Ann Oper Res (2013) 207:121–136
DOI 10.1007/s10479-012-1288-x

Single machine total tardiness maximization problems: complexity and algorithms

Evgeny R. Gafarov · Alexander A. Lazarev · Frank Werner

Published online: 28 December 2012

Abstract In this paper, we consider some scheduling problems on a single machine, where weighted or unweighted total tardiness has to be maximized in contrast to usual minimization problems. These problems are theoretically important and have also practical interpretations. For the total weighted tardiness maximization problem, we present an NP-hardness proof and a pseudo-polynomial solution algorithm. For the unweighted total tardiness maximization problem with release dates, NP-hardness is proven. Complexity results for some other classical objective functions (e.g., the number of tardy jobs, total completion time) and various additional constraints (e.g., deadlines, weights and/or release dates of jobs may be given) are presented as well.

Keywords Scheduling · Single machine maximization problems · Complexity · Dynamic programming

1 Introduction

Most scheduling problems consider the minimization of a specific objective function. For instance, the minimization of makespan is a very popular optimization criterion. The minimization of a sum function such as total completion time, total tardiness or the number of

E.R. Gafarov · A.A. Lazarev
Institute of Control Sciences of the Russian Academy of Sciences, Profsoyuznaya st. 65, 117997 Moscow, Russia

E.R. Gafarov
e-mail: axel73@mail.ru

A.A. Lazarev
e-mail: jobmath@mail.ru

F. Werner (✉)
Fakultät für Mathematik, Otto-von-Guericke-Universität Magdeburg, PSF 4120, 39016 Magdeburg, Germany
e-mail: frank.werner@mathematik.uni-magdeburg.de

tardy jobs are other typical optimization criteria. In this paper, we consider single machine problems with an *opposite* criterion, namely we consider the maximization of total tardiness, the maximization of the number of tardy jobs and the maximization of total completion time.

The problems under consideration can be formulated as follows. We are given a set $N = \{1, 2, \ldots, n\}$ of n independent jobs that must be processed on a single machine. Job preemption is not allowed. The machine can handle only one job at a time. For each job $j \in N$, a processing time $p_j > 0$, a due date $d_j \geq 0$, a weight $w_j \geq 0$, a release date $r_j \geq 0$ (i.e., the earliest possible starting time) and/or a deadline $D_j > 0$ (i.e., the latest possible completion time) can be given.

Each feasible schedule starts at time 0 and does not have any idle time between the processing of the jobs (note that the maximization problem considered in this paper would be trivial when allowing arbitrarily inserted idle times, since the maximal objective function value can become arbitrarily large in this case). In particular, if release dates are given, it is assumed that, without loss of generality, the smallest release date is equal to 0 and that there exists a feasible schedule. Thus, a feasible solution is described by a permutation $\pi = (j_1, j_2, \ldots, j_n)$ of the jobs of the set N from which the corresponding schedule can be uniquely determined by starting each job as early as possible. Let $S_{j_k}(\pi) = \sum_{l=1}^{k} p_{j_l}$ be the starting time of job j_k in the schedule resulting from the sequence π if release dates are not defined. Moreover, let $C_{j_k}(\pi) = S_{j_k}(\pi) + p_{j_k}$ be the completion time of job j_k in this schedule. If release dates are given, then $S_{j_k}(\pi) = \max\{r_{j_k}, C_{j_{k-1}}(\pi)\}$. If $C_j(\pi) > d_j$, then job j is tardy and we have $U_j(\pi) = 1$, otherwise $U_j(\pi) = 0$. If $C_j(\pi) \leq d_j$, then job j is on-time. Moreover, let $T_j(\pi) = \max\{0, C_j(\pi) - d_j\}$ be the tardiness and $E_j(\pi) = \max\{0, d_j - C_j(\pi)\}$ be the earliness of job j according to the sequence π. We denote by $C_{\max} = C_{j_n}(\pi)$ the makespan associated with the sequence π and by $L_j(\pi) = C_j(\pi) - d_j$ the lateness of job j according to π.

For the single machine problem of maximizing the weighted total tardiness, the objective is to find an optimal sequence π^* that maximizes the total weighted tardiness, i.e., $F(\pi) = \sum_{j=1}^{n} w_j T_j(\pi)$. We denote this problem by $1(\textit{no-idle})||\max \sum w_j T_j$ ('*no-idle*' means that there is no machine idle time in a feasible schedule (Aloulou et al. 2007)) according to the traditional three-field notation $\alpha|\beta|\gamma$ for scheduling problems proposed by Graham et al. (1979) and adapted by Aloulou et al. (2007), where α describes the machine environment, β gives the job characteristics and further constraints and γ describes the objective function. For the single machine total tardiness maximization problem subject to given release dates the objective is to maximize $F(\pi) = \sum_{j=1}^{n} T_j(\pi)$, and the notation is $1(\textit{no-idle})|r_j|\max \sum T_j$. The problems with deadlines or release dates to maximize total weighted completion time or the number of tardy jobs considered in this paper are denoted by $1(\textit{no-idle})|r_j|\max \sum w_j C_j$, $1(\textit{no-idle})|D_j|\max \sum w_j C_j$ and $1(\textit{no-idle})|D_j|\max \sum U_j$.

On the one side, the investigation of problems with an *opposite* optimization criterion itself is an important theoretical task (Aloulou et al. 2007). Algorithms for such maximization problems can be used to compute parts of optimal schedules for the original problems, or to cut bad sub-problems in the branching tree of branch-and-bound algorithms (Gafarov et al. 2010a, 2012). In Aloulou and Artigues (2010), maximization problems were used for solving bi-criteria problems using branch and bound algorithms. On the other side, such problems separately have practical interpretations (for the problem $1(\textit{no-idle})||\max \sum T_j$, they have been discussed e.g. in Aloulou et al. 2007, Gafarov et al. 2012) and applications.

Next, we mention some related results from the literature. Both the problems $1(\textit{no-idle})||\max \sum E_j$ and $1(\textit{no-idle})||\max \sum T_j$ were considered by Lawler and Moore (1969), where a pseudo-polynomial algorithm with time complexity $O(nd_{\max})$ was presented for the problem $1(\textit{no-idle})||\max \sum w_j T_j$, where $d_{\max}$ is the maximal due date. In

Aloulou et al. (2007), Gafarov et al. (2010a, 2010b), the complexity of single machine scheduling problems with classical objective functions and opposite optimization criteria has been investigated.

Other models with semi-active and non-delay schedules have been considered in Aloulou et al. (2004, 2007). In a semi-active schedule, a job cannot be started earlier without changing the job sequence or violating the feasibility. Such problems are denoted as $1(sa)||\gamma$. A schedule is called non-delay if the machine does not stand idle at any time when there is a job available for processing at this time (Baker 1974). Such problems are denoted as $1(nd)||\gamma$. If no release dates are given, then both the problems $1(\textit{no-idle})||\max f$ and $1(nd)||\max f$ are equivalent. To show the difference between the two types of schedules, let us consider an instance with two jobs 1 and 2 and $r_1 = 0,\ r_2 = 5,\ p_1 = 4,\ p_2 = 2$. For the job sequence $\pi = (1, 2)$, we have $C_1 = 4 < 5 = S_2$ in a non-delay schedule. So, for this instance, there does not exist a no-idle schedule. However, such an instance can be reduced in $O(n \log n)$ time to an instance for which there is a feasible no-idle schedule in the following way. Renumber the jobs according to the order $r_1 \le r_2 \le \cdots \le r_n$. If for a job sequence $\pi = (1, 2, \ldots, n)$, we have $C_k < r_{k+1}$ in the corresponding non-delay schedule, then we can modify the release dates as follows: $r_i' = r_i - (r_{k+1} - C_k),\ i = k+1, \ldots, n$. So, an *nd* problem can be easily reduced to a *no-idle* one in polynomial time.

It is known (Aloulou et al. 2007) that solving a problem in polynomial time when restricting the search to non-delay schedules allows one to solve the same problem in polynomial time when considering semi-active schedules (by solving at most $O(n)$ non-delay versions of the problem). The same remark holds for the NP-hardness of a problem (Aloulou et al. 2007). From Aloulou et al. (2004) it is known that for the problem $1(sa)|r_j|\max f$, there exists an optimal schedule in which the jobs are processed from some release date r_i without idle times, i.e., for each possible r_i (which is the starting time of the first job in the schedule), we have to solve a problem $1(\textit{no-idle})||\max f$. Thus, if one can solve the *no-idle* version of a problem, this allows one to solve the corresponding problems extending the search to semi-active or non-delay schedules.

The rest of this paper is organized as follows. In Sect. 2, we present several complexity results and solution algorithms for single machine problems with total tardiness objective function. In particular, we prove NP-hardness of the problems $1(\textit{no-idle})|r_j|\max \sum T_j$ and $1(\textit{no-idle})||\max \sum w_j T_j$. As a consequence, the non-delay and semi-active versions of these problems are NP-hard as well (Aloulou et al. 2007). In Sect. 3, some single machine scheduling problems with deadlines are considered.

2 Single machine problems with total tardiness objective function

In this section, we present some complexity results and solution algorithms for single machine maximization problems with total tardiness objective function. First, we propose NP-hardness proofs for the problems $1(\textit{no-idle})|r_j|\max \sum T_j$ and $1(\textit{no-idle})||\max \sum w_j T_j$. Then an exact algorithm for the problem $1(\textit{no-idle})||\max \sum w_j T_j$ is presented. This algorithm is a generalization of the algorithm for the problem $1(\textit{no-idle})||\max \sum T_j$ given in Gafarov et al. (2012).

2.1 NP-hardness proof for the problem $1(\textit{no-idle})|r_j|\max \sum T_j$

In this section, we give a polynomial reduction from the partition problem to a special instance of the problem $1(\textit{no-idle})|r_j|\max \sum T_j$.

Partition problem Given is a set $\overline{N} = \{b_1, b_2, \ldots, b_{\bar{n}}\}$ of numbers $b_1 \geq b_2 \geq \cdots \geq b_{\bar{n}} > 0$ with $b_i \in Z_+$, $i = 1, 2, \ldots, \bar{n}$. Is there a subset $N' \subset \bar{N}$ such that $\sum_{j \in N'} b_j = A = \frac{1}{2}\sum_{i=1}^{\bar{n}} b_i$?

In this special instance we have $n = 2\bar{n} + 1$ jobs. We renumber the jobs of the set $N = \{0, 1, 2, \ldots, 2\bar{n}\}$, as

$$V_0, V_1, V_2, V_3, V_4, \ldots, V_{2i-1}, V_{2i}, \ldots, V_{2\bar{n}-1}, V_{2\bar{n}}.$$

Without loss of generality, assume that $A \geq 10$ and $\bar{n} \geq 4$. Denote

$$b = 2A\bar{n}^4, \qquad M = \bar{n}^3 b, \qquad \varepsilon_1 = \frac{2}{3\bar{n}} \quad \text{and} \quad \varepsilon_2 = \frac{1}{2\max_{i \in \bar{N}} b_i}.$$

Given an instance of the partition problem, we construct the following instance of the problem $1(\textit{no-idle})|r_j|\max\sum T_j$:

$$\begin{cases} p_0 = 4M, & (1.1)\\ p_{2i} = 2M - ib, \quad i = 1, 2, \ldots, \bar{n}, & (1.2)\\ p_{2i-1} = 2M - ib + b_i, \quad i = 1, 2, \ldots, \bar{n}, & (1.3)\\ d_0 = \sum_{j=0}^{2\bar{n}} p_j, & (1.4)\\ d_{2\bar{n}} = (2\bar{n} + 1)M, & (1.5)\\ d_{2i} = d_{2i+2} - (i - 1 + \varepsilon_1)b, \quad i = \bar{n} - 1, \bar{n} - 2, \ldots, 1, & (1.6)\\ d_{2i-1} = d_{2i} - (i - 1)b_i - \varepsilon_2 b_i, \quad i = \bar{n}, \bar{n} - 1, \ldots, 1, & (1.7)\\ r_0 = 2\bar{n}M - \sum_{i=1}^{\bar{n}} (\bar{n} - i)b + A, & (1.8)\\ r_i = 0, \quad i = 1, 2 \ldots, 2\bar{n}. & (1.9) \end{cases}$$

For this instance, we have $p_0 > p_1 > p_2 > \cdots > p_{2\bar{n}}$ and $d_1 < d_2 < \cdots < d_{2\bar{n}}$. The decision version of the problem is as follows: Does there exist a feasible schedule, in which total tardiness is not smaller than $\bar{n}\sum_{i=0}^{2\bar{n}} p_i - \sum_{i=1}^{\bar{n}}(i-1)p_{2i} - \sum_{i=1}^{\bar{n}} d_{2i} + \varepsilon_2 \cdot \frac{1}{2}\sum_{i=1}^{\bar{n}} b_i$. Next, we can present the following properties for the instance (1.1)–(1.9).

Lemma 1 *For the instance* (1.1)–(1.9), *there are exactly* $\bar{n}$ *tardy jobs in any feasible schedules.*

Proof In a feasible schedule, there is no machine idle time. Since $d_0 = \sum_{j=0}^{2\bar{n}} p_j$, the job 0 is not tardy in any feasible schedule. Since $d_1 > \sum_{j \in \widehat{N}} p_j$ for all $\widehat{N} \subset N \setminus \{V_0\}$ with $|\widehat{N}| = \bar{n}$, there are no more than $\bar{n}$ tardy jobs in any feasible schedule. Since $d_1 < d_2 < \cdots < d_{2\bar{n}} = (2\bar{n} + 1)M < (\bar{n} + 1)(2M - \bar{n}b) = (\bar{n} + 1)p_{2\bar{n}}$, where $p_{2\bar{n}}$ is the minimal processing time, we have at least $\bar{n}$ tardy jobs in any no-idle schedule. Thus, the lemma is true. □

Let $(V_{\bar{n},1}, V_{\bar{n}-1,1}, \ldots, V_{i,1}, \ldots, V_{1,1}, V_0, V_{1,2}, \ldots, V_{i,2}, \ldots, V_{\bar{n}-1,2}, V_{\bar{n},2})$ be a *canonical sequence*, where $\{V_{i,1}, V_{i,2}\} = \{V_{2i-1}, V_{2i}\}$, $i = 1, 2, \ldots, \bar{n}$. We note that some canonical sequences are not feasible.

Let us consider two feasible job sequences $\pi' = (\pi_1, j, V_0, \pi_2)$ and $\pi'' = (\pi_1, V_0, j, \pi_2)$. Since the job V_0 is not tardy in all feasible schedules and the maximization of total tardiness is considered, inequality $F(\pi') \leq F(\pi'')$ holds. Thus, to find an optimal schedule, we only need to consider schedules, where job V_0 is scheduled as early as possible, i.e., in an

optimal sequence this job is processed either at position $\bar{n}+1$ (if the completion time of the job processed at position $\bar{n}$ is larger than or equal to r_0) or $\bar{n}+2$ (otherwise). Below (see Lemma 3) we show that it is processed at position $\bar{n}+1$ in any optimal sequence.

Lemma 2 *For the instance* (1.1)–(1.9), *the inequality*

$$\left|F\left(\pi'\right)-F\left(\pi''\right)\right|<A$$

holds for any two canonical sequences π' *and* π''.

Proof Let us consider a canonical sequence

$$\pi=(V_{\bar{n},1},V_{\bar{n}-1,1},\ldots,V_{i,1},\ldots,V_{1,1},V_0,V_{1,2},\ldots,V_{i,2},\ldots,V_{\bar{n}-1,2},V_{\bar{n},2}).$$

It is known that the jobs $V_{1,2},\ldots,V_{i,2},\ldots,V_{\bar{n}-1,2},V_{\bar{n},2}$ are tardy while the remaining jobs are on-time. Then $\sum_{j\in N}T_j(\pi)=\sum_{i=1}^{\bar{n}}T_{V_{i,2}}(\pi)$.

Let $C=\sum_{i=0}^{2\bar{n}}p_i$. Then we have

$$\sum_{i=1}^{\bar{n}}C_{V_{i,2}}(\pi)=nC-\sum_{i=1}^{\bar{n}}(i-1)p_{V_{i,2}}.$$

Denote by $\pi^{\min}$ a canonical sequence, where $V_{i,2}=V_{2i}$ for all $i=1,2,\ldots,\bar{n}$. Then we have the minimal value of the total tardiness among all canonical sequences:

$$\sum_{j\in N}T_j\left(\pi^{\min}\right)=\bar{n}C-\sum_{i=1}^{\bar{n}}(i-1)p_{2i}-\sum_{i=1}^{\bar{n}}d_{2i}=:T^1.$$

Denote by $\pi^{\max}$ a canonical sequence, where $V_{i,2}=V_{2i-1}$ for all $i=1,2,\ldots,\bar{n}$. Then we have the maximal value of the total tardiness among all canonical sequences:

$$\sum_{j\in N}T_j\left(\pi^{\max}\right)=T^1-\sum_{i=1}^{\bar{n}}(i-1)b_i+\left(\sum_{i=1}^{\bar{n}}(i-1)b_i+\varepsilon_2\sum_{i=1}^{\bar{n}}b_i\right)=:T^2.$$

We have $T^2-T^1<A$, i.e., the lemma is true. □

Lemma 3 *For the instance* (1.1)–(1.9), *all optimal sequences are canonical sequences or they can be reduced to canonical sequences if the SPT* (*shortest processing time*) *rule is applied to the first* $\bar{n}$ *jobs*.

Proof The idea of the proof is as follows. In (1) and (2) we show how a non-canonical sequence can be transformed into a canonical one for which the total tardiness is larger. In (1) we prove that the tardy jobs have to be processed in non-decreasing order of a pair to which they belong. In (2) we show that one and only one job from each pair $\{V_{j,1},V_{j,2}\}$, $j=1,\ldots,n$, is tardy in any optimal sequence. Thus, (1) and (2) show that in any optimal sequence the tardy jobs are processed in the same way as the $\bar{n}$ last tardy jobs in a canonical sequence, and the $\bar{n}$ first on-time jobs can be processed in any order.

(1) Let us consider a job $V^j\in\{V_{j,1},V_{j,2}\}$ and a job $V^i\in\{V_{i,1},V_{i,2}\}$, where $j>i>0$. Let in an optimal sequence $\pi=(\pi_1,V^j,V^i,\pi_2)$ both jobs be tardy. We consider the sequence $\pi'=(\pi_1,V^i,V^j,\pi_2)$. It is easy to check that in the sequence π', both jobs are tardy as well (analogously to the proof of Lemma 1). We get

$$F\left(\pi'\right)-F(\pi)\geq p_{2i}-p_{2j}-b_j>b-b_j>0,$$

i.e., the sequence π is not optimal. This means that in any optimal schedule, a tardy job which belongs to a pair with smaller number has to be processed earlier than the other tardy jobs.

(2) Let us consider an optimal sequence $\pi = (\pi_1, V_{2i}, \pi_2, V_0, \pi_3, V_{2j}, \pi_4)$, $|\pi_3| = i-1$, where at position $\bar{n}+1+i$ job V_{2j} is processed, i.e., the sequence π is not canonical because in a canonical sequence, a job $V^i \in \{V_{i,1}, V_{i,2}\} = \{V_{2i-1}, V_{2i}\}$ should be processed at this position. Next, for the cases (2.1) and (2.2) we show that the sequence π can be transformed into a canonical one with a larger total tardiness.

(2.1) Let $i > j$. For the sequence $\pi' = (\pi_1, V_{2j}, \pi_2, V_0, \pi_3, V_{2i}, \pi_4)$, we have

$$\begin{aligned}
&F(\pi') - F(\pi)\\
&\quad = \big(C_{2j}(\pi) - d_{2i}\big) - \big(C_{2j}(\pi) - d_{2j}\big) + (i-1)(p_{2j} - p_{2i})\\
&\quad = d_{2j} - d_{2i} + (i-1)(p_{2j} - p_{2i})\\
&\quad = -\big((j-1+\varepsilon_1) + \big((j-1)-1+\varepsilon_1\big) + \cdots + \big((i-2)-1+\varepsilon_1\big)\\
&\qquad + \big((i-1)-1+\varepsilon_1\big)\big)b + (i-1)(2M - jb - 2M + ib)\\
&\quad = -\big(\big(i-1+\varepsilon_1-(i-j)\big) + \big(i-1+\varepsilon_1-(i-j-1)\big) + \cdots\\
&\qquad \underbrace{+ (i-1+\varepsilon_1-2) + (i-1+\varepsilon_1-1)}_{i-j}\big)b\\
&\qquad + (i-1)(i-j)b\\
&\quad = -(i-1)(i-j)b + \sum_{k=1}^{i-j} kb - (i-j)\varepsilon_1 b + (i-1)(i-j)b\\
&\quad > b - \bar{n}\varepsilon_1 b = \frac{1}{3}b > 0,
\end{aligned}$$

i.e., the sequence π is not optimal.

Analogously, for any optimal sequence $\pi = (\pi_1, V^i, \pi_2, V_0, \pi_3, V^j, \pi_4)$, where $V^i \in \{V_{i,1}, V_{i,2}\}$ and $V^j \in \{V_{j,1}, V_{j,2}\}$, we can construct a feasible sequence $\pi' = (\pi_1, V^j, \pi_2, V_0, \pi_3, V^i, \pi_4)$, for which $F(\pi') - F(\pi) > \frac{1}{3}b - b_{\max} > 0$. To prove this, one can use (1.3), (1.7), $b = 2A\bar{n}^4 \gg 2\bar{n}b_{\max}$ and $\varepsilon_2 b_i < \frac{1}{2}$.

(2.2) Let $j > i$. For the sequence $\pi' = (\pi_1, V_{2j}, \pi_2, V_0, \pi_3, V_{2i}, \pi_4)$, we have

$$\begin{aligned}
F(\pi') - F(\pi) &= d_{2j} - d_{2i} - (i-1)(p_{2i} - p_{2j})\\
&= \big((i-1+\varepsilon_1) + \big((i+1)-1+\varepsilon_1\big) + \cdots + \big((j-2)-1+\varepsilon_1\big)\\
&\quad + \big((j-1)-1+\varepsilon_1\big)\big)b - (i-1)(2M - ib - 2M + jb)\\
&= \big((i-1+\varepsilon_1+0) + (i-1+\varepsilon_1+1) + \cdots + \big(i-1+\varepsilon_1+(j-i-2)\big)\\
&\quad \underbrace{+ \big(i-1+\varepsilon_1-(j-i-1)\big)}_{j-i}\big)b\\
&\quad - (i-1)(j-i)b\\
&= (i-1)(j-i)b + \sum_{k=0}^{j-i-1} kb + (j-i)\varepsilon_1 b - (i-1)(j-i)b\\
&= \sum_{k=0}^{j-1-i} kb + (j-i)\varepsilon_1 b > 0,
\end{aligned}$$

i.e., the sequence π is not optimal.

Analogously, for any optimal sequence $\pi = (\pi_1, V^i, \pi_2, V_0, \pi_3, V^j, \pi_4)$, we can construct a feasible sequence $\pi' = (\pi_1, V^j, \pi_2, V_0, \pi_3, V^i, \pi_4)$, for which $F(\pi') - F(\pi) > \varepsilon_1 b - b_{\max} > 0$. To prove this, one can use (1.3), (1.7), $b = 2A\bar{n}^4 \gg 2\bar{n}b_{\max}$ and $\varepsilon_2 b_i < \frac{1}{2}$.

We note that for both cases (2.1) and (2.2), inequality $F(\pi') - F(\pi) > nA$ holds. By the transformations described in (2.1) and (2.2), we change an initial feasible sequence into a canonical one. By each transformation the total tardiness value is increased by a number larger than nA. By the transformation described in (2.1), we always obtain a feasible sequence π'. By the transformation described in (2.2), in the final canonical sequence π', we can have an idle time between $S_{V_0}(\pi')$ and $C_{V_{1,1}}(\pi')$, i.e., the resulting schedule is not feasible. This infeasible canonical sequence can be easily transformed into a feasible canonical one which has no idle time, i.e., we can schedule the jobs $V_{2\bar{n}-1}, V_{2\bar{n}-3}, \ldots, V_{2i-1}, \ldots, V_1$ at the beginning of the canonical sequence. Denote such a sequence as π''. Then $F(\pi') - F(\pi'') < A$ (see Lemma 2), i.e., the sequence π'' is better than the initial sequence π, since $F(\pi') - F(\pi) > \varepsilon_1 b - b_{\max} > A > F(\pi') - F(\pi'')$.

If we consider an optimal sequence $\overline{\pi} = (\pi_1, V_\alpha, V_\beta, V_0, \pi_3)$, where the job V_0 is processed at position $\bar{n} + 2$, i.e., $|\pi_1| = \bar{n} - 1$ and $C_{V_\alpha} < r_0$, then the sequence $\pi = (\pi_1, V_\alpha, V_0, V_\beta, \pi_3)$ is not feasible. However, $F(\pi) - F(\overline{\pi}) > 0$, and the sequence π can be transformed into a canonical one by the operations described in (2.1) and (2.2), where by each operation the total tardiness value is increased.

To sum up, we can conclude that all optimal sequences are canonical sequences or they can be reduced to canonical sequences if the SPT (shortest processing time) rule is applied to the first $\bar{n}$ jobs, since they are not tardy and can be processed in any order. □

Theorem 1 *The problem* $1(\textit{no-idle})|r_j|\max\sum T_j$ *is NP-hard.*

Proof In this proof, we analyze the maximal possible value of total tardiness among all canonical schedules. This maximal value corresponds to a canonical schedule which is not feasible. Then, we look for the maximal value reached in a feasible canonical schedule if the instance of the partition problem has the answer "YES" and vice versa.

Let us consider a canonical sequence

$$\pi = (V_{\bar{n},1}, V_{\bar{n}-1,1}, \ldots, V_{i,1}, \ldots, V_{1,1}, V_0, V_{1,2}, \ldots, V_{i,2}, \ldots, V_{\bar{n}-1,2}, V_{\bar{n},2}).$$

It is known that the jobs $V_{1,2}, \ldots, V_{i,2}, \ldots, V_{\bar{n}-1,2}, V_{\bar{n},2}$ are tardy while the remaining jobs are on-time. Then $\sum_{j\in N} T_j(\pi) = \sum_{i=1}^{\bar{n}} T_{V_{i,2}}(\pi)$.

Let $C = \sum_{i=0}^{2\bar{n}} p_i$. Then we have

$$\sum_{i=1}^{\bar{n}} C_{V_{i,2}}(\pi) = nC - \sum_{i=1}^{\bar{n}} (i-1)p_{V_{i,2}}.$$

In addition, let us denote

$$\phi(i) = \begin{cases} 1, & \text{if } V_{i,2} = V_{2i-1}, \\ 0, & \text{if } V_{i,2} = V_{2i}. \end{cases}$$

If $V_{i,2} = V_{2i}$ for all $i = 1, 2, \ldots, \bar{n}$, then

$$\sum_{j\in N} T_j(\pi) = \bar{n}C - \sum_{i=1}^{\bar{n}} (i-1)p_{2i} - \sum_{i=1}^{\bar{n}} d_{2i} =: T^1$$

else we have

$$\sum_{j\in N} T_j(\pi) = T^1 - \sum_{i=1}^{\bar{n}} \phi(i)(i-1)b_i + \left(\sum_{i=1}^{\bar{n}} \phi(i)(i-1)b_i + \varepsilon_2 \sum_{i=1}^{\bar{n}} \phi(i)b_i \right).$$

Thus, the maximal objective function value obtained among the canonical sequences is equal to $T^1 + \varepsilon_2 \sum_{i=1}^{\bar{n}} b_i$. However, the sequence

$$(V_{2\bar{n}}, V_{2(\bar{n}-1)}, \ldots, V_{2i}, \ldots, V_2, V_0, V_1, \ldots, V_{2i-1}, \ldots, V_{2(\bar{n}-1)-1}, V_{2\bar{n}-1})$$

is not feasible since there is an idle time before the job V_0. We must construct a *feasible* canonical sequence, where the value $\varepsilon_2 \sum_{i=1}^{\bar{n}} \phi(i)b_i$ is maximized.

If and only if the instance of the partition problem has the answer "YES", we have a feasible canonical sequence with

$$\varepsilon_2 \sum_{i=1}^{\bar{n}} \phi(i)b_i = \varepsilon_2 \cdot \frac{1}{2} \sum_{i=1}^{\bar{n}} b_i$$

and thus, $S_{V_0}(\pi) = r_0$. If we have $b_i \in N'$ in the original instance of the partition problem, then $\phi(i) = 1$. If the answer is "NO", then in an optimal schedule the total tardiness value is less than $T^1 + \varepsilon_2 \cdot \frac{1}{2} \sum_{i=1}^{\bar{n}} b_i$.

This means that the NP-complete partition problem can be reduced to the problem $1(\textit{no-idle})|r_j|\max \sum T_j$ in polynomial time, i.e., the problem under consideration is NP-hard. □

2.2 NP-hardness for the problem $1(\textit{no-idle})||\max \sum w_j T_j$

In this section, we give a polynomial reduction from the partition problem to a special instance of the problem $1(\textit{no-idle})||\max \sum w_j T_j$. For the latter problem, we have to maximize $F(\pi) = \sum_{j=1}^{\bar{n}} w_j T_j(\pi)$.

First, we present a property of an optimal schedule for the problem $1(\textit{no-idle})||\max \sum w_j T_j$.

Lemma 4 *There exists an optimal job sequence π for the problem $1(\textit{no-idle})||\max \sum w_j T_j$ that can be represented as a concatenation (G, H), where all jobs $j \in H$ are tardy and all jobs $i \in G$ are on-time. All jobs from the set G are processed in non-increasing order of the values $\frac{w_j}{p_j}$ and all jobs from the set H are processed in non-decreasing order of the values $\frac{w_j}{p_j}$.*

Proof First, in (1) we prove that in any optimal schedule, the tardy jobs are processed one by one at the end of the schedule. Then, in (2) and (3) we substantiate the order of the tardy and on-time jobs in an optimal schedule.

(1) Assume that there exists an optimal sequence $\pi = (\pi_1, j, \pi_2, i, \pi_3)$, where job j is tardy and job i is on-time. For the sequence $\pi' = (\pi_1, i, j, \pi_2, \pi_3)$, we have

$$F(\pi') - F(\pi) \geq w_j\big(T_j(\pi') - T_j(\pi)\big) + w_i\big(T_i(\pi') - T_i(\pi)\big) = w_j p_i + 0 > 0.$$

Therefore, we have a contradiction since the sequence π' has a larger value of total weighted tardiness, i.e., π' is better and π is not optimal.

(2) We consider an optimal sequence $\pi = (G, H)$, where all jobs $j \in H$ are tardy and all jobs $i \in G$ are on-time. Now we prove that all jobs from H are processed in non-decreasing

order of the values $\frac{w_j}{p_j}$. Assume that there exists an optimal sequence $\pi = (\pi_1, j_1, j_2, \pi_2)$, where the jobs j_1 and j_2 are tardy and $\frac{w_{j_1}}{p_{j_1}} > \frac{w_{j_2}}{p_{j_2}}$. For the sequence $\pi' = (\pi_1, j_2, j_1, \pi_2)$, we have

$$\begin{aligned} F(\pi') - F(\pi) &= w_{j_1}\big(T_{j_1}(\pi') - T_{j_1}(\pi)\big) + w_{j_2}\big(T_{j_2}(\pi') - T_{j_2}(\pi)\big) \\ &= w_{j_1} p_{j_2} - w_{j_2} \min\big\{p_{j_1}, T_{j_2}(\pi)\big\} > 0. \end{aligned}$$

Therefore, we have a contradiction and $\pi = (\pi_1, j_1, j_2, \pi_2)$ is not optimal.

(3) We consider an optimal sequence $\pi = (G, H)$, where all jobs $j \in H$ are tardy and all jobs $i \in G$ are on-time. Now, we prove that all jobs $i \in G$ can be processed in non-increasing order of the values $\frac{w_j}{p_j}$ in an optimal sequence. For all jobs $i \in G$, we have $d_i \geq \sum_{k\in G} p_k$. Otherwise, if $d_i < \sum_{k\in G} p_k$, the sequence $\pi' = (G \setminus \{i\}, i, H)$ is better, and we have a contradiction. Therefore, the jobs from G can be processed in any order. □

The reduction from the partition problem to a special instance of the problem $1(\textit{no-idle})||\max \sum w_j T_j$ can be done as follows. Without loss of generality, we assume that $\bar{n} > 3$ and $\sum_{i=1}^{\bar{n}} b_i > 10$. Given an instance of the partition problem, we construct the following instance of the problem $1(\textit{no-idle})||\max \sum w_j T_j$, where $n = 2\bar{n}$:

$$\begin{cases} w_{2i} = M^i, \quad i = 1, 2, \ldots, \bar{n}, & (2.1) \\ w_{2i-1} = w_{2i} + b_i, \quad i = 1, 2, \ldots, \bar{n}, & (2.2) \\ p_{2i} = \sum_{j=1}^{i-1} w_{2j} + \frac{1}{2}\sum_{j\in\bar{N}\setminus\{i\}} b_j, \quad i = 1, 2, \ldots, \bar{n}, & (2.3) \\ p_{2i-1} = p_{2i} + b_i, \quad i = 1, 2, \ldots, \bar{n}, & (2.4) \\ d_{2i} = d_{2i-1} = P - \sum_{j=i}^{\bar{n}} p_{2j}, \quad i = 1, 2, \ldots, \bar{n}, & (2.5) \end{cases}$$

with $M = (\bar{n}\sum b_i)^{10} = (2A\bar{n})^{10}$ and $P = \sum_{j=1}^{2\bar{n}} p_j$. The decision version is as follows: Does there exist a feasible schedule, in which total tardiness is not smaller than $\sum_{j=1}^{\bar{n}} w_{2j} p_{2j} + \frac{1}{2}A^2$.

The main idea of using M^i is that the weight of a job belongs to a pair with the largest number being greater than the total weight of all jobs from the pairs with smaller numbers, e.g., for the job $V_{2\bar{n}}$, we have: $w_{2\bar{n}} \gg \sum_{i=1}^{\bar{n}-1}(w_{2i-1} + w_{2i})$. If we suppose that all digits used are coded in a binary system with approximately 2^x zero-one symbols per digit, then, to code $M^{\bar{n}}$, we need $2^{x\cdot O(\bar{n})}$ symbols. So, the input length for the resulting input will be $O(\bar{n})$ times larger than the input length of the initial instance of the partition problem.

We renumber the jobs of the set $N = \{1, 2, \ldots, 2\bar{n}\}$ as

$$V_1, V_2, V_3, V_4, \ldots, V_{2i-1}, V_{2i}, \ldots, V_{2\bar{n}-1}, V_{2\bar{n}}.$$

Let $(V_{\bar{n},1}, V_{\bar{n}-1,1}, \ldots, V_{i,1}, \ldots, V_{1,1}, V_{1,2}, \ldots, V_{i,2}, \ldots, V_{\bar{n}-1,2}, V_{\bar{n},2})$ be a *canonical sequence*, where $\{V_{i,1}, V_{i,2}\} = \{V_{2i-1}, V_{2i}\}$, $i = 1, 2, \ldots, \bar{n}$.

Lemma 5 *For the instance* (2.1)–(2.5), *all optimal sequences are canonical sequences, or they can be reduced to a canonical sequence if the first* $\bar{n}$ *jobs are reordered in non-increasing order of the values* $\frac{w_j}{p_j}$.

Proof In this proof, we only consider optimal sequences of the type described in Lemma 4. First, in (1) we prove that one of the jobs V_{2n} and V_{2n-1} is tardy in any optimal sequence.

In (2) we show that only one of them is tardy, and it is processed at the end of any optimal sequence. In (3), we explain that the same property holds also for the other pairs of jobs.

(1) Suppose that both jobs are on-time in an optimal sequence $\pi = (\pi_1, V_{2n}, \pi_2, V_{2n-1}, \pi_3, \pi_4)$, where only the jobs from the set π_4 are tardy. We note that we consider only optimal sequences of the type $\pi = (G, H)$, where all jobs $j \in H$ are tardy and all jobs $i \in G$ are on-time (see Lemma 4). For the sequence $\pi' = (\pi_1, \pi_2, V_{2n-1}, \pi_3, \pi_4, V_{2n})$, we have

$$\begin{aligned}\sum_{j=1}^{n} w_j T_j(\pi') - \sum_{j=1}^{n} w_j T_j(\pi) &\geq w_{2n} T_{2n}(\pi') - p_{2n} \sum_{j\in\pi_4} w_j \\ &\geq p_{2n} M^n - p_{2n} \sum_{i=1}^{n-1} \left(2M^i + b_i\right) \\ &= p_{2n} M^n - p_{2n} \left(2M \frac{M^{n-1}-1}{M-1} + \sum_{i=1}^{n-1} b_i\right) > 0.\end{aligned}$$

Thus, the sequence π is not optimal, i.e., one of the jobs V_{2n} and V_{2n-1} is tardy in any optimal schedule.

(2) Now we prove that the following inequalities hold:

$$\frac{w_2}{p_2} < \frac{w_4}{p_4} < \cdots < \frac{w_{2n}}{p_{2n}}.$$

We have to prove:

$$\frac{M^{i-1}}{\sum_{j=1}^{i-2} w_{2j} + \frac{1}{2}\sum_{j\in N\setminus\{i-1\}} b_j} < \frac{M^i}{\sum_{j=1}^{i-1} w_{2j} + \frac{1}{2}\sum_{j\in N\setminus\{i\}} b_j}.$$

Let $B_i = \frac{1}{2}\sum_{j\in N\setminus\{i\}} b_j$, $i = 1, 2, \ldots, n$. Then, by equivalent transformations, we obtain

$$\begin{aligned}&\frac{M^{i-1}}{M\frac{M^{i-2}-1}{M-1} + B_{i-1}} < \frac{M^i}{M\frac{M^{i-1}-1}{M-1} + B_i} \\ &\iff \frac{1}{\frac{M(M^{i-2}-1)+B_{i-1}(M-1)}{M-1}} < \frac{M}{\frac{M(M^{i-1}-1)+B_i(M-1)}{M-1}} \\ &\iff M\left(M^{i-1}-1\right) + B_i(M-1) < M\left[M\left(M^{i-2}-1\right) + B_{i-1}(M-1)\right] \\ &\iff 0 < M^2(B_{i-1}-1) - M(B_{i-1}+B_i-1) + B_i.\end{aligned}$$

The latter inequality is true since $M^2 > M \cdot 2\sum_{j=1}^{\bar{n}} b_j$ and $(B_{i-1}-1) > 1$. Thus, the above inequalities hold. Analogously, we can prove that

$$\frac{w_{2(i-1)-1}}{p_{2(i-1)-1}} < \frac{w_{2i}}{p_{2i}}, \qquad \frac{w_{2(i-1)}}{p_{2(i-1)}} < \frac{w_{2i-1}}{p_{2i-1}} \quad \text{and} \quad \frac{w_{2(i-1)-1}}{p_{2(i-1)-1}} < \frac{w_{2i-1}}{p_{2i-1}}$$

for each $i = 2, 3, \ldots, \bar{n}$.

Thus, we conclude that in any optimal sequence, one of the jobs $V_{2\bar{n}}$ or $V_{2\bar{n}-1}$ is the last tardy job. We remind that we only consider optimal sequences of the type $\pi = (G, H)$, where all jobs $j \in H$ are tardy, all jobs $i \in G$ are on-time and all jobs from the set H are processed in non-decreasing order of the values $\frac{w_j}{p_j}$.

Since $d_{2\bar{n}} = d_{2\bar{n}-1} = P - \sum_{j=\bar{n}}^{\bar{n}} p_{2j} = P - p_{2\bar{n}} > P - p_{2\bar{n}-1}$, only one of the jobs $V_{2\bar{n}}$ or $V_{2\bar{n}-1}$ can be tardy in any feasible schedule. Then, one and only one of the jobs $V_{2\bar{n}}$ and $V_{2\bar{n}-1}$ is the last tardy job in any optimal schedule. The on-time jobs can be processed

in an optimal schedule in any order (see point 3 in the proof of Lemma 4). Therefore, in the following, we consider only optimal sequences of the type $(V_{\bar{n},1}, \pi_\alpha, V_{\bar{n},2})$, where $\{V_{\bar{n},1}, V_{\bar{n},2}\} = \{V_{2\bar{n}-1}, V_{2\bar{n}}\}$.

(3) Analogously to (1) and (2), we can prove that for each $i = \bar{n}-1, \bar{n}-2, \ldots, 1$, one and only of the jobs V_{2i} and V_{2i-1} is tardy and precedes only the jobs $V_{i+1,2}, \ldots, V_{\bar{n},2}$.

Thus, the lemma is true. □

Theorem 2 *The problem* $1(\textit{no-idle})||\max\sum w_j T_j$ *is NP-hard in the ordinary sense.*

Proof For the sequence $\pi = (V_{2\bar{n}-1}, V_{2(\bar{n}-1)-1}, \ldots, V_3, V_1, V_2, V_4, \ldots, V_{2(\bar{n}-1)}, V_{2\bar{n}})$, we obtain the objective function value

$$F(\pi) = \sum_{j=1}^{\bar{n}} w_{2j} T_{2j}(\pi) = \sum_{j=1}^{\bar{n}} w_{2j} p_{2j}.$$

Now we consider the canonical sequence

$$\pi' = (V_{\bar{n},1}, V_{\bar{n}-1,1}, \ldots, V_{i,1}, \ldots, V_{1,1}, V_{1,2}, \ldots, V_{i,2}, \ldots, V_{\bar{n}-1,2}, V_{\bar{n},2}).$$

In addition, let us denote

$$x_i = \begin{cases} 1, & \text{if } V_{i,2} = V_{2i-1}, \\ 0, & \text{if } V_{i,2} = V_{2i}. \end{cases}$$

Then we get

$$\begin{aligned}
F(\pi') &= F(\pi) + \sum_{i=1}^{\bar{n}} x_i \left[(w_{2i-1} - w_{2i}) \cdot T_{V_{2i}}(\pi) - (p_{2i-1} - p_{2i}) \left(\sum_{j=1}^{i-1} w_{V_{j,2}} \right) \right] \\
&= F(\pi) + \sum_{i=1}^{\bar{n}} x_i \left[b_i \cdot p_{2i} - b_i \cdot \left(\sum_{j=1}^{i-1} w_{V_{j,2}} \right) \right] \\
&= F(\pi) + \sum_{i=1}^{\bar{n}} x_i \left[b_i \cdot p_{2i} - b_i \cdot \left(\sum_{j=1}^{i-1} w_{2j} + \sum_{j=1}^{i-1} x_j b_j \right) \right] \\
&= F(\pi) + \sum_{i=1}^{\bar{n}} x_i b_i \left[p_{2i} - \sum_{j=1}^{i-1} w_{2j} - \frac{1}{2} \sum_{j \in \bar{N} \setminus \{i\}} x_j b_j \right] \\
&= F(\pi) + \frac{1}{2} \sum_{i=1}^{\bar{n}} x_i b_i \left(\sum_{j \in \bar{N} \setminus \{i\}} b_j - \sum_{j \in \bar{N} \setminus \{i\}} x_j b_j \right) \\
&= F(\pi) + \frac{1}{2} \sum_{i=1}^{\bar{n}} \sum_{j \in \bar{N} \setminus \{i\}} x_i \cdot b_i \cdot b_j \cdot (1 - x_j).
\end{aligned}$$

The equality

$$\sum_{i=1}^{\bar{n}} x_i b_i \left(\sum_{j=1}^{i-1} x_j b_j \right) = \sum_{i=1}^{\bar{n}} x_i b_i \left(\frac{1}{2} \sum_{j \in \bar{N} \setminus \{i\}} x_j b_j \right)$$

can be explained as follows. In the left part of the equality, for any $i \in N$, we have the following combinations $x_i b_i \sum_{j=1}^{i-1} x_j b_j + x_{i+1} b_{i+1} x_i b_i + x_{i+2} b_{i+2} x_i b_i + \cdots + x_{\bar{n}} b_{\bar{n}} x_i b_i$. This

means that in the left equality for each pair (i, j), $i \in \bar{N}$, $j \in \bar{N} \setminus \{i\}$, there is one and only one summand $x_j b_j x_i b_i$. This summand can be presented as a sum of two summands $\frac{1}{2}x_j b_j x_i b_i + \frac{1}{2}x_j b_j x_i b_i$ to be taken into account twice in the right part of the equality.

$F(\pi')$ reaches its maximal value if there exists a subset $N' \subset \bar{N}$ such that $\sum_{i \in N'} b_i = A$, i.e.,

$$F(\pi') = F(\pi) + \frac{1}{2}\sum_{i \in N'} \sum_{j \in \bar{N} \setminus N'} b_i \cdot b_j = F(\pi) + \frac{1}{2}A \cdot A.$$

Otherwise, we have

$$\begin{aligned} F(\pi') &= F(\pi) + \frac{1}{2}\sum_{i=1}^{\bar{n}} \sum_{j \in \bar{N} \setminus \{i\}} x_i \cdot b_i \cdot b_j \cdot (1 - x_j) \\ &= F(\pi) + \frac{1}{2}(A - y)(A + y) \\ &= F(\pi) + \frac{1}{2}A^2 - \frac{1}{2}y^2, \end{aligned}$$

where $y > 0$.

This means that, if and only if there exists a subset $N' \subset \bar{N}$ for the instance of the partition problem such that $\sum_{i \in N'} b_i = A$, then, for an optimal canonical sequence π^*, we have

$$F(\pi^*) = F(\pi) + \frac{1}{2}A^2,$$

where $x_i = 1$ for $i \in N'$. □

In the following section, a pseudo-polynomial solution algorithm for the problem $1(\textit{no-idle})||\max \sum w_j T_j$ is presented. Thus, the problems $1(\textit{no-idle})||\max \sum w_j T_j$ and $1(sa)||\max \sum w_j T_j$ are NP-hard in the ordinary sense but not in the strong sense.

2.3 Solution algorithms for the problem $1(\textit{no-idle})||\max \sum w_j T_j$

Assume that the jobs are numbered such that

$$\frac{w_1}{p_1} \le \frac{w_2}{p_2} \le \cdots \le \frac{w_n}{p_n}.$$

As a corollary from Lemma 4, there exists an optimal schedule, in which all jobs $l \in \{1, 2, \ldots, j\}$ are processed from time t one by one and there is no job $i \in \{j + 1, j + 2, \ldots, n\}$, which is processed between these jobs. Thus, we can propose a dynamic programming algorithm based on Lemma 4. In this algorithm, for each set of jobs $\{1, 2, \ldots, j\}$ (i.e., at each stage j) and for each possible starting time t (i.e., each possible state), we construct a partial optimal job sequence $\pi_l(t)$. In addition, $F_l(t)$ denotes the total weighted tardiness value of this sequence. $\Phi^1(t)$ and $\Phi^2(t)$ are temporary functions, which are used to compute $F_l(t)$.

Algorithm 1

1. Number the jobs such that $\frac{w_1}{p_1} \le \frac{w_2}{p_2} \le \cdots \le \frac{w_n}{p_n}$;
2. $\pi_1(t) := (1)$, $F_1(t) := w_1 \max\{0, t + p_1 - d_1\}$ for all $t \in Z$ with $t \in [0, \sum_{i=2}^{n} p_i]$;
3. FOR $l := 2$ TO n DO

FOR $t := 0$ TO $\sum_{i=l+1}^{n} p_i$ $(t \in Z)$ DO
$\pi^1 := (l, \pi_{l-1}(t+p_l))$, $\pi^2 := (\pi_{l-1}(t), l)$;
$\Phi^1(t) := w_l \max\{0, t + p_l - d_l\} + F_{l-1}(t + p_l)$;
$\Phi^2(t) := F_{l-1}(t) + w_l \max\{0, t + \sum_{i=1}^{l} p_i - d_l\}$;
If $\Phi^1(t) > \Phi^2(t)$ then $F_l(t) := \Phi^1(t)$ and $\pi_l(t) := \pi^1$, else $F_l(t) := \Phi^2(t)$ and $\pi_l(t) := \pi^2$;

4. $\pi_n(0)$ is an optimal job sequence with the objective function value $F_n(0)$.

Theorem 3 *Algorithm 1 constructs an optimal job sequence in* $O(n \sum p_j)$ *time.*

The proof is very similar to that given in Gafarov et al. (2012) based on Lemma 4 adapted to unit weights, and it is also presented in Gafarov et al. (2010b) for the problem under consideration.

In all partial schedules starting from a point $t \geq d_{\max}$, all jobs are tardy in each partial schedule which starts from time t. Thus, we can reduce the time complexity to $O(nd_{\max})$. We note that a similar algorithm for this problem with time complexity $O(nd_{\max})$ has been presented in Lawler and Moore (1969).

To reduce the running time of Algorithm 1, we can use a modification of the dynamic programming algorithm presented in Gafarov et al. (2012) for the special case of the problem $1(\textit{no-idle})||\max \sum T_j$, where $w_j = 1$, $j = 1, 2, \ldots, n$. In this modification, the functions $F_l(t)$, $l = 1, 2, \ldots, n$, are defined for any $t \in (-\infty, +\infty)$ (not only for integer t). Then it is easy to prove that the functions $F_l(t)$ are piecewise-linear, convex, continuous and non-decreasing functions (Gafarov et al. 2012) and that the t-axis can be divided into intervals, on which function $F_l(t)$ is a linear function of the form $F_l(t) = F_l^k(t) = u_l^k \cdot (t - t_l^{k-1}) + b_l^k$, where k is the number of the interval $[t_l^{k-1}, t_l^k)$, u_l^k is the slope of the function and $b_l^k = F_l(t_l^{k-1})$. The points t_l^k which separate the intervals are called *break points*.

Let $t' = d_l - p_l$ and $t'' = d_l - \sum_{i=1}^{l} p_i$. In addition, assume that $t_{l-1}^s - p_l < t' < t_{l-1}^{s+1} - p_l$, $s + 1 \leq v_{l-1}$ and $t_{l-1}^h < t'' < t_{l-1}^{h+1}$, $h + 1 \leq v_{l-1}$, where v_{l-1} is the number of intervals.

The function $\Phi^1(t)$ is obtained from function $F_{l-1}(t)$ by shifting the diagram of function $F_{l-1}(t)$ to the left by the value p_l, adding a new break point t' and increasing the values $u_{l-1}^{s+1}, u_{l-1}^{s+2}, \ldots, u_{l-1}^{v_{l-1}+1}$ by w_l, i.e., the weighted number of tardy jobs (and thus the slope of the function) increases. The function $\Phi^2(t)$ is obtained from function $F_{l-1}(t)$ by adding a column which results from the new break point t'' and increasing the values $u_{l-1}^{h+1}, u_{l-1}^{h+2}, \ldots, u_{l-1}^{v_{l-1}+1}$ by w_l, i.e., the weighted number of tardy jobs increases. So, we transform $F_{l-1}(t)$ into $\Phi^1(t)$ and $\Phi^2(t)$ analytically. The function $F_l(t) = \max\{\Phi^1(t), \Phi^2(t)\}$ can be computed analytically as well. Then the processing time of this modification linearly depends on the number of break points. Since the functions $F_l(t)$ are convex and the slopes u_l^k are equal to the weighted number of tardy jobs in a partial schedule, the number of break points is less than or equal to $O(\sum_{j=1}^{l} w_j)$ at stage l. Moreover, the number of break points is less than or equal to $O(d_{\max})$ for instances with integer parameters. Then the time complexity of this *graphical modification* is equal to $O(n \min\{d_{\max}, \sum w_j\})$.

A more detailed description, the complete proofs and a numerical example of such an algorithm for the problem $1(\textit{no-idle})||\max \sum T_j$ with unit weights have been presented in Gafarov et al. (2012). In particular, it has been shown that for such an algorithm, $F_l(t)$ is a continuous, piecewise-linear and convex function $F_l(t)$ which is also true for the problem under consideration. The time complexity of the graphical modification for the special case $1(\textit{no-idle})||\max \sum T_j$ is equal to $O(n^2)$ (Gafarov et al. 2012).

As a consequence, we can prove that the problem $1(sa)|r_j|\max \sum T_j$, the complexity status of which was open (Aloulou et al. 2007), is polynomially solvable. From (Aloulou

et al. 2004), it is known that for the problem $1(sa)|r_j|\max\sum T_j$, there exists an optimal schedule in which the jobs are processed from some release date r_i without idle times, i.e., for each possible r_i (which is the starting time of the first job in the schedule), we have to solve the problem $1(no\text{-}idle)||\max\sum T_j$, i.e., the resulting solution algorithm has a time complexity of $O(n^3)$.

3 Complexity results for single machine problems with deadlines and/or other objective functions

First, we present two new complexity results for single machine maximization problems.

Lemma 6 *The problems* $1(no\text{-}idle)|D_j|\max\sum T_j$ *and* $1(no\text{-}idle)|D_j|\max\sum U_j$ *are NP-hard.*

Proof We give a reduction from the partition problem. Given an instance of the partition problem, we construct an instance of the scheduling problems with $n=\bar{n}+1$ jobs, where $p_j=b_j$ and $d_j=D_j=\sum_{j=1}^{\bar{n}}p_j+1,\ j=1,2,\ldots,\bar{n}$. In addition, let $p_{\bar{n}+1}=1,\ d_{\bar{n}+1}=A$ and $D_{\bar{n}+1}=A+1$. It is obvious that in any feasible schedule, the jobs $1,2,\ldots,\bar{n}$ are on-time.

If the instance of the partition problem has the answer "YES", then there exists an optimal sequence $\pi=(\pi_1,\bar{n}+1,\pi_2)$, where $\{\pi_1\}=N'$, $\sum_{j\in N'}p_j=A$, $\{\pi_2\}=\bar{N}\setminus N'$ and $\sum_{j=1}^{\bar{n}+1}T_j(\pi)=1$. If the answer is "NO", then in all feasible sequences $\sum_{j=1}^{\bar{n}+1}T_j(\pi)=0$. Analogously, $\sum_{j=1}^{\bar{n}+1}U_j(\pi)=1$ and $\sum_{j=1}^{\bar{n}+1}U_j(\pi)=0$, respectively, hold. □

Lemma 7 *The problems* $1(no\text{-}idle)|r_j|\max\sum w_jC_j$ *and* $1(no\text{-}idle)|D_j|\max\sum w_jC_j$ *are NP-hard.*

Proof We give a reduction from the partition problem. Given an instance of the partition problem with $\bar{n}$ numbers, we construct an instance of the scheduling problem with $n=\bar{n}+1$ jobs, where $p_j=w_j=b_j$ and $r_j=0,\ j=1,2,\ldots,\bar{n}$. In addition, let $p_{\bar{n}+1}=1, w_{\bar{n}+1}=0$ and $r_{\bar{n}+1}=A$.

If the instance of the partition problem has the answer "YES", then there exists an optimal sequence $\pi=(\pi_1,\bar{n}+1,\pi_2)$, where $\{\pi_1\}=N'$, $\sum_{j\in\pi_1}p_j=A$, $\{\pi_2\}=\bar{N}\setminus N'$, $\sum_{j\in\pi_2}p_j=\sum_{j\in N}b_j-A$ and

$$\sum_{j=1}^{\bar{n}+1}w_jC_j(\pi)=\sum_{1\le i\le j\le\bar{n}}b_ib_j+\Bigg(\sum_{j\in\bar{N}}b_j-A\Bigg),$$

since in the sequence $\pi'=(\pi_1,\pi_2,\bar{n}+1)$, the equality

$$\sum_{j=1}^{\bar{n}+1}w_jC_j(\pi)=\sum_{1\le i\le j\le\bar{n}}b_ib_j$$

holds (Lenstra et al. 1977). The jobs from the partial sequences π_1 and π_2 can be processed in an arbitrary order (Lenstra et al. 1977). If the answer is "NO", then

$$\sum_{j=1}^{\bar{n}+1}w_jC_j(\pi)<\sum_{1\le i\le j\le\bar{n}}b_ib_j+\Bigg(\sum_{j\in\bar{N}}b_j-A\Bigg).$$

Table 1 Complexity results

Problem	Maximization version	Minimization version
Total tardiness problems		
$1(\textit{no-idle})\|\|\max\sum w_jT_j$	NP-hard in the ordinary sense (Theorem 2), Solution algorithm with complexity $O(n\min\{\sum w_j, d_{\max}\})$	strongly NP-hard
$1(\textit{no-idle})\|r_j\|\max\sum T_j$	NP-hard (Theorem 1)	strongly NP-hard
$1(\textit{no-idle})\|\|\max\sum T_j$	Solution algorithm with complexity $O(n\sum p_j)$ (Gafarov et al. 2010a). Solution algorithm with complexity $O(n^2)$ (Gafarov et al. 2012)	NP-hard in the ordinary sense. Solution algorithm with complexity $O(n^4\sum p_j)$ (Lawler 1977)
$1(\textit{no-idle})\|D_j\|\max\sum T_j$	NP-hard (Lemma 6)	Special case $1\|\|\sum T_j$ is NP-hard in the ordinary sense
$1(\textit{sa})\|r_j\|\max\sum T_j$	Polynomially solvable in $O(n^3)$ time (see Sect. 2.2)	
$1(\textit{sa})\|r_j\|\max\sum w_jT_j$	NP-hard (Theorem 2)	
$1(\textit{nd})\|r_j\|\max\sum w_jT_j$	strongly NP-hard (Aloulou et al. 2007)	strongly NP-hard
Other maximization problems		
$1(\textit{no-idle})\|D_j\|\max\sum U_j$	NP-hard (Lemma 6)	NP-hard (van den Akker and Hoogeveen 2004)
$1(\textit{no-idle})\|D_j\|\max\sum w_jC_j$	NP-hard (Lemma 7)	strongly NP-hard (Lenstra et al. 1977)
$1(\textit{no-idle})\|r_j\|\max\sum w_jC_j$	NP-hard (Lemma 7)	strongly NP-hard (Lenstra et al. 1977)

Thus, the problem $1(\textit{no-idle})|r_j|\max\sum w_jC_j$ is NP-hard.

In an analogous way, for the problem $1(\textit{no-idle})|D_j|\max\sum w_jC_j$, we consider an instance with $n=\bar{n}+1$ jobs, where $p_j=b_j, w_j=0,\ D_j=\sum_{i=1}^{\bar{n}+1}p_i,\ j=1,2,\ldots,\bar{n}$. In addition, let $p_{\bar{n}+1}=1, w_{\bar{n}+1}=1$ and $D_{\bar{n}+1}=A+1$. Now it is obvious that the instance of the partition problem has the answer "YES" if and only if $C_{\bar{n}+1}(\pi)=A+1$. □

Next, in Table 1, we give an overview on new results which we presented in this paper and a comparison with known results for the corresponding minimization problems. Here, NP-hard means that the existence of a pseudo-polynomial algorithm is unknown.

4 Concluding remarks

In this paper, we considered several scheduling problems, where the optimization criterion is opposite to the classical problems. Mainly, we considered complexity issues of such problems with total tardiness criterion. NP-hardness proofs for 6 open problems were presented as well as a pseudo-polynomial solution algorithm for the total weighted tardiness maximization problem.

For future research, it is interesting to compare the complexity of maximization and minimization versions for problems with other objective functions. Another subject can be the improvement of algorithms for the original problems by analyzing the solutions found in the maximization versions.

Acknowledgements This work has been partially supported by DAAD (Deutscher Akademischer Austauschdienst): A/08/80442/Ref. 325 and by RFBR (Russian Foundation for Basic Research): 11-08-13121. The authors are grateful to the anonymous referees for their constructive suggestions which helped us to improve the presentation.

References

Aloulou, M. A., & Artigues, C. (2010). Flexible solutions in disjunctive scheduling: general formulation and study of the flow-shop case. *Computers & Operations Research*, *37*(5), 890–898.

Aloulou, M. A., Kovalyov, M. Y., & Portmann, M.-C. (2004). Maximization problems in single machine scheduling. *Annals of Operations Research*, *129*, 21–32.

Aloulou, M. A., Kovalyov, M. Y., & Portmann, M.-C. (2007). Evaluation flexible solutions in single machine scheduling via objective function maximization: the study of computational complexity. *RAIRO. Recherche Opérationnelle*, *41*, 1–18.

Baker, K. R. (1974). *Introduction to sequencing and scheduling*. New York: Wiley.

Gafarov, E. R., Lazarev, A. A., & Werner, F. (2010a). Algorithms for maximizing the number of tardy jobs or total tardiness on a single machine. *Automation and Remote Control*, *71*(10), 2070–2084.

Gafarov, E. R., Lazarev, A. A., & Werner, F. (2010b). *Classical combinatorial and single machine scheduling problems with opposite optimality criteria*. Preprint 11/10, FMA, Otto-von-Guericke-Universität Magdeburg.

Gafarov, E. R., Lazarev, A. A., & Werner, F. (2012). Transforming a pseudo-polynomial algorithm for the single machine total tardiness maximization problem into a polynomial one. *Annals of Operations Research*, *196(1)*, 247–261.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Rinnooy Kan, A. H. G. (1979). Optimization and approximation in deterministic machine scheduling: a survey. *Annals of Discrete Mathematics*, *5*, 287–326.

Lawler, E. L. (1977). A pseudopolynomial algorithm for sequencing jobs to minimize total tardiness. *Annals of Discrete Mathematics*, *1*, 331–342.

Lawler, E. L., & Moore, J. M. (1969). A functional equation and its application to resource allocation and sequencing problems. *Management Science*, *16*(1), 77–84.

Lenstra, J. K., Rinnooy Kan, A. H. G., & Brucker, P. (1977). Complexity of machine scheduling problems. *Annals of Discrete Mathematics*, *1*, 343–362.

van den Akker, M., & Hoogeveen, H. (2004). Minimizing the number of tardy jobs. In Y.-T. Leung (Ed.), *Handbook of scheduling: algorithms, models and performance analysis*. London: Chapman & Hall.