European Journal of Operational Research 232 (2014) 435–441

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

Order acceptance and scheduling with machine availability constraints

Xueling Zhong [a,1], Jinwen Ou [b,*], Guoqing Wang [b,1]

[a] Department of Computer, Guangdong University of Finance, Guangzhou 510520, People's Republic of China
[b] Department of Administrative Management, Jinan University, Guangzhou 510632, People's Republic of China

ARTICLE INFO

Article history:
Received 22 May 2012
Accepted 22 July 2013
Available online 27 July 2013

Keywords:
Order acceptance and scheduling
Machine availability constraints
Job rejection
Approximation

ABSTRACT

We consider an order acceptance and scheduling model with machine availability constraints. The manufacturer (machine) is assumed to be available to process orders only within a number of discontinuous time intervals. To capture the real-life behavior of a typical manufacturer who has restrictions of time availability to process orders, our model allows the manufacturer to reject or outsource some of the orders. When an order is rejected or outsourced, an order-dependent cost of penalty will occur. The objective is to minimize the makespan of all accepted orders plus the total penalty of all rejected/outsourced orders. We study the approximability of the model and some of its important special cases.

1. Introduction

Order acceptance and scheduling (OAS) has attracted considerable attention from scheduling researchers as well as production managers who practice it in the past a few decades. The key issue in OAS is to balance order acceptance (to increase revenue and guarantee satisfaction to customers) and order rejection (to reduce production load so as to meet production capacity restrictions). In traditional OAS models the manufacturer (machine) is usually assumed to have the capacity of processing orders continuously. But in reality, production could be interrupted due to some potential reasons such as machine breakdown/maintenance, production setup, and worker vacation, which leads to the scenario that there are only a number of discontinuous time intervals available to process the orders in question. When production capacity and machine available time have restrictions, to reduce production load, the manufacturer may have to reject some orders which have long processing times but contribute relatively small profits, or outsource the production of some orders, in which case the profit of those outsourced orders is reduced. In this paper we study a single-machine OAS model with machine availability constraints. In the proposed model, the manufacturer (machine) is available to process orders only within a number of given discontinuous time intervals, and she is allowed to reject some orders. When an order is rejected, an order-dependent penalty will occur. The manufacturer needs to balance the production capacity (the makespan of completing all accepted orders) and the total penalty of all rejected jobs.

OAS has been studied extensively in the production scheduling literature. Guerrero and Kern (1988) provided the rational how to accept and reject orders effectively. Slotnick and Morton (1996) and Ghosh (1997) studied single-machine OAS with the objective of maximizing revenue minus weighted lateness penalties. Slotnick and Morton (2007) extended their previous work (Slotnick & Morton, 1996) by replacing lateness with tardiness in the objective function. Rom and Slotnick (2009) developed a genetic algorithm for the same problem and compared it with the myopic heuristic given in Slotnick and Morton (2007). Talla Nobibon and Leus (2011) studied an OAS model with "firm planned orders as well as potential orders". Oguz, Salman, and Yalcin (2010) included sequence-dependent setup times and two-level due-dates in an OAS model with weighted tardiness. Cesaret, Oguz, and Salman (2012) developed a tabu algorithm for the same problem and compared it with the two heuristics proposed in Oguz et al. (2010). Yang and Geunes (2007) considered an OAS model with controllable processing times. Chen and Li (2008), Lee and Sung (2008, 2008) and Qi (2008, 2009, 2011) considered OAS models with outsourcing options. An excellent literature survey on the topic of OAS is provided by Slotnick (2011) recently.

OAS is actually equivalent to machine scheduling with rejection (MSR) in mathematics. Bartal, Leonardi, Marchetti-Spaccamela, Sgall, and Stougie (2000) first introduced a multi-processor MSR model to minimize the makespan of all accepted jobs plus the total rejection penalty of all rejected jobs. Seiden (2001) and Hoogeveen, Skutella, and Woeginger (2003) studied the on-line version and the off-line version in Bartal et al. (2000) when job preemption is

* Corresponding author. Tel.: +86 20 3820 2796.
E-mail addresses: zhongxuel@hotmail.com (X. Zhong), toujinwen@jnu.edu.cn (J. Ou), tgqwang@jnu.edu.cn (G. Wang).
[1] Tel.: +86 20 8522 3243.

0377-2217/$ - see front matter
http://dx.doi.org/10.1016/j.ejor.2013.07.032

allowed, respectively. Epstein, Noga, and Woeginger (2002) considered on-line MSR with unit-processing-time jobs and the total completion time of all accepted jobs. Engels et al. (2003) studied single-processor MSR with total weighted completion times of all accepted jobs, and recently, Kellerer and Strusevich (2013) improved the related results. MSR models with industrial applications are studied by Cheng and Sun (2009), Zhang, Lu, and Yuan (2009, 2010), Lu, Zhang, and Yuan (2008), Lu, Cheng, Yuan, and Zhang (2009), among others. A very recent survey on MSR is provided by Shabtay, Gaspar, and Kaspi (2013). All of the OAS and MSR models mentioned above have the potential assumption that machines can process jobs continuously, which differ from our model.

Our model is an extension of machine scheduling with availability constraints (MSAC). MSAC also has been studied extensively. The surveys of MSAC are provided by Schmidt (2000) and Ma, Chu, and Zuo (2010). In this paper we consider a single machine model and the criterion of all accepted jobs is to minimize the makespan. Therefore, in the following we only review the previous work of MSAC with a single machine, and whose objective is limited to minimization of makespan. For convenience, define MSAC-k to be the MSAC problem to minimize makespan with a single machine and k time intervals during which the machine is not allowed to process jobs. Lee (1996) showed that MSAC-1 has been NP-hard, and that MSAC-k is NP-hard in the strong sense if k is arbitrary. He, Ji, and Cheng (2005) proposed a fully polynomial time approximation scheme (FPTAS) for MSAC-1. Breit, Schmidt, and Strusevich (2003) showed that no polynomial time approximation algorithm with a fixed performance ratio exists for MSAC-2. When every job also has a delivery time, Yuan, She, and Ou (2008) showed that MSAC-k can be solved by a pseudo-polynomial time algorithm if k is fixed. They also developed a polynomial time approximation scheme (PTAS) for MSAC-1. Kacem (2009) studied a similar problem to Yuan et al. (2008), and proposed an FPTAS by exploiting the well-known approach of Ibarra and Kim (1975). Wu and Lee (2003), Gawiejnowicz (2007), and Ji, He, and Cheng (2006) studied MSAC with deterioration, where the processing time of a job is a linear function of its starting time. To the best of our knowledge, our model is the first to examine OAS with machine availability constraints.

We now describe our problem formally as follows. Given a set of n jobs $J = \{J_1, J_2, \ldots, J_n\}$ and a single machine, each job $J_j \in J$ is associated with a processing time p_j and a rejection penalty w_j. Job J_j is either accepted and then processed on the machine, or it is rejected by the machine and then a rejection penalty w_j is paid. Penalty w_j is regarded as the cost of losing or outsourcing job J_j. Also given $2m+1$ integers $a_0, a_1, \ldots, a_{2m}$ such that $0 = a_0 < a_1 < \cdots < a_{2m}$, we assume that in the planning horizon the machine is available to process jobs within time intervals $[a_{2m}, +\infty)$ and $[a_{2i-2}, a_{2i-1}]$ for $i = 1,2,\ldots,m$, but it is not allowed to process any job during time interval (a_{2i-1}, a_{2i}) for any $i = 1,2,\ldots,m$. In order words, there are m UIs (time intervals during which the machine is not allowed to process jobs) and $m+1$ AIs (time intervals during which the machine is allowed to process jobs) in the planning horizon. Let A be the set of jobs accepted, and $R = J \backslash A$ be the set of jobs rejected. Define C_{max} to be the makespan of A, i.e., the completion time of the last accepted job. The problem is to determine A and a feasible schedule of jobs in A on the machine so as to minimize the objective function $Z = C_{max} + \sum_{J_j \in R} w_j$, i.e., the makespan of all accepted jobs plus the total penalty of all rejected jobs. Denote the proposed problem with m UIs (or, equivalently, $m+1$ AIs) by $\mathbf{P}^{(m)}$.

In our model, we assume that the machine can process at most one job at a time, that job preemption is not allowed, and that p_j and w_j are non-negative integers for $j = 1,2,\ldots,n$. For convenience, we introduce the following notations, which will be used throughout the paper:

$P = \sum_{J_j \in J} p_j$	the total processing time of all the jobs;
$U = \{J_j \in J \mid w_j > p_j\}$	the set of jobs with processing times less than their rejection penalty;
$V = \{J_j \in J \mid w_j \leqslant p_j\}$	the set of jobs with processing times no less than their rejection penalty;
$P(U) = \sum_{J_j \in U} p_j$	the total processing time of all the jobs in U;
σ^*	the optimal solution;
A^*	the set of all accepted jobs in σ^*;
$R^* = J \backslash A^*$	the set of all rejected jobs in σ^*;
C^*_{max}	the makespan of A^*, i.e., the completion time of the last accepted job in A^*.
$Z^* = C^*_{max} + \sum_{J_j \in R^*} w_j$	the objective function value of σ^*;
$a_{2m+1} = a_{2m} + P$	the largest possible makespan of all accepted jobs in σ^*;
$L_i = a_{2i} - a_{2i-1}$	the length of the ith UI ($i = 1,2,\ldots,m$);
$L'_i = a_{2i-1} - a_{2i-2}$	the length of the ith AI ($i = 1,2,\ldots,m+1$).

Note that if $P(U) \leqslant a_1$, it is easy to obtain an optimal solution where all the jobs in U are accepted and all the jobs in V are rejected. Also note that if there exists some $\alpha \in \{1,2,\ldots,m\}$ such that $L'_\alpha \geqslant P$, then it is easy to show that no job should be processed after $a_{2\alpha-1}$ in any optimal solution. Thus, we can ignore the UIs after $a_{2\alpha-1}$ without destroying optimality if such α exists. Without loss of generality, in the remainder of this paper we assume $P(U) > a_1$ and

$$L'_i < P \quad (i = 1,2,\ldots,m). \tag{1}$$

The rest of the paper is organized as follows. In Section 2 we develop a pseudo-polynomial algorithm to solve problem $\mathbf{P}^{(m)}$ when m is fixed. In Section 3 we provide non-approximability results to problem $\mathbf{P}^{(m)}$ when $m > 1$. In Section 4 we present an FPTAS for the special case $\mathbf{P}^{(1)}$. In Section 5 we develop a $(2+\epsilon)$-approximation for the special case with $L'_i = L'$ and $L_i = L$ for $i = 1,2,\ldots,m$. Some concluding remarks are provided in Section 6.

2. A pseudo-polynomial algorithm when m is fixed

Note that if m is arbitrarily, then it is easy to show that problem $\mathbf{P}^{(m)}$ is NP-hard in the strong sense by reducing from 3-PARTITION, and thus an optimal pseudo-polynomial algorithm for $\mathbf{P}^{(m)}$ does not exist (Garey & Johnson, 1979). Therefore, in the remainder of this section, we assume that the value of m is fixed. We will present a pseudo-polynomial algorithm to solve $\mathbf{P}^{(m)}$ when m is fixed.

Consider C^*_{max}, the makespan of all accepted jobs in optimal solution σ^*. Assume that

$$a_{2\mu-2} < C^*_{max} \leqslant a_{2\mu-1}$$

for some $\mu \in \{1,2,\ldots,m+1\}$, i.e., the last accepted job in σ^* is processed and completed within the μth AI $[a_{2\mu-2}, a_{2\mu-1}]$. Provided the value of μ, we will present a pseudo-polynomial algorithm to solve $\mathbf{P}^{(m)}$ in the following.

We first study the situation when $\mu = m+1$, in which case we will develop an $O(nP^m)$ optimal dynamic program (DP) to solve $\mathbf{P}^{(m)}$. Define J_0 to be a dummy job with zero processing time, and fix J_0 to be accepted and processed on the machine at time a_{2m} (i.e., J_0 is the first accepted job processed within the last AI). We schedule J_0 before the scheduling of any job in our DP. Scheduling

J_0 has a contribution of a_{2m} to the objective function value. Due to the existence of J_0, for any job J_j ($j = 1,2,\ldots,n$), no matter it is rejected, or processed within the $(m+1)$th AI, its contribution to the value of the objective function is equal to

$$p'_j = \min\{p_j, w_j\}.$$

Based on such an insight, we are now ready to present our DP for the case with $\mu = m+1$. Our DP will keep track of the total processing time of the jobs scheduled within each of the first m AIs when scheduling the n jobs one by one. Denote the problem instance containing only the first j jobs $J_1, J_2, \ldots, J_j$ by $\mathbf{I}_j$. For any $j = 1,2,\ldots,n$ and $g_i = 0,1,\ldots,L'_i$, define $G_j(g_1,\ldots,g_m)$ as the minimum total cost to $\mathbf{I}_j$ such that the total processing time of all accepted jobs assigned to AI $[a_{2i-2}, a_{2i-1}]$ is equal to g_i for $i = 1,2,\ldots,m$. To develop recursive relations, consider the following two cases:

Case a. Job J_j is rejected, or it is processed within the $(m+1)$th AI. In this case, we have $G_j(g_1,\ldots,g_m) = G_{j-1}(g_1,\ldots,g_m) + p'_j$.
Case b. Job J_j is accepted and scheduled to the ith AI for some $i \in \{1,2,\ldots,m\}$. In this case, we should have $g_i \geqslant p_j$ and $G_j(g_1,\ldots,g_m) = G_{j-1}(g_1,\ldots,g_i - p_j,\ldots,g_m)$ as J_j does not contribute to the value of the objective function.

Combining the two cases above, we have the following dynamic program $\mathbf{DP}_1$:

(I) Recurrence relation: For $j = 1,2,\ldots,n$, $i = 1,2,\ldots,m$ and $g_i = 0,1,\ldots,L'_i$,

$$G_j(g_1,\ldots,g_m) = \min\left\{ G_{j-1}(g_1,\ldots,g_m) + p'_j,\ \min_{\substack{1\leqslant i\leqslant m \\ \text{s.t. } g_i \geqslant p_j}} \{G_{j-1}(g_1,\ldots,g_{i-1},g_i - p_j,g_{i+1},\ldots,g_m)\} \right\}.$$

(II) Boundary condition:

$$G_0(g_1,\ldots,g_m) = \begin{cases} a_{2m}, & \text{if } g_1 = g_2 = \cdots = g_m = 0; \\ +\infty, & \text{otherwise.} \end{cases}$$

(III) Objective:

$$\min\left\{ G_n(g_1,\ldots,g_m) \mid 1 \leqslant i \leqslant m,\ 0 \leqslant g_i \leqslant L'_i,\ \sum_{i=1,2,\ldots,m} g_i \leqslant P \right\}.$$

Consider the complexity of $\mathbf{DP}_1$. Note that $L'_i < P$ for $i = 1,2,\ldots,m$ (by (1)). The recursive function has at most $O(nP^m)$ states for $G_j(g_1,\ldots,g_m)$, and each recursion can be calculated in $O(m) = O(1)$ time. Therefore, $\mathbf{DP}_1$ has a complexity of $O(nP^m)$.

We now study the situation when $\mu \leqslant m$. For this case, we develop another DP to solve the problem. The new DP is similar to $\mathbf{DP}_1$, but it needs to keep track of the total processing time of the jobs processed within each of the first μ AIs. It is not difficult to design such a DP with complexity of $O(nP^\mu)$. We would like to point out that the new DP is actually applicable to the situation when $\mu = m+1$, but the corresponding time complexity is $O(nP^{m+1})$. To reduce complexity, we develop $\mathbf{DP}_1$ to handle the situation when $\mu = m+1$ additionally, so that the overall time complexity of our algorithm is bounded by $O(nP^m)$. We thus have the following theorem.

Theorem 1. *Problem $\mathbf{P}^{(m)}$ can be solved in $O(nP^m)$ time when m is fixed.*

3. Non-approximability results

In this section, we study the approximability of problem $\mathbf{P}^{(m)}$. We will show that $\mathbf{P}^{(m)}$ does not admit a polynomial time approximation algorithm with a constant worst-case bound when $m > 1$, and that a 2-approximation exists for the special case with $P(U) \geqslant a_{2m}$. As we will use the well-known NP-hard problem PARTITION (Garey & Johnson, 1979) in our analysis, for convenience, we state PARTITION in advance here: Given a set of t positive integers $S = \{e_1, e_2, \ldots, e_t\}$ such that $\sum_{i=1}^{t} e_i = 2B$. The question is to decide if there is a subset $S' \subseteq S$ such that $\sum_{e_i \in S'} e_i = \sum_{e_i \in S\setminus S'} e_i = B$.

Theorem 2. *When $m > 1$, problem $\mathbf{P}^{(m)}$ does not admit a polynomial time approximation algorithm with worst-case bound better than 2^n unless $P = NP$.*

Proof. We use the gap reduction from PARTITION. Given an instance I of PARTITION, construct an instance I' of $\mathbf{P}^{(m)}$ as follows. There are $n = t$ jobs, where the processing time and rejection penalty of job J_j are $(p_j, w_j) = (e_j, 2^n(2B+1))$ for $j = 1,\ldots,t$. There are $m = 2$ UIs: $(a_1, a_2) = (B, B+1)$ and $(a_3, a_4) = (2B+1, 2^n(2B+1))$. Such an instance can be constructed in polynomial time. Consider the optimal solution value of instance I'. It is obvious that the optimal solution value of I' is no less than $2B+1$. If we reject one job, the total penalty is no less than $2^n(2B+1)$. Also, if there exists a job processed after time $2^n(2B+1)$, the corresponding makespan also will be no less than $2^n(2B+1)$. Thus, to obtain a solution value of $2B+1$ is equivalent to find a subset of jobs with a total processing time of B, so that all of the jobs in the subset are processed on the machine within interval $[0, B]$ without any idle time, while all of the rest of the other jobs are processed on the machine within interval $[B+1, 2B+1]$ also without any idle time. This is equivalent to a solution to PARTITION. If it fails to find out such a subset of jobs, then solution value is at least $2^n(2B+1)$, which is 2^n times of $2B+1$. This completes the proof. □

Through such the proof above, we can see that if a_{2m}, the ending point of the last UI, is much larger than total processing time $P(U)$, then it is impossible to develop a polynomial approximation algorithm with a constant worst-case bound. However, as we will show it later, if $P(U) > a_{2m}$, i.e., the total processing time in U is greater than the ending point of the last UI, then a very simple heuristic with a worst-case bound of 2 can be developed. The heuristic is denoted by H_1, which is described as follows: Reject all jobs in V, and let all jobs in U be accepted and processed within time interval $[a_{2m}, +\infty)$ as early as possible in any job sequence. Let Z_{H_1} be the objective function value of the solution generated by heuristic H_1. We have the following theorem.

Theorem 3. *$Z_{H_1} \leqslant 2Z^*$ if $P(U) > a_{2m}$.*

Proof. Note that we have $Z^* > a_{2m}$ if $P(U) > a_{2m}$. Remember that A^* and R^* are the set of all accepted jobs and the set of all rejected jobs in the optimal solution, respectively. We then have

$$\begin{aligned} Z^* &\geqslant \sum_{J_j \in A^*} p_j + \sum_{J_j \in R^*} w_j \geqslant \sum_{J_j \in A^*} \min\{p_j, w_j\} + \sum_{J_j \in R^*} \min\{p_j, w_j\} \\ &= \sum_{J_j \in J} \min\{p_j, w_j\} = \sum_{J_j \in U} \min\{p_j, w_j\} + \sum_{J_j \in V} \min\{p_j, w_j\} \\ &= \sum_{J_j \in U} p_j + \sum_{J_j \in V} w_j = P(U) + \sum_{J_j \in V} w_j. \end{aligned}$$

Thus,

$$Z_{H_1} = a_{2m} + P(U) + \sum_{J_j \in V} w_j < 2P(U) + \sum_{J_j \in V} w_j \leqslant 2\left(P(U) + \sum_{J_j \in V} w_j \right) \leqslant 2Z^*.$$

This completes the proof. □

When $P(U) > a_{2m}$, the following theorem shows that the worst-case bound of 2 cannot be improved unless $P = NP$. Let $\epsilon > 0$ be any given small constant such that $\frac{1}{\epsilon}$ is a positive integer.

Theorem 4. *If $P(U) > a_{2m}$, problem $\mathbf{P}^{(m)}$ does not admit a polynomial time approximation algorithm with worst-case bound better than $2-\epsilon$ unless P = NP, even when m = 2.*

Proof. We use the gap reduction from PARTITION again. Given an instance I of PARTITION, construct an instance I'' of $\mathbf{P}^{(m)}$ as follows. Let $d = \frac{2B+4}{\epsilon}$. There are $n = t+1$ jobs, where the processing time and rejection penalty of J_j are $(p_j, w_j) = (e_j, 2B+1+d)$ for $j = 1,\ldots,t$, and $(p_{t+1}, w_{t+1}) = (d+2, d+3)$. There are $m = 2$ UIs: $(a_1, a_2) = (B, B+1)$ and $(a_3, a_4) = (2B+1, 2B+1+d)$. Such an instance construction can be done in polynomial time. It is obvious that $U = \{J_1, J_2, \ldots, J_{t+1}\}$ and $V = \emptyset$. Hence, we have $P(U) = 2B+2+d > 2B+1+d = a_4 = a_{2m}$.

Suppose that there is a polynomial time approximation algorithm **A** for solving I'' such that $Z' \leqslant (2-\epsilon)Z^*$, where Z' is the objective function value of the solution generated by **A**.

Consider the answer to instance I. On one hand, if instance I has a "YES" answer, then there exists a subset $S' \subseteq S$ such that $\sum_{e_i \in S'} e_i = \sum_{e_i \in S \setminus S'} e_i = B$. Consider the following solution to instance I'': Accept all of jobs corresponding to S' and schedule them within AI $[0, B]$; also accept all of jobs corresponding to $S \setminus S'$ and schedule them within AI $[B+1, 2B+1]$; reject job J_{t+1}. It is easy to check that the objective function value of such a solution is equal to $2B+4+d$, which is optimal. Note that $d \cdot \epsilon = 2B+4$. We thus have

$$Z' \leqslant (2-\epsilon)Z^* = (2-\epsilon)(2B+4+d) = 2B+4+2d-\epsilon(2B+4) < 2B+4+2d.$$

On the other hand, if instance I has a "NO" answer, then there does not exist a subset $S_1 \subseteq S$ such that $\sum_{e_i \in S'} e_i = \sum_{e_i \in S \setminus S'} e_i = B$. As a result, at least one job among $J_1, J_2, \ldots, J_t$ is scheduled to neither interval $[0, B]$ nor $[B+1, 2B+1]$. Without loss of generality, assume such a job to be J_k, where $k \in \{1,2,\ldots,t\}$. Then, there are two possible cases: Either J_k is rejected, or J_k is accepted and processed within interval $[2B+1+d, +\infty)$ on the machine. If job J_k is rejected, we have

$$Z' \geqslant w_k + w_{t+1} = (2B+1+d) + (d+3) = 2B+4+2d.$$

If J_k is accepted and processed within interval $[2B+1+d, +\infty)$, it is easy to check that J_{t+1} should also be processed within interval $[2B+1+d, +\infty)$. The corresponding objective function value determined by algorithm **A** should have

$$Z' \geqslant a_{2m} + p_{t+1} + p_k = (2B+1+d) + (d+2) + p_k = 2B+3+2d+p_k.$$

Note that $p_k = e_k$ is a positive integer. Thus,

$$Z' \geqslant 2B+4+2d.$$

In any case, we have $Z' \geqslant 2B+4+2d$ when instance I has a "NO" answer.

The above analysis indicates that if $Z' < 2B+4+2d$, then instance I has a "YES" answer; Otherwise, I has a "NO" answer. Consequently, we can use approximation algorithm **A** to solve PARTITION in polynomial time, which is a contradiction unless P = NP. □

4. An FPTAS to $\mathbf{P}^{(1)}$

In this section we study problem $\mathbf{P}^{(1)}$, where there is only one UI in the time horizon. Denote the unique UI by (a, b). Note that $\mathbf{P}^{(1)}$ has been shown to be NP-hard when job rejection is not allowed (see Lee, 1996), which indicates that $\mathbf{P}^{(1)}$ is NP-hard in general. We will present an FPTAS to $\mathbf{P}^{(1)}$. A family of algorithms $\{A_\epsilon | \epsilon > 0\}$ is called an FPTAS of a minimization problem if A_ϵ is a $(1+\epsilon)$-approximation running in polynomial time in the input size and $\frac{1}{\epsilon}$ for each given ϵ.

We will use the classical 0–1 Min-Knapsack Problem (Min-KP) (Kellerer, Pferschy, & Pisinger, 2004) in our analysis. For convenience, we state Min-KP in advance here: Given a knapsack and a set of items; the size of the knapsack is given; associated with each item is a given size and a given profit. The problem is to select a subset of items into the knapsack so as to minimize the total profit of all unselected items. It is well-known that an FPTAS exists for solving Min-KP (see Kellerer et al., 2004).

We now present an FPTAS to $\mathbf{P}^{(1)}$ based on the FPTAS to Min-KP. Our analysis is partitioned into the following two cases: (i) $C^*_{max} > b$ and (ii) $C^*_{max} \leqslant a$.

Case (i): $C^*_{max} > b$. In this case, it is easy to see that all jobs in U are accepted and processed on the machine in σ^*, the optimal solution to $\mathbf{P}^{(1)}$. However, some jobs in V might be scheduled within interval $[0, a]$ in σ^*. In other words, we have

$$U \subseteq A^* \tag{2}$$

and

$$R^* \subseteq V. \tag{3}$$

Let $Q \subseteq A^*$ be the set of jobs accepted and processed on the machine after time b in σ^*. Note that $Q \neq \emptyset$ as $C^*_{max} > b$. It is clear that

$$Z^* = b + \sum_{J_j \in Q} p_j + \sum_{J_j \in R^*} w_j \tag{4}$$

and

$$\sum_{J \setminus (Q \cup R^*)} p_j = \sum_{A^* \setminus Q} p_j \leqslant a. \tag{5}$$

In order to develop an FPTAS for $\mathbf{P}^{(1)}$ in this case, we define the following auxiliary problem $\mathbf{P}_a$: (i) define a knapsack with a capacity of a; (ii) for each job $J_j \in U$, define an item j with profit p_j and size p_j; and (iii) for each job $J_j \in V$, define an item j with profit w_j and size p_j. The question to select some items to the knapsack such that the total size of all selected items is no greater than a, while the total profit of the unselected items is minimized. Clearly, auxiliary problem $\mathbf{P}_a$ is equivalent to Min-KP. Let $\hat{\sigma}_a$ be the optimal solution to $\mathbf{P}_a$, and Z^*_a be the value of $\hat{\sigma}_a$. We will show that

$$Z^* = b + Z^*_a. \tag{6}$$

Let Q_a be the set of all unselected items corresponding to U in $\hat{\sigma}_a$, and R_a be the set of all unselected items corresponding to V in $\hat{\sigma}_a$. By the definition of $\mathbf{P}_a$, we have

$$Z^*_a = \sum_{j \in Q_a} p_j + \sum_{j \in R_a} w_j \tag{7}$$

and

$$\sum_{j \in \{1,2,\ldots,n\} \setminus (Q_a \cup R_a)} p_j \leqslant a. \tag{8}$$

Let $Q'_a \subseteq U$ be the job set corresponding to Q_a, and let $R'_a \subseteq V$ be the job set corresponding to R_a. To prove Eq. (6), on one hand, if $b + Z^*_a < Z^*$, we can construct a better solution to $\mathbf{P}^{(1)}$ as follows: Reject all jobs in R'_a, accept all jobs in $J \setminus R'_a$, process all jobs in Q'_a on the machine after time b as early as possible in any sequence, and process all jobs in $J \setminus (R'_a \cup Q'_a)$ on the machine within time interval $[0, a]$ in any sequence. Consider the feasibility of such a solution to $\mathbf{P}^{(1)}$. Note that

$$Q'_a \subseteq U = J \setminus V \subseteq J \setminus R'_a,$$

i.e., it is feasible to partition $J \setminus R'_a$ into two subsets: Q'_a and $J \setminus (Q'_a \cup R'_a)$. By (8), we have

$$\sum_{J_j\in J\setminus\left(Q'_a\cup R'_a\right)} p_j = \sum_{j\in\{1,2,\ldots,n\}\setminus(Q_a\cup R_a)} p_j \leqslant a,$$

i.e., such a solution to $\mathbf{P}^{(1)}$ is feasible. However, by (7), the objective function value of such a solution to $\mathbf{P}^{(1)}$ is equal to

$$b+\sum_{J_j\in Q'_a} p_j+\sum_{J_j\in R'_a} w_j = b+\sum_{j\in Q_a} p_j+\sum_{j\in R_a} w_j = b+Z^*_a < Z^*,$$

which is a contradiction to the optimality of σ^*. On the other hand, if $b+Z^*_a > Z^*$, we can construct a better solution to $\mathbf{P}_a$ as follows: Select all items corresponding to $J\setminus(Q\cup R^*)$ to the knapsack. By (5) and the definition of $\mathbf{P}_a$, such a solution is feasible. By (2)–(4) and the definition of $\mathbf{P}_a$, the objective function value of such a solution to $\mathbf{P}_a$ is equal to

$$\sum_{J_j\in Q} p_j+\sum_{J_j\in R^*} w_j = Z^*-b < Z^*_a,$$

which is a contradiction to the optimality of $\hat{\sigma}_a$. Thus, Eq. (6) holds.

We now show that based on the FPTAS to Min-KP, we can develop an FPTAS to $\mathbf{P}^{(1)}$ when $C^*_{max} > b$ easily. Assume that $\widetilde{A}_\epsilon$ is a $(1+\epsilon)$-approximation algorithm to Min-KP. Apply algorithm $\widetilde{A}_\epsilon$ to solve auxiliary problem $\mathbf{P}_a$. Let $Z(\widetilde{A}_\epsilon)$ be the objective function value of the solution after applying $\widetilde{A}_\epsilon$ to $\mathbf{P}_a$. We have

$$Z(\widetilde{A}_\epsilon) \leqslant (1+\epsilon)Z^*_a. \tag{9}$$

Note that based on the solution generated by algorithm $\widetilde{A}_\epsilon$, we can obtain a feasible solution with an objective function value of $b+Z(\widetilde{A}_\epsilon)$ for problem $\mathbf{P}^{(1)}$. Thus, using such an approach to obtain a solution to $\mathbf{P}^{(1)}$, by (6) and (9), we have

$$\frac{b+Z(\widetilde{A}_\epsilon)}{Z^*} = \frac{b+Z(\widetilde{A}_\epsilon)}{b+Z^*_a} \leqslant \frac{b+(1+\epsilon)Z^*_a}{b+Z^*_a} = 1+\frac{\epsilon Z^*_a}{b+Z^*_a} \leqslant 1+\epsilon,$$

i.e., such an approach is a $(1+\epsilon)$-approximation algorithm to problem $\mathbf{P}^{(1)}$, and the time complexity is the same as the one of $\widetilde{A}_\epsilon$. In other words, when $C^*_{max} > b$, there exists an FPTAS to $\mathbf{P}^{(1)}$ with the same time complexity as the FPTAS to Min-KP.

Case (ii): $C^*_{max} \leqslant a$. In this case, it is easy to see that it is optimal to reject all the jobs in V, and that $P(U) > a$. Without loss of generality, we assume $V=\emptyset$ in this case. Note that in the optimal solution to $\mathbf{P}^{(1)}$ in this case,

$$\begin{aligned} Z^* &= \sum_{J_j\in A^*} p_j+\sum_{J_j\in R^*} w_j = \sum_{J_j\in U} p_j-\sum_{J_j\in U\setminus A^*} p_j+\sum_{J_j\in R^*} w_j \\ &= P(U)-\sum_{J_j\in R^*} p_j+\sum_{J_j\in R^*} w_j = P(U)+\sum_{J_j\in R^*}(w_j-p_j). \end{aligned} \tag{10}$$

Also note that $w_j-p_j>0$ for any $J_j\in U$. This indicates that to obtain the optimal objective function value Z^*, it is equivalent to determine a set of accepted jobs A^* such that $\sum_{J_j\in A^*} p_j \leqslant a$ and the value of $\sum_{J_j\in U\setminus A^*}(w_j-p_j)=\sum_{J_j\in R^*}(w_j-p_j)$ is minimal. Based on such an insight, we define the following auxiliary problem $\mathbf{P}'_a$: (i) define a knapsack with a capacity of a and (ii) for each job $J_j\in U$, define an item with profit w_j-p_j and size p_j. The question to select some items to the knapsack such that the total size of all selected items is not greater than a while the total profit of the unselected items is minimal. Clearly, auxiliary problem $\mathbf{P}'_a$ is equivalent to Min-KP. It is easy to see that an optimal solution to $\mathbf{P}'_a$ can be transferred into an optimal solution to $\mathbf{P}^{(1)}$ immediately when $C^*_{max} \leqslant a$. Similar to Case (i), we now show that based on the FPTAS to Min-KP, we can develop an FPTAS to $\mathbf{P}^{(1)}$ easily. Remember that $\widetilde{A}_\epsilon$ is a $(1+\epsilon)$-approximation algorithm to Min-KP. Let $Z'(\widetilde{A}_\epsilon)$ be the objective function value of the solution generated by applying $\widetilde{A}_\epsilon$ to solve $\mathbf{P}'_a$. Then,

$$Z'(\widetilde{A}_\epsilon) \leqslant (1+\epsilon)\sum_{J_j\in R^*}(w_j-p_j). \tag{11}$$

Let $\widetilde{Z}'$ be the objective function value of the solution to $\mathbf{P}^{(1)}$ transferred from the solution generated by $\widetilde{A}_\epsilon$. Thus, by (10) and (11), we have

$$\begin{aligned} \widetilde{Z}' &= P(U)+Z'(\widetilde{A}_\epsilon) \leqslant P(U)+(1+\epsilon)\sum_{J_j\in R^*}(w_j-p_j) \\ &\leqslant (1+\epsilon)\left[P(U)+\sum_{J_j\in R^*}(w_j-p_j)\right] = (1+\epsilon)Z^*. \end{aligned}$$

This indicates that there exists a $(1+\epsilon)$-approximation algorithm to $\mathbf{P}^{(1)}$, and the corresponding time complexity is the same as the one of $\widetilde{A}_\epsilon$. In other words, when $C^*_{max} \leqslant a$, there also exists an FPTAS to $\mathbf{P}^{(1)}$ with the same time complexity as the FPTAS to Min-KP.

Summarizing the analysis of Cases (i) and (ii), we have presented an FPTAS to $\mathbf{P}^{(1)}$ with the same time complexity as the FPTAS to Min-KP. So far, the best FPTAS to Min-KP is designed by Kellerer and Pferschy (1999, 2004, 2004) with the time complexity of $O(n/\epsilon)$. Therefore, we have the following theorem.

Theorem 5. *There exists an FPTAS to problem $\mathbf{P}^{(1)}$ in $O(n/\epsilon)$ time.*

Remark 1. For the variant of $\mathbf{P}^{(m)}$ when job preemption is allowed, it is easy to show that such a variant is actually equivalent to problem $\mathbf{P}^{(1)}$. Thus, $\mathbf{P}^{(m)}$ is still NP-hard even when job preemption is allowed, and an FPTAS exists for such a variant.

5. When $L_i = L$ and $L'_i = L'$

In this section we consider the special case of $\mathbf{P}^{(m)}$ when the length of each AI is identical, and the length of each UI is also identical, i.e., $L'_i = L'$ and $L_i = L$ for $i = 1,2,\ldots,m$. This case happens when the working hour on each day is fixed, or when machine maintenance activities are periodic (Ji, He, & Cheng, 2007). Denote such a special case by $\overline{\mathbf{P}}_e$. Without loss of generality, we assume $p_j \leqslant L'$ and $w_j > 0$ for any $J_j\in J$ in problem $\overline{\mathbf{P}}_e$ (note that if $p_j > L'$ or $w_j = 0$, then it is optimal to reject J_j). We still let the optimal solution to $\overline{\mathbf{P}}_e$ be σ^*, and let C^*_{max} and Z^* be the makespan and the objective function value of σ^*, respectively. Let ϵ be any given small constant such that $1/\epsilon$ is a positive integer. We will present an efficient heuristic to solve $\overline{\mathbf{P}}_e$ with a tight worst-case bound of $2+\epsilon$ in the remainder of this section.

In order to develop the desired heuristic, we first partition J into $1+n/\epsilon$ subsets. We define

$$\overline{S}_0 = \{J_j\in J | w_j > L'+L\}$$

and

$$\overline{S}_i = \left\{J_j\in J \middle| \frac{(i-1)(L'+L)\epsilon}{n} < w_j \leqslant \frac{i(L'+L)\epsilon}{n}\right\}$$

for $i = 1,2,\ldots,n/\epsilon$. For any $J_j\in\overline{S}_i,\ i=1,2,\ldots,n/\epsilon$, let

$$\tilde{w}_j = \frac{(i-1)(L'+L)\epsilon}{n}$$

be the modified rejection penalty. Clearly,

$$0 \leqslant \tilde{w}_j \leqslant w_j \leqslant \tilde{w}_j+\frac{(L'+L)\epsilon}{n}. \tag{12}$$

Define $\overline{\mathbf{I}}$ to be the problem instance the same as the original problem $\overline{\mathbf{P}}_e$, except that the rejection penalty of each job $J_j\in\overline{S}_i\subseteq J$ is replaced by $\tilde{w}_j=\frac{(i-1)(L'+L)\epsilon}{n}$, $i=1,2,\ldots,n/\epsilon$. Define $\overline{\mathbf{I}}'$ to be the problem instance the same as $\overline{\mathbf{I}}$ except that jobs in J are

resumable, i.e., job preemption is allowed when a job encounters a UI during processing on the machine. Clearly, the objective function value of the optimal solution to $\bar{\mathbf{I}}'$ is a lower bound of the optimal solution value to $\bar{\mathbf{P}}_e$. Based on such an insight, the major flow of our approach is as follows. We first develop a polynomial-time algorithm to determine an optimal solution to $\bar{\mathbf{I}}'$, based on which we construct a feasible solution to $\bar{\mathbf{I}}$. Based on the constructed solution to $\bar{\mathbf{I}}$, we can obtain a feasible solution to the original problem $\bar{\mathbf{P}}_e$ by replacing the scaled rejection penalties by the original ones for all rejected jobs in the solution.

To start our approach, we first develop a polynomial time algorithm to solve $\bar{\mathbf{I}}'$ optimally. Let

$$\bar{J} = J \setminus \bar{S}_0$$

and let

$$N = |\bar{J}|$$

be the number of jobs in $\bar{J}$. Without loss of generality, we assume that the jobs in $\bar{J}$ are the first N jobs in J, i.e., $\bar{J} = \{J_1, J_2, \ldots, J_N\}$. Define

$$K = \frac{(L' + L)\epsilon}{n}$$

and

$$\Delta(x) = x - \left\lfloor \frac{x}{L + L'} \right\rfloor (L + L')$$

for any $x \geqslant 0$. For $j = 1,2,\ldots,N$ and $t = 0,1,\ldots,nN/\epsilon$, define $\varphi_j(t)$ to be the objective function value of the optimal solution to the instance containing only the first j jobs of $\bar{J}$ in $\bar{\mathbf{I}}'$, such that the total modified penalty of the first j jobs is equal to tK. To develop recursive relations, consider the following two cases:

Case 1. Job J_j is rejected. In this case, we have $\varphi_j(t) = \varphi_{j-1}\left(t - \frac{\tilde{w}_j}{K}\right) + \tilde{w}_j$.

Case 2. Job J_j is accepted and processed on the machine. Note that jobs are resumable in problem instance $\bar{\mathbf{I}}'$. In this case, the makespan of the first $j-1$ jobs on the machine is equal to $\varphi_{j-1}(t) - tK$. Then, $L' - \Delta(\varphi_{j-1}(t) - tK)$ represents the remaining space of the AI corresponding to the makespan of the first $j-1$ jobs. If $L' - \Delta(\varphi_{j-1}(t) - tK) \geqslant p_j$, then the processing of J_j can be completed within the same AI, in which case the makespan of the first j jobs is equal to $\varphi_{j-1}(t) - tK + p_j$, and the total cost is equal to $\varphi_{j-1}(t) + p_j$. However, if $L' - \Delta(\varphi_{j-1}(t) - tK) > p_j$, then the processing of J_j can be completed within the next AI of the current one (note that $p_j \leqslant L'$), in which case the makespan of the first j jobs is equal to $\varphi_{j-1}(t) - tK + p_j + L$, and the total cost is equal to $\varphi_{j-1}(t) + p_j + L$.

Combining the analysis of Case 1 and Case 2, we have the following dynamic program $\mathbf{DP}_2$:

(I) Recurrence relation: For $j = 1,2,\ldots,N$ and $t = 0,1,\ldots,nN/\epsilon$,

$$\varphi_j(t) = \begin{cases} \min\left\{\varphi_{j-1}(t) + p_j, \varphi_{j-1}\left(t - \frac{\tilde{w}_j}{K}\right) + \tilde{w}_j\right\}, & \text{if } L' - \Delta(\varphi_{j-1}(t) - tK) \geqslant p_j; \\ \min\left\{\varphi_{j-1}(t) + p_j + L, \varphi_{j-1}\left(t - \frac{\tilde{w}_j}{K}\right) + \tilde{w}_j\right\}, & \text{if } L' - \Delta(\varphi_{j-1}(t) - tK) < p_j. \end{cases}$$

(II) Boundary condition:

$$\varphi_0(t) = \begin{cases} 0, & \text{if } t = 0; \\ +\infty, & \text{otherwise.} \end{cases}$$

(III) Objective: $\min\{\varphi_N(t) |\ t = 0,1,\ldots,nN/\epsilon\}$.

The time complexity of $\mathbf{DP}_2$ is bounded by $O(N^2 n/\epsilon) = O(n^3/\epsilon)$. Assume that

$$\varphi_N(\bar{t}) = \min\{\varphi_N(t) | t = 0,1,\ldots,nN/\epsilon\},$$

where $\bar{t} \in \{0,1,\ldots,nN/\epsilon\}$. Let $\bar{R}_{\bar{t}}$ be the set of all rejected jobs in the optimal solution corresponding to $\varphi_N(\bar{t})$. Based on the solution generated by $\mathbf{DP}_2$, we further assign the jobs in $\bar{S}_0$ one by one to the machine as early as possible, so as to generate a solution to $\bar{\mathbf{I}}'$. Denote such a solution to $\bar{\mathbf{I}}'$ by $\bar{\sigma}'_{\bar{t}}$, and let $\bar{Z}'_{\bar{t}}$ be its objective function value. Note that $\bar{Z}'_{\bar{t}} - \bar{t}K$ represents the makespan of all accepted jobs in solution $\bar{\sigma}'_{\bar{t}}$. It is easy to check that no job in $\bar{S}_0$ should be rejected in the optimal solution to $\bar{\mathbf{I}}'$. Thus, $\bar{\sigma}'_{\bar{t}}$ is optimal to $\bar{\mathbf{I}}'$, which further indicates that

$$\bar{Z}'_{\bar{t}} \leqslant Z^*. \tag{13}$$

Based on solution $\bar{\sigma}'_{\bar{t}}$, we now construct a feasible solution $\bar{\sigma}_{\bar{t}}$ to problem instance $\bar{\mathbf{I}}$ as follows: In $\bar{\sigma}_{\bar{t}}$, all jobs in $\bar{R}_{\bar{t}}$ are rejected and all jobs in $J \setminus \bar{R}_{\bar{t}}$ are accepted; The jobs in $J \setminus \bar{R}_{\bar{t}}$ are assigned to the machine as early as possible in the longest-processing-time-first (LPT) order. Denote the objective function value of $\bar{\sigma}_{\bar{t}}$ by $\bar{Z}_{\bar{t}}$. Note that $\bar{Z}_{\bar{t}} - \bar{t}K$ represents the makespan of all accepted jobs in solution $\bar{\sigma}_{\bar{t}}$. We have the following property.

Lemma 1. $\bar{Z}_{\bar{t}} - \bar{t}K \leqslant 2(\bar{Z}'_{\bar{t}} - \bar{t}K)$.

Proof. We assume that

$$\sum_{J_j \in J \setminus \bar{R}_{\bar{t}}} p_j = \lambda L' + \delta,$$

where $0 < \delta \leqslant L'$ and λ is a nonnegative integer. We then have

$$\bar{Z}'_{\bar{t}} - \bar{t}K = \lambda(L' + L) + \delta.$$

If $\lambda = 0$, then $\bar{Z}_{\bar{t}} - \bar{t}K = \delta = \bar{Z}'_{\bar{t}} - \bar{t}K$, and the result holds. We only consider the case with $\lambda \geqslant 1$ in the following analysis. Suppose $\bar{Z}_{\bar{t}} - \bar{t}K > 2\lambda(L' + L) + \delta$. Then, there must exist at least one job in $J \setminus \bar{R}_{\bar{t}}$ that is assigned to the $(2\lambda + 1)$th AI by the LPT rule. Assume $J_\eta \in J \setminus \bar{R}_{\bar{t}}$ to be the last job assigned to the $(2\lambda + 1)$th AI. If $p_\eta \leqslant L'/2$, then the total processing time of the jobs that have been assigned to any of the first 2λ AIs is greater than $L'/2$ as J_η fails to be assigned to any of them. If $p_\eta > L'/2$, then according to the LPT rule, any of the first 2λ AIs has been assigned a job with a processing time no less than p_η. Thus, in each of the two cases, the total processing time of the jobs assigned to the first 2λ AIs is greater than $\lambda L'$. This indicates that after assigning the jobs to the first 2λ AIs, the total processing time of the jobs remaining in $J \setminus \bar{R}_{\bar{t}}$ is less than $\lambda L' + \delta - \lambda L' = \delta \leqslant L'$. This further indicates that all of the jobs remaining in $J \setminus \bar{R}_{\bar{t}}$ can be assigned to the $(2\lambda + 1)$th AI with a job completion time less than $2\lambda(L' + L) + \delta$. This is a contradiction. Thus, we have $\bar{Z}_{\bar{t}} - \bar{t}K \leqslant 2\lambda(L' + L) + \delta \leqslant 2(\bar{Z}'_{\bar{t}} - \bar{t}K)$. □

Based on solution $\bar{\sigma}_{\bar{t}}$, we now construct a feasible solution $\sigma_{\bar{t}}$ to the original problem $\bar{\mathbf{P}}_e$ by replacing rejection penalty $\tilde{w}_j$ by the original one w_j for any job $J_j \in \bar{R}_{\bar{t}}$. Denote the objective function value of $\sigma_{\bar{t}}$ by $Z_{\bar{t}}$. We also have the following property.

Lemma 2. *If* $C^*_{max} \geqslant L' + L$, *then* $\bar{Z}_{\bar{t}} \leqslant (2 + \epsilon)Z^*$.

Proof. Note that $\bar{t}K = \sum_{J_j \in \bar{R}_{\bar{t}}} \tilde{w}_j$. Thus, if $Z^* \geqslant C^*_{max} \geqslant L' + L$, then by Lemmas 1 and (12),

$$Z_{\bar{t}} = \bar{Z}_{\bar{t}} - \bar{t}K + \sum_{J_j \in \bar{R}_{\bar{t}}} w_j \leqslant 2(\bar{Z}'_{\bar{t}} - \bar{t}K) + \sum_{J_j \in \bar{R}_{\bar{t}}} \left(\tilde{w}_j + \frac{(L + L')\epsilon}{n}\right) \leqslant 2\bar{Z}'_{\bar{t}} + \frac{N(L + L')\epsilon}{n}$$
$$\leqslant 2Z^* + (L + L')\epsilon \leqslant 2Z^* + \epsilon Z^* = (2 + \epsilon)Z^*. \qquad \square$$

By Lemma 2, if C^*_{max}, the makespan in the optimal solution, is no less than $L + L'$, then the proposed approach is a $(2 + \epsilon)$-approximation to $\bar{\mathbf{P}}_e$ with the running time bounded by $O(n^3/\epsilon)$. Note that if $C^*_{max} < L + L'$, or equivalently, $C^*_{max} \leqslant L'$, then we can apply the FPTAS to $\mathbf{P}^{(1)}$ in Case (ii) to solve $\bar{\mathbf{P}}_e$ with a worst-case bound of $1 + \epsilon$ and time complexity of $O(n/\epsilon)$. We thus have the following theorem.

Theorem 6. *There exists a $(2+\epsilon)$-approximation to problem $\overline{\mathbf{P}}_e$ with the time complexity of $O(n^3/\epsilon)$.*

Note that when job rejection is not allowed, $\overline{\mathbf{P}}_e$ is the same as the one in Ji et al. (2007). Ji et al. (2007) have shown that the worst-case bound of the LPT rule to solve the corresponding problem is 2, and the bound is tight. Therefore, our worst-case bound of $2+\epsilon$ is also tight for solving problem $\overline{\mathbf{P}}_e$.

6. Conclusion

In this paper, we study an order acceptance and scheduling model with machine availability constraints. The objective is to minimize the makespan of all accepted orders plus the total rejection penalty of all rejected orders. We develop a pseudo-polynomial algorithm for the case with a fixed number of UIs (time intervals during which the machine is not allowed to process jobs). We show that no aproximation algorithm with constant worst-case bound exists for the problem in general. We also show that if $P(U)$, the total processing time of all the jobs each of which has a processing time less than its rejection penalty, is greater than the ending point of the last UI, then a simple heuristic with a worst-case bound of 2 can be easy to develop, and such a worst-case bound is shown to be best possible. We further give an FPTAS for the case with a single UI by applying the existing FPTAS for the 0–1 Min-Knapsack Problem. Finally, we present an efficient $(2+\epsilon)$-approximation for the special case in which UIs and AIs are identical and periodic, where ϵ can be any small constant.

For future research, it is an interesting research topic of whether the worst-case bound of $2+\epsilon$ for problem $\overline{\mathbf{P}}_e$ could be improved to 2. It is also interesting to develop efficient exact algorithms to solve the proposed problem. The model with other objective functions is also worth considering.

Acknowledgments

The authors thank the AE and the anonymous referees for their helpful comments and suggestions. This research was supported in part by NSFC 71101064 and Fundamental Research Funds for the Central Universities 12JNQM003. The first author was also supported in part by Humanities and Social Sciences Research Foundation of Ministry of Education of China 13YJC630239 and Research Project of Guangdong University of Finance 12XJ03-12.

References

Bartal, Y., Leonardi, S., Marchetti-Spaccamela, A., Sgall, J., & Stougie, L. (2000). Multiprocessor scheduling with rejection. *SIAM Journal on Discrete Mathematics, 13*, 64–78.

Breit, J., Schmidt, G., & Strusevich, V. A. (2003). Non-preemptive two-machine open shop scheduling with non-availability constraints. *Mathematical Methods of Operations Research, 57*, 217–234.

Cesaret, B., Oguz, C., & Salman, F. S. (2012). A tabu search algorithm for order acceptance and scheduling. *Computers and Operations Research, 39*, 1197–1205.

Cheng, Y. S., & Sun, S. J. (2009). Scheduling linear deteriorating jobs with rejection on a single machine. *European Journal of Operational Research, 194*, 18–27.

Chen, Z.-L., & Li, C.-L. (2008). Scheduling with subcontracting options. *IIE Transactions, 40*, 1171–1184.

Engels, D. W., Karger, D. R., Kolliopoulos, S. G., Sengupta, S., Uma, R. N., & Wein, J. (2003). Techniques for scheduling with rejection. *Journal of Algorithms, 49*, 175–191.

Epstein, L., Noga, J., & Woeginger, G. J. (2002). On-line scheduling of unit time jobs with rejection: Minimizing the total completion time. *Operations Research Letters, 30*, 415–420.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. New York: Freeman.

Gawiejnowicz, S. (2007). Scheduling deteriorating jobs subject to job or machine availability constraints. *European Journal of Operational Research, 180*, 472–478.

Ghosh, J. B. (1997). Job selection in a heavily loaded shop. *Computers and Operations Research, 24*, 141–145.

Guerrero, H. H., & Kern, G. M. (1988). How to more effectively accept and refuse orders. *Production and Inventory Management Journal, 29*, 59–63.

He, Y., Ji, M., & Cheng, T. C. E. (2005). Single machine scheduling with a restricted rate-modifying activity. *Naval Research Logistics, 52*, 361–369.

Hoogeveen, H., Skutella, M., & Woeginger, G. J. (2003). Preemptive scheduling with rejection. *Mathematical Programming, 94*, 361–374.

Ibarra, O., & Kim, C. E. (1975). Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM, 22*, 463–468.

Ji, M., He, Y., & Cheng, T. C. E. (2006). Scheduling linear deteriorating jobs with an availability constraint on a single machine. *Theoretical Computer Science, 362*, 115–126.

Ji, M., He, Y., & Cheng, T. C. E. (2007). Single-machine scheduling with periodic maintenance to minimize makespan. *Computers and Operations Research, 34*, 176–1770.

Kacem, I. (2009). Approximation algorithms for the makespan minimization with positive tails on a single machine with a fixed non-availability interval. *Journal of Global Optimization, 17*, 117–133.

Kellerer, H., & Pferschy, U. (1999). A new fully polynomial time approximation scheme for the knapsack problem. *Journal of Combinatorial Optimization, 3*, 59–71.

Kellerer, H., & Pferschy, U. (2004). Improved dynamic programming in connection with an FPTAS for the knapsack problem. *Journal of Combinatorial Optimization, 8*, 5–11.

Kellerer, H., Pferschy, U., & Pisinger, D. (2004). *Knapsack Problems*. Berlin: Springer.

Kellerer, H., & Strusevich, V. A. (2013). Fast approximation schemes for Boolean programming and scheduling problems related to positive convex half-product. *European Journal of Operational Research, 228*, 24–32.

Lee, C. Y. (1996). Machine scheduling with an availability constraint. *Journal of Global Optimization, 9*, 395–416.

Lee, I. S., & Sung, C. S. (2008). Minimizing due date related measures for a single machine scheduling problem with outsourcing allowed. *European Journal of Operational Research, 186*, 931–952.

Lee, I. S., & Sung, C. S. (2008). Single machine scheduling with outsourcing allowed. *International Journal of Production Economics, 111*, 623–634.

Lu, L. F., Cheng, T. C. E., Yuan, J. J., & Zhang, L. Q. (2009). Bounded single-machine parallel-batch scheduling with release dates and rejection. *Computers and Operation Research, 36*, 2748–2751.

Lu, L. F., Zhang, L. Q., & Yuan, J. J. (2008). The unbounded parallel batch machine scheduling with release dates and rejection to minimize makespan. *Theoretical Computer Science, 396*, 283–289.

Ma, Y., Chu, C. B., & Zuo, C. R. (2010). A survey of scheduling with deterministic machine availability constraints. *Computers and Industrial Engineering, 58*, 199–211.

Oguz, C., Salman, F. S., & Yalcin, Z. B. (2010). Order acceptance and scheduling decisions in make-to-order systems. *International Journal of Production Economics, 125*, 200–211.

Qi, X. (2008). Coordinated logistics scheduling for in-house production and outsourcing. *IEEE Transactions on Automation Science and Engineering, 5*, 188–192.

Qi, X. (2009). Two-stage production scheduling with an option of outsourcing from a remote supplier. *Journal of Systems Science and Systems Engineering, 18*, 1–15.

Qi, X. (2011). Outsourcing and production scheduling for a two-stage flow shop. *International Journal of Production Economics, 129*, 43–50.

Rom, W. O., & Slotnick, S. A. (2009). Order acceptance using genetic algorithms. *Computers and Operations Research, 36*, 1758–1767.

Schmidt, G. (2000). Scheduling with limited machine availability. *European Journal of Operational Research, 121*, 1–15.

Seiden, S. S. (2001). Preemptive multiprocessor scheduling with rejection. *Theoretical Computer Science, 2621*, 437–458.

Shabtay, D., Gaspar, N., & Kaspi, M. (2013). A survey on offline scheduling with rejection. *Journal of Scheduling, 16*, 3–28.

Slotnick, S. A. (2011). Order acceptance and scheduling: A taxonomy and review. *European Journal of Operational Research, 212*, 1–11.

Slotnick, S. A., & Morton, T. E. (1996). Selecting jobs for a heavily loaded shop with lateness penalties. *Computers and Operations Research, 23*, 131–140.

Slotnick, S. A., & Morton, T. E. (2007). Order acceptance with weighted tardiness. *Computers and Operations Research, 34*, 3029–3042.

Talla Nobibon, F., & Leus, R. (2011). Exact algorithms for a generalization of the order acceptance and scheduling problem in a single-machine environment. *Computers and Operations Research, 38*, 367–378.

Wu, C. C., & Lee, W. C. (2003). Scheduling linear deteriorating jobs to minimize makespan with an availability constraint on a single machine. *Information Processing Letters, 87*, 89–93.

Yang, B., & Geunes, J. (2007). A single resource scheduling problem with job-selection flexibility, tardiness costs and controllable processing times. *Computers and Industrial Engineering, 53*, 420–432.

Yuan, J. J., She, L., & Ou, J. W. (2008). Single machine scheduling with forbidden intervals and job delivery times. *Asia-Pacific Journal of Operational Research, 25*, 317–325.

Zhang, L. Q., Lu, L. F., & Yuan, J. J. (2009). Single machine scheduling with release dates and rejection. *European Journal of Operational Research, 198*, 975–978.

Zhang, L. Q., Lu, L. F., & Yuan, J. J. (2010). Single-machine scheduling under the job rejection constraint. *Theoretical Computer Science, 411*, 1877–1882.