The Tenth International Symposium on Operations Research and Its Applications (ISORA 2011)
Dunhuang, China, August 28–31, 2011

Scheduling Problem with Discretely Compressible Release Dates

Shu-Xia Zhang*[1] Bao-Qiang Fan[2]

[1]Department of Watercraft Command
Zhenjiang Watercraft College, Zhenjiang 212003, China
[2]Department of Mathematics and Information
Ludong University, Yantai, Shandong 264025, China

Abstract In this paper, we address the scheduling model with discretely compressible release dates, where processing any job with a compressed release date incurs a corresponding compression cost. We consider the following problem: scheduling with discretely compressible release dates to minimize the sum of makespan plus total compression cost. We show its NP-hardness, and design an approximation algorithm with worst-case performance ratio 2.

Keywords Scheduling; Discretely compressible release dates; NP-hardness; Approximation algorithm

1 Introduction

In this paper, we study the scheduling problem with discretely compressible release dates. In the classical scheduling, it is always assumed that the parameters of a job, say its processing time, its release date etc. are all fixed. However, in the real world, the processing of jobs is not only determined by the machine speed, but also by other resources such as labor, funds etc., therefore the parameters of a job may not be fixed. For example, we can compress the original release dates. Of course, processing any job with a compressed release date incurs a compression cost.

The scheduling problem with compressible release dates has its deep root in the real world. It commonly arises in manufacturing systems where the preprocessing of the jobs depends on a common resource such as fuel, catalyzer, raw materials, etc. Real-life examples of such problems are given in Janiak ([1], [2], [3]) in the context of steel production which involves preheating of iron ingots ([4]).

Let $J = \{1,2,\cdots,n\}$ denote a list of given jobs. We write SCR as an abbreviation of the scheduling problem with compressible release dates. We denote by TRC the total compression cost in SCR. There are two variants for SCR, the continuous one and the discrete one, which are denoted by SCCR and SDCR, respectively. In SCCR, any job J_j can be processed with a release date $r_j \in [l_j,u_j]$ and a corresponding compression cost $c_j(u_j - r_j)$ is incurred, where c_j is the cost coefficient. And in SDCR, the value of r_j can

*Corresponding author. E-mail: zsxhdsd@163.com

be selected from among $\{r_{j1}, r_{j2}, \cdots, r_{jk}\}$, and the corresponding compression costs are $e_{j1}, e_{j2}, \cdots, e_{jk}$.

There are the following four models for SCR:

(P1) To minimize $F_1 + F_2$;

(P2) To minimize F_1 subject to $F_2 \leq a$;

(P3) To minimize F_2 subject to $F_1 \leq b$;

(P4) To identify the set of Pareto-optimal points for (F_1, F_2).

Where F_1 is the original objective function, and F_2 is TRC. In the objective function field of the notation of Graham et al.([5]), we write the above four model as $F_1 + F_2$, F_1/F_2, F_2/F_1 and (F_1, F_2), respectively. We use cr and dr to characterize SCCR and SDCR, respectively.

Nowicki and Zdrzalka ([6]) showed that the problem $1|r_j, cr|C_{\max} + TRC$ is strongly NP-hard and designed an approximation algorithm with worst-case ratio 2 which is the best possible. Sun ([7]) concentrated on the problems $1|r_j, p_j = 1, cr|(C_{\max}, TRC)$, $1|r_j, p_j = 1, c_j = 1, cr|(L_{\max}, TRC)$ and $1|r_j, p_j = 1, c_j = 1, cr|(\sum \omega_j C_j, TRC)$. Sun and R.J.Kibet ([7]) also studied the problem $1|r_j, p_j = 1, cr|(\sum C_j, TRC)$. Cheng and Shakhlevich ([8]) studied the problems $1|r_j, p_j = 1, cr|C_{\max} + TRC$, $1|r_j, p_j = 1, cr|C_{\max}/TRC$, $1|r_j, p_j = 1, cr|TRC/C_{\max}$, and the cases with integer compression amounts for the above three problems. They also discussed the problems $1|r_j, p_j = 1, c_j = 1, cr|C_{\max} + TRC$, $1|r_j, p_j = 1, c_j = 1, cr|C_{\max}/TRC$ and $1|r_j, p_j = 1, c_j = 1, cr|TRC/C_{\max}$. And furthermore, they showed that $1|r_j, p_j = 1, cr|\sum w_j C_j + TRC$, $1|r_j, p_j = 1, cr|\sum w_j C_j/TRC$ and $1|r_j, p_j = 1, cr|TRC/\sum w_j C_j$ are all NP-hard in the ordinary sense.

The major work in the area of scheduling with compressible parameters considered either compressible processing times and fixed release dates ([11, 12, 14]), or compressible release dates and fixed processing times ([6, 7, 8, 10, 15, 16]), Cheng, Kovalyov and Shakhlevich ([13]) examined the problem of optimal scheduling the jobs on a single-machine when job processing times and release dates are conpressible parameters and the objective is to minimize the makespan together with the linear compression cost function. They constructed a reduction to the assignment problem for the case of equal release date compression costs and develop an $O(n^2)$ algorithm for the case of equal release date compression costs and the equal processing time compression costs.

In this paper, we address the P1 model for SDCR, $1|r_j, dr|C_{\max} + TRC$ following the notation of Graham et al. The rest of this paper is organized as follows. Some preliminaries are given in Section 2. In section 3, we show it is strongly NP-hard. In section 4, we present an approximation algorithm with worst-case ratio 2. Conclusion and remarks are given in Section 5.

2 Preliminaries

An algorithm A is a ρ-approximation algorithm for a minimization problem if it produces a solution which is at most ρ times the optimal one, with a running time bounded by a polynomial in the input size. We also say ρ is the worst-case ratio of algorithm A. The worst-case ratio is the most frequently used measure for the quality of an approximation algorithm for a scheduling problem: the smaller the ratio is, the better the approximation algorithm is.

In the following, we introduce some basic notations and formulate the problem considered in this paper. Each job $j \in J$ is characterized by a processing time p_j, a set of k ($k \geq 1$ and is given) potential release dates: $r_{j1}, r_{j2}, \cdots, r_{jk}$ with $r_{j1} > r_{j2} > \cdots > r_{jk}$, where r_{j1} is called the normal release date and r_{jk} the minimum possible release date, and a set of k potential compression costs: $e_{j1}, e_{j2}, \cdots, e_{jk}$. Whenever r_{ji}, $j \in J, 1 \leq i \leq k$, is selected as the actual release date of job j, a corresponding compression cost e_{ji} is paid. Our work is to select a release date for each job and schedule these jobs such that the objective value $C_{\max} + TRC$ is minimized. It is also assumed that $0 = e_{j1} < e_{j2} < \cdots < e_{jk}$. This assumption is sound as the more release date we compress the more cost we should pay.

Let $r = (r_1, r_2, \cdots, r_n)$ be a vector of actual release dates where $r_j \in \{r_{j1}, r_{j2}, \cdots, r_{jk}\}$, and π, a schedule of J defining the job processing order. We denote by R the set of all feasible actual release dates, $R = \{r : r_j \in (r_{j1}, r_{j2}, \cdots, r_{jk})\}$, and by Π the set of all schedules of J. We assume that a pair (r, π) uniquely determines a completion time $C_j(r, \pi)$ of each job $j \in J$. Two general performance measures are considered in our problem in this paper. The first one is based on the completion time $C_j(r, \pi)$ and therefore it will be called a completion cost. Given r and π, the completion cost, which is denoted by $F_1(r, \pi)$, is the maximum completion time of all the jobs, i.e. $F_1(r, \pi) = \max\{C_j(r, \pi) : 1 \leq j \leq n\}$. The second measure is the total cost of compressions $F_2(r) = \sum_{j=1}^{n} e_{ji} (1 \leq i \leq k)$. $F_2(r)$ will be called a compression cost. For a pair (r, π), a total scheduling cost $M(r, \pi)$ is defined by $M(r, \pi) = F_1(r, \pi) + F_2(r)$. Denote by $r_{\pi(j), \sigma(j, \pi)}$ the actual release date of job j in π, where $\pi(j)$ is the jth processed job in π and $\sigma(j, \pi)$ the index of its release time, $1 \leq \sigma(j, \pi) \leq k$. The problem considered in this paper is formulated as follows. Find $\pi^* \in \Pi$ and $r^* \in R$ minimizing

$$M(r, \pi) = \max_{1 \leq j \leq n} (r_{\pi(j), \sigma(j, \pi)} + \sum_{t=j}^{n} p_{\pi(t)}) + \sum_{j=1}^{n} e_{\pi(j), \sigma(j, \pi)} \tag{1}$$

subjectto $r \in R$ and $\pi \in \Pi$.

3 Proof of NP-hardness

Nowicki and Zdrzalka ([6]) showed that the problem $1|r_j, cr|C_{\max} + TRC$ is strongly NP-hard by a reduction to the problem $1||\sum \omega_j T_j$, which is strongly NP-hard. We remark that their proof also applies to the special case that all the parameters are nonnegative integers. Next, we will prove by reduction to this case that $1|r_j, dr|C_{\max} + TRC$ is also strongly NP-hard.

Theorem 1. *$1|r_j, dr|C_{\max} + TRC$ is strongly NP-hard.*

Proof. For any integral instance of $1|r_j, cr|C_{\max} + TRC$, $I = (p_1, p_2, \cdots, p_n; l_1, l_2, \cdots, l_n; u_1, u_2, \cdots, u_n, c_1, c_2, \cdots, c_n)$, where all the numbers are nonnegative integers, construct an instance I' of $1|r_j, dr|C_{\max} + TRC$ as follows: there are also n jobs in total, for each job j, $1 \leq j \leq n$, its release time has $u_j - l_j + 1$ choices, $r_{j1} = u_j, r_{j2} = u_j - 1, \cdots, r_{j, u_j - l_j + 1} = l_j$, and the corresponding costs are $e_{j1} = 0, e_{j2} = c_j, \cdots, e_{j, u_j - l_j + 1} = (u_j - l_j) c_j$.

Obviously I and I' have exactly the same optimal objective values as well as the same optimal schedules. Since the construction can be done in pseudo-polynomial time of the size of I, the proof is done. □

4 An approximation algorithm

In what follows, we discuss approximation algorithms for our problem. First note that compressing the release date may decrease the completion times of the jobs but incurs additional costs, so we here assume without loss of generality that $e_{j2} < r_{j1} - r_{j2}, e_{j3} < r_{j1} - r_{j3}, \cdots, e_{jk} < r_{j1} - r_{jk}$ for any job $j \in J$.

Algorithm $G(General\ approximation\ algorithm)$

Step 1. Choose a schedule π^G.
Step 2. Determine r^G minimizing $M(r, \pi^G)$ subject to $r \in R$.

Suppose that π^G is chosen according to one of the following rules: arbitrary schedule (G0), nondecreasing r_{j1} (G1), nondecreasing r_{jk} (G2), nondecreasing $r_{jk} + e_{jk}$ (G3), nondecreasing $r_{j1} + p_j$ (G4), nondecreasing $r_{jk} + p_j$ (G5), nondecreasing $r_{jk} + e_{jk} + p_j$ (G6), nonincreasing p_j (G7), nonincreasing $r_{j1} + p_j$ (G8), nondecreasing $(r_{jk} + e_{jk})/p_j$ (G9). If there is a choice in G1-G3, then take the job with the largest processing time.

For the minimizing problem in Step 2, we design an optimal algorithm OMC. Before describing it, we state the following lemma which plays an important role in design and analysis of the algorithm OMC.

Lemma 2. *There always exists an optimal schedule with the following properties:*
(1)*There is no idle time between adjacent jobs.*
(2)*There exists at least one job whose release time is exactly its start time (we call this job critical job).*

Proof sketch. If there exists some idle time between two adjacent blocks, here *block* means a largest possible set of jobs in the optimal schedule with the property that the completion time of the former job is just the start time of the latter one, we can just postpone the start time of the former block such that its completion time is exactly the start time of the latter block, and so that there is no idle time between these two blocks. Obviously this will not change the objective function value, thus property (1) holds. Property (2) is straightforward. □

Obviously, after determining the critical job and its release date, we can easily calculate the objective value of the schedule with no idle time between adjacent jobs, which is also the main idea of our optimal algorithm. To simplify our discussion, we may assume without loss of generality that the jobs are re-indexed so that $\pi^G = (1, 2, \cdots, n)$ after choosing π^G in Step 1. Suppose job J_{j_0} is the critical job we determine, and its actual release date is denoted by $r_{j_0 i_0}$.

For $1 \le x < j_0$, we define

$$r(j_0, i_0, x) = \max\{r_{xi} : 1 \le i \le k, r_{xi} \le r_{j_0,i_0} - \sum_{t=x}^{j_0-1} p_t\}$$

For $j_0 < y \le n$, we define

$$R(j_0, i_0, y) = \max\{r_{yi} : 1 \le i \le k, r_{yi} \le r_{j_0,i_0} + \sum_{t=j_0}^{y-1} p_t\}$$

Denote $c(j_0,i_0,x)$ and $c(j_0,i_0,y)$ as the corresponding indexes of release dates, namely $r(j_0,i_0,x) = r_{x,c(j_0,i_0,x)}$, $R(j_0,i_0,y) = r_{y,c(j_0,i_0,y)}$.

Algorithm OMC(*Optimal minimum cost*)

Input an instance $\mathscr{J}$.
Let $H_{j_0} := \infty$, $B_{j_0} := \infty$
For $j_0 = 1$ **to** n **do**
  For $i_0 = 1$ **to** k **do**
  Calculate $c(j_0,i_0,1), \cdots, c(j_0,i_0,j_0-1), c(j_0,i_0,j_0+1), \cdots, c(j_0,i_0,n)$
   If all $c(j_0,i_0,h)(h \in [1, j_0-1] \cup [j_0+1, n])$ exist **then** let

$$H_{j_0} := \min\{H_{j_0}, \sum_{x=1}^{j_0-1} e_{x,c(j_0,i_0,x)}\},$$

$$B_{j_0} := \min\{B_{j_0}, r_{j_0 i_0} + \sum_{t=j_0}^{n} p_t + e_{j_0} + \sum_{y=j_0+1}^{n} e_{y,c(j_0,i_0,y)}\}$$

  EndFor
EndFor
Output $\min\{H_i + B_i : 1 \le i \le nk\}$ and the corresponding schedule.

It's not hard to calculate that the running time is $O(n^2k)$.

In what follows we show results of the worst-case analysis for various variants of **Algorithm** G, obtained from the general scheme by applying various rules for choosing π^G in Step 1. The rules G3, G6, G9 are justified by the following lower bound. By (1), we get

$$M(r^*,\pi^*) \ge \min_{\pi\in\Pi} \max_{1\le j\le n} \{ \min_{1\le\sigma(j)\le k} (r_{j,\sigma(j)} + \sum_{t=j}^{n} p_t + \sum_{j=1}^{n} e_{j,\sigma(j)})\}$$
$$\ge \min_{\pi\in\Pi} \max_{1\le j\le n} (r_{jk} + e_{jk} + \sum_{t=j}^{n} p_t)$$

Thus we can get the lower bound on M^* by scheduling the jobs in the order of nondecreasing $r'_{jk} = r_{jk} + e_{jk}$, in the classical problem $1|r'_{jk}|C_{\max}$.

Theorem 3. *The worst-case ratio of* **Algorithm** *G is 2.*

Proof. Denote $C_{\max}(r,\pi) = \max_{1\le j\le n} (r_{j,\sigma(j)} + \sum_{t=j}^{n} p_t)$. Let $\tilde{\pi}(r)$ be a schedule minimizing $C_{\max}(r,\pi)$ over $\pi \in \Pi$. For any $\pi \in \Pi$, $C_{\max}(r,\pi) \le \max_{1\le j\le n} r_{j,\sigma(j)} + \sum_{j=1}^{n} p_j$, and $C_{\max}(r,\tilde{\pi}(r)) \ge \max_{1\le j\le n} r_{j,\sigma(j)}$, and $C_{\max}(r,\tilde{\pi}(r)) \ge \sum_{j=1}^{n} p_j$. We have the following observation, i.e., for any $\pi \in \Pi$,

$$C_{\max}(r,\pi) \le 2C_{\max}(r,\tilde{\pi}(r)) \tag{2}$$

Since $M(r^*,\pi^*) = M(r^*,\tilde{\pi}(r^*))$, we thus get from (2) that

$$\begin{aligned} M^G &\leq M(r^*,\pi^G) = C_{\max}(r^*,\pi^G) + \sum_{j=1}^{n} e^*_{j,\sigma(j)} \\ &\leq 2C_{\max}(r^*,\tilde{\pi}(r^*)) + \sum_{j=1}^{n} e^*_{j,\sigma(j)} \\ &\leq 2M(r^*,\pi^*) = 2M^* \end{aligned}$$

□

5 Conclusion and remarks

In this paper, we have discussed the scheduling problem with discretely compressible release times. This model is of interest both in the real world and in the sense of theory and it has attracted relatively little attention compared with traditional scheduling problems. We address the P1 model for scheduling with discretely compressible release times to minimize makespan. We show that it is strongly NP-hard, and present an approximation algorithm with worst-case ratio 2.

Acknowledgements

This research was supported by the National Science Foundation of China under Grant No. 11001117.

References

[1] A. Janiak: Time-optimal control in a single machine problem with resource constraints. *Automatica*, **22**, 1986, 745-747.

[2] A. Janiak: Single machine scheduling problem with a common deadline and resource dependent release dates. *European Journal of Operations Research*, **53**, 1991, 317-325.

[3] A. Janiak: Single machine sequencing with linear models of release dates. *Naval Research Logistics*, **45**, 1998, 99-113.

[4] T.J. Williams: *Analysis and Design of Hierarchical Control Systems: with Special Reference of Steel Plant Operations*, 1986, North-Holland, Amsterdam.

[5] R.L. Graham, E.L. Lawler, J.K. Lenstra and A.H.G. Rinnooy Kan: Optimization and approximation in deterministic sequencing and scheduling. *Annals of Discrete Mathematics*, **5**, 1979, 287-326.

[6] E.Nowicki and S. Zdrzalka: A survey of results for sequencing problems with controllable processing times. *Discrete Applied Mathematics*, **26**, 1990, 271-287.

[7] G. Tang, F. Zhang, Sh. Luo and L. Liu: Theory of modern scheduling. *Shanghai Science Generalization Publisher*, 2003, 30-33.

[8] T.C.E. Cheng, N. V. Shakhlevich : Single machine scheduling of unit-time jobs with controllable release dates. *Journal of Global Optimization*, **27**, 2003, 293-311.

[9] Z. Chen, Q. Lu and G. Tang: Single machine scheduling with discretely controllable processing times. *Operations Research Letters*, **21**, 1997, 69-76.

[10] T.C.E. Cheng, A. Janiak: Resource optimal control in some single-machine scheduling problems.*IEEE Transactions on Automatic Control*, **39**, 1994, 1243-1246.

[11] T.C.E. Cheng, A. Janiak: A permutation flow-shop scheduling problem with convex models of operation processing times. *Annals of Operations Research*, **96**, 2000, 39-60.

[12] T.C.E. Cheng, A. Janiak, M.Y.Kovalyov: Bicriterion single machine scheduling with resource dependent processing times. *SIAM Journal of Optimization*, **8**, 1998, 617-630.

[13] T.C.E. Cheng, M.Y.Kovalyov, N.V.Shakhlevich: scheduling with controllable release dates and processing times:Makespan minimization. *European Journal of Operations Research*, **175**, 2006, 751-768.

[14] A. Janiak, M.Y.Kovalyov, W.Kubiak, F.Werner: Positive half-products and scheduling with controllable processing times. *European Journal of Operations Research*, **165**, 2005, 416-422.

[15] A. Janiak, Ch.-L.Li: Scheduling to minimize the total weighted completion time with a constraint on the release time resource consumption. *Mathematics of Computation Modelling*, **20**, 1994, 53-58.

[16] A. Janiak: Computational complexity analysis of single machine scheduling problems with job release dates dependent on resources. *In Proceedings of the Symposium on Operations Research (SOR96), Braunschweig, September 3-6*, Springer-Verlag, 1997, 203-207.

[17] Z. Cao, Z. Wang, Y. Zhang and S. Liu: On several scheduling problems with rejection or discretely compressible processing times. *Lecture Notes in Computer Science*, **3959**, 2006, 90-98.