Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 7527862, 9 pages
https://doi.org/10.1155/2020/7527862

Research Article

A Competitive Online Algorithm for Minimizing Total Weighted Completion Time on Uniform Machines

Xuyang Chu and Jiping Tao

School of Aerospace Engineering, Xiamen University, Xiamen 361005, China

Correspondence should be addressed to Jiping Tao; jipingtao@gmail.com

Received 29 September 2019; Revised 7 March 2020; Accepted 18 March 2020; Published 14 April 2020

Academic Editor: Francesco Soldovieri

We consider the classic online scheduling problem on m uniform machines in the online setting where jobs arrive over time. Preemption is not allowed. The objective is to minimize total weighted completion time. An online algorithm based on the directly waiting strategy is proposed. Its competitive performance is proved to be $\max\{2s_{\max}(1-(1/(2\sum s_i))), (2s_{\max}/(1+s_{\max}))(2.5-(1/2m))\}$ by the idea of instance reduction, where s_m is the fastest machine speed after being normalized by the slowest machine speed.

1. Introduction

Scheduling is a commonly used term in several different contexts including production management, service operations, and computer systems. In a general sense, scheduling is a process of allocating some scarce resources to different activities. It aims to improve the effectiveness and/or efficiency to implement these activities. It is for this fact that scheduling has attracted numerous interests of both practitioners and researchers. In academia, thousands of papers have been published with scheduling as topics. These works can be roughly classified into two categories, namely, application-oriented research and theoretical research.

From a perspective of applications, researchers try to provide a solution method for a given scheduling problem by utilizing mathematical or computation techniques. The effectiveness and efficiency of the proposed approach are often judged by its performance on randomly generated or benchmark instances or some case data. Along this line, a rich variety of scheduling problems have been investigated, motivated by different applications from real-world situations. Among these problems, a great many are focused on scheduling arising in industries of production and manufacturing. In this field, most of the work can be divided into four types in terms of solution methodology, which are analytical approaches (e.g., branch and bound [1], Lagrangian relaxation [2], and column generation [3]), heuristic dispatch rules [4], metaheuristics (e.g., simulated annealing [5] and genetic algorithms [6]), and machine learning methods (e.g., neural network [7], decision tree [8], and ensemble methods [9]). The detailed discussion of these methods is beyond the scope of this paper. The abovementioned references are just some of the landmark or representative papers in each field. Interested readers can further refer to some classical books (e.g., Brucker [10] and Pinedo [11]). In addition to the abovementioned work about shop-level scheduling methods, some attention has been paid on scheduling-related problems from the perspective of planning and management. For example, Sastoque Pinilla et al. [12] conducted a case study about technology readiness level (TRL) 5C7 project management in the aircraft industry. In addition, with the new requirements and challenges occurring in the context of Industry 4.0, some new technologies and ideas such as Internet of things, cyber-physical systems, and big data analysis have been introduced into the scheduling research [13–15].

In contrast, from a theoretical perspective, researchers first try to understand the computational complexity of a given scheduling problem. If the problem is computationally tractable, they then develop a polynomial algorithm that can give optimal solutions in polynomial time for any instance of the problem. If the problem is computationally intractable, they design an algorithm and evaluate it by its theoretically guaranteed performance rather than its performance in numeric simulations. The theoretically guaranteed performance often refers to the worst-case performance. Along this line, most of the work is focused on some classical and simplified scheduling models that are abstracted from similar applications, which can be often denoted by the widely used three-field classification scheme [16]. These models can be further divided into two categories in terms of whether the full information about jobs is known at the outset or not, namely, off-line models and online models. For the former category, some pioneering works can be found in [17] and [18]. For the online category, some representative works include Hoogeveen and Vestjens [19], Anderson and Potts [20], and Correa and Wagner [21]. Interested readers can further refer to a review by Potts and Strusevich [22]. Following this line of theoretical research, we consider a classical online scheduling problem in this work.

1.1. Problem Formulation. Formally, there is a sequence of jobs denoted by $J_1, J_2, \ldots, J_n$, which must be scheduled on one of m uniform machines without interruption. Here, uniform means that each machine has a job-independent speed. Each job J_j is characterized by a release time r_j, a processing requirement p_j, and a positive weight w_j. When a job is assigned on a machine, the processing time is its processing requirement divided by the machine speed. Without loss of the generality, we assume that the machines are ordered by a nondecreasing speed $1 = s_1 \leq s_2 \leq \cdots \leq s_m$. The objective is to minimize the total weighted completion time of all the jobs. We focus our attention on the online setting, which means that all the characteristics about one job is not revealed to the scheduling algorithm until it is released at its release time. The problem can be denoted by $Q_m|r_j, \text{online}|\sum w_j C_j$.

1.2. Related Work. In the online setting, parallel machine scheduling problem has obtained much attention. For the case of identical machines, many efficient online algorithms were presented [21, 23, 24]. When it comes to the setting of uniform machines, i.e., each machine has a different and job-independent speed, most of the work was focused on the objective of minimizing the makespan [25–27]. For the objective of minimizing the total completion time, to the best of our knowledge, the only constant competitive algorithm was the one presented by Hall et al. [18]. They designed a $8 + \varepsilon$-competitive algorithm. The result had remained unchanged for more than ten years until Liu et al. [28] extended the result about a two-machine scheduling problem [29] and proposed an online algorithm with a competitive performance of $(\sqrt{4m-3} + 3)/2$. It performs better than the 8-competitive algorithm when the machine number is not too larger. However, the competitive ratio tends to infinity with m increasing. As mentioned by Hoogeveen and Vestjens [19], for the single-machine case, any online algorithm that schedules a job as soon as the machine is available has an unbounded worst-case performance. The conclusion also holds up for the case of uniform machines. In other words, a waiting strategy is necessary in order to guarantee bounded competitive performance. Two kinds of techniques are commonly used to design waiting strategies. The first one is to shift releasing times forward. The revised releasing times are derived either by a heuristic method [23, 30, 31] or from a preemptive schedule by solving a related relaxation problem [21, 32, 33]. The other technique is to apply a directly waiting strategy, which makes the decision to insert appropriate waiting time or to immediately schedule a job by directly comparing processing time with the current time [19, 20, 34]. Hoogeveen and Vestjens [19] first developed the technique and proposed the delayed shortest processing rule (D-SPT) for the single-machine scheduling to minimize the total completion time. They proved that the D-SPT rule is the best online algorithm for $1|r_j, \text{online}|\sum C_j$. The idea behind the D-SPT rule and its competitive analysis was generalized to the weighted case by Anderson and Potts [20]. They proposed the delayed shortest weighted processing time (D-SWPT) rule and proved it is optimal for $1|r_j, \text{online}|\sum w_j C_j$. The idea of the directly waiting strategy was further extended to design online algorithms for $P_m|r_j, \text{online}|\sum w_j C_j$ by Tao [24], Tao et al. [35], and Ma and Tao [36]. The authors designed several waiting strategies and proved their competitive performance.

1.3. Our Contribution. For $Q_m|r_j, \text{online}|\sum w_j C_j$, we construct a deterministic online algorithm and prove that it has a competitive ratio of no more than $\max\{2s_{\max}(1-(1/(2\sum s_i))), (2s_{\max}/(1+s_{\max}))(2.5-(1/2m))\}$. To the best of our knowledge, this is the first online algorithm where the approximation ratio depends on the machine speeds. If the fastest machine is not too fast, say less than 4, the algorithm beats the $8 + \varepsilon$-competitive algorithm [18], which is the best constant competitive algorithm so far. In addition, the performance ratio for our algorithm decreases as the machine number increases. It is contrary to the result in [28], where the competitive ratio increases as the machine number increases. Thus, our algorithm also outperforms the one in [28] in the case with a larger number of machines.

The remaining sections are organized as follows. In Section 2, the online algorithm is presented. The competitive analysis is detailed in Section 3. Conclusions are given in Section 4.

2. The LW-SWPR Rule

Inspired by the same idea of the directly waiting strategy mentioned above in the subsection of related work, we construct an online algorithm for $Q_m \mid r_j, \text{online} \mid \sum w_j C_j$. We call the proposed algorithm the limited waiting shortest weighted processing requirement rule. It is therefore abbreviated to LW-SWPR, which can be described as follows with some notations listed in Table 1:

Table 1: Symbol/notation description.

Notation	Description
t	The current decision time
$\sigma(\cdot)$	The schedule constructed by the LW-SWPR rule for a given instance. It also refers to the objective value of the schedule when no confusion arises
S_j	The starting time of job J_j in the online schedule $\sigma(\cdot)$
C_j	The completion time of job J_j in the online schedule $\sigma(\cdot)$
$\pi(\cdot)$	The optimal schedule for a given instance. It also refers to the objective value of the schedule when no confusion arises

2.1. The LW-SWPR Rule. Whenever some idle machines and some jobs are available at time t, choose the machine with the highest speed and choose a job with the smallest value of the ratio p_j/w_j among all the arrived and unscheduled jobs. If ties occur, choose the smallest index. Say that M_i and J_j are chosen. Calculate the total processing requirement at all the busy machines at time t, which can be written as $\sum_{S_i\le t, C_i>t} p_i$. Then, if

$$\frac{p_j + \sum_{S_i\le t, C_i>t} p_i}{\sum_{i=1}^{m} s_i} \le t, \tag{1}$$

schedule J_j on the machine M_i; otherwise, wait until a new job is released or the above inequality is satisfied.

We note that the LW-SWPR rule can be reduced to the AD-SWPT rule in Tao [24] when all the machines have the same speed, i.e., the model is reduced to the identical-machine model. Next, we will show that the LW-SWPR rule has a machine number independent performance guarantee.

Theorem 1. *For the online scheduling problem of* $Q_m|r_j, online|\sum w_j C_j$, *the LW-SWPR rule is* $\max\{2s_m(1-(1/(2\sum s_i))), (2s_m/(1+s_m))(2.5-(1/2m))\}$-*competitive, where* s_m *is the fastest machine speed after being normalized by the slowest machine speed.*

3. The Competitive Analysis of the LW-SWPR Algorithm

3.1. Instance Reduction. Since it is very difficult to directly analyze the performance for an arbitrary instance, we wish to reduce the searching space for the worst-case instances. This is just the intuition behind the instance reduction method commonly used by Tao et al. [37, 38] and Tao [24]. The basic idea is to modify an arbitrary instance to a new instance such that it has a worse performance ratio as well as a more special structure of which we can take advantage to analyze the performance ratio. To simplify the presentation, we list some instances with specified structure in Table 2. We also use these notations to refer to the instance set with the specified structure when no confusion arises. For ease of presentation, we denote that one machine is "*idle*"/"*busy*" at the time t if the machine remains idle/busy during the interval of $(t-\varepsilon, t+\varepsilon)$, where ε is an infinitely small positive value. In order to differentiate the switching time points between the *busy* and *idle* states, we further state that the time t is a "*starting point of busy time*"(abbreviated to SPoint) if the machine remains idle in $(t-\varepsilon, t)$ and busy in $(t, t+\varepsilon)$ and that the time t is an "*ending point of busy time*"(EPoint) if the machine is busy in $(t-\varepsilon, t)$ and idle in $(t, t+\varepsilon)$.

First, we note that the worst-case instances can be obtained among the set of I_1. The reason follows. For any instance I which does not belong to the set of I_1, there exists a time t between the earliest SPoint and the latest EPoint when all the machines are *idle*. Thus, we can split the instance I into two smaller instances that consist of jobs scheduled, respectively, before and after t. Denote the two instances by I' and I'', respectively. According to the LW-SWPR rule, we can readily discover that $\sigma(I')$ and $\sigma(I'')$ maintain the starting times of all the jobs same as in $\sigma(I)$, i.e.,

$$\sigma(I) = \sigma(I') + \sigma(I''). \tag{2}$$

Given any feasible schedule of I, we can construct two feasible schedules for I' and I'', respectively, by keeping the starting times unchanged. Because the optimal schedule is the one with the minimum objective value among all the feasible schedules, it follows that

$$\pi(I) \ge \pi(I') + \pi(I''). \tag{3}$$

Combining (2) with (3), we can obtain

$$\frac{\sigma(I)}{\pi(I)} \le \frac{\sigma(I') + \sigma(I'')}{\pi(I') + \pi(I'')} \le \max\left\{\frac{\sigma(I')}{\pi(I')}, \frac{\sigma(I'')}{\pi(I'')}\right\}, \tag{4}$$

i.e., at least one of the two smaller instances can achieve a performance ratio not less than that of the original instance I. Therefore, from the perspective of worst-case instances, we need to focus only on the set of I_1.

Next, we show in Lemma 1 that I_1 can be further reduced to one of the two new instances I_2 or I_3 with the performance ratio not decreasing. Here, we only give some intuitive explanation while referring readers to [24] for the detailed proof. As mentioned above, jobs within each subqueue are ordered according to the WSPR rule. For I_1, we can multiply the weights of some jobs by a parameter δ with not changing the mutual relations of the weighted processing requirements among jobs. The result is that jobs are scheduled in the same time intervals as in $\sigma(I_1)$ after this modification. The fact means that the objective value of the online schedule for the modified instance is a monotonously increasing linear function with respect to δ. At the same time, the optimal objective value is a concave function with respect to δ because any feasible schedule remains feasible after modification and the optimal schedule is the one with the minimum objective value among all the feasible schedules. Combining the above observations with Lemma

Table 2: Three types of instances with specified structure.

I_1:	Any instance for which there does not exist a time t between the earliest SPoint and the latest EPoint in the online schedule by the LW-SWPR rule, such that all the machines remain *idle* at t
I_2:	An instance which not only possesses the same structure as I_1 but also satisfies that each job has the same weighted processing time
I_3:	An instance which not only possesses the same structure as I_1 but also satisfies that jobs in the last subqueue in the online schedule have the same weighted processing time with weights tending to positive infinity

3.1 in [37] which states that the ratio of a convex function to a concave function is maximized at one endpoint of the supported interval, we can modify I_1 to an intermediate instance with a worse performance ratio by setting δ to a specified value. By repeatedly applying the modification, we can reduce I_1 to I_2 or I_3.

Lemma 1 (see [24]). *For any instance I_1, a new instance I_2 or I_3 can be constructed by modifying the weights in I_1, such that*

$$\frac{\sigma(I_1)}{\pi(I_1)} \le \max\left\{\frac{\sigma(I_2)}{\pi(I_2)}, \frac{\sigma(I_3)}{\pi(I_3)}\right\}. \tag{5}$$

3.2. Performance Analysis of I_2 and I_3. Based on the special structure that I_2 and I_3 possess and the LP lower bound, we can analyze the performance ratio of I_2 and I_3. First, an appropriate lower bound on the optimal schedule has to be established in order to further analyze the performance ratios of I_2 and I_3. Because the instances I_2 and I_3 are associated with the special structure, we can only consider the lower bound on these special instances. This can be readily obtained from related results provided by Gu and Lu [39] and Chou et al. [40].

Lemma 2 (see [39, 40]). *For the $Q_m|r_j, online|\sum w_jC_j$ with $p_j = w_j$ for each job, the optimal schedule satisfies an inequality of*

$$\mu(I) \ge r_{\min}\sum_{j=1}^{n} p_j + \frac{1}{2\sum s_i}\left(\sum_{j=1}^{n} p_j\right)^2 + \frac{1}{2s_m}\sum_{j=1}^{n} p_j^2, \tag{6}$$

where $r_{\min}$ is the earliest release time.

Next, we analyze the performance ratios of I_2 and I_3 by two lemmas.

Lemma 3. *For any instance I_2, the online scheduling by the LW-SWPR rule satisfies*

$$\frac{\sigma(I_2)}{\pi(I_2)} \le \max\left\{2s_m\left(1 - \frac{1}{2\sum s_i}\right), \frac{2s_m}{1+s_m}\left(2.5 - \frac{1}{2m}\right)\right\}. \tag{7}$$

Proof. It does not change the performance ratio to multiply the weights of all the jobs by a positive constant. All the jobs in I_2 have the same weighted processing time and we can normalize the ratio of p_j/w_j to 1, i.e., w_j equals p_j.

Consider the latest SPoint in $\sigma(I_2)$, and denote it as r_L. The "*latest*" implies that jobs are continuously processed after r_L at each machine without idle time between jobs. Now, we analyze the performance ratio by two cases.

Case 1: there does not exist a job which is released before r_L and is scheduled at, or after, r_L in $\sigma(I_2)$. Consider these jobs that start at or after r_L. According to the increasing order of their starting times, denote these jobs by $J_1, J_2, \ldots, J_n$. The assumption in this case implies that these jobs must be released at, or after, r_L. Furthermore, these jobs have no effect on jobs starting before r_L. Construct an intermediate instance I_2' which includes all the other jobs in I_2 except $J_1, J_2, \ldots, J_n$. Then, we have

$$\sigma(I_2) \le \sigma(I_2') + \sum_{j=1}^{n}(S_j + p_j)w_j. \tag{8}$$

Jobs are continuously processed after r_L at each machine. So, we can limit the starting time of the j^{th} job in $\{J_1, J_2, \ldots, J_n\}$ as

$$S_j \le r_L + \frac{\sum_{S_j \le r_L, C_j > r_L} p_j + \sum_{1\le i<j} p_i}{\sum_{i=1}^{m} s_i}, \qquad i = 1, 2, \ldots, m,\ j = 1, 2, \ldots, n, \tag{9}$$

where the second term is to average the total processing time that have to be finished between r_L and S_j over all the machines and $\sum_{S_j\le r_L, C_j>r_L} p_j$ represents the total remaining processing time at all the machines at time r_L. Let $\sum_{S_j\le r_L, C_j>r_L} p_j := A$ and $\sum_{j=1}^{n} p_j := B$. Along with (7), we can limit $\sigma(I_2)$ by an upper bound as

$$\begin{aligned}
\sigma(I_2) &\le \sigma(I_2') + \sum_{j=1}^{n}(S_j + p_j)w_j \\
&\le \sigma(I_2') + \sum_{j=1}^{n}\left(r_L + \frac{A + \sum_{i=1}^{j-1} p_i}{\sum s_i} + p_j\right)p_j \\
&= \sigma(I_2') + \sum_{j=1}^{n}\left(r_L + \frac{A}{\sum s_i} + \frac{\sum_{i=1}^{j} p_i}{\sum s_i}\right)p_j + \left(1 - \frac{1}{\sum s_i}\right)\sum_{j=1}^{n} p_j^2 \\
&= \sigma(I_2') + \left(r_L + \frac{A}{\sum s_i}\right)B + \frac{B^2}{2\sum s_i} + \left(1 - \frac{1}{2\sum s_i}\right)\sum_{j=1}^{n} p_j^2.
\end{aligned} \tag{10}$$

The second term in (10) can be regarded as the total weighted completion time of the jobs of $J_1, J_2, \ldots, J_n$, which are continuously processed starting from the time $r_L + (A/\sum s_i)$ on a single machine, with the processing time of each job multiplied by a constant of $1/\sum s_i$. Consider the set of $\{J_1, J_2, \dots, J_n\}$ as a separate instance and further relax the release times of all the jobs to r_L; then, we can develop a lower bound of the optimal schedule $\pi(I_2)$ according to Lemma 2.

$$\begin{aligned}\pi(I_2) &\geq \pi(I_2') + \pi(\{J_1, J_2, \dots, J_n\}) \\ &\geq \pi(I_2') + \mathrm{LB}^m(\{J_1, J_2, \dots, J_n\}) \\ &= \pi(I_2') + r_{min}\sum_{j\in I} p_j + \frac{1}{2\sum s_i}\left(\sum_{j\in I} p_j\right)^2 + \frac{1}{2s_m}\sum_{j\in I} p_j^2.\end{aligned} \quad (11)$$

According to the LW-SWPR rule, we have $(\sum_{S_j\leq r_L, C_j>r_L} p_j/\sum s_i) \leq r_L$, i.e., $(A/\sum s_i) \leq r_L$. Combining (9) and (10), we have

$$\begin{aligned}\frac{\sigma(I_2)}{\pi(I_2)} &\leq \max\left\{\frac{\sigma(I_2')}{\pi(I_2')}, \frac{(r_L + (A/\sum s_i))B + B^2/2\sum s_i + (1-(1/2\sum s_i))\sum_{j=1}^n p_j^2}{r_L B + (B^2/2\sum s_i) + (1/2s_m)\sum_{j=1}^n p_j^2}\right\} \\ &\leq \max\left\{\frac{\sigma(I_2')}{\pi(I_2')}, 2, 2s_m\left(1 - \frac{1}{2\sum s_i}\right)\right\} \\ &\leq \max\left\{\frac{\sigma(I_2')}{\pi(I_2')}, 2s_m\left(1 - \frac{1}{2\sum s_i}\right)\right\}.\end{aligned} \quad (12)$$

Case 2: there exists at least a job J_k which is released before r_L and is scheduled at, or after, r_L in $\sigma(I_2)$. According to the LW-SWPR rule, J_k must satisfy

$$\frac{p_k + \sum_{S_j\leq r_L, C_j>r_L} p_j}{\sum s_i} \geq r_L. \quad (13)$$

Otherwise, J_k would be scheduled before r_L.

Consider the jobs that are completed after r_L. According to the increasing order of their starting times, denote these jobs by $J_1, J_2, \dots, J_n$. Construct an intermediate instance I_2' which includes all the other jobs in I_2 except $J_1, J_2, \dots, J_n$. Divide the set of $\{J_1, J_2, \dots, J_n\}$ into two subsets as follows:

$$\begin{aligned}Q_1 &= \left\{J_j \,\middle|\, S_j < r_L, C_j > r_L\right\} \cup \{J_k\}, \\ Q_2 &= \left\{J_j \,\middle|\, S_j \geq r_L\right\} \backslash \{J_k\}.\end{aligned} \quad (14)$$

Let $\sum_{J_j\in Q_1} p_j := A$ and $\sum_{J_j\in Q_2} p_j := B$.

Then, similar to the derivation of (9), we can limit $\sigma(I_2)$ as

$$\sigma(I_2) \leq \sigma(I_2') + r_L(A+B) + \frac{(A+B)^2}{2\sum s_i} + \left(1 - \frac{1}{2\sum s_i}\right)\sum_{J_j\in Q_1\cup Q_2} p_j^2. \quad (15)$$

By relaxing the releasing times of jobs in $\{J_1, J_2, \dots, J_n\}$ to 0, similar to the analysis of (10), we can limit $\pi(I_2)$ as

$$\pi(I_2) \geq \pi(I_2') + \frac{(A+B)^2}{2\sum s_i} + \frac{1}{2s_m}\sum_{J_j\in Q_1\cup Q_2} p_j^2. \quad (16)$$

We can derive $(A/\sum s_i) \geq r_L$ from (12). Furthermore, we can obtain $\sum_{J_j\in Q_1} p_j^2 \geq A^2/m$ because there are at most m jobs in Q_1. In addition, we have the inequality of $\sum_{i=1}^m s_i \geq m$. Combining these relations with (13) and (15), we can limit the performance ratio of I_2 as

$$\begin{aligned}
\frac{\sigma(I_2)}{\pi(I_2)} &\le \max\left\{\frac{\sigma(I_2')}{\pi(I_2')}, \frac{r_L(A+B)+\left((A+B)^2/2\sum s_i\right)+\left(1-\left(1/2\sum s_i\right)\right)\sum_{J_j\in Q_1\cup Q_2}p_j^2}{\left((A+B)^2/2\sum s_i\right)+(1/2s_m)\sum_{J_j\in Q_1\cup Q_2}p_j^2}\right\}\\
&\le \max\left\{\frac{\sigma(I_2')}{\pi(I_2')}, \frac{\left(A(A+B)/\sum s_i\right)+\left((A+B)^2/2\sum s_i\right)+\left(1-\left(1/2\sum s_i\right)\right)\sum_{J_j\in Q_1\cup Q_2}p_j^2}{\left(A^2/2\sum s_i\right)+(1/2s_m)\sum_{J_j\in Q_1\cup Q_2}p_j^2}\right\}\\
&\le \max\left\{\frac{\sigma(I_2')}{\pi(I_2')}, 2s_m+\frac{\left(((3/2)-s_m)/\sum s_i\right)A^2-\left(1/2\sum s_i\right)\sum_{J_j\in Q_1\cup Q_2}p_j^2}{\left(A^2/2\sum s_i\right)+(1/2s_m)\sum_{J_j\in Q_1\cup Q_2}p_j^2}\right\} \qquad (17)\\
&\le \max\left\{\frac{\sigma(I_2')}{\pi(I_2')}, 2s_m+\frac{\left(((3/2)-s_m)/\sum s_i\right)A^2-\left(1/2\sum s_i\right)\left(A^2/m\right)}{\left(A^2/2\sum s_i\right)+(1/2s_m)\left(A^2/\sum s_i\right)}\right\}\\
&= \max\left\{\frac{\sigma(I_2')}{\pi(I_2')}, \frac{2s_m}{s_m+1}\left(2.5-\frac{1}{2m}\right)\right\}.
\end{aligned}$$

The above two cases show that we can bound the performance ratio of I_2 from above equation by $2s_m(1-(1/2\sum s_i))$ or $(2s_m/(s_m+1))(2.5-(1/2m))$ or the performance ratio of an intermediate instance I_2'. Furthermore, in $\sigma(I_2')$, r_L is not the latest SPoint anymore. Rewrite I_2' as I_2 and repeat the above analysis until I_2' becomes an empty set. Ultimately the performance ratio of I_2 can be bounded from the above equation by the larger value in $2s_m(1-(1/2\sum s_i))$ and $(2s_m/(s_m+1))(2.5-(1/2m))$. □

Lemma 4. *For any instance I_3, the online schedule by the LW-SWPR rule satisfies*

$$\frac{\sigma(I_3)}{\pi(I_3)} \le \max\left\{2s_m\left(1-\frac{1}{2\sum s_i}\right), \frac{2s_m}{1+s_m}\left(2.5-\frac{1}{2m}\right)\right\}. \qquad (18)$$

Proof. Jobs in the last subqueue of I_3 have the same weighted processing time with weights tending to infinity. Without loss of generality, let $w_j=\delta p_j$ for these jobs with δ tending to infinity. In the following calculation of performance ratios, they are all carried out in the sense of limit when δ tends to infinity, with the sign of limit omitted.

Denote by Q_∞ the set including all the jobs in the last subqueue of I_3. Denote by r_f the earliest releasing time of jobs in Q_∞, and by r_L the latest SPoint in I_3. Next, we analyze the performance ratio of I_3 by the following three cases:

Case 1: $r_L \le r_f$. Considering the time r_f, according to the LW-SWPR rule, we have

$$\frac{\sum_{S_j\le r_f, C_j>r_f}p_j}{\sum s_i} \le r_f. \qquad (19)$$

After the jobs which start before r_f are completed, the jobs in Q_∞ are continuously processed. Assume that the jobs in Q_∞ start being processed in the order of $J_1, J_2, \dots, J_n$. Let $\sum_{J_j\in Q_\infty}p_j := B$. Similar to the analysis in Case 1 in the proof of Lemma 3.5, along with (19), we can derive an upper bound of $\sigma(I_3)$ as

$$\begin{aligned}
\sigma(I_3) &= \sigma(I_3') + \sum_{j=1}^{n}(S_j+p_j)w_j\\
&\le \sigma(I_3') + \sum_{j=1}^{n}\left(r_f+\frac{\sum_{S_j\le r_f, C_j>r_f}p_j+\sum_{i=1}^{j-1}p_i}{\sum s_i}+p_j\right)\delta p_j\\
&\le \sigma(I_3') + \delta\left(2r_fB+\frac{B^2}{2\sum s_i}+\left(1-\frac{1}{2\sum s_i}\right)\sum_{j=1}^{n}p_j^2\right),
\end{aligned} \qquad (20)$$

where $\pi(I_3')$ indicates a limited value. By relaxing the releasing times of jobs in Q_∞ to r_f, we can similarly derive a lower bound of $\pi(I_3)$ as

$$\pi(I_3) \ge \pi(I_3') + \delta\left(r_fB+\frac{B^2}{2m}+\frac{1}{2s_m}\sum_{j=1}^{n}p_j^2\right). \qquad (21)$$

When δ tends to infinity, the relations above immediately imply

$$\frac{\sigma(I_3)}{\pi(I_3)} \le \max\left\{\frac{\sigma(I_3')}{\pi(I_3')}, 2s_m\left(1-\frac{1}{2\sum s_i}\right)\right\}. \qquad (22)$$

Case 2: $r_L > r_f$. Furthermore, jobs being processed at r_L in $\sigma(I_3)$ all belong to Q_∞. Similar to the proof of Lemma 3, we can construct an intermediate instance I_3'

by deleting some jobs from I_3 such that r_L is not the latest SPoint in $\sigma(I_3')$ anymore. Furthermore, it holds that

$$\frac{\sigma(I_3)}{\pi(I_3)} \leq \max\left\{\frac{\sigma(I_3')}{\pi(I_3')}, \frac{2s_m}{s_m+1}\left(2.5 - \frac{1}{2m}\right)\right\}. \tag{23}$$

Case 3: $r_L > r_f$. Furthermore, there are one or more jobs which do not belong to Q_∞ and are being processed at r_L in $\sigma(I_3)$. According to the improved AD-SWPT rule, these jobs must start before r_f. We further analyze the performance ratio in terms of two subcases.

Case 3.1: there does not exist a job in Q_∞ which is released before r_L and is scheduled at, or after, r_L in $\sigma(I_3)$. This case implies that jobs starting at, or after, r_L are all released at, or after, r_L. Except these jobs, we can construct an intermediate instance I_3', which includes all the other jobs in I_3. Let $\sum_{S_j \geq r_L} p_j = B$. Similar to the analysis in Case 1 in the proof of Lemma 3, we can derive

$$\begin{aligned}\frac{\sigma(I_3)}{\pi(I_3)} &\leq \frac{\sigma(I_3') + \delta\left(\left(r_L + \left(\sum_{S_j<r_L} p_j / \sum s_i\right)\right)B + \left(B^2/2\sum s_i\right) + \left(1 - \left(1/2\sum s_i\right)\right)\sum_{S_j \geq r_L} p_j^2\right)}{\pi(I_3') + \delta\left(r_L B + \left(B^2/2\sum s_i\right) + \left(1/2s_m\right)\sum_{S_j\geq r_L} p_j^2\right)} \\ &\leq \max\left\{\frac{\sigma(I_3')}{\pi(I_3')}, 2s_m\left(1 - \frac{1}{2\sum s_i}\right)\right\}.\end{aligned} \tag{24}$$

Case 3.2: there exists at least a job J_k in Q_∞ which is released before r_L and is scheduled at, or after, r_L in $\sigma(I_3)$. First, consider these jobs which do not belong to Q_∞ and are being processed at r_L in $\sigma(I_3)$. Denote the set including these jobs by Q'. Let $\sum_{j\in Q', S_j \leq r_L, C_j > r_L} p_j: = A'$. According to the LW-SWPR rule, these jobs must start being processed before r_f. It follows that

$$\frac{A'}{\sum s_i} \leq r_f. \tag{25}$$

Because J_k is released before r_L and is scheduled at, or after, r_L in $\sigma(I_3)$, we have

$$\frac{p_k + \sum_{S_j \leq r_L, C_j > r_L} p_j}{\sum s_i} \geq r_L. \tag{26}$$

Consider the jobs in Q_∞ which are completed after r_L. Define two sets as follows:

$$\begin{aligned} Q_1 &= \left\{J_j \in Q_\infty \,\middle|\, S_j < r_L, C_j > r_L\right\} \cup \{J_k\}, \\ Q_2 &= \left\{J_j \in Q_\infty \,\middle|\, S_j \geq r_L\right\} \backslash \{J_k\}. \end{aligned} \tag{27}$$

Construct an intermediate instance I_3', which includes all the jobs in I_3 except jobs in Q_1 and Q_2. Let $\sum_{J_j \in Q_1} := A$ and $\sum_{J_j\in Q_2} := B$. Similar to the analysis in Case 2 in the proof of Lemma 3, considering that jobs in Q_1 and Q_2 can be continuously processed after jobs in Q' are completed, we can derive an upper bound on $\sigma(I_3)$ as

$$\begin{aligned}\sigma(I_3) \leq \sigma(I_3') + \delta\Bigg(\left(r_L + \frac{A'}{\sum s_i}\right)(A+B) \\ + \frac{(A+B)^2}{2\sum s_i} + \left(1 - \frac{1}{2\sum s_i}\right)\sum_{J_j \in Q_1 \cup Q_2} p_j^2\Bigg).\end{aligned} \tag{28}$$

By relaxing the releasing times of jobs in Q_1 and Q_2 to r_f, we can also derive a lower bound on $\pi(I_3)$ as

$$\pi(I_3) \geq \pi(I_3') + \delta\left(r_f(A+B) + \frac{(A+B)^2}{2\sum s_i} + \sum_{J_j\in Q_1\cup Q_2} \frac{p_j^2}{2}\right). \tag{29}$$

Equation (26) implies that $(A+A')/\sum s_i \geq r_L$. Furthermore, $\sum_{J_j\in Q_1} p_j^2 \geq A^2/\sum s_i$ because there are at most m jobs in Q_1. Combining these relations with (24), (26), and (28), we have

$$\frac{\sigma(I_3)}{\pi(I_3)} \leq \max\left\{\frac{\sigma(I_3')}{\pi(I_3')}, 2, \frac{\left(A/\sum s_i\right)(A+B) + \left(\left(A+B\right)^2/2\sum s_i\right) + \left(1 - \left(1/2\sum s_i\right)\right)\sum_{J_j\in Q_1\cup Q_2} p_j^2}{\left(\left(A+B\right)^2/2\sum s_i\right) + \left(1/2s_m\right)\sum_{J_j\in Q_1\cup Q_2} p_j^2}\right\}$$

$$
\begin{aligned}
&\le \max\left\{\frac{\sigma(I_3')}{\pi(I_3')}, 2, 2s_m + \frac{(((3/2)-s_m)/\sum s_i)A^2 - (1/2\sum s_i)\sum_{J_j\in Q_1\cup Q_2} p_j^2}{(A^2/2\sum s_i) + (1/2s_m)\sum_{J_j\in Q_1\cup Q_2} p_j^2}\right\}\\
&\le \max\left\{\frac{\sigma(I_3')}{\pi(I_3')}, 2, 2s_m + \frac{(((3/2)-s_m)/\sum s_i)A^2 - (1/2\sum s_i)(A^2/m)}{(A^2/2\sum s_i) + (1/2s_m)(A^2/\sum s_i)}\right\}\\
&\le \max\left\{\frac{\sigma(I_3')}{\pi(I_3')}, \frac{2s_m}{s_m+1}\left(2.5 - \frac{1}{2m}\right)\right\}.
\end{aligned}
\tag{30}
$$

The first inequality is derived by applying $(A + A')/\sum s_i \ge r_L$ and $A'/\sum s_i \le r_f$. The second inequality is obtained by relaxing Q_2 to an empty set and then applying $\sum_{J_j\in Q_1} p_j^2 \ge A^2/m$.

The above three cases show that we can bound the performance ratio from the above equation by the maximal value among $2s_m(1/(1-2\sum s_i))$ and $(2s_m/(s_m+1))(2.5 - (1/2m))$ and the performance ratio of the intermediate instance I_3'. Furthermore, in $\sigma(I_3')$, r_L is not the latest SPoint anymore. Rewrite I_3' as I_3 and repeat the above analysis. Ultimately the performance ratio of I_3 can be bounded from the above equation by $\max\{2s_m(1-(1/2\sum s_i)), (2s_m/(1+s_m))(2.5-(1/2m))\}$.

Following Lemmas 1, 3, and 4, we can readily obtain that LW-SWPR is $\max\{2s_m(1-(1/2\sum s_i)), (2s_m/(1+s_m))(2.5-(1/2m))\}$-competitive. Thus, Theorem 1 is proved. □

4. Conclusions

In this work, we design an online algorithm for $Q_m|r_j, \text{online}|\sum w_j C_j$ and prove that it is $\max\{2s_m(1-(1/2\sum s_i)), (2s_m/(1+s_m))(2.5-(1/2m))\}$-competitive. The result is a generalization from the identical-machine scheduling considered in [24]. Although the competitive performance is not a constant because it tends to infinity when the fastest machine speed s_m tends to infinity, it still makes improvements on the existed results in some extents. The first one is that it improves the result provided by Hall et al. [18] when the fastest machine is not too fast, say less than 4. The second one is that it tends to an s_m-related constant when the machine number tends to infinity. In this sense, it also improves the result provided by Liu et al. [28]. Furthermore, it is very hopeful to improve the performance guarantee to an s_m-independent value if all the machines are considered besides the idle ones in the scheduling rule. The same analysis method can be employed. It deserves to be further investigated in our future work.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Science Foundation of China (no. 51975496), the Fundamental Research Funds for the Central Universities of China (no. 20720180076), and the Natural Science Foundation of Fujian Province, China (no. 2018J01083).

References

[1] M. Singer and M. Pinedo, "A computational study of branch and bound techniques for minimizing the total weighted tardiness in job shops," *IIE Transactions*, vol. 30, no. 2, pp. 109–118, 1998.

[2] D. J. Hoitomt, P. B. Luh, and K. R. Pattipati, "A practical approach to job-shop scheduling problems," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 1, pp. 1–13, 1993.

[3] Z.-L. Chen and W. B. Powell, "Solving parallel machine scheduling problems by column generation," *INFORMS Journal on Computing*, vol. 11, no. 1, pp. 78–94, 1999.

[4] A. P. J. Vepsalainen and T. E. Morton, "Priority rules for job shops with weighted tardiness costs," *Management Science*, vol. 33, no. 8, pp. 1035–1047, 1987.

[5] P. J. M. Van Laarhoven, E. H. L. Aarts, and J. K. Lenstra, "Job shop scheduling by simulated annealing," *Operations Research*, vol. 40, no. 1, pp. 113–125, 1992.

[6] E. S. H. Hou, N. Ansari, and H. Hong Ren, "A genetic algorithm for multiprocessor scheduling," *IEEE Transactions on Parallel and Distributed Systems*, vol. 5, no. 2, pp. 113–120, 1994.

[7] A. El-Bouri, S. Balakrishnan, and N. Popplewell, "Sequencing jobs on a single machine: a neural network approach," *European Journal of Operational Research*, vol. 126, no. 3, pp. 474–490, 2000.

[8] X. Li and S. Olafsson, "Discovering dispatching rules using data mining," *Journal of Scheduling*, vol. 8, no. 6, pp. 515–527, 2005.

[9] P. Priore, B. Ponte, J. Puente, and A. Gómez, "Learning-based scheduling of flexible manufacturing systems using ensemble methods," *Computers & Industrial Engineering*, vol. 126, pp. 282–291, 2018.

[10] P. Brucker, *Scheduling Algorithms*, Springer, Berlin, Germany, 2004.

[11] M. Pinedo, *Scheduling: Theory, Algorithms, and Systems*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2002.

[12] L. Sastoque Pinilla, R. Llorente Rodríguez, N. Toledo Gandarias, L. N. López de Lacalle, and M. Ramezani Farokhad, "TRLs 5-7 advanced manufacturing centres, practical model to boost Technology transfer in manufacturing," *Sustainability*, vol. 11, no. 18, p. 4890, 2019.

[13] D. A. Rossit, F. Tohmé, and M. Frutos, "Industry 4.0: smart scheduling," *International Journal of Production Research*, vol. 57, no. 12, pp. 3802–3813, 2019.

[14] G. Urbikain and L. N. López de Lacalle, "Monithor: a complete monitoring tool for machining data acquisition based on fpga programming," *SoftwareX*, vol. 11, Article ID 100387, 2020.

[15] J. Zhang, G. Ding, Y. Zou, S. Qin, and J. Fu, "Review of job shop scheduling research and its new perspectives under industry 4.0," *Journal of Intelligent Manufacturing*, vol. 30, no. 4, pp. 1809–1830, 2019.

[16] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. R. Kan, "Optimization and approximation in deterministic sequencing and scheduling: a survey," *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979.

[17] R. L. Graham, "Bounds for certain multiprocessing anomalies," *Bell System Technical Journal*, vol. 45, no. 9, pp. 1563–1581, 1966.

[18] L. A. Hall, A. S. Schulz, D. B. Shmoys, and J. Wein, "Scheduling to minimize average completion time: off-line and on-line approximation algorithms," *Mathematics of Operations Research*, vol. 22, no. 3, pp. 513–544, 1997.

[19] J. A. Hoogeveen and A. P. A. Vestjens, "Optimal on-line algorithms for single-machine scheduling," in *Proceedings of the Lecture Notes in Computer Science 1084*, pp. 404–414, Vancouver, Canada, May 1996.

[20] E. J. Anderson and C. N. Potts, "Online scheduling of a single machine to minimize total weighted completion time," *Mathematics of Operations Research*, vol. 29, no. 3, pp. 686–697, 2004.

[21] J. R. Correa and M. R. Wagner, "LP-based online scheduling: from single to parallel machines," *Mathematical Programming*, vol. 119, no. 1, pp. 109–136, 2009.

[22] C. N. Potts and V. A. Strusevich, "Fifty years of scheduling: a survey of milestones," *Journal of the Operational Research Society*, vol. 60, no. 1, pp. S41–S68, 2009.

[23] N. Megow and A. S. Schulz, "On-line scheduling to minimize average completion time revisited," *Operations Research Letters*, vol. 32, no. 5, pp. 485–490, 2004.

[24] J. Tao, "A better online algorithm for the parallel machine scheduling to minimize the total weighted completion time," *Computers & Operations Research*, vol. 43, pp. 215–224, 2014.

[25] C. Koulamas and G. J. Kyparisis, "Makespan minimization on uniform parallel machines with release times," *European Journal of Operational Research*, vol. 157, no. 1, pp. 262–266, 2004.

[26] C.-H. Lin and C.-J. Liao, "Makespan minimization for multiple uniform machines," *Computers & Industrial Engineering*, vol. 54, no. 4, pp. 983–992, 2008.

[27] M. Liu, Y. Xu, C. Chu, and F. Zheng, "Online scheduling on two uniform machines to minimize the makespan," *Theoretical Computer Science*, vol. 410, no. 21–23, pp. 2099–2109, 2009.

[28] M. Liu, C. Chu, Y. Xu, and F. Zheng, "Online scheduling on m uniform machines to minimize total (weighted) completion time," *Theoretical Computer Science*, vol. 410, no. 21–23, pp. 3875–3881, 2009.

[29] P. Liu and X. Lu, "On line scheduling of two uniform machines to minimize total completion times," *Journal of Industrial & Management Optimization*, vol. 5, no. 1, pp. 95–102, 2009.

[30] X. Lu, R. A. Sitters, and L. Stougie, "A class of on-line scheduling algorithms to minimize total completion time," *Operations Research Letters*, vol. 31, no. 3, pp. 232–236, 2003.

[31] C. Phillips, C. Stein, and J. Wein, "Minimizing average completion time in the presence of release dates," *Mathematical Programming*, vol. 82, pp. 199–223, 1998.

[32] M. X. Goemans, "Improved approximation algorithms for scheduling with release dates," in *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 591–598, Barcelona Spain, January 1997.

[33] M. X. Goemans, M. Queyranne, A. S. Schulz, M. Skutella, and Y. Wang, "Single machine scheduling with release dates," *SIAM Journal on Discrete Mathematics*, vol. 15, no. 2, pp. 165–192, 2002.

[34] P. Liu and X. Lu, "On-line scheduling of parallel machines to minimize total completion times," *Computers & Operations Research*, vol. 36, no. 9, pp. 2647–2652, 2009.

[35] J. Tao, R. Huang, and T. Liu, "A 2.28-competitive algorithm for online scheduling on identical machines," *Journal of Industrial & Management Optimization*, vol. 11, no. 1, pp. 185–198, 2015.

[36] R. Ma and J. Tao, "An improved 2.11-competitive algorithm for online scheduling on parallel machines to minimize total weighted completion time," *Journal of Industrial & Management Optimization*, vol. 14, no. 2, pp. 497–510, 2018.

[37] J. Tao, Z. Chao, and Y. Xi, "A semi-online algorithm and its competitive analysis for a single machine scheduling problem with bounded processing times," *Journal of Industrial & Management Optimization*, vol. 6, no. 2, pp. 269–282, 2010.

[38] J. Xi, Z. Chao, Y. Xi, and Y. Tao, "An optimal semi-online algorithm for a single machine scheduling problem with bounded processing time," *Information Processing Letters*, vol. 110, no. 8-9, pp. 325–330, 2010.

[39] M. Gu and X. Lu, "Asymptotical optimality of WSEPT for stochastic online scheduling on uniform machines," *Annals of Operations Research*, vol. 191, no. 1, pp. 97–113, 2011.

[40] M. C. Chou, M. Queyranne, and D. Simchi-levi, "The asymptotic performance ratio of an on-line algorithm for uniform parallel machine scheduling with release dates," *Mathematical Programming*, vol. 106, no. 1, pp. 137–157, 2006.