Asian Research Journal of Mathematics

18(4): 1-6, 2022; Article no.ARJOM.85321

ISSN: 2456-477X

A Single-machine Scheduling with Generalized Due Dates to Minimize Total Weighted Late Work

Myoung-Ju Park [a] **and Byung-Cheon Choi** [b*]

[a] *Department of Industrial and Management Systems Engineering, Kyung Hee University, 1732, Deogyeong-daero, Giheung-gu, Yongin-si, Kyunggi-do 17104, Korea.*
[b] *School of Business, Chungnam National University, 99 Daehak-ro, Yuseong-gu, Daejeon 34134, Korea.*

Authors' contributions

This work was carried out in collaboration between both authors. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/ARJOM/2022/v18i430368

Open Peer Review History:

This journal follows the Advanced Open Peer Review policy. Identity of the Reviewers, Editor(s) and additional Reviewers, peer review comments, different versions of the manuscript, comments of the editors, etc are available here: https://www.sdiarticle5.com/review-history/85321

Received: 02 February 2022
Accepted: 05 April 2022
Published: 11 April 2022

Short Communication

Abstract

In the paper, we consider a single-machine scheduling problem with generalized due dates, in which the objective is to minimize total weighted work. This problem was proven to be NP-hard by Mosheiov et al. [1]. However, the exact complexity remains open. We show that the problem is strongly NP-hard, and is weakly NP-hard if the lengths of the intervals between the consecutive due dates are identical.

Keywords: Scheduling; total late work; generalized due dates; computational complexity.

1 Introduction

Consider a scheduling problem such that the due date is assigned not to the specific job but to the job position. Such a due date is referred to as the *generalized due date (GDD)*. Since the scheduling problem with GDD was initiated from Hall [2], much research has been done in [3, 4, 5, 6, 7, 8]. Recently, Mosheiov et al. [1] considered single-machine scheduling problems with GDD to minimize total late work. They showed that the problem can be solved by the Shortest Processing Time first

**Corresponding author: E-mail: polytime@cnu.ac.kr;*

(SPT) rule, while it is NP-hard if each job has a different weight. Note that it is unknown whether the case with the different weights is strongly NP-hard or not. We establish the exact complexity for the case with the different weights.

The remainder of this paper is organized as follows. Sections 2 and 3 defines the problem formally and establishes the computational complexity.

2 Problem Definition

Our problem can be formally stated as follows: For each job $j \in \mathcal{J} = \{1, 2, ..., n\}$, let p_j and w_j be the processing time and the weight, respectively. Let $\pi = \Big(\pi(1), \pi(2), ..., \pi(n)\Big)$ be a schedule, where $\pi(j)$ is the jth job. For each $j \in \mathcal{J}$, let $S_j(\pi)$ and $C_j(\pi)$ be the start and completion times of job j in π, respectively, and $\pi^{-1}(j)$ be the position of job j in π. In our model, unlike the traditional scheduling problem, the due date d_i is assigned not to the specific job, but to the job positioned ith for each due date $i \in \mathcal{D} = \{1, 2, ..., n\}$. For simplicity, assume that $d_0 = 0$ and

$$d_1 \leq d_2 \leq \cdots \leq d_n.$$

GDD has two special cases depending on the condition of the due dates. The first and the second cases have a common due date with

$$d_i = d \quad \text{for} \quad i \in \mathcal{D}, \tag{2.1}$$

and identical lengths of the intervals between the consecutive due dates, that is,

$$d_i = i\delta \quad \text{and} \quad d_i - d_{i-1} = \delta \quad \text{for} \quad i \in \mathcal{D}, \tag{2.2}$$

respectively. Let the due dates with relations (2.1) and (2.2) be referred to as the *common due dates* (CDD) and *periodic due dates* (PDD), respectively. For each $j \in \mathcal{J}$, let $T_j(\pi)$ and $Y_j(\pi)$ be the tardiness and late work of a job j in π, respectively, which are calculated as

$$T_j(\pi) = \max\big\{0, L_j(\pi)\big\} \quad \text{and} \quad Y_j(\pi) = \min\big\{p_j, T_j(\pi)\big\},$$

where $L_j(\pi) = C_j(\pi) - d_{\pi^{-1}(j)}$. The objective is to find a schedule π to minimize total weighted late work, which is calculated as

$$z(\pi) = \sum_{j \in \mathcal{J}} w_j Y_j(\pi).$$

We follows the standard three-field notation $1|\beta|\sum_{j\in\mathcal{J}} w_jY_j$ introduced by Graham et al. [9], where $\beta \in \{CDD, PDD, GDD\}$ describes the characteristics of the due dates. This paper establishes the complexities of three cases.

Table 1 summarizes our results (note that 'wNP-hard' and 'sNP-hard' stand for weakly and strongly NP-hard, respectively).

Table 1. Complexity for $1|\beta|\gamma$

$\gamma \setminus \beta$	CDD	PDD	GDD
$\sum w_jT_j$	wNP-hard [10, 8]	wNP-hard [3]	sNP-hard [4]
$\sum w_jY_j$	polynomially solvable [1]	wNP-hard (Cor. 3.2)	sNP-hard (Thm. 1)

3 Computational Complexity

In this section, we show that $1|GDD|\sum w_jY_j$ and $1|PDD|\sum w_jY_j$ are strongly and weakly NP-hard, respectively.

Theorem 1. $1|GDD|\sum w_jY_j$ is strongly NP-hard.

Proof Gao and Yuan [4] showed that $1|GDD|\sum w_jT_j$ is strongly NP-hard. It is observed from the reduced instance in their proof that $T_j = Y_j$ holds for each job $j \in \mathcal{J}$ in the optimal schedule. Thus, $1|GDD|\sum w_jY_j$ is strongly NP-hard. ■

Theorem 2. $1|PDD|\sum w_jY_j$ is NP-hard.

Proof For simplicity, for $1|CDD|\sum w_jT_j$, let $\bar{p}_j$ and $\bar{w}_j$ be the processing time and weight of job $j \in \{1, 2, ..., n\}$, respectively, and d be the common due date. Yuan [8] showed that $1|CDD|\sum w_jT_j$ is NP-hard, even if

$$\sum_{j=1}^{n} \bar{p}_j \leq 2d + 1. \tag{3.1}$$

Given an instance of $1|CDD|\sum w_jT_j$, we can construct an instance of $1|PDD|\sum w_jY_j$ with $(n+1)$ jobs in $\mathcal{J} = \{0, 1, ..., n\}$ such that

- $p_0 = 0$ and $w_0 = 1 + \sum_{j=1}^{n} \bar{w}_j$;
- $p_j = d + \bar{p}_j$ and $w_j = \bar{w}_j$, $j = 1, 2, ..., n$;
- $\delta = d$.

It is observed that job 0 is processed at the first position in any optimal schedule for the reduced instance of $1|PDD|\sum w_jY_j$. Thus, we consider only a schedule π for the reduced instance with $\pi(1) = 0$, that is, a schedule $\pi = (0, \bar{\pi})$, where $\bar{\pi}$ is the schedule for a given instance of $1|CDD|\sum w_jT_j$. Note that the kth job in $\bar{\pi}$ is the $(k+1)$th job in π. Then, we have

$$C_{\pi(k+1)}(\pi) = \sum_{h=2}^{k+1} p_{\pi(h)} = \sum_{h=1}^{k} (d + p_{\bar{\pi}(h)}) = kd + C_{\bar{\pi}(k)}(\bar{\pi}), \tag{3.2}$$

where the first equality holds due to $p_{\pi(1)} = 0$. If job j is the kth job in $\bar{\pi}$, then we have, by equation (3.2),

$$L_j(\pi) = kd + C_{\bar{\pi}(k)}(\bar{\pi}) - (k+1)\delta = C_j(\bar{\pi}) - d = L_j(\bar{\pi})$$

and

$$T_j(\pi) = T_j(\bar{\pi}).$$

By inequality (3.1), we have $T_j(\bar{\pi}) \leq \sum_{j=1}^{n} \bar{p}_j - d \leq d + 1 \leq d + \bar{p}_j$. Then

$$Y_j(\pi) = \min\{p_j, T_j(\pi)\} = \min\{d + \bar{p}_j, T_j(\bar{\pi})\} = T_j(\bar{\pi}).$$

Since job 0 is not tardy in π and $w_j = \bar{w}_j$, $j = 1, 2, ..., n$, the objective values of the two schedules π and $\bar{\pi}$ in each instance are the same. This implies that $1|CDD|\sum w_jT_j$ is special case of $1|PDD|\sum w_jY_j$. Thus, Theorem 2 holds. ■

Let a job j be referred to as *small* if $p_j \leq \delta$, and *large*, otherwise. Let $\mathcal{S}$ and $\mathcal{L}$ be the sets of small and large jobs, respectively. Let

$$a_j = \begin{cases} \delta - p_j & \text{for } j \in \mathcal{S} \\ p_j - \delta & \text{for } j \in \mathcal{L}. \end{cases}$$

Furthermore, let a_j be referred to as *auxiliary processing time* for $j \in \mathcal{L}$. Under a schedule π, let a job j be referred to as *early* if $Y_j(\pi) = 0$, *partially late* if $0 < Y_j(\pi) < p_j$, and *fully late* if $Y_j(\pi) = p_j$. In $1|PDD|\sum w_jY_j$, an optimal schedule π can be represented as

$$\pi = (\pi_s, \pi_e, \pi_p, \pi_f),$$

where π_s, π_e, π_p and π_f are sequences of small, early, partially late, and fully late jobs, respectively. Furthermore, the jobs in π_i for $i \in \{s, e, f\}$ are sequenced arbitrarily. By Observation 3, henceforth, we construct only a schedule for large jobs. Let $d = \sum_{j\in\mathcal{S}} a_j$ and $[h]$ be the hth large job in π. Note that

$$T_{[h]}(\pi) = \max\left\{0, \sum_{i=1}^{h} a_{[i]} - d\right\} \quad \text{and} \quad Y_{[h]}(\pi) = \min\left\{p_{[h]}, T_{[h]}(\pi)\right\}. \tag{3.3}$$

Let $\mathcal{P}$ and x be the set of partially late jobs and the first partially late job in the optimal schedule, respectively. Let x be referred to as a *straddling* job.

Lemma 1. In an optimal schedule π, jobs in $\mathcal{P} \setminus \{x\}$ are sequenced in non-increasing order of w_j/a_j.

Proof Suppose that there exist two jobs $i = [k]$ and $j = [k+1]$ in $\mathcal{P} \setminus \{x\}$ with

$$\frac{w_i}{a_i} < \frac{w_j}{a_j}. \tag{3.4}$$

Note that by $[k-1] \in \mathcal{P}$, $T_{[k-1]}(\pi) > 0$. Then, by $\{i, j\} \subset \mathcal{P}$ and (3.3),

$$w_iY_i(\pi) + w_jY_j(\pi) = w_i\big(T_{[k-1]}(\pi) + a_i\big) + w_j\big(T_{[k-1]}(\pi) + a_i + a_j\big). \tag{3.5}$$

Let $\bar{\pi}$ be the schedule constructed by interchanging the positions of jobs i and j from π. Then,

$$w_jY_j(\bar{\pi}) + w_iY_i(\bar{\pi}) \leq w_j\big(T_{[k-1]}(\pi) + a_j\big) + w_i\big(T_{[k-1]}(\pi) + a_j + a_i\big). \tag{3.6}$$

By (3.4)-(3.6), we have

$$z(\pi) - z(\bar{\pi}) \geq w_ja_i - w_ia_j > 0.$$

This contradicts to the optimality of π. ■

Theorem 3. $1|PDD|\sum w_jY_j$ can be solved in pseudo-polynomial time.

Proof We present a DP based on Observation 3 and Lemma 1. Suppose that in an optimal schedule, the auxiliary processing time and the weight of the straddling job x are a and w, respectively. Renumber the remaining large jobs such that

$$\frac{w_1}{a_1} \geq \frac{w_2}{a_2} \geq \cdots \geq \frac{w_m}{a_m},$$

where $m = |\mathcal{L}| - 1$. Then, we construct a schedule of jobs in $\{1, 2, ..., m\}$ by applying Algorithm 3.1. For each $k \in \{1, 2, ..., m\}$, the kth phase of Algorithm 3.1 produces a set $\mathcal{S}_k$ of states. Each state in $\mathcal{S}_k$ is expressed as a vector $S = [s_1, s_2, s_3, s_4, s_5]$ representing the information of a partial schedule for the first k jobs, where

· The component s_1 is total auxiliary processing time of early jobs;

· The components s_2 and s_3 are total auxiliary processing time and total weight of partially late jobs, respectively;

· The component s_4 is the last partially late job in the current partial schedule;

· The component s_5 is total weighted late work of a partial schedule.

The initial set $\mathcal{S}_0$ contains only one state $[0, 0, 0, 0, 0]$. For each $k \in \{1, 2, ..., m\}$, $\mathcal{S}_k$ is obtained from $\mathcal{S}_{k-1}$ through three mappings, F_1, F_2, and F_3, which translate $S := [s_1, s_2, s_3, s_4, s_5] \in \mathcal{S}_{k-1}$ into the states in $\mathcal{S}_k$ as follows:

i) Calculate F_1 defined by

$$F_1(a_k, w_k, S) = [s_1, s_2, s_3, s_4, s_5 + w_k(a_k + \delta)].$$

Note that job k becomes a fully late job through mapping F_1;

ii) Calculate F_2 defined by

$$F_2(a_k, w_k, S) = [s_1, s_2 + a_k, s_3 + w_k, k, s_5 + w_k(s_2 + a_k)].$$

Note that job k becomes a partially late job through mapping F_2;

iii) If $s_1 + a_k < d$, then calculate F_3 defined by

$$F_3(a_k, w_k, S) = [s_1 + a_k, s_2, s_3, s_4, s_5].$$

Note that job k becomes an early job through mapping F_3.

After completing the mth phase, we place the straddling job x if jobs x and s_4 can be the first and last partially late jobs, respectively. That is, shift all (partially and fully) late jobs to the right by $(s_1 + a - d)$ and insert the straddling job x on interval $[s_1, s_1 + a]$ if the state $S \in \mathcal{S}_m$ belongs to the following set from (3.3):

$$\mathcal{Q} = \{S \in \mathcal{S}_m \mid s_1 \leq d < s_1 + a \text{ and } \delta \leq s_1 + a + s_2 - d < a_{s_4} + \delta\}.$$

At this time, total weighted late work of a feasible schedule is calculated as

$$G(S) = s_5 + (s_3 + w)(s_1 + a - d) \quad \text{for} \quad S \in \mathcal{Q}.$$

Algorithm 3.1 outputs a schedule with the minimum $G(S)$ among $S \in \mathcal{Q}$.

Algorithm 3.1 (t). $\mathcal{S}_0 \leftarrow \{[0, 0, 0, 0, 0]\}$ $k \leftarrow 1$ ***to*** m ***each*** $S := [s_1, s_2, s_3, s_4, s_5] \in \mathcal{S}_{k-1}$ $\mathcal{S}_k \leftarrow \mathcal{S}_k \cup F_1(a_k, w_k, S) \cup F_2(a_k, w_k, S) \cup F_3(a_k, w_k, S)$ $\mathcal{Q} = \{S \in \mathcal{S}_m \mid s_1 \leq d < s_1 + a$ *and* $\delta \leq s_1 + a + s_2 - d < a_{s_4} + \delta\}$ ***each*** $S := [s_1, s_2, s_3, s_4, s_5] \in \mathcal{Q}$ $G(S) \leftarrow s_5 + (s_3 + w)(s_1 + a - d)$ $\min\{G(S) \mid S \in \mathcal{Q}\}$ *DP for* $1|PDD|\sum w_j Y_j$ *with a fixed straddling job.*

Note that the number of states in the algorithm is bounded by $O(lA^2WT)$, where $l = |\mathcal{L}|$, $A = \sum_{j\in\mathcal{L}} a_j$, $W = \sum_{j\in\mathcal{L}} w_j$, and $T = \sum_{j\in\mathcal{L}} w_j p_j$. Hence, Algorithm 3.1 is a pseudo-polynomial algorithm. Since the possible number of straddling job is l, $1|PDD|\sum w_j Y_j$ can be solved in pseudo-polynomial time. ■

Corollary 3.2. $1|PDD|\sum w_j Y_j$ *is weakly NP-hard.*

Proof It immediately holds by Theorems 2 and 3. ■

4 Concluding Remarks

We consider a single-machine scheduling problem with generalized due dates and total weighted late work criterion. Although the problem has been known to be NP-hard, its exact complexity is not established. We prove its strong NP-hardness, and weak NP-hardness of the case with periodic due dates.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Mosheiov G, Oron D, Shabtay D. Minimizing total late work on a single machine wiht generalized due dates. European Journal of Operational Research. 2021;293:837-846.

[2] Hall NG. Scheduling problems with generalized due dates. IIE Transactions. 1986;18:220-222.

[3] Choi BC, Kim KM, Min Y, Park MJ. Scheduling with generalized and periodic due dates under single- and two-machine environments. Optimization Letters. 2022;16:623-633.

[4] Gao Y, Yuan JJ. Unary NP-hardness of minimizing total weighted tardiness with generalized due dates. Operations Research Letters. 2016;44:92-95.

[5] Hall NG, Sethi SP, Sriskandarajah C. On the complexity of generalized due date scheduling problems. European Journal of Operational Research. 1991;51:100-109.

[6] Park MJ, Choi BC, Kim KM, Min Y. Two-machine ordered flow shop scheduling wiht generalized due dates. Asia-Pacific Journal of Operational Research. 2020;37:1-16.

[7] Srikandarajah S. A note on the generalized due dates scheduling Problem. Naval Research Logistics. 1990;37:587-597.

[8] Yuan JJ. The NP-hardness of the single machine common due date weighted tardiness problem. Journal of Systems Science and Complexity. 1992;5:328-333.

[9] Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG. Optimization and approximation in deterministic sequencing and scheduling: A survey. Annals of Discrete Mathematics. 1979;3:287-326.

[10] Lawler EL, Moore JM. A functional equation and its application to resource allocation and sequencing problems. Management Science. 1969;16:77-84.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
https://www.sdiarticle5.com/review-history/85321