SEMI-ONLINE SCHEDULING ON TWO UNIFORM PARALLEL MACHINES
WITH INITIAL LOOKAHEAD

FEIFENG ZHENG¹, YUHONG CHEN¹ AND MING LIU²,*

Abstract. This work studies the problem of semi-online scheduling on two uniform parallel machines
with speeds 1 and \( s \geq 2 \), respectively. We introduce a novel concept of initial lookahead in which any
deterministic online algorithm has the full knowledge of the first \( k \) jobs at the beginning, while the
remaining jobs are released one-by-one in the online over-list mode. The objective of the considered
problem is to minimize the makespan. We focus on the case where the first \( k \) jobs are of a total
processing time not less than \( (s + 1)\Delta \) where \( \Delta \) is the largest job length, and it is assumed that \( s \)
is an integer. We prove a lower bound of \( \frac{s^2 + s + 1}{2s^2 + s} \), and propose a deterministic semi-online
algorithm with competitive ratio of \( \frac{(s+1)^2}{2s^2 + s} \). The ratio is at most 9/7 and much less than that of 1.618 for the
results demonstrate that a finite ability of initial lookahead can greatly improve the competitiveness of
online algorithms.

Mathematics Subject Classification. 90B35.

Received November 18, 2022. Accepted February 11, 2024.

1. Introduction

Online parallel machine scheduling depicts a real-time scheduling environment in practice where a manufac-
turer (or decision maker) receives jobs and assigns them to parallel machines for processing one by one in either
over list or over time mode [20]. One of the merits of the online processing pattern in the operational level is
its small time complexity in generating a processing schedule. On the release of each job, the decision on the
processing of the job is usually made in a simple rule with no need of knowing any future jobs, and thus it is
easy for workers to fulfill the processing schedule and handle the jobs in real-time.

On the other hand, with regard to the objective value of the whole job sequence, the above processing
schedule generated in the online way may be far away from an optimal one due to the lack of knowledge of
future jobs when processing any released job. Therefore, researchers seek various approaches to improve the
competitive performance of deterministic online algorithms, and there have emerged many variants of semi-
online scheduling in recent decades, such as accessing partial knowledge of future jobs in advance [17, 36], job rearrangement [11, 16], augment of manufacturing resource [5], job rejection [8, 19], job splitting [22, 30], etc.

Among the above extensive semi-online scheduling variants, one interesting measure is to adopt the function of lookahead [13]. Since there are generally two online versions, i.e., over-list and over-time, in the literature, there are two lookahead modes accordingly. In the online over-list mode, an online player foresees a finite number of nearest future jobs on the release of any job, while in the online over-time mode the online player acquires at any time point the knowledge of all the jobs arriving in the next fixed-length time interval [24, 27]. Some previous studies have revealed that a sufficiently large ability of lookahead can boost the competitiveness of online algorithms [39].

In this work we focus on the online over-list mode of uniform parallel machine scheduling. As pointed out in Zheng et al. [41], the function of lookahead might consume an unacceptable extra cost for repeatedly acquiring the information of near-future jobs at any decision-making point. Moreover, accessing the knowledge of future jobs in a dynamic way may also enhance the implementation complexity of production scheduling. Therefore, this work introduces a new lookahead fashion which only foresees the information of the first several jobs (named initial jobs) at the beginning. All the remaining jobs cannot be observed by the online player unless they are released. We refer to the above new lookahead fashion as initial lookahead. With the initial lookahead ability, the online player can produce a locally optimal processing schedule for the jobs foreseen by the initial lookahead. For the rest jobs, however, they have to be processed as in the classic online scheduling fashion. Thus, this work aims to reveal how a limited ability of initial lookahead helps to improve the competitiveness of online algorithms. We focus on the semi-online over-list scheduling on two uniform parallel machines with initial lookahead.

For online algorithms, their performance is usually measured by competitive ratio [2]. Given any job sequence \( \sigma \) in an online scheduling problem with a minimization objective. Denote by \( C_\mathcal{A}(\sigma) \) the objective value of the schedule produced by an online algorithm \( \mathcal{A} \), and \( C_{OPT}(\sigma) \) that of the optimal algorithm OPT. We say algorithm \( \mathcal{A} \) is \( \rho \)-competitive if the following inequality holds for any sequence \( \sigma \).

\[
C_\mathcal{A}(\sigma) \leq \rho \cdot C_{OPT}(\sigma) + b
\]

where \( b \geq 0 \) is a constant irrelevant to sequence \( \sigma \). Obviously \( \rho \geq 1 \) is true. The minimum of \( \rho \) is defined as the competitive ratio of algorithm \( \mathcal{A} \). Moreover, the infimum of competitive ratio of the best online algorithm is referred to as the lower bound of competitive ratio, which is denoted by LB. \( \mathcal{A} \) is called an optimal online algorithm if its competitive ratio is equal to the lower bound, i.e., \( \rho = LB \).

The remainder of this work is organized as follows. Section 2 reviews previous related studies of online parallel machine scheduling. In Section 3 we present the problem of semi-online over-list scheduling on two uniform parallel machines. We prove a lower bound of competitiveness for the considered problem in Section 4, and propose a competitive semi-online algorithm named BAL in Section 5. Finally, Section 6 concludes this work and gives some potential directions in future research.

2. Literature review

This work investigates semi-online over-list scheduling on two uniform parallel machines with the initial lookahead ability. In the following we review three parts of the most relevant previous studies, i.e., online over-list scheduling on uniform parallel machines, semi-online identical parallel machine scheduling with lookahead, and semi-online over-list scheduling on two uniform parallel machines with partial knowledge.

2.1. Online over-list scheduling on uniform parallel machines

For the online over-list uniform parallel machine scheduling problem, there have been extensive results in the literature [3, 9, 15, 26, 38, 40].

With regard to the objective of makespan minimization, Cho and Sahni [7] address the online uniform parallel machine scheduling problem, and show that the greedy List Scheduling (LS) algorithm is currently best for the scenario with a small number of machines. LS always assigns the next job to a machine so as to complete it as
early as possible. They prove that LS has a competitive ratio of \( \phi = \frac{1 + \sqrt{5}}{2} \approx 1.618 \) for the case \( m = 2 \), and the ratio is at most \( 1 + \sqrt{(m-1)/2} \) for \( m \geq 3 \). This bound is proved tight for \( 3 \leq m \leq 6 \). Musitelli and Nicoletti [34] consider the case where \( m - 1 \) machines are of speed 1 and the last machine is of speed \( s \) (\( 0 \leq s \leq 1 \)). They first prove that LS is the best deterministic algorithm for some specific values of \( s \) and \( m = 3 \). They further develop a randomized online algorithm with competitive ratio less than that of LS for any \( s \) when \( m = 3 \). Jeż et al. [23] provide lower bounds for online scheduling on \( m \) uniform parallel machines and focus on the case with \( m \leq 11 \). Especially they prove lower bounds of 2.141391 and 2.314595 for \( m = 4 \) and 5, respectively, and computer-assisted new lower bounds for 6 \leq m \leq 11. For the cases where \( m = 2, 3 \), it has been shown that the lower bounds are equal to 1.618 and 2, respectively, and LS algorithm is optimally 1.618 and 2-competitive for \( m = 2, 3 \) [18]. Liu et al. [29] investigate online scheduling on two uniform machines under a grade of service (GoS). They first give a lower bound of \( 1 + \frac{2s}{s+2} \) when \( 0 < s \leq 1 \) and \( 1 + \frac{s+1}{s(2s+1)} \) when \( s > 1 \). They present an online algorithm High Speed machine First (HSF) which is \((s+1)\)-competitive for \( 0 < s \leq 1 \), \( \frac{s+1}{s} \)-competitive for the case \( s > 1 \) and \( \sum_1 < \sum_2 / s \), and 1-competitive for the case \( s > 1 \) and \( \sum_1 \geq \sum_2 / s \) where \( \sum_1 \) and \( \sum_2 \) are the total processing times of jobs that request higher GoS and respectively normal GoS. They also propose another algorithm EX-ONLINE (Extended ONLINE) which is optimally \( (1 + \frac{2s}{s+2}) \)-competitive for the case \( 2(\sqrt{2} - 1) \leq s \leq 1 \). Dolgui et al. [10] study the online uniform parallel machine scheduling scenario where the first \( k \) machines are of speed \( s > 1 \) and the rest \( m - k \) machines are of speed 1. They mainly propose a parametric scheme with competitive ratio of 2.618 when the ratio of \( m/k \) tends to infinity.

According to the above studies of online uniform parallel machines scheduling, when there are only \( m = 2 \) machines, online algorithm LS has an optimal competitive ratio of 1.618, while the ratio is at least 2 for the case with three or more machines. Therefore, many authors try various methods to break the competitive ratios, including using the function of lookahead or acquiring partial knowledge of future jobs in advance.

2.2. Semi-online identical parallel machine scheduling with lookahead

Some researchers employ the function of lookahead to improve the competitiveness of online algorithms. Whenever there releases a job, an online player with the lookahead ability can acquire either the knowledge of a finite number of nearest future jobs in the over-list setting or that of the jobs released in the following fixed-length time interval in the over-time setting [13].

Mandelbaum and Shabtay [33] investigate the semi-online over-list parallel machine scheduling with lookahead with the objective of makespan minimization. They assume that jobs with unit length are of different types, and each job can only be processed on a subset of machines. Given that the online player can foresee \( k \) nearest future jobs on the release of any job, they propose a \( 1 \)-competitive semi-online algorithm when there are only two types of jobs. They further show that no such optimal online algorithms exist if there are more than two types of jobs. Pan and Xu [37] consider the semi-online over-list scheduling problem of berth and quay cranes integrated allocation at a container terminal with lookahead, in which quay cranes can be roughly regarded as identical parallel machines. When there releases one vessel request, the online player can acquire the information of the next single request. For the objective of makespan minimization, they propose an optimal semi-online algorithm with competitive ratio of \((1 + \sqrt{2})/2\) and \(5/4\) for the case with 4 and 5 quay cranes, respectively. Li et al. [28] revisit the problem of Pan and Xu [37] and extend it to the scenario where the online player can foresee the next \( k \) (\( \geq 2 \)) requests at a time. They mainly present an optimal \( 7/6 \)-competitive semi-online algorithm for the case with \( k = 2 \), and provide a lower bound for the general case with \( k \geq 2 \).

Recently Zheng et al. [41] introduce the concept of workload fence in the semi-online identical parallel machine scheduling problem, in which the workload fence is equivalent to the initial lookahead in this work. They show that if the total length of jobs being foreseen by the initial lookahead is exactly equal to the largest processing time of job, then the lookahead ability can improve the competitiveness of online algorithms, and they provide an optimally \( 4/3 \)-competitive semi-online algorithm. When the total job length in the lookahead is either less or more than the longest job length, they prove that the initial lookahead ability seems helpless. Zheng et al. [42] investigate semi-online two identical parallel machines scheduling with initial-lookahead. They prove that
the classical LS algorithm is optimally \((k + 3)/(k + 2)\)-competitive when the first \(k\) initial jobs are all of the largest length \(\Delta\). When the \(k\) jobs are of a total length at least \(\alpha \Delta\) where \(\alpha \geq 2\) is an integer, they present a \((\alpha + 2)/\alpha - 1\)-competitive semi-online algorithm. For the case where \(\alpha = 1\), they show that the initial lookahead ability is useless and any online algorithm has a competitive ratio at least \(3/2\).

Some authors also study the semi-online over-time mode of identical parallel machine scheduling, and present interesting results on the competitiveness of deterministic online algorithms [6,25]. We observe that all the above works are for the identical parallel machines environment where machines are of the same processing speed. To the best of our knowledge, there are no any results for semi-online over-list scheduling with lookahead in the uniform parallel machines environment.

### 2.3. Semi-online over-list scheduling on two uniform parallel machines with partial knowledge

Although there are no results for applying lookahead in semi-online over-list uniform parallel machine scheduling, many authors have caught interest in the semi-online variation of uniform parallel machines scheduling where partial knowledge of future jobs is known beforehand to an online player. Among these studies many are for the setting with two uniform parallel machines [1,21].

Most previous studies focus on the objective of makespan minimization, and it is assumed that the two machines are of speeds 1 and \(s \geq 1\), respectively. Luo et al. [32] study the problem of semi-online scheduling on two uniform parallel machines where the largest job length is known in advance. They mainly propose an online algorithm with competitive ratio of \((2s + 1)/(s + 1)\). Ng et al. [35] consider two semi-online scheduling problems on two uniform parallel machines. For the first problem with known optimal value, they provide an optimal algorithm for \(s \in [(1 + \sqrt{3})/2, (1 + \sqrt{2})/4]\), and improved algorithms or/and lower bounds for \(s \in [(1 + \sqrt{2})/4, 3]\)

As a result, the largest gap between the competitive ratio and the lower bound decreases to 0.02192. For the second problem with known total sum, they also present algorithms and lower bounds. The largest gap between the competitive ratio and the lower bound is 0.01762, and the length of the interval over which the optimal algorithm is unknown is 0.47382. Cai and Yang [4] investigate semi-online scheduling on two uniform machines with known largest job length. When \(1 \leq s \leq \sqrt{2}\), they present an optimal online algorithm with competitive ratio of \(2(s + 1)/(s + 1)\), and propose an online algorithm with competitive ratio strictly less than that of the classical LS algorithm for the case \(\sqrt{2} < s < 1 + \sqrt{3}\). Dósa et al. [12] address the semi-online two uniform machines scheduling problem with known optimal makespan. They construct tight lower bounds for all values of \(s\) in the two intervals \([1.3956, 1.4430]\) and \([1.6667, 1.7258]\).

Lu and Liu [31] study semi-online over-list scheduling on two uniform machines under a GoS provision, where one machine is available for all jobs and the other one is only available for partial jobs. They consider three variants, where the optimal makespan, the total size of jobs, and the largest job size are known in advance, respectively. For the former two models, they propose optimally \(\min\{(1 + 2s)/(1 + s), (1 + s)/(s)\}\)-competitive algorithms, and for the third model, their algorithm is of competitive ratio \(\min\{1 + s, (1 + \sqrt{3})/2\}\) if \(0 < s < 1\), and \(\min\{(1 + \sqrt{1 + 4s})/2, (1 + s)/s\}\) if \(s \geq 1\).

In this work, we investigate the semi-online over-list two uniform parallel machines scheduling problem with initial lookahead, and similarly assume that the faster machine is of speed \(s \geq 2\). The model can also be regarded as a variant of semi-online scheduling with partial knowledge. That is, the knowledge of the first several jobs is known in advance. We target to uncover how a limited lookahead ability can improve the competitiveness of deterministic online algorithms.

### 3. Problem description

There are two uniform parallel machines \(M_1, M_2\) to process a sequence of jobs \(\sigma = (J_1, J_2, \ldots, J_n)\) that are released one by one in the online over-list mode. \(M_1\) and \(M_2\) are of processing speeds 1 and \(s \geq 2\), respectively, and they are called slow and fast machines, respectively. An online player has to irrecoverably assign each released job to either machine before observing the next job. With the initial lookahead ability, the online
player obtains the full knowledge of the first $k$ ($\geq 1$) jobs in the sequence at the beginning. More precisely, the processing time $p_j$ ($>0$) of job $J_j$ for $1 \leq j \leq k$ as well as their total length is known before the release of the first job. For the remaining jobs, if any, their processing times can only be observed on their release. Accordingly, the online player cannot access the value of $n$ unless the last job $J_n$ has been assigned. Denote by $\Delta = \max_{1 \leq j \leq n} p_j$ the longest processing time of job in the job sequence. Similarly the online player may not have the knowledge of $\Delta$ in advance. We consider the objective of the makespan minimization, i.e., minimizing the largest completion time of job, in this work.

Using the classic three-field notation in the scheduling area, we denote the problem under consideration as $Q_2|\text{online-over-list}, \text{LD}(k)|C_{\text{max}}$, where $Q_2$ means two uniform parallel machines, online-over-list denotes the online over-list mode, and $\text{LD}(k) = \sum_{j=1}^{k} p_j$ represents the initial lookahead model in which the processing times as well as the total length of the first $k$ jobs are foreseen by the online player. Especially, we focus on the scenario where $s$ is a positive integer greater than or equal to 2, i.e., $s = 2, 3, 4, \ldots$. Note that for the case with $s = 1$, it reduces to the two identical parallel machines setting and the case has been well investigated in Zheng et al. [42]. With regard to the value of $\text{LD}(k)$, if $\text{LD}(k) < (s+1)\Delta$, it contains the traditional online case without initial lookahead, i.e., $\text{LD}(k) = 0$, and the previous results apply to this case [7, 18]. Therefore, in the remainder of this work we focus on the case where $\text{LD}(k) \geq (s+1)\Delta$. We first prove a lower bound of competitive ratio, and then propose a deterministic semi-online algorithm.

4. A LOWER BOUND

Cho and Sahni [7] prove a lower bound of 1.618 for the two uniform parallel machines scheduling problem without the ability of lookahead. The following theorem shows that given a certain initial lookahead ability, the lower bound of competitive ratio can be improved to be $\frac{s^2+s+1}{s^2+s}$ (where $\frac{s^2+s+1}{s^2+s} \leq 7/6 < 1.618$). It implies that deterministic online algorithms may improve their competitiveness with the initial lookahead ability. Besides, the larger the $s$ is, the smaller the lower bound can be.

**Theorem 4.1.** For problem $Q_2|\text{online-over-list}, \text{LD}(k) \geq (s+1)\Delta|C_{\text{max}}$ where $s \geq 2$, any deterministic semi-online algorithm cannot have a competitive ratio less than $\frac{s^2+s+1}{s^2+s}$.

**Proof.** We construct a job input instance $\sigma = (J_1, J_2, \ldots, J_n)$ with at least $s+3$ jobs, and show that any deterministic algorithm $A$ produces a processing schedule with its objective value at least $\frac{s^2+s+1}{s^2+s}$ times the optimal one. Denote by $C_A(\sigma)$ and $C_{\text{OPT}}(\sigma)$ the objective values of the processing schedules produced by LS and OPT, respectively, for the sequence $\sigma$.

The first $s+2$ jobs are of lengths $p_j = \Delta$ for $1 \leq j \leq s$ and $p_{s+1} = p_{s+2} = (s+1)\Delta/(2s)$. As $\text{LD}(k) \geq (s+1)\Delta$, $\sum_{j=1}^{s+1} p_j = \sum_{j=1}^{s+2} p_j = \sum_{j=1}^{s+2} p_j = \frac{s+1}{s+2} \Delta < (s+1)\Delta$ and $\sum_{j=1}^{s+2} p_j = \sum_{j=1}^{s+2} p_j = \sum_{j=1}^{s+2} p_j = \sum_{j=1}^{s+2} p_j = \sum_{j=1}^{s+2} p_j = \frac{s+1}{s+2} \Delta > (s+1)\Delta$, we set $k = s+2$ in this instance. That is, $A$ observes the first $s+2$ jobs at the beginning. Consider the following three cases.

**Case 1.** $A$ assigns at least two jobs to the slow machine $M_1$. No more jobs arrive in this case. $C_A(\sigma) \geq p_{s+1} + p_{s+2} = (1+1/s)\Delta$. For OPT, it assigns one of the longest jobs to machine $M_1$ and the rest $s+1$ jobs to the other machine. $C_{\text{OPT}}(\sigma) = [(s-1)\Delta+(s+1)\Delta/s]/s = (1+1/s^2)\Delta$. The ratio $\frac{C_A(\sigma)}{C_{\text{OPT}}(\sigma)} = \frac{(1+1/s)\Delta}{(1+1/s^2)\Delta} = \frac{s^2+s}{s^2+1}$.

**Case 2.** $A$ assigns all the jobs to the fast machine $M_2$. Again no more jobs arrive. $C_A(\sigma) = \sum_{j=1}^{s+2} p_j/s = [s+(s+1)/s^2] \Delta/s = (1+(s+1)/s^2)\Delta$, while $C_{\text{OPT}}(\sigma) = (1+1/s^2)\Delta$ as in Case 1. $\frac{C_A(\sigma)}{C_{\text{OPT}}(\sigma)} = \frac{(1+(s+1)/s^2)}{(1+1/s^2)\Delta} = \frac{s^2+s+1}{s^2+1}$ in this case.

**Case 3.** $A$ assigns only one job with length either $\Delta$ or $(s+1)\Delta/(2s)$ to the slow machine $M_1$. In this case the last job $J_{s+3}$ with length $p_{s+3} = \Delta$ arrives. $A$ at best assigns the last job to machine $M_2$ and $C_A(\sigma) \geq \sum_{j=2}^{s+2} p_j/s = \sum_{j=2}^{s+2} p_j/s = (1+(s+1)/s^2)\Delta$. OPT will assign jobs $J_{s+1}, J_{s+2}$ to $M_1$ and the rest $s+1$ jobs with uniform length of $\Delta$ to $M_2$, and $C_{\text{OPT}}(\sigma) = [(s+1)/s] \Delta$. $\frac{C_A(\sigma)}{C_{\text{OPT}}(\sigma)} = \frac{1+(s+1)/s^2}{(s+1)/s} = \frac{s^2+s+1}{s^2+1}$ in this case.

As $\frac{s^2+s+1}{s^2+1} < \frac{s^2+s}{s^2+1} < \frac{s^2+s+1}{s^2+1}$ due to $s > 1$, we have $\frac{C_A(\sigma)}{C_{\text{OPT}}(\sigma)} \geq \frac{s^2+s+1}{s^2+1}$ for all the above cases. The theorem is proved. □
According to the formula of lower bound in the above theorem, its value decreases with the processing speed $s$ of fast machine and approaches 1 as $s$ goes to infinity.

5. A semi-online algorithm and its competitive analysis

In this section we mainly present a deterministic semi-online algorithm and prove its competitiveness. Since any semi-online algorithm has the full knowledge of the first $k$ released jobs at the beginning, the jobs can be assigned to the two machines the same as the offline local optimal solution, following the constraint that their processing sequence is in accordance with their release sequence. Therefore, to any semi-online algorithm, the release sequence of the first $k$ jobs does not impact the workloads of the two machines for the jobs. For notational convenience, we assume without loss of generality that in any job instance $\sigma = (J_1, J_2, \ldots, J_n)$, the first $k$ jobs $J_1, J_2, \ldots, J_k$ are in the non-increasing order of processing time, i.e., $p_j \geq p_{j+1}$ for $1 \leq j < k$.

Let $J_1$ be the job such that $\frac{T_1}{s} > T_2$ while $\frac{T_1 - p_k}{s} \leq T_2 + p_l$ where $T_1 = \sum_{j=1}^{k} p_j$ and $T_2 = \sum_{j=1}^{l} p_j$. We first consider a special case where $T_2 = 0$. In this case, $l = k$ and we claim that $T_1 = (s + 1)\Delta$. Otherwise if $T_2 > (s + 1)\Delta$, then $T_1 - p_k > s\Delta$ due to $p_k \leq \Delta$, and then $\frac{T_1 - p_k}{s} > \Delta$ due to the inequality $\frac{T_1 - p_k}{s} \leq T_2 + p_l$. A contradiction to the inequality $\frac{T_1 - p_k}{s} \leq T_2 + p_l$ where $l = k$.

According to the conclusion $T_1 = (s + 1)\Delta$, we further claim that $k = s + 1$ when $T_2 = 0$. Notice that it is not the case $k < s + 1$ since $T_1 = \sum_{j=1}^{k} p_j = (s + 1)\Delta$ where $p_j \leq \Delta$. If otherwise $k > s + 1$, then $p_k < \Delta$ due to the fact that $J_k$ is a smallest job among the first $k$ jobs and $T_1 = (s + 1)\Delta$. If follows $T_1 - p_k > s\Delta$ and again $\frac{T_1 - p_k}{s} > \Delta$ due to $p_k = T_2 + p_l$. A contradiction to the inequality $\frac{T_1 - p_k}{s} \leq T_2 + p_l$ again. Combining $T_1 = (s + 1)\Delta$ and $k = s + 1$, it follows $p_j = \Delta (1 \leq j \leq k)$ for the special case $T_2 = 0$.

Below we propose a semi-online algorithm BAL (for Balance), which tries to make a balanced workload assignment for the first $k$ jobs, and then assigns the rest jobs as the classical algorithm LS does. The details of the algorithm is as follows.

Algorithm BAL

**Step 1.** For the first $k$ jobs, if $0 \leq T_2 \leq p_1$, assign the longest job $J_1$ to slow machine $M_1$ and the rest jobs to fast machine $M_2$. Otherwise if $p_1 < T_2$, assign the smallest $k - l$ jobs $J_{l+1}, \ldots, J_k$ with a total length of $T_2$ to $M_1$ and the first $l$ jobs $J_1, \ldots, J_l$ with a total length of $T_1$ to $M_2$.

**Step 2.** For any other job $J_j$ ($j > k$), if any, assign it to the machine which is of a smaller current workload after the assignment of $J_j$. Ties are broken by assigning the job to slow machine $M_1$.

**Theorem 5.1.** For problem $Q_2|\text{online-over-list}, \text{LD}(k) \geq (s + 1)\Delta|C_{\text{max}}$ where $s \geq 2$, BAL has a competitive ratio of $\frac{(s + 1)^2}{2s + 1}$.

**Proof.** Given any job instance $\sigma = (J_1, J_2, \ldots, J_n)$. Let $C_j$ be the completion time of job $J_j$, and $C_1(j), C_2(j)$ be the workloads of machines $M_1$ and $M_2$, respectively, on the assignment of job $J_j$ ($1 \leq j \leq n$) in the processing schedule of algorithm BAL. Let $C_{\text{BAL}}(\sigma)$ be the makespan of the processing schedule produced by BAL and $C_{\text{OPT}}(\sigma)$ that of the optimal schedule by OPT.

**Case 1.** $n = k$, and then $p_1 = \Delta$. There are two cases by the value of $T_2$. Notice that if $T_2 = 0$, then we have $T_1 = (s + 1)\Delta, k = s + 1$ and $p_j = \Delta$ for $1 \leq j \leq k$ by previous argument. Thus $C_1(k) = C_2(k) = \Delta$ by assigning only $J_1$ to $M_1$ in the BAL schedule, and $C_{\text{BAL}}(\sigma)/C_{\text{OPT}}(\sigma) = 1$.

**Case 1.1.** $0 \leq T_2 \leq p_1$. In this case, BAL assigns the longest job $J_1$ to machine $M_1$ and assigns the rest jobs to $M_2$. Machines $M_1$ and $M_2$ are of workloads equal to $p_1$ and $T_1 + T_2 - p_1$, respectively. Since $T_1 + T_2 \geq (s + 1)\Delta$, we have $p_1 = \Delta \leq (T_1 + T_2 - p_1)/s$, and thus $C_{\text{BAL}}(\sigma) = (T_1 + T_2 - p_1)/s$. For OPT, $C_{\text{OPT}}(\sigma) \geq (T_1 + T_2)/(s + 1)$. Combining $\frac{T_1 - p_k}{s} \leq T_2 + p_l$ and $p_l \leq p_1$, we have $T_1 \leq (T_2 + p_1)s + p_1 \leq (T_2 + p_1)s + p_1$. By the case condition $T_2 \leq p_1$, $T_1 + T_2 \leq (T_2 + p_1)s + p_1 + T_2 \leq 2(s + 1)p_1$. Therefore,

$$\frac{C_{\text{BAL}}(\sigma)}{C_{\text{OPT}}(\sigma)} \leq \frac{(T_1 + T_2 - p_1)/s}{(T_1 + T_2)/(s + 1)}$$
Case 2. $p_1 < T_2$. In this case, BAL assigns the last $k - l$ jobs $J_{l+1}, \ldots, J_k$ with total length of $T_2$ to machine $M_1$ and assigns the first $l$ jobs $J_1, \ldots, J_l$ with total length of $T_1$ to machine $M_2$. As in Case 1, we have by $\frac{T_1 - p_1}{s} \leq T_2 + p_1$ and $p_1 \leq p_1$ that $T_1 \leq (T_2 + p_1)s + p_1$. Since $T_1/s > T_2$, we have $C_{BAL}(\sigma) = T_1/s$. For OPT, again $C_{OPT}(\sigma) \geq (T_1 + T_2)/(s + 1)$.

\[
\frac{C_{BAL}(\sigma)}{C_{OPT}(\sigma)} \leq \frac{T_1/s}{(T_1 + T_2)/(s + 1)} \leq \frac{(T_2 + p_1)s + p_1}{(T_2 + p_1)s + p_1 + T_2} \cdot \frac{s + 1}{s} = \frac{sT_2 + (s + 1)p_1}{sT_2 + sp_1} < \frac{2s + 1}{2s}
\]

where the second inequality is by $T_1 \leq (T_2 + p_1)s + p_1$, and the third inequality is by the case condition $p_1 < T_2$.

Case 2. $n > k$. We first bound the lower bounds of $C_1(k)$ and $C_2(k)$. By the analysis of Case 1, if $T_2 \leq p_1$ (i.e., it is in Case 1.1), then $C_1(k) = p_1$ and $C_2(k) = (T_1 + T_2 - p_1)/s \geq ((s + 1)\Delta - p_1)/s \geq \Delta$. Otherwise if $T_2 > p_1$ (i.e., it is in Case 1.2), then $C_1(k) = T_2 > p_1$ and $C_2(k) = T_1/s > T_2$. As $C_1(k) + C_2(k) = T_1 + T_2 \geq (s + 1)\Delta$, together with $C_1(k) = T_2, C_2(k) = T_1/s$, we claim that $C_2(k) \geq \Delta$ in the latter case. Assume otherwise that $C_2(k) < \Delta$, then $T_1 = C_2(k) \cdot s < s\Delta$. If $C_1(k) = T_2 \leq \Delta$, then $T_1 + T_2 < (s + 1)\Delta$, a contradiction to the fact that $T_1 + T_2 \geq (s + 1)\Delta$; otherwise if $C_1(k) = T_2 > \Delta$, then it contradicts to the fact that $T_1/s > T_2$. Therefore, $C_1(k) \geq p_1$ and $C_2(k) \geq \Delta$ no matter $T_2 \leq p_1$ or $T_2 > p_1$.

Consider the assignment of the last job $J_n$ in $\sigma$. Assume without loss of generality that the completion time of job $J_n$ is equal to the makespan of the processing schedule. Otherwise if $C_{BAL}(\sigma) > C_n$, then deleting $J_n$ does not reduce the value of $C_{BAL}(\sigma)$ while the value of $C_{OPT}(\sigma)$ can be decreased. It implies an increase of the ratio of $C_{BAL}(\sigma)/C_{OPT}(\sigma)$.

Case 2.1. $J_n$ is assigned to $M_1$. It implies that $C_1(n - 1) + p_n \leq C_2(n - 1) + p_n/s$. In this case, $C_{BAL}(\sigma) = C_1(n - 1) + p_n$ by the previous argument. For OPT, $C_{OPT}(\sigma) \geq (C_1(n - 1) + sC_2(n - 1) + p_n)/(s + 1)$.

\[
\frac{C_{BAL}(\sigma)}{C_{OPT}(\sigma)} \leq \frac{C_1(n - 1) + p_n}{(C_1(n - 1) + sC_2(n - 1) + p_n)/(s + 1)} \leq \frac{(C_2(n - 1) + p_n/s + sC_2(n - 1))/(s + 1)}{(C_2(n - 1) + p_n/s + sC_2(n - 1))/(s + 1)} = \frac{(s + 1)^2}{s^2 + s + 1}
\]

where the second inequality is by $C_1(n - 1) + p_n \leq C_2(n - 1) + p_n/s$, and the last inequality is due to $C_2(n - 1) \geq C_2(k) \geq \Delta$ and $p_n \leq \Delta$.

Case 2.2. $J_n$ is assigned to $M_2$, implying that $C_1(n - 1) + p_n > C_2(n - 1) + p_n/s$. In this case, $C_{BAL}(\sigma) = C_2(n - 1) + p_n/s$. Again $C_{OPT}(\sigma) \geq (C_1(n - 1) + sC_2(n - 1) + p_n)/(s + 1)$.

\[
\frac{C_{BAL}(\sigma)}{C_{OPT}(\sigma)} \leq \frac{C_2(n - 1) + p_n/s}{(C_1(n - 1) + sC_2(n - 1) + p_n)/(s + 1)}
\]
where the second inequality is by $C_1(n - 1) + p_n > C_2(n - 1) + p_n / s$, and the last inequality is by $C_2(n - 1) \geq C_2(k) \geq \Delta$ and $p_n \leq \Delta$.

As $\frac{2s+1}{2s} < \frac{(s+1)^2}{s^2+s+1}$ by $s \geq 2$, $\frac{C_{\text{BAL}}(\sigma)}{C_{\text{OPT}}(\sigma)} \leq \frac{(s+1)^2}{s^2+s+1}$ holds for any case. The proof of the theorem is established. □

Combining Theorems 4.1 and 5.1, we observe that there is a gap of $\frac{s}{s^2+s+1} - \frac{1}{2s}$ between the upper bound and lower bound for the problem $Q_2|\text{online-over-list}, \text{LD}(k)|C_{\text{max}}$. The maximum gap is equal to 0.1474 when $s = 3$, and the gap gradually reduces to 0.0451 when $s = 20$. Figure 1 illustrates the trends of the upper and lower bounds with the increase of $s$. Furthermore, the competitive ratio of BAL is at most $9/7 \approx 1.286$, which is much smaller than the ratio of 1.618 in Cho and Sahni [7] for the online model without lookahead. It indicates that an appropriate initial lookahead can really help enhance the competitive performance of online algorithms.

6. Conclusion

In this work we introduce a new concept of initial lookahead in the semi-online scheduling on two uniform parallel machines. Different from the traditional semi-online model with lookahead, semi-online algorithms with initial lookahead only foresee the processing times of the first $k$ jobs at the beginning. The rest jobs are released in the online over-list fashion and observed on their arrivals. We focus on the case where the total processing time of the first $k$ jobs is at least $(s+1)\Delta$ where $s \geq 2$ is the processing speed of the fast machine and the $\Delta$ is the largest processing time of job. A lower bound of $\frac{s^2+s+1}{s^2+s+1}$ is provided, and then a deterministic semi-online algorithm is proposed and proved $\frac{(s+1)^2}{s^2+s+1}$-competitive. The competitive ratio obtained in this work is noticeably smaller than the previous result in the scenario without lookahead ability.

There are some meaningful future research directions. First, as we focus on the scenario where the speed of the fast machine $s$ is an integer, it is interesting to exploit the competitiveness of online algorithms for the case where $s$ is continuous (i.e., $s > 1$). Second, it is meaningful to investigate more general situation where
the initial lookahead ability is less than \((s + 1)\Delta\), especially to determine the least total processing time of the initially foreseen jobs that can help improve the competitiveness of online algorithms. Third, it is worthy to investigate the corresponding scenario where jobs are of GoS and present competitive algorithms. Fourthly, it is expected that the analysis and conclusion of this work can be extended to the case with \(m \geq 2\) uniform parallel machines with initial lookahead.

**Acknowledgements**

This work has been partially supported by the National Natural Science Foundation of China (72271051, 71832001, 72071144) and the Fundamental Research Funds for the Central Universities (2232018H-07).

**References**


