Available online at www.sciencedirect.com
ScienceDirect
Computers & Operations Research 34 (2007) 1764 – 1770
computers & operations research
www.elsevier.com/locate/cor

Single-machine scheduling with periodic maintenance to minimize makespan

Min Ji[a, b], Yong He[a], T.C.E. Cheng[b, *]

[a]*Department of Mathematics, and State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310027, PR China*
[b]*Department of Logistics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*

Available online 22 August 2005

Abstract

We consider a single-machine scheduling problem with periodic maintenance activities. Although the scheduling problem with maintenance has attracted researchers' attention, most of past studies considered only one maintenance period. In this research several maintenance periods are considered where each maintenance activity is scheduled after a periodic time interval. The objective is to find a schedule that minimizes the makespan, subject to periodic maintenance and nonresumable jobs. We first prove that the worst-case ratio of the classical *LPT* algorithm is 2. Then we show that there is no polynomial time approximation algorithm with a worst-case ratio less than 2 unless $P = NP$, which implies that the *LPT* algorithm is the best possible.

Keywords: Single-machine scheduling; Periodic maintenance; Nonresumable jobs; Approximation algorithm; Non-approximability

1. Introduction

Most literature on scheduling theory assumes that machines are continuously available. However, this assumption may not be valid in a real production situation due to preventive maintenance (a deterministic event) or breakdown of machines (a stochastic phenomenon). It is not uncommon to observe in practice that machines are awaiting maintenance while there are jobs waiting to be processed by these machines. This is due to a lack of coordination between maintenance planning and production scheduling. Uncertain

* Corresponding author.
E-mail addresses: jimkeen@math.zju.edu.cn (M. Ji), mathhey@zju.edu.cn (Y. He), LGTcheng@polyu.edu.hk (T.C.E. Cheng).

0305-0548/$ - see front matter
doi:10.1016/j.cor.2005.05.034

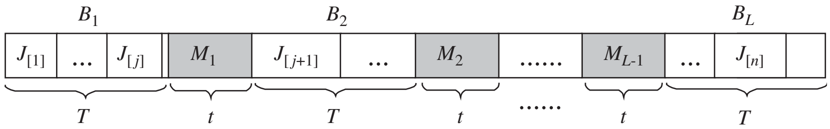

Fig. 1. An illustration of the problem under consideration, where $J_{[j]}$ denotes the job placed in the jth position of the given schedule.

machine breakdowns will make the shop behavior hard to predict, thus reducing the efficiency of the production system. Maintenance can reduce the breakdown rate with minor sacrifices in production time. The importance of maintenance has increasingly been recognized by decision makers. Therefore, scheduling the maintenance of manufacturing systems has gradually become a common practice in many companies. The work of periodic maintenance includes periodic inspections, periodic repairs, and preventive maintenance. With proper planning of periodic maintenance, the shop can improve production efficiency and safety, resulting in increased productivity and heightened safety awareness [1].

As maintenance is scheduled periodically in many manufacturing systems, there is a need to develop an approach to handle the scheduling of jobs for processing in systems with periodic maintenance, which usually have more than one maintenance period. However, to the best of our knowledge, only Liao and Chen [2] have considered such a scheduling problem for the objective of minimizing the maximum tardiness. They proposed a branch-and-bound algorithm to derive the optimal schedule and a heuristic algorithm for large-sized problems. They also provided computational results to demonstrate the efficiency of their heuristic. In this paper we consider the scheduling problem with periodic maintenance to minimize the makespan.

Formally, the considered problem can be described as follows: We are given n independent nonresumable jobs $\mathscr{J} = \{J_1, J_2, \dots, J_n\}$, which are processed on a single machine. Here *nonresumable* means that if a job cannot be finished before a maintenance activity, it has to restart. The processing time of job J_i is p_i. All the jobs are available at time zero. The amount of time to perform each maintenance activity is t. Let the length of the time interval between two consecutive maintenance periods be T. We assume that $T \geqslant p_i$ for every $i = 1, 2, \dots, n$, for otherwise there is trivially no feasible schedule. We think of each interval between two consecutive maintenance activities as a batch with a capacity T. Thus, a schedule π can be denoted as $\pi = (B_1, M_1, B_2, M_2, \dots, M_{L-1}, B_L)$, where M_i is the ith maintenance activity, L is the number of batches, and B_i is the ith batch of jobs. An illustration of the considered problem in the form of a Gantt chart is given in Fig. 1. Let C_i be the completion time of job J_i, then the objective is to minimize the makespan, which is defined as $C_{\max} = \max_{i=1,2,\dots,n} C_i$. Using the three-field notation of Graham et al. [3], we denote this scheduling problem as $1/nr - pm/C_{\max}$. It can easily be shown that this problem is strongly *NP*-hard [4], but no approximation algorithm has been provided and analyzed in the literature.

We use the worst-case ratio to measure the quality of an approximation algorithm. Specifically, for the makespan problem, let C_A denote the makespan produced by an approximation algorithm A, and C_{OPT} the makespan produced by an optimal off-line algorithm. Then the *worst-case ratio* of algorithm A is defined as the smallest number c such that for any instance I, $C_A \leqslant c C_{OPT}$.

The single-machine scheduling problem with single maintenance and nonresumable jobs has been well studied. For the objective of minimizing the makespan, Lee [4] showed that the longest processing time (*LPT*) rule has a tight worst-case ratio of $\frac{4}{3}$, and He et al. [5] presented a fully polynomial time approximation scheme. For the objective of minimizing the total completion time, Lee and Liman [6] proved that the worst-case ratio of the shortest processing time (*SPT*) rule is $\frac{9}{7}$. Sadfi et al. [7] proposed a modified algorithm *MSPT* with a worst-case ratio of $\frac{20}{17}$. He et al. [8] presented a polynomial time approximation scheme. Moreover, Lee [4] presented heuristics for other objectives, such as minimizing the maximum lateness, the number of tardy jobs, and the total weighted completion time, etc. Graves and Lee [9] extended the problem to consider semiresumable jobs. For details on related research, the reader may refer to the survey paper [10].

In this paper we first show that the worst-case ratio of the classical *LPT* algorithm is 2. Then we prove that there is no polynomial time approximation algorithm with a worst-case ratio of less than 2, which implies that *LPT* is the best possible algorithm. Finally, we present some concluding remarks.

2. The LPT algorithm and its worst-case ratio

In this section we analyze the *LPT* algorithm, which is a classical heuristic for solving scheduling problems. It can be formally described as follows.

Algorithm *LPT*. Re-order all the jobs such that $p_1 \geqslant p_2 \geqslant \cdots \geqslant p_n$; then process the jobs consecutively as early as possible.

Note that if we take each batch as a bin, then the *LPT* algorithm is equivalent to the first fit decreasing (*FFD*) algorithm, which is a classical heuristic for solving the bin-packing problem. The worst-case ratio for the *FFD* is $\frac{3}{2}$ [11], i.e.,

$$b \leqslant \tfrac{3}{2} b^*, \tag{1}$$

where b is the number of bins (i.e., batches) obtained by the *FFD* (i.e., *LPT*) algorithm and b^* is the optimal number of bins (batches) for the bin-packing (scheduling) problem. Before analyzing the *LPT* algorithm, we first present some properties and lemmas, which are all straightforward.

Property 1. *The optimal schedule must have the minimum number of batches, i.e., it corresponds to an optimal solution for the bin-packing problem.*

Lemma 1 (*see Baase and Gelder [12, p. 574]*). *In the LPT schedule, if $b > b^*$, then the processing time of each job in batches $B_{b^*+1}, B_{b^*+2}, \ldots, B_b$ is not larger than $T/3$.*

Lemma 2 (*see Baase and Gelder [12, p. 574–575]*). *In the LPT schedule, if $b > b^*$, then the total number of jobs in batches $B_{b^*+1}, B_{b^*+2}, \ldots, B_b$ is not greater than $b^* - 1$.*

Let the total processing times of the jobs in the last batch of the optimal schedule and the *LPT* schedule be x and y, respectively. Then from Property 1, the makespan of the optimal schedule is

$$C_{OPT} = (b^* - 1)(T + t) + x, \tag{2}$$

while the makespan of the *LPT* schedule is

$$C_{LPT} = (b - 1)(T + t) + y. \tag{3}$$

(2) implies that

$$b^* = 1 + \frac{C_{OPT} - x}{T + t}. \tag{4}$$

Substituting (4) into (1), we obtain

$$b \leqslant \frac{3}{2}\left(1 + \frac{C_{OPT} - x}{T + t}\right). \tag{5}$$

Substituting (5) into (3), we establish

$$\begin{aligned} C_{LPT} &\leqslant \left[\frac{3}{2}\left(1 + \frac{C_{OPT} - x}{T + t}\right) - 1\right](T + t) + y \\ &= \frac{3}{2}C_{OPT} + \frac{1}{2}(T + t) - \frac{3}{2}x + y \\ &\leqslant \frac{3}{2}C_{OPT} + \frac{1}{2}(T + t) + y. \end{aligned} \tag{6}$$

On the other hand, it is clear that $y \leqslant T$. Combining it with (6), we obtain

$$C_{LPT} \leqslant \tfrac{3}{2}C_{OPT} + \tfrac{1}{2}(3T + t). \tag{7}$$

Now we are ready to obtain the worst-case ratio of the *LPT* algorithm.

Theorem 3. *For the problem* $1/nr - pm/C_{\max}$, *the worst-case ratio of the LPT algorithm is* 2.

Proof. We first claim that $b^* > 1$. Otherwise, we have $C_{LPT} = C_{OPT}$, and we are done. If $b = b^*$, then from (2) and (3), we see that

$$C_{LPT} = C_{OPT} + y - x \leqslant C_{OPT} + T < 2C_{OPT},$$

where the last inequality holds because $b^* > 1$, i.e., $T < C_{OPT}$. So, we assume in the following that $b > b^*$.

Case 1: $b^* \geqslant 4$. Thus, from (2), we have $C_{OPT} \geqslant 3(T + t) > 3T + t$. Combining it with (7), we obtain $C_{LPT} \leqslant 2C_{OPT}$.

Case 2: $b^* = 3$. Then, from Lemma 2, we know that in the *LPT* schedule, the total number of jobs in batches $B_{b^*+1}, B_{b^*+2}, \ldots, B_b$ is not greater than 2. Combining it with Lemma 1, we conclude that $y \leqslant \frac{2}{3}T$. As $b \leqslant \frac{3}{2}b^* = \frac{9}{2}$ and $b > b^*$, we know that $b = 4$. By (3) and $y \leqslant \frac{2}{3}T$, we have $C_{LPT} = 3(T + t) + y < 4(T + t)$. (2) states that $C_{OPT} = 2(T + t) + x > 2(T + t)$. Hence, we have $C_{LPT} < 2C_{OPT}$.

Case 3: $b^* = 2$. By the same reasoning as Case 2, we conclude that $b = 3$, the number of jobs in the 3rd batch of the *LPT* schedule is 1, and $y \leqslant T/3$. Thus, we have $C_{OPT} = (T + t) + x$ and $C_{LPT} = 2(T + t) + y$.

Denote $P = \sum_{i=1}^{n} p_i$. Let A be the total processing time of the jobs in the first batch of the optimal schedule, and B, C be the total processing times of the jobs in the first and the second batches of the *LPT* schedule, respectively. Then, we have $B + C + y = P = A + x$. Combining it with $A \leqslant T$, we have

$$B + C + y \leqslant T + x. \tag{8}$$

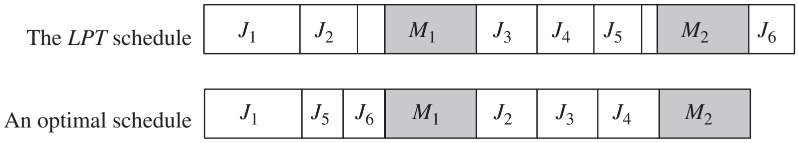

Fig. 2. An instance showing the tightness of the worst-case ratio.

On the other hand, by the *LPT* rule, we have $B + y > T$ and $C + y > T$, and hence $B + C + y > 2T - y$. Combining it with (8), we obtain $x > T - y$, and hence $x > \frac{2}{3}T$ (since $y \leqslant T/3$). Therefore, we obtain $C_{OPT} = (T + t) + x > \frac{5}{3}T + t$, and $C_{LPT} = 2(T + t) + y \leqslant \frac{7}{3}T + 2t$, implying $C_{LPT} < 2C_{OPT}$.

Hence, we have completed the proof that the worst-case ratio of the *LPT* algorithm is not greater than 2. To show that the ratio cannot be smaller than 2, consider the following instance: Let $T = 12$, $p_1 = 6$, $p_2 = p_3 = p_4 = 4$, $p_5 = p_6 = 3$, and t be an arbitrary integer. Applying *LPT*, we obtain $B_1 = \{J_1, J_2\}$, $B_2 = \{J_3, J_4, J_5\}$, $B_3 = \{J_6\}$, and the makespan is $C_{LPT} = 2t + 27$. However, an optimal solution has two batches, where the first batch contains J_1, J_5, J_6 while the second batch contains J_2, J_3, J_4. Hence, $C_{OPT} = t + 24$. It follows that $C_{LPT}/C_{OPT} = (2t + 27)/(t + 24) \to 2$ as $t \to \infty$. The *LPT* schedule and an optimal schedule are illustrated in Fig. 2. □

3. Non-approximability

It is well known that it is impossible to have a polynomial time approximation algorithm for the bin-packing problem with a worst-case ratio of less than $\frac{3}{2}$ unless $P = NP$ [13]. However, for our problem, the lower bound can be larger. In fact, we show that if there is an approximation algorithm with a worst-case ratio of $2 - \varepsilon$ for any $0 < \varepsilon < 1$, then it can be used to establish a polynomial time algorithm for solving the PARTITION problem, which is *NP*-hard [13], leading to a contradiction (if $P \neq NP$). Hence, such an algorithm for the considered problem cannot exist unless $P = NP$.

PARTITION: Given n positive integers $h_1, h_2, \ldots, h_n$ with $\sum_{i=1}^{n} h_i = 2H$, does there exist a set $U \subseteq \{1, 2, \ldots, n\}$ with $\sum_{i \in U} h_i = H$?

For any fixed positive number $\varepsilon < 1$ and an instance *I* of the PARTITION problem, we construct an instance *II* of our scheduling problem as follows: There are n jobs: $J_1, \ldots, J_n$, and the processing time of job J_i is $p_i = h_i$. Let the time of the maintenance period be $t = \lceil 2H(1 - \varepsilon)/\varepsilon \rceil$, and the time of the interval between two consecutive maintenance periods be $T = H$. It is clear that this construction can be performed in polynomial time. We first give some lemmas.

Lemma 4. *If there exists a solution to the instance I, then the optimal makespan for the instance II is* $C_{OPT} = 2H + t$.

Proof. Suppose that there exists such a subset U for the instance I such that $\sum_{i \in U} h_i = H$. We process jobs $\{J_i | i \in U\}$ in the first batch and all the remaining jobs in the second batch. Hence, the corresponding makespan equals $2H + t$, which achieves the trivial lower bound for the optimal makespan and is thus optimal. □

Lemma 5. *If there is no solution to the instance I, then the optimal solution value for the instance II satisfies the inequality* $C_{OPT} \geqslant 2H + 2t + 1$.

Proof. If there is no solution to the instance *I*, then it can easily be verified that the optimal schedule for instance *II* has to use at least three batches. Therefore, the optimal value $C_{OPT} \geqslant 2(T+t)+1 = 2(H+t)+1$. □

Lemma 6. *If there exists a polynomial time approximation algorithm* A_ε *with a worst-case ratio of* $2-\varepsilon$ *for some positive number* $\varepsilon < 1$, *then there exists a polynomial time algorithm for the PARTITION problem.*

Proof. Given any instance *I* of the PARTITION problem, we construct the corresponding instance *II* of our scheduling problem in polynomial time as above. Define an upper threshold $Z = 2(H+t)$. Then the instance *I* of the PARTITION problem can be answered by merely comparing the values of C_{A_ε} and Z.

To see this is the case, let us apply A_ε to the instance *II*. If $C_{A_\varepsilon} \leqslant Z = 2(H+t)$, then $C_{OPT} \leqslant C_{A_\varepsilon} \leqslant 2(H+t)$. By Lemma 5, we deduce that there is a solution to the instance *I* of the PARTITION problem. On the other hand, if $C_{A_\varepsilon} > Z$, since $C_{OPT} \geqslant C_{A_\varepsilon}/(2-\varepsilon)$ by the assumption that A_ε has a worst-case ratio of $2-\varepsilon$, and since $t = \lceil 2H(1-\varepsilon)/\varepsilon \rceil$ implies $\varepsilon \geqslant (2H/(2H+t))$, we have

$$C_{OPT} > \frac{2(H+t)}{2-\varepsilon} \geqslant \frac{2(H+t)}{2-(2H/(2H+t))} = 2H+t.$$

Combining it with Lemma 4, we deduce that there is no solution to the instance *I*.

Hence, we have shown that the schedule for the instance *II* produced by A_ε gives us a right answer about whether there exists a solution to the instance *I*. Since the times for constructing instance *II* and A_ε are all polynomial, A_ε can solve *I*, an arbitrary instance of PARTITION problem in polynomial time. This completes the proof. □

By Lemma 6, and the fact that any *NP*-hard problem cannot be solved by a polynomial time algorithm unless $P = NP$, we establish the following main result.

Theorem 7. *Unless* $P = NP$, *the LPT algorithm is the best possible polynomial time approximation algorithm for the problem* $1/nr - pm/C_{\max}$.

4. Conclusions

We showed that the worst-case ratio of the classical *LPT* algorithm is 2 for the problem $1/nr - pm/C_{\max}$. We also showed that 2 is the best possible for all the polynomial time approximation algorithms unless $P = NP$. So in future research, it is worth considering the design of approximation algorithms with a lower time complexity than the *LPT* algorithm, while maintaining the worst-case ratio of 2. It is also worth considering the problem with respect to other objectives and in parallel-machine systems.

Acknowledgements

This research was supported in part by the National Natural Science Foundation of China under Grant Numbers (10271110, 60021201); the Teaching and Research Award Program for Outstanding Young

Teachers in Higher Education Institutions of the MOE, China; and The Hong Kong Polytechnic University under Grant Number G-T997.

References

[1] Art RHPM, Knapp GM, Lawrence MJ. Some aspects of measuring maintenance in the process industry. Journal of Quality in Maintenance Engineering 1998;4:6–11.
[2] Liao CJ, Chen WJ. Single-machine scheduling with periodic maintenance and nonresumable jobs. Computers and Operations Research 2003;30:1335–47.
[3] Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG. Optimization and approximation in deterministic sequencing and scheduling: a survey. Annals of Operations Research 1979;5:287–326.
[4] Lee CY. Machine scheduling with an availability constraint. Journal of Global Optimization 1996;9:395–416.
[5] He Y, Ji M, Cheng TCE. Single machine scheduling with a restricted rate-modifying activity. Naval Research Logistics 2005;52:361–9.
[6] Lee CY, Liman SD. Single machine flow-time scheduling with scheduled maintenance. Acta Informatica 1992; 29:375–82.
[7] Sadfi C, Penz B, Rapine C, Błazewicz J, Formanowicz P. An improved approximation algorithm for the single machine total completion time scheduling problem with availability constraints. European Journal of Operational Research 2005; 161:3–10.
[8] He Y, Zhong WY, Gu HK. Improved algorithms for two single machine scheduling problems, AAIM 2005. Lecture Notes in Computer Science 2005;3521:66–76.
[9] Graves GH, Lee CY. Scheduling maintenance and semiresumable jobs on a single machine. Naval Research Logistics 1999;46:845–63.
[10] Schmidt G. Scheduling with limited machine availability. European Journal of Operational Research 2000;121:1–15.
[11] Simchi-Levi D. New worst-case results for the bin packing problem. Naval Research Logistics 1994;41:579–85.
[12] Baase S, Gelder AV. Computer algorithms: introduction to design and analysis. 3rd ed., USA: Addison-Wesley; 2000.
[13] Garey MR, Johnson DS. Computers and intractability: a guide to the theory of *NP*-completeness. New York: W.H. Freeman and Company; 1979.