_{n-1}, S_n]$ and have the processing time p', select one which arrives latest as the job J'. Let r' and q' be the arrival time and the delivery time of J', respectively. Among the jobs which arrive in the time interval $(S_{n-1}, S_n]$ and have the delivery time q'', select one which arrives latest as the job J''. Let r'' and q'' be the arrival time and the delivery time of J'', respectively. Thus we have $p' \geq p''$ and $q' \leq q''$. Considering that both J' and J'' arrive after the time S_{n-1}, $p' \geq p''$ and $q' \leq q''$, then by Lemma 2.1, we have $L_{\text{opt}} \geq S_{n-1} + p' + q''$. Since B_n is a restricted

batch, by Step 4.2 and 4.3 of Algorithm H, we have $p(B_{n-1}) < p' < \frac{3}{2}p(B_{n-1})$ or $\frac{3}{4}p(B_{n-1}) < p' \leq p(B_{n-1})$. Note that $r' > S_{n-1} \geq \frac{1}{2}p(B_{n-1})$ by Observation 2.3.

Case 1. $p(B_{n-1}) < p' < \frac{3}{2}p(B_{n-1})$.

By Step 4.2 of Algorithm H, we have $p(B_n) = \max\{p(B_{n-1}), p'\} = p'$, $S_n = \max\{p', r'\}$ and $q(B_n) = \max\{q'', q(B_{n-1})\}$.

As $p(B_{n-1}) < p'$, considering that all jobs have agreeable processing times and delivery times, then the largest delivery time of jobs in B_{n-1} is not more than q'. Thus $q(B_{n-1}) \leq q' \leq q''$ and $q(B_n) = \max\{q'', q(B_{n-1})\} = q''$. Now $L_{\text{on}} = S_n + p(B_n) + q(B_n) = S_n + p' + q''$. Now we distinguish the following cases.

If $r' > p'$, then $S_n = r'$ and $L_{\text{on}} = S_n + p(B_n) + q(B_n) = r' + p' + q''$. Note that $q' \leq q''$. If $q' = q''$, then $L_{\text{opt}} \geq r' + p' + q' = r' + p' + q'' = L_{\text{on}}$, which implies $L_{\text{on}} = L_{\text{opt}}$. If $q' < q''$, noting that all jobs have agreeable processing times and delivery times, then $p' \leq p''$. Also considering that $p' \geq p''$, we have $p' = p''$. By the definitions of J' and J'' and $p' = p''$, we have $r' \geq r''$. Since B_n interrupts B_{n-1} at time $S_n = r'$ and $r' > p'$, then by Step 4.2 of Algorithm H, we have $r' < \frac{5}{4}p(B_{n-1})$. Then $r'' \leq r' < \frac{5}{4}p(B_{n-1})$. If $r'' < r'$ and $r'' \geq p''$, considering that $p' = p''$ and $r'' < \frac{5}{4}p(B_{n-1})$, then by Step 4.2 of Algorithm H, B_n will interrupt B_{n-1} at time r'', a contradiction to $S_n = r'$ and $r'' < r'$. If $r'' < r'$ and $r'' < p''$, considering that $p'' = p'$, $r' > p'$ and p' is the longest processing time of jobs arriving in the time interval $(S_{n-1}, S_n] = (S_{n-1}, r']$, then by Step 4.2 of Algorithm H, B_n will interrupt B_{n-1} at time $p'' = p'$, a contradiction to the fact that $S_n = r'$ and $r' > p'$. So the assumption of $r'' < r'$ is not right and we have $r'' = r'$. So if $q' < q''$, we have $p' = p''$ and $r'' = r'$. Then $L_{\text{opt}} \geq r'' + p'' + q'' = r' + p' + q'' = L_{\text{on}}$, which implies $L_{\text{on}} = L_{\text{opt}}$.

If $r' \leq p'$, then $S_n = p'$ and $L_{\text{on}} = S_n + p(B_n) + q(B_n) = 2p' + q''$. Thus $\frac{L_{\text{on}}}{L_{\text{opt}}} \leq \frac{2p'+q''}{S_{n-1}+p'+q''} \leq \frac{2p'+q''}{\frac{1}{2}p(B_{n-1})+p'+q''} \leq \frac{2p'}{\frac{1}{2}p(B_{n-1})+p'} \leq \frac{2\times\frac{3}{2}p(B_{n-1})}{\frac{3}{2}p(B_{n-1})+\frac{1}{2}p(B_{n-1})} = \frac{3}{2}$.

Case 2. $\frac{3}{4}p(B_{n-1}) < p' \leq p(B_{n-1})$.

By Step 4.3 of Algorithm H, we have $p(B_n) = \max\{p', p(B_{n-1})\} = p(B_{n-1})$, $r' < \frac{5}{4}p(B_{n-1})$ and $S_n = \max\{2p(B_{n-1}) - p', r'\}$. Note that $S_n < \frac{5}{4}p(B_{n-1})$ as $\frac{3}{4}p(B_{n-1}) < p'$ and $r' < \frac{5}{4}p(B_{n-1})$. Then $L_{\text{on}} = S_n + p(B_n) + q(B_n) = S_n + p(B_{n-1}) + q(B_n) < \frac{9}{4}p(B_{n-1}) + q(B_n)$.

Subcase 2.1 $q'' > q(B_{n-1})$.

Then $q(B_n) = \max\{q(B_{n-1}), q''\} = q''$. Now $L_{\text{on}} = S_n + p(B_n) + q(B_n) = S_n + p(B_{n-1}) + q'' < \frac{9}{4}p(B_{n-1}) + q''$.

Note that q^p_{n-1} is the delivery time of J^p_{n-1} and $p(B_{n-1})$ is the processing time of J^p_{n-1}. Considering that $q'' > q(B_{n-1}) \geq q^p_{n-1}$, we have $p'' \geq p(B_{n-1})$. Considering that $p'' \leq p'$, $p' \leq p(B_{n-1})$ and $p'' \geq p(B_{n-1})$, we have $p'' = p' = p(B_{n-1})$. Then $L_{\text{opt}} \geq S_{n-1} + p' + q'' \geq \frac{1}{2}p(B_{n-1}) + p(B_{n-1}) + q'' = \frac{3}{2}p(B_{n-1}) + q''$ by Observation 2.3. Thus $\frac{L_{\text{on}}}{L_{\text{opt}}} \leq \frac{\frac{9}{4}p(B_{n-1})+q''}{\frac{3}{2}p(B_{n-1})+q''} \leq \frac{3}{2}$.

Subcase 2.2 $q'' \leq q(B_{n-1})$.

Then $q(B_n) = \max\{q(B_{n-1}), q''\} = q(B_{n-1})$. Now $L_{\text{on}} = S_n + p(B_n) + q(B_n) = S_n + p(B_{n-1}) + q(B_{n-1})$. Since B_n interrupts B_{n-1}, we have $p' \geq \frac{3}{4}p(B_{n-1}) +$

$\frac{1}{4}q(B_{n-1})$ by Step 4.3 of Algorithm H. As $p' \geq \frac{3}{4}p(B_{n-1}) + \frac{1}{4}q(B_{n-1})$ and $p' \leq p(B_{n-1})$, we have $q(B_{n-1}) \leq p(B_{n-1})$.

By Lemma 2.2, for the batch B_{n-1}, at least one of the jobs J_{n-1}^p and J_{n-1}^q has the processing time $p(B_{n-1})$ and the delivery time $q(B_{n-1})$ and denote the job by J_{n-1}^*.

If $S_n = 2p(B_{n-1}) - p'$, then $L_{\text{on}} = S_n + p(B_{n-1}) + q(B_{n-1}) = 2p(B_{n-1}) - p' + p(B_{n-1}) + q(B_{n-1}) \leq 2p(B_{n-1}) - (\frac{3}{4}p(B_{n-1}) + \frac{1}{4}q(B_{n-1})) + p(B_{n-1}) + q(B_{n-1}) = \frac{9}{4}p(B_{n-1}) + \frac{3}{4}q(B_{n-1})$. Note that J' arrives after time S_{n-1}, and the processing time of J_{n-1}^* is $p(B_{n-1})$, and the delivery time of J_{n-1}^* is $q(B_{n-1})$. Thus $L_{\text{opt}} \geq \min\{S_{n-1} + p(B_{n-1}) + q(B_{n-1}), p(B_{n-1}) + p'\}$ whether J' and J_{n-1}^* are in a common batch in the optimal off-line schedule π or not. Then $L_{\text{opt}} \geq \min\{S_{n-1} + p(B_{n-1}) + q(B_{n-1}), p(B_{n-1}) + p'\} \geq \min\{\frac{1}{2}p(B_{n-1}) + p(B_{n-1}) + q(B_{n-1}), p(B_{n-1}) + \frac{3}{4}p(B_{n-1}) + \frac{1}{4}q(B_{n-1})\} \geq \min\{\frac{3}{2}p(B_{n-1}) + q(B_{n-1}), p(B_{n-1}) + \frac{2}{4}p(B_{n-1}) + \frac{2}{4}q(B_{n-1})\} = \frac{3}{2}p(B_{n-1}) + \frac{1}{2}q(B_{n-1})$. Here the last but one inequality hods as $p(B_{n-1}) \geq q(B_{n-1})$. Thus $\frac{L_{\text{on}}}{L_{\text{opt}}} \leq \frac{\frac{9}{4}p(B_{n-1}) + \frac{3}{4}q(B_{n-1})}{\frac{3}{2}p(B_{n-1}) + \frac{1}{2}q(B_{n-1})} = \frac{3}{2}$.

If $S_n = r'$, then $r' \geq 2p(B_{n-1}) - p'$ and $L_{\text{opt}} \geq r' + p' \geq 2p(B_{n-1})$. Now $L_{\text{on}} = S_n + p(B_{n-1}) + q(B_{n-1}) = r' + p(B_{n-1}) + q(B_{n-1})$. Considering that $p' \geq \frac{3}{4}p(B_{n-1}) + \frac{1}{4}q(B_{n-1})$ and $q(B_{n-1}) \leq p(B_{n-1})$, we have $\frac{L_{\text{on}}}{L_{\text{opt}}} \leq \frac{r'+p(B_{n-1})+q(B_{n-1})}{r'+p'} = 1 + \frac{p(B_{n-1})+q(B_{n-1})-p'}{r'+p'} \leq 1 + \frac{\frac{1}{4}p(B_{n-1})+\frac{3}{4}q(B_{n-1})}{2p(B_{n-1})} \leq 1 + \frac{p(B_{n-1})}{2p(B_{n-1})} = \frac{3}{2}$. The result follows. □

Proof of Lemma 2.5 As $S_n = S_{n-1} + p(B_{n-1})$, then B_n is a free batch and all jobs in B_n arrive after the time S_{n-1}. Now $L_{\text{on}} = S_n + p(B_n) + q(B_n) = S_{n-1} + p(B_{n-1}) + p(B_n) + q(B_n)$. By Lemma 2.2, $L_{\text{opt}} \geq r(B_n) + p(B_n) + q(B_n) > S_{n-1} + p(B_n) + q(B_n)$. Thus

$$L_{\text{on}} - L_{\text{opt}} \leq p(B_{n-1}). \tag{1}$$

Let p' be the longest processing time of jobs arriving in the time interval $(S_{n-2}, S_{n-1}]$. Among the jobs which arrive in the time interval $(S_{n-2}, S_{n-1}]$ and have a processing time p', select one which arrives latest as the job J'. Let r' and q' be the arrival time and the delivery time of J', respectively. Considering that B_{n-1} is a restricted batch, then by Algorithm H, we have $p(B_{n-2}) < p' < \frac{3}{2}p(B_{n-2})$ or $\frac{3}{4}p(B_{n-2}) < p' \leq p(B_{n-2})$. By Lemma 2.2, at least one of the jobs J_n^p and J_n^q has the processing time of $p(B_n)$ and the delivery time of $q(B_n)$. Let J_n^* be one of J_n^p and J_n^q whose processing time is $p(B_n)$ and delivery time is $q(B_n)$.

Case 1. $p(B_{n-2}) < p' < \frac{3}{2}p(B_{n-2})$.

Then $p(B_{n-1}) = \max\{p(B_{n-2}), p'\} = p'$. By Step 4.2 of Algorithm H, $S_{n-1} = \max\{p', r'\}$.

If $r' \geq p'$, we have $S_{n-1} = r'$ and $L_{\text{opt}} \geq r' + p' \geq 2p' = 2p(B_{n-1})$. Then by the inequality (1), we have $\frac{L_{\text{on}} - L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{p(B_{n-1})}{2p(B_{n-1})} = \frac{1}{2}$.

If $r' < p'$, we have $S_{n-1} = p' = p(B_{n-1})$. If J' starts not before J_n^* in the optimal off-line schedule π, considering that J_n^* arrives after the time S_{n-1}, then

$L_{\text{opt}} \geq S_{n-1} + p' = 2p(B_{n-1})$. Thus we have $L_{\text{on}} - L_{\text{opt}} \leq p(B_{n-1}) \leq \frac{1}{2}L_{\text{opt}}$. If J' starts before J_n^* in the schedule π, then $L_{\text{opt}} \geq r' + p' + p(B_n) + q(B_n) > S_{n-2} + p(B_{n-1}) + p(B_n) + q(B_n)$. Considering that $\frac{1}{3}p(B_{n-1}) = \frac{1}{3}p' < \frac{1}{2}p(B_{n-2})$ and $S_{n-2} \geq \frac{1}{2}p(B_{n-2})$, we have $S_{n-2} \geq \frac{1}{3}p(B_{n-1})$. Then $L_{\text{opt}} > S_{n-2} + p(B_{n-1}) + p(B_n) + q(B_n) \geq \frac{1}{3}p(B_{n-1}) + p(B_{n-1}) = \frac{4}{3}p(B_{n-1})$ and $L_{\text{on}} - L_{\text{opt}} \leq S_{n-1} + p(B_{n-1}) + p(B_n) + q(B_n) - (S_{n-2} + p(B_{n-1}) + p(B_n) + q(B_n)) = S_{n-1} - S_{n-2} \leq p(B_{n-1}) - \frac{1}{3}p(B_{n-1}) = \frac{2}{3}p(B_{n-1})$. Thus $\frac{L_{\text{on}} - L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{\frac{2}{3}p(B_{n-1})}{\frac{4}{3}p(B_{n-1})} = \frac{1}{2}$.

Case 2. $\frac{3}{4}p(B_{n-2}) < p' \leq p(B_{n-2})$.

Then $p(B_{n-1}) = \max\{p(B_{n-2}), p'\} = p(B_{n-2})$. By Step 4.3 of Algorithm H, we have $S_{n-1} = \max\{2p(B_{n-2}) - p', r'\}$. Then $L_{\text{on}} = S_{n-1} + p(B_{n-1}) + p(B_n) + q(B_n) = S_{n-1} + p(B_{n-2}) + p(B_n) + q(B_n)$.

If $r' \geq 2p(B_{n-2}) - p'$, then $L_{\text{opt}} \geq r' + p' \geq 2p(B_{n-2}) = 2p(B_{n-1})$. Thus $L_{\text{on}} - L_{\text{opt}} \leq p(B_{n-1}) \leq \frac{1}{2}L_{\text{opt}}$ by the inequality (1).

If $r' < 2p(B_{n-2}) - p'$, then $S_{n-1} = 2p(B_{n-2}) - p' < \frac{5}{4}p(B_{n-2})$. Now we distinguish the following cases. If J_{n-2}^p or J' starts not before J_n^* in the schedule π, considering that J_n^* arrives after the time S_{n-1}, then $L_{\text{opt}} \geq S_{n-1} + \min\{p(B_{n-2}), p'\} = S_{n-1} + p' = 2p(B_{n-2}) = 2p(B_{n-1})$. Thus $L_{\text{on}} - L_{\text{opt}} \leq p(B_{n-1}) \leq \frac{1}{2}L_{\text{opt}}$. If both J_{n-2}^p and J' start before J_n^* in the schedule π, then J_n^* starts not earlier than the time $\min\{p(B_{n-2}) + p', r' + p(B_{n-2})\}$ in the schedule π whether J_{n-2}^p and J' are in a common batch or not in the schedule π. Note that $\min\{p(B_{n-2}) + p', r' + p(B_{n-2})\} \geq \min\{p(B_{n-2}) + \frac{3}{4}p(B_{n-2}), S_{n-2} + p(B_{n-2})\} \geq \min\{p(B_{n-2}) + \frac{3}{4}p(B_{n-2}), \frac{1}{2}p(B_{n-2}) + p(B_{n-2})\} = \frac{3}{2}p(B_{n-2})$. Then J_n^* starts not earlier than the time $\frac{3}{2}p(B_{n-2})$ in the schedule π and $L_{\text{opt}} \geq \frac{3}{2}p(B_{n-2}) + p(B_n) + q(B_n)$. Thus $\frac{L_{\text{on}}}{L_{\text{opt}}} \leq \frac{S_{n-1} + p(B_{n-2}) + p(B_n) + q(B_n)}{\frac{3}{2}p(B_{n-2}) + p(B_n) + q(B_n)} \leq \frac{\frac{5}{4}p(B_{n-2}) + p(B_{n-2}) + p(B_n) + q(B_n)}{\frac{3}{2}p(B_{n-2}) + p(B_n) + q(B_n)} \leq \frac{3}{2}$. The result follows. □

Proof of Lemma 2.6 Since $S_n = S_{n-1} + p(B_{n-1})$, then B_n is a free batch, and $r(B_n) > S_{n-1}$, and $L_{\text{on}} = S_{n-1} + p(B_{n-1}) + p(B_n) + q(B_n)$. By Lemma 2.2, $L_{\text{opt}} \geq r(B_n) + p(B_n) + q(B_n) > S_{n-1} + p(B_n) + q(B_n)$. Then

$$L_{\text{on}} - L_{\text{opt}} \leq p(B_{n-1}). \tag{2}$$

Case 1. $p(B_n) \geq \frac{3}{2}p(B_{n-1})$.

Note that $L_{\text{on}} - L_{\text{opt}} \leq p(B_{n-1})$ and $S_{n-1} \geq \frac{1}{2}p(B_{n-1})$ by Observation 2.3. Then $L_{\text{opt}} > S_{n-1} + p(B_n) + q(B_n) \geq \frac{1}{2}p(B_{n-1}) + \frac{3}{2}p(B_{n-1}) = 2p(B_{n-1})$. Thus $\frac{L_{\text{on}} - L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{p(B_{n-1})}{2p(B_{n-1})} = \frac{1}{2}$.

Case 2. $p(B_{n-1}) < p(B_n) < \frac{3}{2}p(B_{n-1})$.

Note that the arrival time of J_n^p is r_n^p and the processing time of J_n^p is $p(B_n)$. Now we prove that $r_n^p \geq p(B_{n-1})$. If $r_n^p < p(B_{n-1})$, noting that $r_n^p < p(B_{n-1}) < p(B_n)$, then B_n will interrupt B_{n-1} at time $p(B_n)$ by Step 4.2 of Algorithm H, a contradiction.

So we have $r_n^p \geq p(B_{n-1})$. Then $L_{\text{opt}} \geq r_n^p + p(B_n) \geq 2p(B_{n-1})$. Thus by the inequality (2), $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{p(B_{n-1})}{2p(B_{n-1})} = \frac{1}{2}$.

Case 3. $p(B_n) \leq p(B_{n-1})$.

If $r_n^p + p(B_n) \geq 2p(B_{n-1})$, we have $L_{\text{opt}} \geq r_n^p + p(B_n) \geq 2p(B_{n-1})$. Then $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{p(B_{n-1})}{2p(B_{n-1})} = \frac{1}{2}$. Now we assume that $r_n^p + p(B_n) < 2p(B_{n-1})$, i.e., $r_n^p < 2p(B_{n-1}) - p(B_n)$.

Subcase 3.1. $p(B_n) = p(B_{n-1})$.

If $p(B_{n-1}) > q(B_{n-1})$, then $\frac{3}{4}p(B_{n-1})+\frac{1}{4}q(B_{n-1}) < p(B_{n-1}) = p(B_n)$. Considering that $r_n^p < 2p(B_{n-1}) - p(B_n) = p(B_{n-1}) = p(B_n)$ and $p(B_n) > \frac{3}{4}p(B_{n-1}) + \frac{1}{4}q(B_{n-1})$, then B_n will interrupt B_{n-1} at time $2p(B_{n-1}) - p(B_n) = p(B_{n-1})$ by Step 4.3 of Algorithm H, a contradiction. So we have $p(B_{n-1}) \leq q(B_{n-1})$. By Lemma 2.2, $L_{\text{opt}} \geq p(B_{n-1}) + q(B_{n-1}) \geq 2p(B_{n-1})$. Then by the inequality (2), we have $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{p(B_{n-1})}{2p(B_{n-1})} = \frac{1}{2}$.

Subcase 3.2. $p(B_n) < p(B_{n-1})$.

By Lemma 2.2, for the batch B_{n-1}, at least one of the jobs J_{n-1}^p and J_{n-1}^q has the processing time of $p(B_{n-1})$ and the delivery time of $q(B_{n-1})$. So let J_{n-1}^* be one of the jobs J_{n-1}^p and J_{n-1}^q whose processing time is $p(B_{n-1})$ and delivery time is $q(B_{n-1})$. Similarly, let J_n^* be one of the jobs J_n^p and J_n^q whose processing time is $p(B_n)$ and delivery time is $q(B_n)$.

If $p(B_{n-1}) \leq q(B_{n-1})$, by Lemma 2.2, we have $L_{\text{opt}} \geq p(B_{n-1}) + q(B_{n-1}) \geq 2p(B_{n-1})$. Then $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{p(B_{n-1})}{2p(B_{n-1})} = \frac{1}{2}$. Now we assume that $p(B_{n-1}) > q(B_{n-1})$.

If $\frac{3}{4}p(B_{n-1})+\frac{1}{4}q(B_{n-1}) < p(B_n) < p(B_{n-1})$, considering that $r_n^p < 2p(B_{n-1}) - p(B_n) < \frac{5}{4}p(B_{n-1})$, then B_n will interrupt B_{n-1} by Step 4.3 of Algorithm H, a contradiction. So we have $p(B_n) \leq \frac{3}{4}p(B_{n-1}) + \frac{1}{4}q(B_{n-1})$.

If J_{n-1}^* starts not earlier than the time S_{n-1} in the schedule π, then $L_{\text{opt}} \geq S_{n-1} + p(B_{n-1})+q(B_{n-1})$. Note that the processing time of J_n^q is p_n^q and the processing time of J_{n-1}^p is $p(B_{n-1})$. Then $p_n^q \leq p(B_n) < p(B_{n-1})$. Considering that $p_n^q < p(B_{n-1})$, then the delivery time of J_n^q is not more than the delivery time of J_{n-1}^p, i.e., $q(B_n) \leq q_{n-1}^p$. Then $q(B_n) \leq q_{n-1}^p \leq q(B_{n-1})$. Thus $L_{\text{on}} - L_{\text{opt}} \leq S_{n-1} + p(B_{n-1}) + p(B_n) + q(B_n) - (S_{n-1} + p(B_{n-1}) + q(B_{n-1})) \leq p(B_n) \leq \frac{3}{4}p(B_{n-1}) + \frac{1}{4}q(B_{n-1})$. Note that $L_{\text{opt}} \geq S_{n-1} + p(B_{n-1}) + q(B_{n-1}) \geq \frac{1}{2}p(B_{n-1}) + p(B_{n-1}) + q(B_{n-1})$. Then $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \leq \frac{1}{2}$.

Now we assume that J_{n-1}^* starts earlier than the time S_{n-1} in the schedule π. Considering that all jobs in B_n arrive after the time S_{n-1}, then J_{n-1}^* starts before J_n^* in the schedule π. Then $L_{\text{opt}} \geq r(B_{n-1}) + p(B_{n-1}) + p(B_n) + q(B_n)$.

As B_{n-1} is a free batch, by Observation 2.1, we have $S_{n-1} = \frac{1}{2}p(B_{n-1})$, or $S_{n-1} = r(B_{n-1})$, or $S_{n-1} = S_{n-2} + p(B_{n-2})$.

If $S_{n-1} = r(B_{n-1})$ or $S_{n-1} = \frac{1}{2}p(B_{n-1})$, then $L_{\text{on}} - L_{\text{opt}} \le S_{n-1} + p(B_{n-1}) + p(B_n) + q(B_n) - (r(B_{n-1}) + p(B_{n-1}) + p(B_n) + q(B_n)) = S_{n-1} - r(B_{n-1}) \le \frac{1}{2}p(B_{n-1})$ and $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \le \frac{1}{2}$.

From now on we assume that $S_{n-1} = S_{n-2}+p(B_{n-2})$. So $L_{\text{on}} = S_{n-2}+p(B_{n-2})+p(B_{n-1})+p(B_n)+q(B_n)$. Since B_{n-1} is a free batch, we have $r(B_{n-1}) > S_{n-2}$. Thus $L_{\text{opt}} \ge r(B_{n-1}) + p(B_{n-1}) + p(B_n) + q(B_n) > S_{n-2} + p(B_{n-1}) + p(B_n) + q(B_n)$ and $L_{\text{on}} - L_{\text{opt}} \le p(B_{n-2})$. Noting that $L_{\text{on}} - L_{\text{opt}} \le p(B_{n-1})$, then

$$L_{\text{on}} - L_{\text{opt}} \le \min\{p(B_{n-1}),\, p(B_{n-2})\}. \tag{3}$$

By Lemma 2.2, one job of the batch B_{n-2} has the processing time of $p(B_{n-2})$ and the delivery time of $q(B_{n-2})$. Let J^*_{n-2} be such a job in B_{n-2} with the processing time $p(B_{n-2})$ and the delivery time $q(B_{n-2})$.

If B_{n-2} is a restricted batch, then we have $S_{n-2} \ge p(B_{n-2})$ by Observation 2.2. Thus $L_{\text{opt}} \ge r(B_n) > S_{n-1} = S_{n-2} + p(B_{n-2}) \ge 2p(B_{n-2})$. Hence $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \le \frac{\min\{p(B_{n-1}),p(B_{n-2})\}}{2p(B_{n-2})} \le \frac{1}{2}$ by the inequality (3).

If B_{n-2} is a free batch, we distinguish the following cases. If $p(B_{n-1}) \ge \frac{3}{2}p(B_{n-2})$, we have $L_{\text{opt}} \ge r(B_{n-1}) + p(B_{n-1}) > S_{n-2} + p(B_{n-1}) \ge \frac{1}{2}p(B_{n-2}) + \frac{3}{2}p(B_{n-2}) = 2p(B_{n-2})$. Thus $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \le \frac{\min\{p(B_{n-1}),p(B_{n-2})\}}{2p(B_{n-2})} \le \frac{1}{2}$. If $p(B_{n-2}) < p(B_{n-1}) < \frac{3}{2}p(B_{n-2})$ or $[\frac{3}{4}p(B_{n-2}) < p(B_{n-1}) \le p(B_{n-2})$ and $p(B_{n-1}) \ge \frac{3}{4}p(B_{n-2}) + \frac{1}{4}q(B_{n-2})]$, noting that B_{n-1} does not interrupt the free batch B_{n-2}, then $r^p_{n-1} \ge \frac{5}{4}p(B_{n-2})$ by Step 4.2 and 4.3 of Algorithm H. Note that the arrival time of J^p_{n-1} is r^p_{n-1} and the processing time of J^p_{n-1} is $p(B_{n-1})$. Thus $L_{\text{opt}} \ge r^p_{n-1} + p(B_{n-1}) \ge \frac{5}{4}p(B_{n-2}) + \frac{3}{4}p(B_{n-2}) \ge 2\min\{p(B_{n-1}),\, p(B_{n-2})\}$. So $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \le \frac{\min\{p(B_{n-1}),p(B_{n-2})\}}{2\min\{p(B_{n-1}),p(B_{n-2})\}} \le \frac{1}{2}$. If $[\frac{3}{4}p(B_{n-2}) < p(B_{n-1}) \le p(B_{n-2})$ and $p(B_{n-1}) < \frac{3}{4}p(B_{n-2}) + \frac{1}{4}q(B_{n-2})]$ or $p(B_{n-1}) \le \frac{3}{4}p(B_{n-2})$, we distinguish the following two cases. If J^p_{n-1} and J^*_{n-2} are not in a common batch in the schedule π, $L_{\text{opt}} \ge p(B_{n-1}) + p(B_{n-2}) \ge 2\min\{p(B_{n-1}),\, p(B_{n-2})\}$. Thus $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \le \frac{\min\{p(B_{n-1}),p(B_{n-2})\}}{2\min\{p(B_{n-1}),p(B_{n-2})\}} = \frac{1}{2}$. If J^p_{n-1} and J^*_{n-2} are in a common batch in the schedule π, then $L_{\text{opt}} \ge r^p_{n-1}+p(B_{n-2})+q(B_{n-2}) > S_{n-2}+p(B_{n-2})+q(B_{n-2}) \ge \frac{1}{2}p(B_{n-2}) + p(B_{n-2}) + q(B_{n-2}) = \frac{3}{2}p(B_{n-2}) + q(B_{n-2})$. Thus $\frac{L_{\text{on}}-L_{\text{opt}}}{L_{\text{opt}}} \le \frac{p(B_{n-1})}{\frac{3}{2}p(B_{n-2})+q(B_{n-2})} \le \frac{\frac{3}{4}p(B_{n-2})+\frac{1}{4}q(B_{n-2})}{\frac{3}{2}p(B_{n-2})+q(B_{n-2})} \le \frac{1}{2}$. The result follows. □

References

1. Akker, M.V.D., Hoogeveen, H., Vakhania, N.: Restarts can help in the online minimization of the maximum delivery time on a single machine. J. Sched. **3**, 333–341 (2003)

2. Epstein, L., Van Stee, R.: Lower bounds for on-line single-machine scheduling. Theor. Comput. Sci. **299**, 439–450 (2003)
3. Fu, R.Y., Tian, J., Yuan, J.J., He, C.: On-line scheduling on a batch machine to minimize makespan with limited restarts. Oper. Res. Lett. **36**, 255–258 (2008)
4. Fu, R.Y., Cheng, T.C.E., Ng, C.T., Yuan, J.J.: Online scheduling on two parallel batching machines with limited restarts to minimize the makespan. Inf. Process. Lett. **110**, 444–450 (2010)
5. Hoogeveen, H., Potts, C.N., Woeginger, G.J.: On-line Scheduling on a single machine: maximizing the number of early jobs. Oper. Res. Lett. **27**, 193–197 (2000)
6. Hoogeveen, J.A., Vestjens, A.P.A.: A best possible deterministic on-line algorithm for minimizing maximum delivery time on a single machine. SIAM J. Discrete Math. **13**, 56–63 (2000)
7. Lee, C.Y., Uzsoy, R., Martin-Vega, L.A.: Efficient algorithms for scheduling semi-conductor burn-in operations. Oper. Res. **40**, 764–775 (1992)
8. Liu, P.H., Lu, X.W.: Online unbounded batch scheduling on parallel machines with delivery times. J. Comb. Optim. **29**(1), 228–236 (2015)
9. Liu, H.L., Yuan, J.J., Li, W.J.: Online scheduling of equal length jobs on unbounded parallel batch processing machines with limited restart. J. Comb. Optim. **31**, 1609–1622 (2016)
10. Tian, J., Cheng, T.C.E., Ng, C.T., Yuan, J.J.: An improved on-line algorithm for single parallel-batch machine scheduling with delivery times. Discrete Appl. Math. **160**, 1191–1210 (2012)
11. Tian, J., Fu, R.Y., Yuan, J.J.: Online over time scheduling on parallel batch machines: a survey. J. Oper. Res. Soc. China **2**, 445–454 (2014)
12. Tian, J., Wang, Q., Fu, R.Y., Yuan, J.J.: Online scheduling on the unbounded drop-line batch machines to minimize the maximum delivery completion time. Theor. Comput. Sci. **617**, 65–68 (2016)
13. Uzsoy, R., Lee, C.Y., Martin-Vega, L.A.: A review of production planning and scheduling models in the semiconductor industry, part I: system characteristics, performance evaluation and production planning. IIE Trans. Sched. Logist. **24**, 47–61 (1992)
14. Uzsoy, R., Lee, C.Y., Martin-Vega, L.A.: A survey of production planning and scheduling models in the semiconductor industry, part II: shop-floor control. IIE Trans. Sched. Logist. **26**, 44–55 (1994)
15. Van Stee, R., Poutré, H.L.: Minimizing the total completion time on-line on a single machine, using restarts. J. Algorithms. **57**, 95–129 (2005)
16. Wei, Z.G.: Scheduling on a batch machine with item-availability to minimize total weighted completion time. Master Degree Thesis, Zhengzhou University, (2011)
17. Yuan, J.J., Li, S.S., Tian, J., Fu, R.Y.: A best on-line scheduling for the single machine parallel-batch scheduling with restricted delivery time. J. Comb. Optim. **17**, 206–213 (2009)
18. Yuan, J.J., Fu, R.Y., Ng, C.T., Cheng, T.C.E.: A best online algorithm for unbounded parallel-batch scheduling with restarts to minimize makespan. J. Sched. **14**, 361–369 (2011)