Information Processing Letters 116 (2016) 503–507

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Faster algorithms for single machine scheduling with release dates and rejection

Jinwen Ou [a,1], Xueling Zhong [b,2], Chung-Lun Li [c,*]

[a] Department of Administrative Management, Jinan University, Guangzhou, 510632, People's Republic of China
[b] Department of Internet Finance and Information Engineering, Guangdong University of Finance, Guangzhou 510520, People's Republic of China
[c] Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 9 March 2015
Received in revised form 23 February 2016
Accepted 23 February 2016
Available online 27 February 2016
Communicated by M. Chrobak

Keywords:
Scheduling
Release dates
Rejection penalty
Algorithms
Worst-case analysis

ABSTRACT

We consider the single machine scheduling problem with release dates and job rejection with an objective of minimizing the makespan of the job schedule plus the total rejection penalty of the rejected jobs. Zhang et al. [6] have presented a 2-approximation algorithm with an $O(n^2)$ complexity for this problem and an exact algorithm with an $O(n^3)$ complexity for the special case with identical job processing times. In this note, we show that the 2-approximation algorithm developed by Zhang et al. [6] can be implemented in $O(n \log n)$ time. We also develop a new exact algorithm with an improved complexity of $O(n^2 \log n)$ for the special case with identical job processing times. The second algorithm can be easily extended to solve the parallel-machine case with the same running time complexity, which answers an open question recently raised by Zhang and Lu [5].

1. Introduction

In this note, we consider the following single-machine scheduling problem with release dates and job rejection decisions: There are a single machine and a set of n jobs $J = \{J_1, J_2, \ldots, J_n\}$. Each job $J_j \in J$ has a processing time $p_j \geq 0$, a release date $r_j \geq 0$, and a rejection penalty $w_j > 0$. Job J_j is either rejected, which incurs a rejection penalty w_j, or accepted and processed by the machine at or after its release date r_j. The machine can process at most one job at a time, and preemption is not allowed during job processing. The objective is to determine a subset of jobs to be accepted and processed on the machine, so as to minimize the makespan of the job schedule plus the total rejection penalty of all rejected jobs. This problem was introduced by Zhang et al. [6]. Following their notation, this problem is denoted as $1|r_j, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$, and the special case with identical job processing times is denoted as $1|r_j, p_j = p, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$, where R is the set of rejected jobs.

Zhang et al. [6] have shown the NP-hardness of problem $1|r_j, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$ and provided several exact and approximation algorithms for the problem. One of their main results is a 2-approximation algorithm with an $O(n^2)$ running time for this problem. Another main result is an exact algorithm with a running time of $O(n(r_{\max} + P))$, where $r_{\max} = \max_j\{r_j\}$ and $P = \sum_j p_j$, and this exact algorithm has an $O(n^3)$ running time when applied to the special case $1|r_j, p_j = p, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$. Although a lot of studies on related topics have been pub-

* Corresponding author. Tel.: +852 2766 7410.
E-mail addresses: toujinwen@jnu.edu.cn (J. Ou), zhongxuel@hotmail.com (X. Zhong), chung-lun.li@polyu.edu.hk (C.-L. Li).
[1] Tel.: +86 20 8522 3243.
[2] Tel.: +86 20 3820 2796.

http://dx.doi.org/10.1016/j.ipl.2016.02.008
0020-0190/

lished recently (see [3] and [4] for recent reviews), to the best of our knowledge, no improved algorithms for these two specific problems have appeared in the literature, except for the recent work by He et al. [1] (see Remark 1 below). In the following sections, we first show that the 2-approximation algorithm developed by Zhang et al. [6] for problem $1|r_j, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$ can be implemented in $O(n \log n)$ time. We then develop a new exact algorithm with a reduced complexity of $O(n^2 \log n)$ for problem $1|r_j, p_j = p, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$. This algorithm can be extended to solve the parallel machine problem $P|r_j, p_j = p, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$, which answers an open question recently raised by Zhang and Lu [5].

2. Faster approximation algorithm for $1|r_j, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$

Zhang et al. [6] have developed an approximation algorithm with an $O(n^2)$ complexity for problem $1|r_j, \text{reject}|C_{\max} + \sum_{J_j \in R} w_j$. Denoting $p(S) = \sum_{J_j \in S} p_j$ for any job subset S, their algorithm is given as follows:

Algorithm A. (See Zhang et al. [6].)

Step 1. For each $t \in \{r_j \mid j = 1, 2, \ldots, n\}$, we divide the jobs into three sets such that $S_1(t) = \{J_j \mid r_j \le t \text{ and } p_j \le w_j\}$, $S_2(t) = \{J_j \mid r_j \le t \text{ and } p_j > w_j\}$, and $S_3(t) = \{J_j \mid r_j > t\}$.

Step 2. Accept all jobs in $S_1(t)$ and reject the jobs in $S_2(t) \cup S_3(t)$. Assign the accepted jobs to be processed in time interval $[t, t + p(S_1(t))]$ on the machine. The resulting schedule is denoted by $\pi(t)$.

Step 3. Let $Z(t)$ be the value of the objective function for each $\pi(t)$. Among all the schedules obtained above, select the one with the minimum $Z(t)$ value.

Zhang et al. [6, Thm. 4.1] have shown that Algorithm A has a performance ratio of 2. In their proof, they have implicitly assumed that at least one job must be accepted. If we allow the solution to reject all jobs, then Algorithm A does not have a constant performance ratio. To see this, consider an example with $n = 1$, $p_1 = 1$, $w_1 = 2$, and $r_1 = M$, where M is a large number. In this example, $S_1(M) = \{J_1\}$ and $S_2(M) = S_3(M) = \emptyset$. The only schedule generated by Steps 1–2 is $\pi(M)$, which processes job J_1 in the time interval $[M, M+1]$. The objective value of this solution is $M + 1$. On the other hand, the optimal solution is to reject J_1, and has an objective value of 2. Thus, the performance ratio of the solution generated by Algorithm A is $\frac{M+1}{2}$, which approaches infinity as $M \to \infty$. To ensure that the algorithm can also handle the case where all jobs are rejected, we consider a modified version of Algorithm A by adding the following step.

Step 4. Compute $W = \sum_{j=1}^{n} w_j$. If the minimum $Z(t)$ value obtained by Step 3 is greater than W, then reject all jobs.

We denote this modified algorithm as Algorithm A1. It is easy to check that after adding Step 4 to Algorithm A, the proof of Theorem 4.1 in [6] is valid even if the optimal solution is to reject all jobs. Hence, Algorithm A1 is a 2-approximation algorithm.

In the following, we show that Algorithm A1 can be implemented in $O(n \log n)$ time. We first re-index the jobs such that $r_1 \le r_2 \le \cdots \le r_n$. This requires $O(n \log n)$ time. Denote $P_k = p(S_1(r_k))$ and $\bar{W}_k = \sum_{J_j \in J \setminus S_1(r_k)} w_j$, for $k = 1, 2, \ldots, n$. It is easy to check that $Z(r_k) = r_k + P_k + \bar{W}_k$. Next, we show that the values of $Z(r_1), Z(r_2), \ldots, Z(r_n)$ can be computed recursively in $O(n)$ time.

Consider any $k = 2, 3, \ldots, n$. If $p_k > w_k$, then $S_1(r_k) = S_1(r_{k-1})$, $P_k = P_{k-1}$, $\bar{W}_k = \bar{W}_{k-1}$, and therefore $Z(r_k) = r_k + P_k + \bar{W}_k = r_k + P_{k-1} + \bar{W}_{k-1} = r_k + Z(r_{k-1}) - r_{k-1}$. If $p_k \le w_k$, then $S_1(r_k) = S_1(r_{k-1}) \cup \{J_k\}$, $P_k = P_{k-1} + p_k$, $\bar{W}_k = \bar{W}_{k-1} - w_k$, and therefore $Z(r_k) = r_k + P_k + \bar{W}_k = r_k + P_{k-1} + p_k + \bar{W}_{k-1} - w_k = r_k + Z(r_{k-1}) - r_{k-1} + p_k - w_k$. Thus, in both cases,

$$Z(r_k) = r_k + Z(r_{k-1}) - r_{k-1} + \min\{p_k - w_k, 0\}. \tag{1}$$

Clearly, $Z(r_1)$ can be determined in $O(n)$ time. By repeatedly applying equation (1), the values of $Z(r_2), Z(r_3), \ldots, Z(r_n)$ can be computed in $O(n)$ time. In other words, when the jobs are indexed in nondecreasing release dates, $Z(r_1), Z(r_2), \ldots, Z(r_n)$ can be obtained without determining the sets $S_1(t)$, $S_2(t)$, and $S_3(t)$ for $t = r_1, r_2, \ldots, r_n$. After computing $Z(r_1), Z(r_2), \ldots, Z(r_n)$, we determine $S_1(r_{k'})$, where $k' = \arg\min_{k=1,\ldots,n}\{Z(r_k)\}$. The schedule in Step 3 is obtained by accepting the jobs in $S_1(r_{k'})$ and assigning them to time interval $[r_{k'}, r_{k'} + P_{k'}]$. This step, as well as Step 4, requires $O(n)$ time. Hence, we have the following result.

Theorem 1. *Algorithm A1 can be implemented in $O(n \log n)$ time.*

To illustrate this computation process, consider an example with $n = 5$, $(r_1, r_2, r_3, r_4, r_5) = (0, 2, 6, 10, 16)$, $(p_1, p_2, p_3, p_4, p_5) = (7, 1, 10, 4, 3)$, and $(w_1, w_2, w_3, w_4, w_5) = (8, 4, 12, 3, 7)$. In this example, $P_1 = 7$, $\bar{W}_1 = 26$, and therefore $Z(r_1) = r_1 + P_1 + \bar{W}_1 = 33$. By equation (1),

$$\begin{aligned}
Z(r_2) &= r_2 + Z(r_1) - r_1 + \min\{p_2 - w_2, 0\} \\
&= 2 + 33 - 0 + \min\{-3, 0\} = 32; \\
Z(r_3) &= r_3 + Z(r_2) - r_2 + \min\{p_3 - w_3, 0\} \\
&= 6 + 32 - 2 + \min\{-2, 0\} = 34; \\
Z(r_4) &= r_4 + Z(r_3) - r_3 + \min\{p_4 - w_4, 0\} \\
&= 10 + 34 - 6 + \min\{1, 0\} = 38; \\
Z(r_5) &= r_5 + Z(r_4) - r_4 + \min\{p_5 - w_5, 0\} \\
&= 16 + 38 - 10 + \min\{-4, 0\} = 40.
\end{aligned}$$

Thus, $k' = 2$, and the solution generated by Step 3 is to accept the jobs in $S_1(r_2) = \{J_1, J_2\}$, assign them to the time interval $[r_2, r_2 + P_2] = [2, 10]$, and reject the other jobs.