Online scheduling of unit jobs on three machines with rejection: A tight result

Leah Epstein* Hanan Zebedat-Haider †

Abstract

We design an algorithm of the best possible competitive ratio for preemptive and non-preemptive scheduling of unit size jobs with rejection on three identical machines. The algorithm does not use preemption even for the preemptive variant, and it has the interesting feature that one of its parameters is not fixed in advance, and it is defined based on the properties of the first input job having a sufficiently large rejection penalty.

Keywords: on-line algorithms, scheduling, competitive ratio.

1 Introduction

We deal with the problem of online scheduling with rejection of jobs of processing time 1 on three identical machines. Such jobs are called *unit jobs*. Jobs are presented to an online algorithm one by one. There is a set J of arriving jobs, where the j-th job in the input sequence is denoted by j. Each job $j \in J$ is characterized by a value $w_j \geq 0$, where w_j is the rejection penalty of j (and the processing time of j is equal to 1). For each arriving job, the algorithm decides whether it will be rejected or accepted. If it is rejected, then its rejection penalty will be added to the cost of the algorithm. If it is accepted, then it is assigned to be processed by the machines. The makespan (the last completion time) of the final schedule will be added to the cost of the algorithm at termination. Each machine can run at most one job at each time. In the non-preemptive variant, each accepted job must be assigned to run during a specific continuous time slot on one machine. In the preemptive variant, a job can be split between several time slots (possibly on different machines), under the restriction that the parts of one job cannot be run in parallel on different machines.

Multiprocessor scheduling with rejection was first introduced by Bartal et al. [1]. Non-preemptive and preemptive online models for minimizing the makespan plus the total rejection penalty have been studied since then [12, 8, 2, 5, 6, 4]. The non-preemptive scheduling problem

*Department of Mathematics, University of Haifa, Haifa, Israel. lea@math.haifa.ac.il.
†Department of Mathematics, University of Haifa, Haifa, Israel. hnan_haider@hotmail.com.

of unit jobs without rejection is trivial even as an online problem (jobs are scheduled in a round-robin manner, and the makespan is $\lceil \frac{n}{m} \rceil$ for m machines and n jobs). In the case of preemptive scheduling, the cost of an optimal solution is $\max\{1, \frac{n}{m}\}$ [9]. For the preemptive case, the best possible competitive ratio was analyzed for all numbers of machines $m \geq 2$ by Seiden, Sgall, and Woeginger [13] (for $m = 3$ the competitive ratio is equal to $\frac{5}{4}$). In the variants with rejection (and in particular, variants with unit jobs), a suitable rejection policy is a crucial part of the algorithm, and it is often the case that the scheduling algorithm of the accepted jobs is not very advanced [1, 8, 2, 5, 7].

While the non-preemptive scheduling problem of unit jobs without rejection is simple, the same problem with rejection is a non-trivial problem [1, 4]. Bartal at al. [1] presented a sequence of lower bounds on the optimal competitive ratio for fixed values of m, where this monotonically increasing sequence of lower bounds on the competitive ratios tends to 2 for large values of m. These lower bounds were proved using unit jobs, and they are valid for non-preemptive and preemptive algorithms. For $m = 2$, the value of this lower bound is $\phi = \frac{\sqrt{5}+1}{2} \approx 1.61803$, and an algorithm whose competitive ratio is ϕ (for jobs of arbitrary processing times) was also presented in [1]. For $m = 3$, the value of the lower bound of [1] is approximately $R = 1.83929$, and two algorithms whose competitive ratios are at most 2 (for $m = 3$) were designed in [1]. The precise value of R is defined as follows. First, let $z = (17+3\sqrt{33})^{1/3}$. We let $R = \frac{3z}{z^2-z-2}$, and it can be easily seen that $\frac{1}{R} = \frac{z-1}{3} - \frac{2}{3z} \approx 0.54369$. We will use a parameter $\alpha = \frac{1}{R}$ in our algorithm. In the same paper [1], the case of arbitrary values of m was studied, and tight bounds of $1 + \phi \approx 2.61803$ on the competitive ratio were given (this value is known to be tight in the overall sense, but not for specific values of m).

We studied several models for scheduling unit jobs with rejection [7, 6, 4]. In particular, the cases where the rejection penalties of jobs are either non-increasing or non-decreasing were analyzed completely [7]. In the case of non-decreasing penalties, there are two models, depending on whether the number of jobs is known in advance. The best possible competitive ratios for the three models and three machines are ϕ (for non-increasing penalties), and approximately 1.5133 and 1.7801 (for the case where the number of jobs is known in advance, and the case where it is not given in advance, respectively, and non-decreasing penalties). The algorithms are based on thresholds. Roughly speaking, for each index of a job there is a threshold that determines (by comparing the sum of the rejection penalty of the job and previously rejected jobs to the threshold) whether the job will be accepted.

In this paper, we present an optimal algorithm with competitive ratio approximately 1.839287, a value which matches the special case of the lower bound (for $m = 3$) that was proven by using a sequence of jobs of unit size for the general problem (arbitrary job sizes with or without preemption) of online scheduling with rejection on three identical machines [1]. The algorithm is non-preemptive, but we will show that it has the best possible competitive ratio not only for non-preemptive algorithms but also among preemptive algorithms.

2 The algorithm

In the algorithm, we will use a constant parameter α, such that $0 < \alpha < 1$. The value of α was defined earlier, and we will also state it again later. We will show that the competitive ratio of the algorithm does not exceed R, also defined earlier, and mentioned again later. Moreover, during its execution, the algorithm will define a parameter β satisfying $\alpha^2 < \beta < \alpha$ based on its input.

Our online algorithm consists of a rejection strategy and a scheduling algorithm for the accepted jobs. The scheduling algorithm simply assigns accepted jobs to machines using a round-robin policy. Thus, even in the preemptive variant, the algorithm does not use preemption. The rejection strategy for deciding which jobs are rejected and which jobs are accepted is more complicated. In particular, the threshold β that is used for such decisions in the second stage of the algorithm is defined in the first stage of the algorithm, based on the input. The *special job* will be defined as the first input job j^* such that $w_{j^*} \geq \alpha^2$ (if such a job exists), and the value of β is based on w_{j^*}. After this job arrives (and it is either rejected or scheduled), the algorithm moves to a second stage and stays there until the input is terminated. The algorithm is called ***MSR*3**, which is an abbreviation of MULTIPROCESSOR SCHEDULING WITH REJECTION ON THREE MACHINES.

Algorithm $\boldsymbol{MSR3(\alpha)}$

- Let $j = 1$.

Stage 1

- If the input job j exists, act as follows (and otherwise halt).
- If $w_j < \alpha^2$, then reject j, let $j = j + 1$, and go to stage 1.
- Otherwise, define $j^* = j$, that is, define j to be *the special job*.
- If $w_{j^*} < \alpha$, reject j^*, and let $\beta = \alpha - (1 - \alpha)w_{j^*}$.
- Otherwise, accept j^*, schedule it on the machine of smallest current completion time of the minimum index (that is, on machine 1, as no jobs were scheduled so far), and let $\beta = \frac{1}{3}$.
- Let $j = j + 1$, and go to stage 2.

Stage 2

- If the input job j exists, act as follows (and otherwise halt).
- If $w_j \leq \beta$, reject j.

- Otherwise, schedule it on the machine of smallest current completion time of the minimum index.

- Let $j = j + 1$ and go to stage 2.

We choose $\alpha \approx 0.54369$, which is the solution of the following equation with the variable x: $x^3 + x^2 + x = 1$. Note that $\alpha^2 \approx 0.29560$. Let $R = \alpha^2 + \alpha + 1 = \frac{1}{\alpha} \approx 1.83929$. In the case where β was not defined by the algorithm, in what follows we let $\beta = \frac{1}{3}$.

Claim 1. *We have $\alpha^2 < \beta < \alpha$, and $\beta \leq 1 - 2\alpha^2 \approx 0.40880$.*

Proof. If $\beta = \frac{1}{3}$, then the claim holds by the value of α. Otherwise, j^* exists, $w_{j^*} \geq \alpha^2$ holds by the choice of j^*, and since the algorithm defined β such that $\beta \neq \frac{1}{3}$, $w_{j^*} < \alpha$ holds as well. We find $\beta = \alpha - (1-\alpha)w_{j^*} \leq \alpha - \alpha^2 + \alpha^3 = 1 - 2\alpha^2 \approx 0.40880$ since $w_{j^*} \geq \alpha^2$, and by the value of α. We get $\beta > \alpha^2$ as $\beta = \alpha - (1-\alpha)w_{j^*}$ and $w_{j^*} < \alpha$. Finally, we get $\beta \leq 1 - 2\alpha^2 < \alpha$, by $\alpha^3 + \alpha = 1 - \alpha^2$ and $\alpha < 1$. □

Let $C = \{j \neq j^* : w_j \leq \beta\}$ and $E = \{j \neq j^* : w_j > \beta\}$, and we call the jobs in the sets C and E cheap and expensive jobs, respectively.

Claim 2. *The algorithm accepts all expensive jobs and it rejects all cheap jobs.*

Proof. The claim holds for all jobs that were considered in stage 2 by the definition of the algorithm. Moreover, any job j considered in stage 1 (excluding j^*) satisfies $w_j < \alpha^2 < \beta$. □

3 Analysis

Theorem 3. *The competitive ratio of the above algorithm is at most R, both for the preemptive variant, and for the non-preemptive variant.*

Proof. We consider both variants together, as the lower bounds that we will use on optimal solutions are valid both for preemptive and non-preemptive algorithms.

We will prove that the competitive ratio is no larger than R via negation. For any input $\tilde{J}$, let $\text{OPT}(\tilde{J})$ be a specific optimal solution for $\tilde{J}$. We denote the cost of the algorithm for the input $\tilde{J}$ by $\text{ALG}(\tilde{J})$. Assume by contradiction that there exists an input J for which $\text{ALG}(J) > R \cdot \text{OPT}(J)$. We assume without loss of generality that J is a minimal counterexample with respect to its cardinality (the number of input jobs), and n is the number of jobs in J. Note that the jobs arriving before the special job are considered independently of each other by the algorithm, and the jobs arriving after the special job are considered independently of each other by the algorithm as well. In particular, if a cheap job $j \neq j^*$ is removed from J, and the algorithm is executed again without this job, then the cost of the algorithm is decreased by w_j. The corresponding property for $\text{OPT}(J)$ is that if we remove a job j rejected by $\text{OPT}(J)$, then the cost is reduced by w_j, so for $J' = J \setminus \{j\}$, $\textsc{opt}(J') \le \textsc{opt}(J) - w_j$ (the resulting solution is not necessarily optimal for the modified input).

Claim 4. *There exists a special job in J.*

Proof. Assume that there is no special job. In this case, all jobs are rejected, and each job has a rejection penalty strictly smaller than α^2. We will show that the algorithm produces an optimal solution by showing that $\textsc{opt}(J)$ also rejects all jobs, and the outputs of the algorithm and of $\textsc{opt}(J)$ are identical. Assume by contradiction that $\textsc{opt}(J)$ schedules $N > 0$ jobs. Modify this solution such that it will reject all jobs. The completion time for any schedule of N jobs is at least $\frac{N}{3}$ (as the total size of jobs is N), while the cost of rejecting these jobs is smaller than $\alpha^2 \cdot N < \frac{N}{3}$. Therefore, the cost strictly decreases as a result, contradicting the optimality of $\textsc{opt}(J)$. □

In what follows we assume that j^* exists.

Claim 5. *For the input J, $\textsc{opt}(J)$ does not reject any cheap jobs.*

Proof. Assume by contradiction that $\textsc{opt}(J)$ rejects at least one cheap job j_1. Let $J' = J \setminus \{j_1\}$. Since j_1 is rejected by $\textsc{opt}(J)$, $\textsc{opt}(J') \le \textsc{opt}(J) - w_{j_1}$. For the algorithm we find $\textsc{alg}(J') = \textsc{alg}(J) - w_{j_1}$. Using the fact that J is a counterexample we find $\textsc{alg}(J') = \textsc{alg}(J) - w_{j_1} > R \cdot \textsc{opt}(J) - w_{j_1} \ge R \cdot \textsc{opt}(J') + R \cdot w_{j_1} - w_{j_1} > R \cdot \textsc{opt}(J')$, as $R > 1$. Thus, $\textsc{alg}(J') > R \cdot \textsc{opt}(J')$ holds, contradicting the minimality of the counterexample, as $|J'| < |J|$. □

Claim 6. *If there exists at least one cheap job, then $\textsc{opt}(J)$ schedules all jobs, possibly except for j^*.*

Proof. If there are no expensive jobs, then we are done by Claim 5. Otherwise, there is at least one cheap job and at least one expensive job. Recall that $\textsc{opt}(J)$ schedules all cheap jobs, and assume by contradiction that $\textsc{opt}(J)$ rejects an expensive job j_1. Let j_2 be a cheap job (which $\textsc{opt}(J)$ schedules). Swapping the roles of these jobs in the optimal solution we get an alternative solution whose cost is smaller by $w_{j_1} - w_{j_2} > 0$, since $w_{j_1} > \beta$ while $w_{j_2} \le \beta$ (as j_1 is expensive and j_2 is cheap). This contradicts the optimality of $\textsc{opt}(J)$. □

Claim 7. *The input J contains at least two jobs, i.e., $n \ge 2$.*

Proof. Assume that $n = 1$. Then, the only job is j^*. If the algorithm rejects it, then $w_{j^*} < \alpha$, $\textsc{opt}(J)$ also rejects j^* (as otherwise its cost would be at least 1), and the algorithm is optimal. If the algorithm schedules j^* and so does $\textsc{opt}(J)$, then the algorithm is optimal again (both of them have makespans of 1). Otherwise, $\textsc{alg}(J) = 1$ while $\textsc{opt}(J) = w_{j^*} \ge \alpha$, and the competitive ratio is at most $\frac{1}{\alpha} = R$. □

Claim 8. *The input J contains no cheap jobs.*

Proof. Assume that J contains at least one cheap job. In this case $\text{OPT}(J)$ does not reject any job (except for possibly j^*) by Claim 6. The cost of $\text{OPT}(J)$ is at least 1, as it accepts at least one job, since $n \geq 2$. Moreover, its cost is at least $\max\{1, \frac{n}{3}\}$, if it accepts j^*, and it is at least $\max\{1, \frac{n-1}{3}\} + w_{j^*}$ otherwise, due to the total sizes of jobs, and $n \geq 2$. In both cases, $\text{OPT}(J) \geq \max\{1, \frac{n-1}{3} + \alpha^2\}$, as $\alpha^2 \leq \min\{\frac{1}{3}, w_{j^*}\}$.

If the algorithm accepts j^*, then $\beta = \frac{1}{3}$. Additionally, $w_{j^*} \geq \alpha > \frac{1}{3}$, so $\text{OPT}(J) \geq \max\{1, \frac{n}{3}\}$, no matter whether $\text{OPT}(J)$ accepts or rejects j^*. The cost of the algorithm for every rejected job is no larger than $\beta = \frac{1}{3}$, and its cost for N accepted jobs is at most $\lceil \frac{N}{3} \rceil \leq \frac{N+2}{3}$. Thus, its total cost is at most $\frac{n+2}{3}$. If $n \geq 3$, then $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{(n+2)/3}{n/3} = 1 + \frac{2}{n} \leq \frac{5}{3} < R$. If $n = 2$, then the makespan of the algorithm is 1, and it rejects at most one job (of rejection penalty at most $\beta = \frac{1}{3}$), so $\text{ALG}(J) \leq \frac{4}{3}$, while $\text{OPT}(J) \geq 1$, and the competitive ratio is below R in this case as well.

We are left with the case where the algorithm rejects j^*, and thus $w_{j^*} < \alpha$. Let $N_e \geq 0$ be the number of expensive jobs, and let $N_c \geq 1$ be the number of cheap jobs (where $N_e + N_c = n-1$). We have $\text{ALG}(J) \leq \lceil \frac{N_e}{3} \rceil + w_{j^*} + \beta \cdot N_c$. Using the definition of β and $0 < \alpha^2 \leq w_{j^*} < \alpha$, $w_{j^*} + \beta \cdot N_c = w_{j^*} + \beta + \beta \cdot (N_c - 1) = (1 + w_{j^*})\alpha + \beta \cdot (N_c - 1) \leq (1+\alpha)\alpha + \beta \cdot (N_c - 1)$. We have $\lceil \frac{N_e}{3} \rceil = 0$ for $N_e = 0$, $\lceil \frac{N_e}{3} \rceil = 1$ for $N_e = 1, 2, 3$, and $\lceil \frac{N_e}{3} \rceil \leq \frac{N_e+2}{3}$.

First, consider the case $N_e = 0$. In this case, $N_c = n-1$. By $\text{ALG}(J) \leq (1+\alpha)\alpha + \beta \cdot (N_c - 1)$ and $\beta < \alpha$, we get $\text{ALG}(J) \leq (1+\alpha)\alpha + \alpha \cdot (n-2) = \alpha^2 + \alpha(n-1)$. By using $\text{OPT}(J) \geq \frac{n-1}{3} + \alpha^2$, we find $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{3\alpha(n-1)+3\alpha^2}{n-1+3\alpha^2} < 3\alpha < R$, since $\alpha > 0$ and $n \geq 2$.

We are left with the case $N_e \geq 1$, and $n \geq 3$ (as $N_c \geq 1$, and $N_c + N_e = n - 1$). If $n \leq 4$, then $N_e, N_c \leq 2$, and $\text{ALG}(J) \leq 1 + \alpha + \alpha^2 + \beta \cdot (N_c - 1)$. If $N_c = 1$, we are done by $\text{OPT}(J) \geq 1$. Otherwise, $N_c = 2$ and $n = 4$ hold, and we have $\text{OPT}(J) \geq 1 + \alpha^2$ while $\text{ALG}(J) \leq 1 + \alpha + \alpha^2 + \beta < 1 + 2\alpha + \alpha^2$, as $\beta < \alpha$. We get $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{1+2\alpha+\alpha^2}{1+\alpha^2} = \alpha^2 + \alpha + 1 = R$, since $\alpha = \alpha^2 + \alpha^3 + \alpha^4$.

Finally, we are left with the case $n \geq 5$, $N_e \geq 1$, and $N_c \geq 1$. Let $\omega = 1 - 2\alpha^2$. Recall that $\beta \leq \omega$ and $\omega > \frac{1}{3}$, by Claim 1. Thus, $\text{ALG}(J) \leq \omega(N_e + 2) + (1+\alpha)\alpha + \omega \cdot (N_c - 1) = n\omega + \alpha + \alpha^2$, while $\text{OPT}(J) \geq \frac{n-1}{3} + \alpha^2$. We get $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{3n\omega+3\alpha+3\alpha^2}{n-1+3\alpha^2} = 3\omega + \frac{3\omega-9\omega\alpha^2+3\alpha+3\alpha^2}{n-1+3\alpha^2} \leq 3\omega + \frac{3\omega-9\omega\alpha^2+3\alpha+3\alpha^2}{4+3\alpha^2} = \frac{15\omega+3\alpha+3\alpha^2}{4+3\alpha^2}$. As $\frac{15\omega+3\alpha+3\alpha^2}{4+3\alpha^2} \leq R = \frac{1}{\alpha}$ is equivalent to $15\omega\alpha + 3\alpha^3 \leq 4$, we will prove the last inequality. Using $\omega \leq 0.41$, $\alpha < 0.55$, and $\alpha^3 < 0.17$ we find that $15\omega\alpha + 3\alpha^3 < 4$, as required. $\square$

Claim 9. *The input J contains no expensive jobs.*

Proof. Assume that J has at least one expensive job. As there are no cheap jobs by Claim 8, the algorithm does not reject any job, possibly except for j^*. Additionally, $w_{j^*} \geq \alpha^2$, the cost of $\text{OPT}(J)$ for each job is at least $\frac{1}{3}$ if this job is accepted, and otherwise (if it is rejected) the cost of $\text{OPT}(J)$ for the job is at least α^2 (as all jobs except for j^* are expensive, and $\beta > \alpha^2$). We find $\text{OPT}(J) > \alpha^2 n$.

If $n \leq 3$, the algorithm completes all its accepted jobs at time 1. Its cost satisfies $\text{ALG}(J) \leq 1 + w_{j^*}$ if j^* is rejected (in which case $w_{j^*} < \alpha$), and $\text{ALG}(J) \leq 1$ otherwise. Thus, $\text{ALG}(J) < 1+\alpha$. If $\text{OPT}(J)$ accepts at least one job, we are done, as in this case $\text{OPT}(J) \geq 1$. Otherwise, if $n = 3$, we have $\text{OPT}(J) \geq 3\alpha^2$, and $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{1+\alpha}{3\alpha^2} < R$ as $3\alpha^2 R = 3\alpha > 1 + \alpha$ since $\alpha > 1/2$. If $n = 2$ and $\text{ALG}(J) = 1$, then using $\text{OPT}(J) \geq 2\alpha^2$, we get $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{1}{2\alpha^2} < R$, as $2\alpha^2 R = 2\alpha > 1$. If $n = 2$ and $\text{ALG}(J) = 1 + w_{j^*}$, then as $\text{OPT}(J)$ rejects both jobs (one expensive job and the job j^*), $\text{OPT}(J) \geq \beta + w_{j^*} = \alpha(1 + w_{j^*})$, by the definition of β in this case, and $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{1}{\alpha} = R$.

If $n = 4$, then $\text{ALG}(J) \leq 1 + w_{j^*} < 2$ if j^* is rejected and otherwise $\text{ALG}(J) = 2$. Using $\text{OPT}(J) \geq 4\alpha^2$ we have $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{1}{2\alpha^2} < R$.

If $n \geq 5$, then $\text{ALG}(J) \leq \lceil \frac{n-1}{3} \rceil + w_{j^*} \leq \frac{n+1}{3} + w_{j^*}$ if j^* is rejected, and otherwise $\text{ALG}(J) \leq \lceil \frac{n}{3} \rceil \leq \frac{n+2}{3}$. In both cases, $\text{ALG}(J) \leq \frac{n+1}{3} + \alpha$. Using $\text{OPT}(J) \geq n\alpha^2$ we have $\frac{\text{ALG}(J)}{\text{OPT}(J)} \leq \frac{n/3+1/3+\alpha}{n\alpha^2} = \frac{1}{3\alpha^2} + \frac{1/3+\alpha}{n\alpha^2} \leq \frac{1}{3\alpha^2} + \frac{1/3+\alpha}{5\alpha^2} = \frac{2+\alpha}{5\alpha^2} < R$ as $5\alpha^2 R = 5\alpha > 2 + \alpha$. □

We found that J cannot have cheap jobs or expensive jobs, but $n \geq 2$ implies that there is at least one job except for j^*, thus, we reached a contradiction. □

References

[1] Y. Bartal, S. Leonardi, A. Marchetti-Spaccamela, J. Sgall, and L. Stougie. Multiprocessor scheduling with rejection. *SIAM Journal on Discrete Mathematics*, 13(1):64–78, 2000.

[2] Gy. Dósa and Y. He. Preemptive and non-preemptive on-line algorithms for scheduling with rejection on two uniform machines. *Computing*, 76(1-2):149–164, 2006.

[3] Gy. Dósa and Y. He. Scheduling with machine cost and rejection. *Journal of Combinatorial Optimization*, 12(4):337–350, 2006.

[4] L. Epstein and H. Zebedat-Haider. Online scheduling with rejection and reordering: exact algorithms for unit size jobs. *Journal of Combinatorial Optimization*, 28(4):875–892, 2014.

[5] L. Epstein and H. Zebedat-Haider. Online scheduling with rejection and withdrawal. *Theoretical Computer Science*, 412(48):6666–6674, 2011.

[6] L. Epstein and H. Zebedat-Haider. Preemptive online scheduling with rejection of unit jobs on two uniformly related machines. *Journal of Scheduling*, 17(1):87–93, 2014.

[7] L. Epstein and H. Zebedat-Haider. Rent or buy problems with a fixed time horizon. *Theory of Computing Systems*, 56(2):309-329, 2015.

[8] Y. He and X. Min. On-line uniform machine scheduling with rejection. *Computing*, 65(1):1–12, 2000.

[9] R. McNaughton. Scheduling with deadlines and loss functions. *Management Science*, 6(1):1–12, 1959.

[10] X. Min, J. Liu, and Y. Wang. Optimal semi-online algorithm for scheduling with rejection on two uniform machines. *Journal of Combinatorial Optimization*, 22(4):674–683, 2011.

[11] X. Min, Y. Wang, J. Liu, and M. Jiang. Semi-online scheduling on two identical machines with rejection. *Journal of Combinatorial Optimization*, 26(3):472–479, 2013.

[12] S. S. Seiden. Preemptive multiprocessor scheduling with rejection. *Theoretical Computer Science*, 262(1):437–458, 2001.

[13] S. Seiden, J. Sgall, and G. Woeginger. Semi-online scheduling with decreasing job sizes. *Operations Research Letters*, 27(5):215–221, 2000.