Article

Single-Machine Scheduling with Rejection and an Operator Non-Availability Interval

Lili Zuo ¹, Zhenxia Sun ¹, Lingfa Lu ^{1,*} and Liqi Zhang ²¹ School of Mathematics and Statistics, Zhengzhou University, Zhengzhou 450001, China² College of Information and Management Science, Henan Agricultural University, Zhengzhou 450002, China

* Correspondence: lulingfa@zzu.edu.cn

Received: 8 July 2019; Accepted: 24 July 2019; Published: 26 July 2019

Abstract: In this paper, we study two scheduling problems on a single machine with rejection and an operator non-availability interval. In the operator non-availability interval, no job can be started or be completed. However, a *crossover* job is allowed such that it can be started before this interval and completed after this interval. Furthermore, we also assume that job rejection is allowed. That is, each job is either accepted and processed in-house, or is rejected by paying a rejection cost. Our task is to minimize the sum of the makespan (or the total weighted completion time) of accepted jobs and the total rejection cost of rejected jobs. For two scheduling problems with different objective functions, by borrowing the previous algorithms in the literature, we propose a pseudo-polynomial-time algorithm and a fully polynomial-time approximation scheme (FPTAS), respectively.

Keywords: scheduling with rejection; machine non-availability; operator non-availability; dynamic programming; FPTAS

1. Introduction

In this section, we introduce some models on scheduling with (machine or operator) non-availability intervals, scheduling with rejection, and scheduling with rejection and non-availability intervals, respectively.

1.1. Scheduling with Non-Availability Intervals

In most scheduling problems, it is assumed that the machines are available at all times. However, in some industrial settings, the assumption might not be true. Some machines or the operator might be unavailable in some time intervals. Recently, some researchers have studied some scheduling problems with the non-availability intervals. Two models of the non-availability interval were studied mainly: one is the machine non-availability (MNA) intervals due to the machine maintenances and the other is the operator non-availability (ONA) intervals because the operator is resting from work. The difference between MNA intervals and ONA intervals is that a crossover job can exist in the ONA interval. However, no job can be processed in the MNA interval.

To the best of our knowledge, the earliest scheduling problem with MNA intervals was studied by Schmidt [1]. He considered a parallel-machine scheduling problem in which each machine has different MNA intervals. The task is to find a feasible preemptive schedule if it exists. A polynomial-time algorithm is presented for the above problem. We first introduce some single-machine scheduling problems with an MNA interval (a, b) . The corresponding problem can be denoted by $1|MNA(a, b)|f$, where “ $MNA(a, b)$ ” means that there is an MNA interval (a, b) and f is the objective function to be minimized. For problem $1|MNA(a, b)|\sum C_j$, Adiri et al. [2] proved that the problem is NP-hard and then presented a $\frac{5}{4}$ -approximation algorithm. For problem $1|MNA(a, b)|C_{\max}$, Lee [3] showed that this problem is binary NP-hard and then provided a $\frac{4}{3}$ -approximation algorithm. For problem

$1|MNA(a, b)|L_{\max}$, Kacem [4] designed a $\frac{3}{2}$ -approximation algorithm and a fully polynomial-time approximation scheme (FPTAS). If there are $k \geq 2$ MNA intervals $(a_1, b_1), (a_2, b_2), \dots, (a_k, b_k)$ on the machine, the corresponding problem $1|MNA(a_i, b_i)|C_{\max}$ is strongly NP-hard (see [3]) when k is arbitrary. Breit et al. [5] showed that, for any $\rho \geq 1$ and $k \geq 2$, there is no ρ -approximation algorithm for problem $1|MNA(a_i, b_i)|C_{\max}$ unless $P = NP$.

When there are $m \geq 2$ parallel machines $M_1, \dots, M_m$ and each machine M_i has an MNA interval (a_i, b_i) , the corresponding problem is denoted by $Pm|MNA(a_i, b_i)|f$. For problem $Pm|MNA(0, b_i)|C_{\max}$, i.e., each machine M_i has a machine release time b_i , Lee [6] provided a modified LPT (MLPT) algorithm with a tight approximation ratio $\frac{4}{3}$. Kellerer [7] improved this bound $\frac{4}{3}$ to $\frac{5}{4}$ by a dual approximation algorithm using a bin packing approach. For problem $Pm|MNA(0, b_i)|\sum C_j$, Schmidt [8] showed that the SPT rule is optimal. Lee [6] also studied the problem $Pm|MNA(a_i, b_i)|C_{\max}$ with the assumption that one machine is always available. He showed that the approximation ratios of LS (List Scheduling) and LPT are m and $\frac{m+1}{2}$, respectively. Furthermore, Liao et al. [9] considered the same problem with $m = 2$ and developed exact algorithms based on the TMO algorithm for problem $P2||C_{\max}$. Aggoune [10] studied the flow-shop scheduling problem with several MNA intervals on each machine. A heuristic algorithm is provided to approximately solve this problem. Burdett and Kozan [11] also addressed some MNA intervals in railway scenarios. They introduced new fixed jobs for the MNA intervals. Some constructive heuristics and meta-heuristic algorithms were proposed in this paper. For more new models and results about this topic, the reader is referred to the survey by Ma et al. [12].

Brauner et al. [13] first studied the scheduling problems with an ONA interval. Similarly, this scheduling model can be denoted by $1|ONA(a, b)|f$. For problem $1|ONA(a, b)|C_{\max}$, Brauner et al. [13] proved that it is binary NP-hard and provided an FPTAS. For problem $1|ONA(a, b)|L_{\max}$, Kacem et al. [14] proposed an FPTAS by borrowing the FPTAS for problem $1|MNA(a, b)|L_{\max}$. Chen et al. [15] considered the problem $1|ONA(a, b)|\sum C_j$ and presented a $\frac{20}{17}$ -approximation algorithm. Wan and Yuan [16] further considered the problem $1|ONA(a, b)|\sum w_j C_j$. They designed a pseudo-polynomial-time dynamic programming (DP) algorithm and then converted the DP algorithm into an FPTAS. Burdett et al. [17] considered the flexible job shop scheduling with operators (FJSOP) for coal export terminals. A hybrid meta-heuristic and a lot of numerical testings were designed for the above problem.

1.2. Scheduling with Rejection

In many practical cases, processing all jobs may occur high inventory or tardiness costs. However, rejecting some jobs can save time and reduce costs. When a job is rejected, a corresponding rejection cost is required. The decision maker needs to determine which jobs should be accepted (and a feasible schedule for accepted jobs), and which jobs should be rejected, such that the production cost and the total rejection cost are minimized. Thus, both from the practical and theoretical point of view, scheduling models with rejection are very interesting. In addition, an important application also occurs in scheduling with outsourcing. If the outsourcing cost is treated as the rejection cost, scheduling with rejection and scheduling with outsourcing are in fact equivalent.

Scheduling models with rejection were first introduced by Bartal et al. [18]. They considered several off-line and on-line scheduling problems on m parallel machines. The task is to minimize the sum of the makespan of accepted jobs and the total rejection cost of rejected jobs. For the on-line version, they designed an on-line algorithm with the best-possible competitive ratio of 2.618. For the off-line version, they provided an FPTAS when m is fixed, and a PTAS when m is arbitrary.

Next, we only introduce some results on the single-machine scheduling with rejection. The corresponding problem can be denoted by $1|rej|f + e(R)$, where f is the objective function on the set A of accepted jobs and $e(R)$ is the total rejection cost on the set R of rejected jobs. For problem $1|r_j, rej|C_{\max} + e(R)$, Cao and Zhang [19] proved that this problem is NP-hard and designed a PTAS. However, they also pointed out that it is open whether this problem is ordinary or

strongly NP-hard. Zhang et al. [20] showed that this problem is binary NP-hard by providing two different pseudo-polynomial-time algorithms. Finally, they also provided a 2-approximation algorithm and an FPTAS for the above problem. For problem $1|\text{rej}|L_{\max} + e(R)$, Sengupta [21] proved that this problem is binary NP-hard. He also proposed two dynamic programming algorithms and converted one of the two algorithms into an FPTAS. Engels et al. [22] studied the problem $1|\text{rej}|\sum w_j C_j + e(R)$. They showed that this problem is binary NP-hard and then provided an FPTAS. They also showed that, when $w_j = 1$, the problem $1|\text{rej}|\sum C_j + e(R)$ is polynomial-time solvable. Recently, Shabtay et al. [23] presented a comprehensive survey for the off-line scheduling problems with rejection. For other models and results on scheduling with rejection, the reader is referred to the survey by Shabtay et al. [23].

1.3. Scheduling with Rejection and Non-Availability Intervals

There are only two articles which considered “scheduling with rejection” and “machine non-availability intervals” together. Zhong et al. [24], and Zhao and Tang [25] considered the problems $1|\text{MNA}(a, b), \text{rej}|C_{\max} + e(R)$ and $1|\text{MNA}(a, b), \text{rej}|\sum w_j C_j + e(R)$, respectively. Both of them presented a pseudo-polynomial dynamic programming algorithm and an FPTAS for the corresponding problem. In addition, Li and Chen [26] investigated several scheduling problems with rejection and a deteriorating maintenance activity on a single machine. In their model, the starting time of the maintenance activity (non-availability intervals) is not fixed and the duration is a linear increasing function of its starting time. Some (pseudo-)polynomial-time algorithms are presented for some different objective functions. However, to the best of our knowledge, no article considered “scheduling with rejection” and “operator non-availability intervals” simultaneously. In this paper, we are the first to consider scheduling with rejection and an operator non-availability interval.

2. Problem Formulation

The single-machine scheduling with rejection and an operator non-availability interval can be stated formally as follows. There are n jobs $J_1, J_2, \dots, J_n$ and a single machine. Each job J_j is available at time 0 and has a processing time p_j , a weight w_j and a rejection cost e_j . Each job J_j is either rejected and the rejection cost e_j has to be paid, or accepted and then processed non-preemptively on the machine. There is an operator non-availability interval (a, b) on the machine, where we assume that $0 < a < b$. This implies that, in any feasible schedule π , no accepted job J_j can be started or be completed in the interval (a, b) . However, a crossover job is allowed such that it can start before this interval and complete after this interval. Without loss of generality, we assume that all the parameters a, b, p_j, w_j and e_j are non-negative integers. Let A and R be the sets of accepted jobs and rejected jobs, respectively. Denote by $C_{\max} = \max\{C_j : J_j \in A\}$, $\sum w_j C_j = \sum_{J_j \in A} w_j C_j$ and $e(R) = \sum_{J_j \in R} e_j$ the makespan of accepted jobs, the total weighted completion time of accepted jobs and the total rejection cost of rejected jobs, respectively. Our task is to find a feasible schedule such that $C_{\max} + e(R)$ or $\sum w_j C_j + e(R)$ is minimized. By using the notation for scheduling problems, the corresponding problems are denoted by $1|\text{ONA}(a, b), \text{rej}|C_{\max} + e(R)$ and $1|\text{ONA}(a, b), \text{rej}|\sum w_j C_j + e(R)$, respectively.

Two similar problems related to the above problems are $1|\text{MNA}(a, b), \text{rej}|C_{\max} + e(R)$ and $1|\text{MNA}(a, b), \text{rej}|\sum w_j C_j + e(R)$. Zhong et al. [24] and Zhao and Tang [25] presented a pseudo-polynomial dynamic programming algorithm and an FPTAS for the corresponding problem, respectively. In this paper, by borrowing the algorithms in [24,25], we also presented a pseudo-polynomial dynamic programming algorithm and an FPTAS for the problems $1|\text{ONA}(a, b), \text{rej}|C_{\max} + e(R)$ and $1|\text{ONA}(a, b), \text{rej}|\sum w_j C_j + e(R)$, respectively.

3. Pseudo-Polynomial-Time Algorithms

Brauner et al. [13] and Chen et al. [15] showed that problems $1|\text{ONA}(a, b)|C_{\max}$ and $1|\text{ONA}(a, b)|\sum C_j$ are NP-hard. Thus, two problems $1|\text{ONA}(a, b), \text{rej}|C_{\max} + e(R)$ and $1|\text{ONA}(a, b), \text{rej}|\sum w_j C_j + e(R)$ are also NP-hard. In this section, we show that the above problems can

be solved in pseudo-polynomial time. That is, both of the problems, $1|ONA(a, b), rej|C_{\max} + e(R)$ and $1|ONA(a, b), rej|\sum w_j C_j + e(R)$, are binary NP-hard.

For problems $1|MNA(a, b), rej|C_{\max} + e(R)$ and $1|MNA(a, b), rej|\sum w_j C_j + e(R)$, Zhong et al. [24] and Zhao and Tang [25] presented a pseudo-polynomial dynamic programming algorithm, respectively. By enumerating the crossover job and its starting time, we show that problem $1|ONA(a, b), rej|f + e(R)$ can be decomposed into many subproblems of $1|MNA(a, b), rej|f + e(R)$. Consequently, by borrowing the algorithms in [24,25], we also presented a pseudo-polynomial algorithm for problems $1|ONA(a, b), rej|C_{\max} + e(R)$ and $1|ONA(a, b), rej|\sum w_j C_j + e(R)$, respectively.

Lemma 1. *If algorithm A^{MNA} solves problem $1|MNA(a, b), rej|f + e(R)$ in $O(T)$ time, then Algorithm 1 yields an optimal schedule for problem $1|ONA(a, b), rej|f + e(R)$ in $O(naT)$ time.*

Algorithm 1 A^{ONA}

Step 1: Let A^{MNA} be a dynamic programming algorithm for problem $1|MNA(a, b), rej|f + e(R)$, where $f \in \{C_{\max}, \sum w_j C_j\}$.

Step 2: Applying algorithm A^{MNA} to the problem $1|MNA(a, b), rej|f + e(R)$, we can obtain a schedule π_0 .

Step 3: For each job J_j with $j = 1, \dots, n$ and each integer t with $0 \leq t \leq a$ and $t + p_j \geq b$, we first schedule J_j in time interval $[t, t + p_j]$. Set $a' = t$, $b' = t + p_j$ and instance $I' = \{J_1, \dots, J_n\} \setminus J_j$. Applying algorithm A^{MNA} to the instance I' of problem $1|MNA(a', b'), rej|f + e(R)$, we can obtain a schedule $\pi(j, t)$.

Step 4: Choose the schedule among all schedules in $\{\pi_0\} \cup \{\pi(j, t) : 1 \leq j \leq n, 0 \leq t \leq a \text{ and } t + p_j \geq b\}$ with the smallest objective value.

Proof. Firstly, we show that Algorithm 1 yields an optimal schedule for problem $1|ONA(a, b), rej|f + e(R)$. Let π^* be an optimal schedule for problem $1|ONA(a, b), rej|f + e(R)$. We distinguish two cases into our discussion.

Case 1: No crossover job exist in π^* .

In this case, interval (a, b) is completely forbidden. That is, an ONA interval is equivalent to an MNA interval. Thus, in this case, π_0 is also an optimal schedule for problem $1|ONA(a, b), rej|f + e(R)$.

Case 2: There is a crossover job in π^* .

Let J_j be the crossover job in π^* and let t be the starting time of J_j in π^* . Clearly, we have $0 \leq t \leq a$ and $t + p_j \geq b$. In this case, no job in $I' = \{J_1, \dots, J_n\} \setminus J_j$ can be processed in the interval $(t, t + p_j)$. Set $a' = t$ and $b' = t + p_j$. The remaining problem is equivalent to the instance I' of the problem $1|MNA(a', b'), rej|f + e(R)$. Thus, in this case, $\pi(j, t)$ is also an optimal schedule for problem $1|ONA(a, b), rej|f + e(R)$.

From the above discussions, Algorithm 1 always yields an optimal schedule for problem $1|ONA(a, b), rej|f + e(R)$. Note that algorithm A^{MNA} is called at most $O(na)$ times. Thus, the time complexity of Algorithm 1 is exactly $O(naT)$. $\square$

Note that Zhong et al. [24] presented an $O(n \sum_{j=1}^n p_j)$ -time dynamic programming algorithm for problem $1|MNA(a, b), rej|C_{\max} + e(R)$. Zhao and Tang [25] presented an $O(na \sum_{j=1}^n p_j)$ -time algorithm for problem $1|MNA(a, b), rej|\sum w_j C_j + e(R)$. Thus, we have the following corollaries.

Corollary 1. Algorithm 1 solves problem $1|ONA(a, b), rej|C_{\max} + e(R)$ in $O(n^2 a \sum_{j=1}^n p_j)$ time and solves problem $1|ONA(a, b), rej| \sum w_j C_j + e(R)$ in $O(n^2 a^2 \sum_{j=1}^n p_j)$ time.

Example 1. A simple example for problem $1|ONA(a, b), rej|C_{\max} + e(R)$ is constructed as follows: Given a positive number $k > 0$, we have three jobs J_1, J_2, J_3 with $(p_1, e_1) = (2k, k)$, $(p_2, e_2) = (4k, 2k)$ and $(p_3, e_3) = (7k, +\infty)$. We also assume that there is a single machine with an ONA interval (a, b) , where $a = 4k$ and $b = 9k$. Note that $p_1 < p_2 < b_k - a_k = 5k < p_3$. Thus, J_3 is the unique candidate for the crossover job. Note further that $e_3 = +\infty$. Thus, J_3 must be accepted in any optimal schedule. We distinguish two cases in our discussion.

Case 1: J_3 is not a crossover job.

In this case, since $p_3 = 7k > a = 4k$, J_3 must be processed at or after time $b = 9k$. Thus, the optimal schedule is to process J_2 and J_3 in $[0, 4k]$ and $[9k, 16k]$, respectively, and reject J_1 . That is, the $C_{\max} + e(R)$ value is $16k + k = 17k$.

Case 2: J_3 is the crossover job and J_3 starts its processing at time t .

Since $t \in [0, 4k]$ and $t + p_3 \geq b = 9k$, we have $2k \leq t \leq 4k$. When $t \in [2k, 4k)$, only J_1 can be processed before J_3 . Thus, the $C_{\max} + e(R)$ value is at least $t + 7k + 2k \geq 11k$. The minimum value $C_{\max} + e(R) = 11k$ can be reached by processing J_1 and J_3 in $[0, 9k]$ and reject J_2 . When $t = 4k$, only one job between J_1 and J_2 can be processed before J_3 . Thus, the $C_{\max} + e(R)$ value is at least $4k + 7k + k \geq 12k$. The minimum value $C_{\max} + e(R) = 12k$ can be reached by processing J_2 and J_3 in $[0, 11k]$ and reject J_1 .

By combining all cases, the optimal $C_{\max} + e(R)$ value is $11k$, which can be reached by processing J_1 and J_3 in $[0, 9k]$ and reject J_2 .

4. Approximation Schemes

In this section, by borrowing the FPTASs in [24,25], we also presented an FPTAS for problems $1|ONA(a, b), rej|C_{\max} + e(R)$ and $1|ONA(a, b), rej| \sum w_j C_j + e(R)$, respectively. Given an $\epsilon = \frac{1}{E}$ for some positive integer E , we set $t_i = \epsilon a$ for each $i = 1, 2, \dots, E$. To propose an FPTAS, we delay the starting of the crossover job slightly such that the crossover job starts its processing at some time t_i . Such a delay may increase the objective value by $1 + \epsilon$ times, we say that it produces a $(1 + \epsilon)$ -loss.

Lemma 2. If a crossover job exists in an optimal schedule, with a $(1 + \epsilon)$ -loss, we can assume that the crossover job starts its processing at some time t_i .

Proof. Let π^* be an optimal schedule for problem $1|ONA(a, b), rej|C_{\max} + e(R)$ or problem $1|ONA(a, b), rej| \sum w_j C_j + e(R)$. Let A^* and R^* be the set of accepted jobs and the set of rejected jobs in π^* , respectively. In addition, we also assume that $J_j \in A^*$ is the j th processed job in π^* . Without loss of generality, we assume that $J_k \in A^*$ is the crossover job in π^* . Furthermore, we assume that the starting time of J_k is t with $0 \leq t \leq a$ and $t + p_k \geq b$. If $t \neq t_i$ for each $i = 1, 2, \dots, E$, then there is some i with $1 \leq i \leq E$ such that $t_{i-1} < t < t_i$, where $t_0 = 0$. By delaying the processing of $J_k, J_{k+1}, \dots, J_{|A^*|}$ by a length of $t_i - t$, we can obtain a schedule π . Note that $t_i \leq a$ and $t_i + p_k \geq t + p_k \geq b$. Thus, π is feasible for problems $1|ONA(a, b), rej|C_{\max} + e(R)$ and $1|ONA(a, b), rej| \sum w_j C_j + e(R)$. Note further that $t_i - t < t_i - t_{i-1} = \epsilon a$ and $C_j(\pi^*) \geq b > a$ for each $j = k, \dots, |A^*|$. Thus, we have $C_j(\pi) = C_j(\pi^*)$ for each $j = 1, \dots, k-1$ and $C_j(\pi) = C_j(\pi^*) + (t_i - t) \leq C_j(\pi^*) + \epsilon a \leq C_j(\pi^*) + \epsilon C_j(\pi^*) = (1 + \epsilon)C_j(\pi^*)$ for each $j = k, \dots, |A^*|$. It follows that

$$C_{\max}(\pi) \leq (1 + \epsilon)C_{\max}(\pi^*) \text{ and } \sum w_j C_j(\pi) \leq (1 + \epsilon) \sum w_j C_j(\pi^*).$$

Notice that both of the total rejection cost in π and π^* are $e(R^*)$. Thus, we can conclude that the objective value is increased by at most $1 + \epsilon$ times. This completes the proof of Lemma 2. $\square$

Next, based on Lemma 2, we propose an FPTAS for problems $1|ONA(a, b), rej|C_{\max} + e(R)$ and $1|ONA(a, b), rej|\sum w_j C_j + e(R)$.

Lemma 3. If algorithm A_ϵ^{MNA} is an FPTAS for problem $1|MNA(a, b), rej|f + e(R)$ with the time complexity of $O(T)$, then Algorithm 2 is also an FPTAS for problem $1|ONA(a, b), rej|f + e(R)$ with the time complexity of $O(\frac{nT}{\epsilon})$.

Algorithm 2 A_ϵ^{ONA}

Step 1: Let A_ϵ^{MNA} be an FPTAS for problem $1|MNA(a, b), rej|f + e(R)$, where $f \in \{C_{\max}, \sum w_j C_j\}$.

Step 2: Applying algorithm A_ϵ^{MNA} to the problem $1|MNA(a, b), rej|f + e(R)$, we can obtain a schedule π_0 .

Step 3: For each job J_j with $j = 1, \dots, n$ and each integer t_i with $0 \leq t_i \leq a$ and $t_i + p_j \geq b$, we first schedule J_j in time interval $[t_i, t_i + p_j]$. Set $a' = t_i$, $b' = t_i + p_j$ and instance $I' = \{J_1, \dots, J_n\} \setminus J_j$. Applying algorithm A_ϵ^{MNA} to the instance I' of problem $1|MNA(a', b'), rej|f + e(R)$, we can obtain a schedule $\pi(j, t_i)$.

Step 4: Choose the schedule among all schedules in $\{\pi_0\} \cup \{\pi(j, t_i) : 1 \leq j \leq n, 0 \leq t_i \leq a \text{ and } t_i + p_j \geq b\}$ with the smallest objective value.

Proof. Firstly, we prove that Algorithm 2 is an FPTAS for $1|ONA(a, b), rej|f + e(R)$. Let π^* be an optimal schedule for problem $1|ONA(a, b), rej|f + e(R)$. Let Z and Z^* be the objective values obtained from Algorithm 2 and the optimal schedule π^* , respectively. We also distinguish two cases in our discussion.

Case 1: No crossover job exists in π^* .

In this case, interval (a, b) is completely forbidden. That is, an ONA interval is equivalent to an MNA interval. Thus, we have $Z \leq Z(\pi_0) \leq (1 + \epsilon)Z(\pi^*) = (1 + \epsilon)Z^*$.

Case 2: There is a crossover job in π^* .

Let J_j be the crossover job in π^* and let t be the starting time of J_j in π^* . Clearly, we have $0 \leq t \leq a$ and $t + p_j \geq b$. In this case, no job in $I' = \{J_1, \dots, J_n\} \setminus J_j$ can be processed in the interval $(t, t + p_j)$. Set $a' = t$ and $b' = t + p_j$. Thus, the remaining problem is equivalent to the instance I' of the problem $1|MNA(a', b'), rej|f + e(R)$. Let $Z^*(j, t)$ be the optimal objective value for problem $1|ONA(a, b), rej|f + e(R)$ under the constraint in which J_j is the crossover job and J_j starts its processing at time t . Note that algorithm A_ϵ^{MNA} is an FPTAS for problem $1|MNA(a, b), rej|f + e(R)$. Thus, we have $Z(\pi(j, t_i)) \leq (1 + \epsilon)Z^*(j, t_i)$. Furthermore, by Lemma 2, we also have $Z^*(j, t_i) \leq (1 + \epsilon)Z^*(j, t) = (1 + \epsilon)Z^*$. It follows that

$$Z \leq Z(\pi(j, t_i)) \leq (1 + \epsilon)Z^*(j, t_i) \leq (1 + \epsilon)^2 Z^*(j, t) = (1 + \epsilon)^2 Z^*.$$

From the above discussions, Algorithm 2 is an FPTAS for $1|ONA(a, b), rej|f + e(R)$. Note that algorithm A_ϵ^{MNA} is called at most $O(\frac{n}{\epsilon})$ times. Thus, the time complexity of Algorithm 2 is exactly $O(\frac{nT}{\epsilon})$. $\square$

Note that Zhong et al. [24] presented an FPTAS with the time complexity of $O(\frac{n}{\epsilon})$ for problem $1|MNA(a, b), rej|C_{\max} + e(R)$. Zhao and Tang [25] presented an FPTAS with the time complexity

$O(\frac{n^4 L^5}{\epsilon^4})$ for problem $1|MNA(a, b), rej| \sum w_j C_j + e(R)$, where $L = \log(\max\{n, \frac{1}{\epsilon}, b, \max e_j, (\max w_j) \cdot (\max p_j)\})$. Thus, we have the following corollaries.

Corollary 2. Algorithm 1 is an FPTAS for problems $1|ONA(a, b), rej|C_{\max} + e(R)$ and $1|ONA(a, b), rej| \sum w_j C_j + e(R)$ with the time complexities of $O(\frac{n^2}{\epsilon^2})$ and $O(\frac{n^5 L^5}{\epsilon^5})$, respectively.

5. Conclusions and Future Work

In this paper, we are the first to consider scheduling with rejection and an operator non-availability interval simultaneously. The objective is to minimize the sum of the makespan (or the total weighted completion time) of the accepted jobs and the total rejection cost of the rejected jobs. Firstly, we build the relation between problem $1|MNA(a, b), rej|f + e(R)$ and problem $1|ONA(a, b), rej|f + e(R)$, where $f \in \{C_{\max}, \sum w_j C_j\}$. That is, by enumerating the crossover job and its starting time, problem $1|ONA(a, b), rej|f + e(R)$ can be decomposed into many subproblems of $1|MNA(a, b), rej|f + e(R)$. Consequently, by borrowing the previous algorithms for problem $1|MNA(a, b), rej|f + e(R)$, we provide a pseudo-polynomial-time algorithm and an FPTAS for problem $1|ONA(a, b), rej|f + e(R)$, respectively.

When there are $k \geq 2$ MNA or ONA intervals $(a_1, b_1), (a_2, b_2), \dots, (a_k, b_k)$ on the machine, the corresponding problem is strongly NP-hard (see [3]) when k is arbitrary. However, when k is a fixed constant, it is possible to propose a pseudo-polynomial-time algorithm with a larger time complexity. Breit et al. [5] showed that, for any $\rho \geq 1$ and $k \geq 2$, there is no ρ -approximation algorithm for problem $1|MNA(a_i, b_i)|C_{\max}$ unless $P = NP$. It is easy to verify that the inapproximability result still holds when there are $k \geq 2$ MNA or ONA intervals on the machine and b_k is sufficiently large.

Note that, for problem $1|ONA(a, b), rej| \sum w_j C_j + e(R)$, the time complexity of the proposed FPTAS is $O(\frac{n^5 L^5}{\epsilon^5})$. That is, the running time is very large and it is not strongly polynomial. Thus, an interesting problem is to design a faster (strongly polynomial) FPTAS for problem $1|ONA(a, b), rej| \sum w_j C_j + e(R)$. Note further that, when there are $k \geq 2$ MNA or ONA intervals and each MNA or ONA interval has a bounded length, it is possible to design an effective approximation algorithm. Thus, another interesting problem is to propose some approximation algorithms for the problems with multiple MNA or ONA intervals. Finally, it is also interesting to consider other objective functions such as $L_{\max} + e(R)$ and other machine setting such as parallel machines or shop machines.

Author Contributions: Conceptualization, methodology and writing-original manuscript, L.Z. (Lili Zuo) and Z.S.; project management, supervision and writing-review, L.L.; investigation, formal analysis and editing, L.Z. (Liqi Zhang).

Funding: This research was funded by the National Natural Science Foundation of China under grant number 11771406 and 11571321.

Acknowledgments: We are grateful to the Associate Editor and three anonymous reviewers for their valuable comments, which helped us significantly improve the quality of our paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Schmidt, G. Scheduling independent tasks with deadlines on semi-identical processors. *J. Oper. Res. Soc.* **1984**, *39*, 271–277. [CrossRef]
- Adiri, I.; Bruno, J.; Frostig, E.; Kan, A.H.G.R. Single machine flowtime scheduling with a single breakdown. *Acta Inform.* **1989**, *26*, 679–696. [CrossRef]
- Lee, C.Y. Machine scheduling with an availability constraints. *J. Glob. Optim.* **1996**, *9*, 363–382. [CrossRef]
- Kacem, I. Approximation algorithms for the makespan minimization with positive tails on a single machine with a fixed nonavailability interval. *J. Comb. Optim.* **2009**, *17*, 117–133. [CrossRef]
- Breit, J.; Schmidt, G.; Strusevich, V.A. Non-preemptive two-machine open shop scheduling with non-availability constraints. *Math. Methods Oper. Res.* **2003**, *57*, 217–234. [CrossRef]

6. Lee, C.Y. Parallel machine scheduling with non-simultaneous machine available time. *Discret. Appl. Math.* **1991**, *30*, 53–61. [\[CrossRef\]](#)
7. Kellerer, H. Algorithms for multiprocessor scheduling with machine release times. *IIE Trans.* **1998**, *30*, 991–999. [\[CrossRef\]](#)
8. Schmidt, G. Scheduling with limited machine availability. *Eur. J. Oper. Res.* **2000**, *121*, 1–15. [\[CrossRef\]](#)
9. Liao, C.J.; Shyur, D.L.; Lin, C.H. Makespan minimization for two parallel machines with an availability constraint. *Eur. J. Oper.* **2005**, *160*, 445–456. [\[CrossRef\]](#)
10. Aggoune, R. Minimising the makespan for the flow shop scheduling problem with availability constraints. *Eur. J. Oper. Res.* **2004**, *153*, 534–543. [\[CrossRef\]](#)
11. Burdett, R.L.; Kozan, E. Techniques for inserting additional trains into existing timetables. *Transp. Res. Part B* **2009**, *43*, 821–836. [\[CrossRef\]](#)
12. Ma, Y.; Chu, C.; Zuo, C. A survey of scheduling with deterministic machine availability constraints. *Comput. Ind. Eng.* **2010**, *58*, 199–211. [\[CrossRef\]](#)
13. Brauner, N.; Frinke, G.; Lebacque, V.; Rapine, C.; Potts, C.; Struservich, V. Operator nonavailability periods. *4OR Q. J. Oper. Res.* **2009**, *7*, 239–253. [\[CrossRef\]](#)
14. Kacem, I.; Kellerer, H.; Seifaddini, M. Efficient approximation schemes for the maximum lateness minimization on a single machine with a fixed operator or machine non-availability interval. *J. Comb. Optim.* **2016**, *32*, 970–981. [\[CrossRef\]](#)
15. Chen, Y.; Zhang, A.; Tan, Z.Y. Complexity and approximation of single machine scheduling with an operator non-availability period to minimize total completion time. *Inf. Sci.* **2013**, *251*, 150–163. [\[CrossRef\]](#)
16. Wan, L.; Yuan, J.J. Single-machine scheduling with operator non-availability to minimize total weighted completion time. *Inf. Sci.* **2018**, *445*, 1–5. [\[CrossRef\]](#)
17. Burdett, R.L.; Corry, P.; Yarlagadda, P.K.D.V.; Eustace, C.; Smith, S. A flexible job shop scheduling approach with operators for coal export terminals. *Comput. Oper. Res.* **2019**, *104*, 15–36. [\[CrossRef\]](#)
18. Bartal, Y.; Leonardi, S.; Marchetti-Spaccamela, A.; Sgall, J.; Stougie, L. Multiprocessor scheduling with rejection. *Siam J. Discret. Math.* **2000**, *13*, 64–78. [\[CrossRef\]](#)
19. Cao, Z.G.; Zhang, Y.Z. Scheduling with rejection and nonidentical job arrivals. *J. Syst. Sci. Complex.* **2007**, *20*, 529–535. [\[CrossRef\]](#)
20. Zhang, L.Q.; Lu, L.F.; Yuan, J.J. Single machine scheduling with release dates and rejection. *Eur. J. Oper. Res.* **2009**, *198*, 975–978. [\[CrossRef\]](#)
21. Sengupta, S. Algorithms and approximation schemes for minimum lateness/tardiness scheduling with rejection. *Lect. Notes Comput. Sci.* **2003**, *2748*, 79–90.
22. Engels, D.W.; Karger, D.R.; Kolliopoulos, S.G.; Sengupta, S.; Uma, R.N.; Wein, J. Techniques for scheduling with rejection. *J. Algorithms* **2003**, *49*, 175–191. [\[CrossRef\]](#)
23. Shabtay, D.; Gaspar, N.; Kaspi, M. A survey on offline scheduling with rejection. *J. Sched.* **2013**, *16*, 3–28. [\[CrossRef\]](#)
24. Zhong, X.L.; Ou, J.W.; Wang, G.W. Order acceptance and scheduling with machine availability constraints. *Eur. J. Oper. Res.* **2014**, *232*, 435–442. [\[CrossRef\]](#)
25. Zhao, C.L.; Tang, H.Y. Single machine scheduling with an availability constraint and rejection. *Asia-Pac. J. Oper. Res.* **2014**, *31*, 1450037. [\[CrossRef\]](#)
26. Li, S.S.; Chen, R.X. Scheduling with rejection and a deteriorating maintenance activity on a single machine. *Asia-Pac. J. Oper. Res.* **2014**, *34*, 1750010. [\[CrossRef\]](#)