Dynamic Programming Approach to Design FPTAS for Single Machine Scheduling Problems

Evgeny R. Gafarov

Ecole Nationale Superieure des Mines, FAYOL-EMSE, CNRS:UMR6158, LIMOS, F-42023 Saint-Etienne, France
Institute of Control Sciences of the Russian Academy of Sciences, Profsoyuznaya st. 65, 117997 Moscow, Russia,
email: axel73@mail.ru

Alexandre Dolgui

Ecole Nationale Superieure des Mines, FAYOL-EMSE, CNRS:UMR6158, LIMOS, F-42023 Saint-Etienne, France
email: dolgui@emse.fr

Frank Werner

Fakultät für Mathematik, Otto-von-Guericke-Universität Magdeburg, PSF 4120, 39016 Magdeburg, Germany,
email: frank.werner@mathematik.uni-magdeburg.de

Abstract

In this paper, we present a fully polynomial-time approximation schema (FPTAS) for some single machine problems, namely:

- minimizing weighted total tardiness when all due dates are equal,
- two cases of the total tardiness problem,
- a special case of the generalized total tardiness problem and
- maximizing weighted total tardiness.

The FPTAS is obtained by converting a graphical algorithm and has the best running time among the known FPTAS for the problems, that is polynomial in n and $1/\varepsilon$.

Keywords: Single machine scheduling, Weighted total tardiness, Graphical algorithm, FPTAS

Introduction

In this paper, several single machine total tardiness problems are considered. These problems 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. All jobs are assumed to be available for processing at time 0. For each job $j \in N$, a processing time $p_j > 0$, a weight $w_j > 0$ and a due date $d_j > 0$ are given.

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 $C_{j_k}(\pi) = \sum_{l=1}^{k} p_{j_l}$ be the completion time of job j_k in the schedule resulting from the sequence π. If $C_j(\pi) > d$, then job j is tardy. 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 of job j in the schedule resulting from sequence π and let $GT_j(\pi) = \min\{\max\{0, C_j(\pi) - d_j\}, p_j\}$.

In the weighted total tardiness minimization problem the objective is to find an optimal job sequence π^* that minimizes weighted total tardiness, i.e., $F(\pi) = \sum_{j=1}^{n} w_j T_j(\pi)$. Similarly, for the total tardiness problem an objective function $F(\pi) = \sum_{j=1}^{n} T_j(\pi)$ and for the generalized total tardiness problem $F(\pi) = \sum_{j=1}^{n} GT_j(\pi)$ have to be minimized. The following special cases of the problems are considered:

- minimizing weighted total tardiness when all due dates are equal, i.e., $d_j = d,\ i = 1, 2, \ldots, n$. This is denoted by $1|d_j = d|\sum w_j T_j$;
- the special case $B-1$ of the total tardiness problem $1||\sum T_j$, where $p_1 \geq p_2 \geq \cdots \geq p_n,\ d_1 \leq d_2 \leq \cdots \leq d_n$ and $d_n - d_1 < p_n$;
- the special case $B-1G$ of the problem $1||\sum T_j$, where $d_{max} - d_{min} < p_{min}$, where $d_{max} = \max_{j \in N}\{d_j\}$, $d_{min} = \min_{j \in N}\{d_j\}$ and $p_{min} = \min_{j \in N}\{p_j\}$.

For the NP-hard problem of maximizing weighted total tardiness $1(\textit{no-idle})||\max \sum w_j T_j$ [16, 18], the objective is to find an optimal job sequence that maximizes weighted total tardiness, where each feasible schedule starts at time 0 and there is no idle time between the processing of jobs. On the one hand, the investigation of a particular problem with the *maximum* criterion is a theoretically significant task. Algorithms for such a problem with the maximum criterion can be used to cut bad sub-problems in the branching tree of branch-and-bound algorithms, to compute upper and lower bounds for bi-criterion problems. On the other hand, these problems have also practical interpretations and applications [16, 18].

The problem $1||\sum T_j$ is NP-hard in the ordinary sense [2, 3]. A pseudo-polynomial dynamic programming algorithm of time complexity $O(n^4 \sum p_j)$ was proposed by Lawler [4]. The algorithms by Szwarc et al. [5] can solve special instances [6] of this problem for $n \leq 500$ jobs. For the NP-hard special case $B-1$ [3], a pseudo-polynomial algorithm is known [10]. A survey on the problem $1||\sum T_j$ is presented in [7]. For the NP-hard problem $1||\sum GT_j$ [1, 17], pseudo-polynomial algorithms are known as well [1, 17].

The above mentioned problems can be considered as subproblems in some situations, where a complex function $\Psi(t) + F(\pi, t)$ has to be minimized. The function $F(\pi, t)$ corresponds to one of the above mentioned functions $F(\pi)$ if the jobs are processed not from time 0, but from time t. As an alternative, we have to partition the t-axis into intervals with the same optimal schedule. For example, the single machine problem of minimizing the number of late jobs, when the starting time of the machine is variable, is considered in [8]. The same situation arises when it is known that some jobs have to be scheduled one by one in a "batch" from an unknown time point $t \in [t_1, t_2]$, e.g., a set N contains two subsets N_1, N_2 and an optimal job sequence can be represented as a concatenation (π_1, π_2, π_3) where $\{\pi_1\} \bigcup \{\pi_3\} = N_1$ and $\{\pi_2\} = N_2$. Jobs from N_2 have to be scheduled according to one of the above mentioned functions. The graphical and approximation algorithms presented in this paper can be used both for the initial problems and for problems with variable starting time.

Since the main topic of this paper is that of an analysis of approximation algorithms, we recall some relevant definitions. For the scheduling problem of minimizing a function $F(\pi)$, a polynomial-time algorithm that finds a feasible solution π' such that $F(\pi')$ is at most $\rho \geq 1$ times the optimal value $F(\pi^*)$ is called a *ρ-approximation algorithm; the value of ρ is called a worst-case ratio bound.* If a problem admits a ρ-approximation algorithm, it is said to be approximable within a factor ρ. A family of ρ-approximation algorithms is called a fully polynomial-time approximation scheme, or an FPTAS, if $\rho = 1+\varepsilon$ for any $\varepsilon > 0$ and the running time is polynomial with respect to both the length of the problem input and $1/\varepsilon$. Notice that a problem that is NP-hard in the strong sense admits no FPTAS unless P = NP.

For the problem $1 \mid\mid \min \sum T_j$, Lawler [9] converts his dynamic programming algorithm into an FPTAS that requires $O(n^7/\varepsilon)$ time. For the problem $1|d_j = d| \sum w_j T_j$, which is NP-hard in the ordinary sense [11], Fathi and Nuttle [12] provide a 2-approximation algorithm that requires $O(n^2)$ time. A FPTAS for the problem with the running time $O((n^6 \log \sum w_j)/\varepsilon^3)$ was presented by Kellerer and Strusevich in [13]. In [14], a heuristic algorithm is presented, which was incorrectly called FPTAS with the running time $O(n^2/\varepsilon)$. This algorithm is based on a wrong assumption that there is an optimal schedule where tardy jobs are scheduled in non-decreasing order of values p_j/w_j (it does not hold for the first tardy job). For the special case $B - 1$, an FPTAS with the running time $O(n^3 \log n + n^3/\varepsilon)$ is mentioned in [7].

For a practical realization of some pseudo-polynomial algorithms, one can use the idea from [15]. This modification of pseudo-polynomial algorithms for combinatorial problems with Boolean variables (e.g.

problems, where a job can be on-time or tardy, or an item is put into the knapsack or not) is called a *graphical approach.* In this paper, we present such a graphical modification of a pseudo-polynomial algorithm for the problem $1|d_j = d|\sum w_j T_j$ and a FPTAS based on this graphical algorithm with the running time $O(n^3/\varepsilon)$. In addition, modifications of these graphical algorithms and FPTAS for the cases $B-1$, $B-1G$ and the problems $1||\sum GT_j$ and $1(\textit{no-idle})||\max\sum w_j T_j$ are presented.

The rest of this paper is organized as follows. In Section 2, an exact pseudo-polynomial algorithm for the problem $1|d_j = d|\sum w_j T_j$ is presented. A graphical algorithm for the problem $1|d_j = d|\sum w_j T_j$ is given in Section 3. Its advantages in contrast to different dynamic programming algorithms are discussed in Section 4. In Section 5, a FPTAS based on the graphical algorithm is presented. Modifications of the graphical algorithm and FPTAS for the rest of the problems are described in Section 6.

1 Dynamic Programming for the Problem $1|d_j = d|\sum w_j T_j$

In this section, we present a property of an optimal sequence and an exact algorithm for the problem $1|d_j = d|\sum w_j T_j$ which are the base for the modification in Section 2.

Lemma 1 *[1, 13] There exists an optimal job sequence π for the problem $1|d_j = d|\sum w_j T_j$ that can be represented as a concatenation (G, x, H), where all jobs $j \in H$ are tardy and $S_j \geq d$, $\forall j \in H$, and all jobs $i \in G$ are on-time. All jobs from the set G are processed in non-increasing order of values $\frac{p_j}{w_j}$ and all jobs from the set H are processed in non-decreasing order of values $\frac{p_j}{w_j}$. The job x starts before time d and is completed no earlier than time d.*

The job x is called *straddling.*

Assume that the jobs are numbered as follows:

$$\frac{p_2}{w_2} \leq \frac{p_3}{w_3} \leq \cdots \leq \frac{p_n}{w_n}, \qquad (1)$$

where the job with number 1 is a straddling job. As a corollary from Lemma 1, there is a straddling job x, $x \in N$, to which the number 1 will be assigned, such that for each $l \in \{1, 2, \ldots, n\}$, there exists an optimal schedule in which all jobs $j \in \{1, 2, \ldots, l\}$ are processed from time t one by one, and there is no job $i \in \{l+1, l+2, \ldots, n\}$ which is processed between these jobs. Thus, we can present a dynamic programming algorithm (DPA) based on Lemma 1. For each x, $x \in N$, we number the jobs from the set $N \setminus \{x\}$ according to the rule (1) and perform Algorithm 1. Then we choose a best found schedule among the n constructed schedules. At each stage $l, 1 \leq l \leq n$, of Algorithm 1, we construct a best partial sequence $\pi_l(t)$ for the set of jobs $\{1, 2, \ldots, l\}$ and for each possible starting time t of the first job (which represents a possible state in the dynamic programming algorithm). $F_l(t)$ denotes the weighted total tardiness value for the job sequence $\pi_l(t)$. $\Phi^1(t)$ and $\Phi^2(t)$ are temporary functions, which are used to compute $F_l(t)$.

Algorithm 1

```
1. Enumerate the jobs according to order (1);
2. FOR $t := 0$ TO $\sum_{j=2}^{n} p_j$ DO
       $\pi_1(t) := (1)$, $F_1(t) := w_1 \max\{0, p_1 + t - d\}$;
3. FOR $l := 2$ TO $n$ DO
       FOR $t := 0$ TO $\sum_{j=l+1}^{n} p_j$ DO
           $\pi^1 := (l, \pi_{l-1}(t + p_l))$, $\pi^2 := (\pi_{l-1}(t), l)$;
           $\Phi^1(t) := w_l \max\{0, p_l + t - d\} + F_{l-1}(t + p_l)$;
           $\Phi^2(t) := F_{l-1}(t) + w_l \max\left\{0, \sum_{j=1}^{l} p_j + t - d\right\}$;
           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 for the chosen job $x$ with the
   objective function value $F_n(0)$.
```

Theorem 1 *By using Algorithm 1 for each x, $x \in N$, an optimal job sequence of the type described in Lemma 1 can be found in $O(n^2 \sum p_j)$ time.*

Proof. We prove the theorem indirectly. Assume that there exists an optimal job sequence of the form $\pi^* = (G, 1, H)$ of the type described in Lemma 1. Assume that $F(\pi^*) < F(\pi_n(0)) = F_n(0)$.

Let $\pi' := \pi^*$. For each $l = 1, 2, \ldots, n$, we successively consider the part $\bar{\pi}_l \in \pi'$ of the schedule with $\{\bar{\pi}_l\} = \{1, 2, \ldots, l\}$. Let $\pi' = (\pi_\alpha, \bar{\pi}_l, \pi_\beta)$ and $t^* = \sum_{i \in \pi_\alpha} p_i$. If $\bar{\pi}_l \neq \pi_l(t^*)$, then $\pi' := (\pi_\alpha, \pi_l(t^*), \pi_\beta)$. It is obvious that $F((\pi_\alpha, \bar{\pi}_l, \pi_\beta)) \geq F((\pi_\alpha, \pi_l(t^*), \pi_\beta))$. Applying this procedure to subsequent values l, we have $F(\pi^*) \geq F(\pi') = F_n(0)$ at the end. Thus, the schedule $\pi_n(0)$ is also optimal for a chosen job $x = 1$.

Obviously, the time complexity of Algorithm 1 is equal to $O(n \sum p_j)$. So, an optimal job sequence can be found in $O(n^2 \sum p_j)$ time $\square$

It is obvious, that for some chosen job $x,\ x \in N$, in a found job sequence $\pi_n(0)$ the job x will be not straddling (i.e., either $S_x \geq d$ or $C_x < d$). This means that there exists another job $x', x' \in N$ for which the value $F_n(0)$ will be less.

Algorithm 1 can be modified by considering for each $l = 1, 2, \ldots, n$, only the interval $[0, d - p_l]$ instead of the interval $[0, \sum\limits_{i=l+1}^{n} p_i]$ since for each $t > d - p_l$, job j is tardy in any partial sequence $\pi_l(t)$ and the partial sequence $\pi^2 := (\pi_{l-1}(t), l)$ is optimal. Thus, the time complexity of the modified Algorithm 1 is equal to $O(nd)$ and the optimal schedule can be found in $O(n^2d)$ time which is equal to the running time of the solution algorithm for the problem presented in [1].

Let UB be an upper bound on the optimal function value for the problem which is found by the 2-approximation algorithm of Fathi and Nuttle [12], i.e., $UB \leq 2F(\pi^*)$. If for some $t_l^{UB} \in (-\infty, +\infty)$ we have $F_l(t_l^{UB}) = UB$, then for each $t > t_l^{UB}$ we have $F_l(t) > UB$ (since t denotes the starting time of an optimal schedule obtained from the job sequence $\pi_l(t)$ for jobs $1, 2, \ldots, l$ and $F_l(t)$ denotes the corresponding objective function value). So, the states $t > t_l^{UB}$ seem to be unpromising, i.e., for any job sequence π, constructed using these states, we will have $F(\pi) > UB$, i.e., π is not optimal. Thus, we can consider different values $F_l(t)$ only for $t \in [0, t_l^{UB}]$ and assume $F_l(t) = +\infty$ for $t \in (t_l^{UB}, +\infty)$. If all parameters $p_j, w_j,\ \forall j \in N$, and d are integer, then there are at most $UB + 2$ different values $F_l(t)$.

In addition, we can note the following. For each function $F_l(t)$, there can be only one interval $[t', t'']$, $t' \leq t''$, where all the values $F_l(t)$ are equal. Moreover, $F_l(t) = 0$. If $F(t'') < F(t'' + 1)$, then $F(t) < F(t + 1)$ for all $t > t''$. Thus, we can modify Algorithm 1 as follows. If we will save at each stage l instead of all states $t \in [t', t'']$ only one state t'', then the number of saved states will be restricted by UB, since for all saved stated t we have $F(t) < F(t+1)$. The running time of the modified algorithm is $O(n \min\{d, UB\})$. If we consider only the states $t \in [d - \sum_{j=1}^{n} p_j, d]$ instead of $[0, d]$ at each stage $l, l = 1, 2, \ldots, n$, then we obtain an optimal solution for the chosen straddling job for each possible starting time $t \in (-\infty, t_n^{UB})$ in $O(nUB)$ time.

Let π' be a job sequence, where all n jobs are processed in non-decreasing order of the values $\frac{p_j}{w_j}$. Denote by $F(\pi', d)$ the weighted total tardiness for the job sequence π', where the processing of the jobs starts at time d. It is obvious that for this starting time the schedule π' is optimal. Then, by the modified Algorithm 1, we can obtain an optimal solution for each possible starting time $t \in (-\infty, +\infty)$ in $O(n^2 F(\pi', d))$ time.

We note that the inequality $F(t) < F(t + 1)$ does not necessarily hold for all $t > t''$ in the problem $1||\sum GT_j$, i.e., the running time of Algorithm 1 is not restricted by UB for the problem $1||\sum GT_j$.

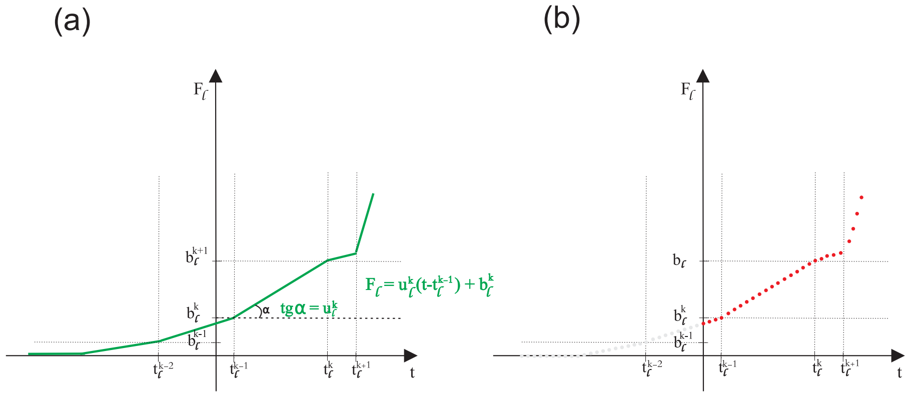

Figure 1: Function $F_l(t)$ in the GrA and in Algorithm 1

2 Graphical Algorithm for the Problem $1|d_j = d|\sum w_j T_j$

In this section, a new graphical algorithm (GrA) is derived for this problem which is based on an idea from [15]. The GrA is a modification of Algorithm 1, in which function $F_l(t)$ is defined for any $t \in (-\infty, +\infty)$ (not only for integer t). However, we need to compute these values only at the *break* points separating intervals in which function $F_l(t)$ is a linear function of the form $F_l(t) = F_l^k(t) = u_l^k \cdot (t - t_l^{k-1}) + b_l^k$. In Theorem 2, we are to prove that $F_l(t)$ is a continuous piecewise linear function whose parameters can be described in a tabular form (like in Table 1).

In each step of the GrA, we store the information on function $F_l(t)$ for a number of intervals (characterized by the same best partial sequence and by the same number of tardy jobs) in a tabular form as given in Table 1.

Table 1: Function $F_l(t)$

k	1	2	...	$m_l + 1$	$m_l + 2$
interval k	$(-\infty, t_l^1]$	$(t_l^1, t_l^2]$	...	$(t_l^{m_l}, t_l^{m_l+1}]$	$(t_l^{m_l+1}, +\infty)$
b_l^k	0	b_l^2	...	$b_l^{m_l+1}$	$+\infty$
u_l^k	0	u_l^2	...	$u_l^{m_l+1}$	0
π_l^k	π_l^1	π_l^2	...	$\pi_l^{m_l+1}$	$(1, 2, \dots, l)$

In Table 1, k denotes the number of the current interval whose values range from 1 to $m_l + 2$ (where the number of intervals $m_l + 2$ is defined for each $l = 1, 2, \dots, n$), $(t_l^{k-1}, t_l^k]$ is the kth interval (where

$t_l^0 = -\infty$, $t_l^{m_l+2} = \infty$ and $t_l^{m_l+1} = t_l^{UB}$), b_l^k, u_l^k are the parameters of the linear function $F_l^k(t)$ defined in the kth interval, and π_l^k is the best sequence of the first l jobs if they are processed from time $t \in (t_l^{k-1}, t_l^k]$.

These data mean the following. For each above interval, we store the parameters b_l^k and u_l^k for describing function $F_l(t)$ and the resulting best partial sequence if the first job starts in this interval. For each starting time $t \in (t_l^{k-1}, t_l^k]$ ($t_l^0 = -\infty$) of the first job, we have a best partial sequence π_l^k of the jobs $1, 2, \ldots, l$ with a total weight of the tardy jobs u_l^k and the function value $F_l(t) = u_l^k \cdot (t - t_l^{k-1}) + b_l^k$ (see Fig. 1). $F_l(t) = 0$, for $t \in (t_l^0, t_l^1]$. Recall that function $F_l(t)$ is defined not only for integers t, but also for real numbers t. For simplicity of the following description, we consider the whole t-axis, i.e., $t \in (-\infty, +\infty)$. In Theorem 2, we prove that this table describes a function $F_l(t)$ which is continuous, piecewise-linear function in interval $(-\infty, t_l^{UB}]$. The points $t_l^1, t_l^2, \ldots, t_l^{m_l+1}$ are called *break points*, since there is a change from value u_l^{k-1} to u_l^k (which means that the slope of the piecewise-linear function changes). Note that some of the break points t_l^k can be non-integer. To describe each linear segment, we store its slope u_l^k and its function value $b_l^k = F_l(t)$ at the point $t = t_l^{k-1}$. So, in the table $b_l^1 < b_l^2 < \cdots < b_l^{m_l+1} < UB$, since $t_l^1 < t_l^2 < \cdots < t_l^{m_l+1}$.

In the GrA, the functions $F_l(t)$ reflect the same functional equations as in Algorithm 1, i.e., for each $t \in Z \bigcap [0, \sum_{j=2}^n p_j]$, the function $F_l(t)$ has the same value as in Algorithm 1 (see Fig. 1), but these functions are now defined for any $t \in (-\infty, +\infty)$. As a result, often a large number of integer states is combined into one interval (describing a new state in the resulting algorithm) with the same best partial sequence. In Fig. 1 (a), the function $F_l(t)$ from the GrA is presented and in Fig. 1 (b), the function $F_l(t)$ from Algorithm 1 is displayed.

Next, the GrA is described. The core is Step 3, where we explain how the states at stage $l, l > 1$, are obtained if the states at stage $l-1$ are known.

Graphical algorithm (GrA)

Step 1. We enumerate the jobs according to order (1);

Step 2. Set $l := 1$, $\pi_1(t) := (1)$, $F_1(t) := w_1 \max\{0, p_1 + t - d\}$ for all t. We represent function $F_1(t)$ in a tabular form as given in Table 2. For all three intervals, there is the same best partial sequence (1). For $t \in (-\infty, d - p_1]$, there is no tardy job in the schedule corresponding to sequence (1) when this job is started at time t and for $t \in (d - p_1, +\infty)$, there is one tardy job. The value t_1^{UB} can be found from the equation $UB = (t_1^{UB} - (d - p_1))w_1 + 0$.

Table 2: Function $F_1(t)$

k	1	2	3
interval k	$(-\infty, d-p_1]$	$(d-p_1, t_1^{UB})$	$(t_1^{UB}, +\infty)$
b_1^k	0	0	$+\infty$
u_1^k	0	w_1	0
π_1^k	(1)	(1)	(1)

Step 3. Let $l > 1$ and assume that function $F_{l-1}(t)$ and the best partial sequences of the jobs $\{1, 2, \ldots, l-1\}$ for all resulting intervals given in Table 3 are known, where $t_{l-1}^{m_{l-1}+1} = t_{l-1}^{UB}$

Table 3: Function $F_{l-1}(t)$

k	1	2	$\ldots$	$m_{l-1}+1$	$m_{l-1}+2$
interval k	$(-\infty, t_{l-1}^1]$	$(t_{l-1}^1, t_{l-1}^2]$	$\ldots$	$(t_{l-1}^{m_{l-1}}, t_{l-1}^{UB}]$	$(t_{l-1}^{UB}, +\infty)$
b_{l-1}^k	0	b_{l-1}^2	$\ldots$	$b_{l-1}^{m_{l-1}+1}$	$+\infty$
u_{l-1}^k	0	u_{l-1}^2	$\ldots$	$u_{l-1}^{m_{l-1}+1}$	0
π_{l-1}^k	π_{l-1}^1	π_{l-1}^2	$\ldots$	$\pi_{l-1}^{m_{l-1}+1}$	$(1, 2, \ldots, l-1)$

In the following, we describe how function $F_l(t)$ is obtained by means of function $F_{l-1}(t)$. Note that we can store the temporary functions $\Phi^1(t)$ and $\Phi^2(t)$ determined in Steps 3.1 and 3.2 also in a tabular form as in Table 1.

Step 3.1. The function $\Phi^1(t)$ is obtained from function $F_{l-1}(t)$ by the following operations. We shift the diagram of function $F_{l-1}(t)$ to the left by the value p_l and in the table for function $F_{l-1}(t)$ and add a column which results from the new break point $t' = d - p_l$. If $t_{l-1}^s - p_l < t' < t_{l-1}^{s+1} - p_l$, $s \leq m_{l-1}$, then we have two new intervals of t in the table for $\Phi^1(t)$: $(t_{l-1}^s - p_l, t']$ and $(t', t_{l-1}^{s+1} - p_l]$ (for $s + 1 = m_{l-1} + 1$, we have $(t_{l-1}^{m_{l-1}+1} - p_l, t']$ and (t', ∞)). This means that we first replace each interval $(t_{l-1}^k, t_{l-1}^{k+1}]$ by $(t_{l-1}^k - p_l, t_{l-1}^{k+1} - p_l]$ in the table for $\Phi^1(t)$, and then replace the column with the interval $(t_{l-1}^s - p_l, t_{l-1}^{s+1} - p_l]$ by two new columns with the intervals $(t_{l-1}^s - p_l, t']$ and $(t', t_{l-1}^{s+1} - p_l]$.

Moreover, we increase the values $u_{l-1}^{s+1}, u_{l-1}^{s+2}, \ldots, u_{l-1}^{m_{l-1}+1}$ by w_l, i.e., the total weight of tardy jobs (and thus the slope of the function) increases. The corresponding partial sequences π^1 are obtained by adding job l as the first job to each previous partial sequence. In this way, we obtain the information on function $\Phi^1(t)$ together with the corresponding partial sequences given in Table 4, which also includes the calculation of the corresponding b values.

Step 3.2. The function $\Phi^2(t)$ is obtained from function $F_{l-1}(t)$ by the following operations. In the table for $F_{l-1}(t)$, we add a column which results from the new break point $t'' = d - \sum_{i=1}^{l} p_i$. If $t_{l-1}^h <$

Table 4: Function $\Phi^1(t)$

interval k	$(-\infty, t_{l-1}^1 - p_l]$	$(t_{l-1}^1 - p_l, t_{l-1}^2 - p_l]$	$\dots$	$(t_{l-1}^s - p_l, t']$	$(t', t_{l-1}^{s+1} - p_l]$	$(t_{l-1}^{s+1} - p_l, t_{l-1}^{s+2} - p_l]$	$\dots$	$(t_{l-1}^{m_{l-1}} - p_l, t_{l-1}^{UB}]$	$(t_{l-1}^{UB} - p_l, +\infty)$
b_k	0	b_{l-1}^2	$\dots$	b_{l-1}^{s+1}	b'	$\widetilde{b}_{l-1}^{s+2}$	$\dots$	$\widetilde{b}_{l-1}^{m_{l-1}+1}$	$+\infty$
u_k	0	u_{l-1}^2	$\dots$	u_{l-1}^{s+1}	$u_{l-1}^{s+1} + w_l$	$u_{l-1}^{s+2} + w_l$	$\dots$	$u_{l-1}^{m_{l-1}+1} + w_l$	0
best partial sequence π^1	(l, π_{l-1}^1)	(l, π_{l-1}^2)	$\dots$	(l, π_{l-1}^{s+1})	(l, π_{l-1}^{s+1})	(l, π_{l-1}^{s+2})	$\dots$	$(l, \pi_{l-1}^{m_{l-1}+1})$	$(1, 2, \dots, l)$

$b' = b_{l-1}^{s+1} + (t' - (t_{l-1}^s - p_l)) \cdot u_{l-1}^{s+1}$, $\widetilde{b}_{l-1}^{s+2} = b' + ((t_{l-1}^{s+1} - p_l) - t') \cdot (u_{l-1}^{s+1} + w_l)$, $\dots$, $\widetilde{b}_{l-1}^{m_{l-1}+1} = \widetilde{b}_{l-1}^{m_{l-1}} + (t_{l-1}^{m_{l-1}} - t_{l-1}^{m_{l-1}-1}) \cdot (u_{l-1}^{m_{l-1}} + w_l)$

Table 5: Function $\Phi^2(t)$

interval k	$(-\infty, t_{l-1}^1]$	$(t_{l-1}^1, t_{l-1}^2]$	$\dots$	$(t_{l-1}^h, t'']$	$(t'', t_{l-1}^{h+1}]$	$(t_{l-1}^{h+1}, t_{l-1}^{h+2}]$	$\dots$	$(t_{l-1}^{m_{l-1}}, t_{l-1}^{UB}]$	$(t_{l-1}^{UB}, +\infty)$
b_k	0	b_{l-1}^2	$\dots$	b_{l-1}^{h+1}	b''	$\widetilde{b}_{l-1}^{h+2}$	$\dots$	$\widetilde{b}_{l-1}^{m_{l-1}+1}$	$+\infty$
u_k	0	u_{l-1}^2	$\dots$	u_{l-1}^{h+1}	$u_{l-1}^{h+1} + w_l$	$u_{l-1}^{h+2} + w_l$	$\dots$	$u_{l-1}^{m_{l-1}+1} + w_l$	0
best partial sequence π^2	(π_{l-1}^1, l)	(π_{l-1}^2, l)	$\dots$	(π_{l-1}^{h+1}, l)	(π_{l-1}^{h+1}, l)	(π_{l-1}^{h+2}, l)	$\dots$	$(\pi_{l-1}^{m_{l-1}+1}, l)$	$(1, 2, \dots, l)$

$b'' = b_{l-1}^{h+1} + (t'' - t_{l-1}^h) \cdot u_{l-1}^{h+1}$, $\widetilde{b}_{l-1}^{h+2} = b'' + (t_{l-1}^{h+1} - t'') \cdot (u_{l-1}^{h+1} + w_l)$, $\dots$, $\widetilde{b}_{l-1}^{m_{l-1}+1} = \widetilde{b}_{l-1}^{m_{l-1}} + (t_{l-1}^{m_{l-1}} - t_{l-1}^{m_{l-1}-1}) \cdot (u_{l-1}^{m_{l-1}} + w_l)$

$t'' < t_{l-1}^{h+1}$, $h+1 \leq m_{l-1}+1$, then there are two new intervals of t in the table for $\Phi^2(t)$: $(t_{l-1}^{h}, t'']$ and $(t'', t_{l-1}^{h+1}]$ (for $h = m_{l-1}+1$, we have $(t_{l-1}^{m_{l-1}+1}, t'']$ and (t'', ∞)).

This means that we replace the column with the interval $(t_{l-1}^{h}, t_{l-1}^{h+1}]$ by two new columns with the intervals $(t_{l-1}^{h}, t'']$ and $(t'', t_{l-1}^{h+1}]$.

Moreover, we increase the values $u_{l-1}^{h+1}, u_{l-1}^{h+2}, \dots, u_{l-1}^{m_{l-1}+1}$ by w_l, i.e., the total weight of tardy jobs increases. The corresponding partial sequences π^2 are obtained by adding job l at the end to each previous partial sequence. In this way, we obtain the information on function $\Phi^2(t)$ together with the corresponding partial sequences given in Table 5.

Step 3.3. Now we construct a table that corresponds to the function

$$F_l(t) = \min\{\Phi^1(t), \Phi^2(t)\}.$$

Consider all resulting intervals from both tables and search for intersection points of the diagrams of functions $\Phi^1(t)$ and $\Phi^2(t)$. Then we construct function $F_l(t)$ as the minimum of both functions obtained.

To be more precise, we construct a list $t_1, t_2, \dots, t_e$, $t_1 < t_2 < \dots, t_e$, of all break points t from the tables for $\Phi^1(t)$ and $\Phi^2(t)$, which are left / right boundary points of the intervals given in these tables. Then we consider each interval $(t_i, t_{i+1}]$, $i = 1, 2, \dots, e-1$, and compare the two functions $\Phi^1(t)$ and $\Phi^2(t)$ over this interval. Let the interval $(t_i, t_{i+1}]$ be contained in the interval $(t_{z-1}, t_z]$ from the table for $\Phi^1(t)$ and in the interval $(t_{y-1}, t_y]$ from the table for $\Phi^2(t)$. Then $\Phi^1(t) = (t - t_{z-1}) \cdot u_z + b_z$ and $\Phi^2(t) = (t - t_{y-1}) \cdot u_y + b_y$. Choose the column from both tables corresponding to the maximum of the two functions in the interval $(t_i, t_{i+1}]$ and insert this column into the table for $F_l(t)$. If there exists an intersection point t''' of $\Phi^1(t)$ and $\Phi^2(t)$ in this interval, then insert two columns with the intervals $(t_i, t''']$ and $(t''', t_{i+1}]$.

This step requires $O(m_{l-1})$ operations.

Step 3.4. Let m be the number of columns in the resulting table of $F_l(t)$ and for k, $1 < k < m$, the inequality $b_l^k < UB \leq b_l^{k+1}$ holds. Then compute from the equality $UB = (t_l^{UB} - t_l^{k-1})u_l^k + b_l^k$ the value t_l^{UB}. In the column with an interval $(t_l^{k-1}, t_l^k]$ (column k) assign $t_l^k = t_l^{UB}$ and substitute all the columns $k+1, k+2, \dots, m$, by one column with the interval $(t_l^{UB}, +\infty)$, $b_l^{k+1} = +\infty$, $u_l^{k+1} = 0$ and $\pi_l^{k+1} = (1, 2, \dots, l)$. So, in the resulting table, there will be no more then $UB + 2$ columns if all the parameters of the problem are integer.

Step 3.5. If $l = n$, then GOTO 4 else $l := l + 1$ and GOTO 3.

Step 4. In the table corresponding to function $F_n(t)$ we determine the column $(t_n^k, t_n^{k+1}]$, where $t_n^k < 0 \leq t_n^{k+1}$. Then we have an optimal sequence $\pi^* = \pi_n^{k+1}$ for a chosen job x and the optimal function value $F(\pi^*) = b_n^{k+1} + (0 - t_n^k) \cdot u_n^{k+1}$.

Theorem 2 *By using the GrA for each x, $x \in N$, an optimal job sequence of the type described in Lemma 1 will be found in $O(n^2F^*)$ time, where F^* is the optimal objective function value.*

Proof.

First, prove that all functions $F_l(t),\ l = 1, 2, \ldots, n$, defined at the beginning of Section 3 are continuous and piecewise linear functions in the interval $(-\infty, t_l^{UB}]$.

It is obvious that function $F_1(t)$ is a continuous and piecewise linear function with two break points in the interval $(-\infty, t_1^{UB}]$. According to the operations described in Step 3, both functions $\Phi^1(t)$ and $\Phi^2(t)$ are also continuous and piecewise linear functions in the intervals $(-\infty, t_1^{UB} - p_2]$ and $(-\infty, t_1^{UB}]$. Thus, the function

$$F_2(t) = \min\{\Phi^1(t), \Phi^2(t)\}$$

and its modification obtained in Step 3.4. are continuous and piecewise linear functions in the interval $(-\infty, t_2^{UB}]$ as well. Continuing in this way, all functions $F_l(t), l = 1, 2, \ldots, n$, have the above properties.

Now assume that we have obtained Table 6 for some function $F_l(t)$ in Step 3.

Table 6: Function $F_l(t)$

k	1	2	$\ldots$	s	$s+1$	$\ldots$	m_l+1	m_l+2
interval k	$(-\infty, t_l^1]$	$(t_l^1, t_l^2]$	$\ldots$	$(t_l^{s-1}, t_l^s]$	$(t_l^s, t_l^{s+1}]$	$\ldots$	$(t_l^{m_l}, t_l^{UB}]$	$(t_l^{UB}, +\infty)$
b_l^k	0	b_l^2	$\ldots$	b_l^s	b_l^{s+1}	$\ldots$	$b_l^{m_l+1}$	$+\infty$
u_l^k	0	u_l^2	$\ldots$	u_l^s	u_l^{s+1}	$\ldots$	$u_l^{m_l+1}$	0
π_l^k	π_l^1	π_l^2	$\ldots$	π_l^s	π_l^{s+1}	$\ldots$	$\pi_l^{m_l+1}$	$(1, 2, \ldots, l)$

We prove that $b_l^1 < b_l^2 < \cdots < b_l^s < b_l^{s+1} < \cdots < b_l^{m_l+1} < UB$ holds. Assume that we have $b_l^s \geq b_l^{s+1}$. Let $\overline{F}(\pi, t)$ be the weighted total tardiness value of the sequence π when the first job starts at time t. For each $t \in (t_l^s, t_l^{s+1}]$, we have $\overline{F}(\pi_l^s, t) \geq \overline{F}(\pi_l^{s+1}, t)$, since for $t \in (t_l^{s+1}, t_l^{s+2}]$ the inequality $\overline{F}(\pi_l^s, t) \geq \overline{F}(\pi_l^{s+1}, t)$ holds, which is a contradiction. So, in Table 6, there are at most $UB + 2$ columns and $l + 1$ break points.

Therefore, Step 3 requires $O(UB)$ operations for each $l = 1, 2, \ldots, n$.

According to the GrA, it is obvious that at each integer point $t \in (-\infty, t_l^{UB}]$, the value $F_l(t)$ is equal to that determined by Algorithm 1 (since in the GrA, the functions $F_l(t)$ represent the same equations as in Algorithm 1 for the integer values t considered).

For some chosen straddling jobs x, it can be that $F_n(0) = +\infty$. This means that the best job sequence with the chosen straddling job x is not better then a job sequence π^{UB} for which $F(\pi^{UB}) < F(\pi_n(0))$.

Therefore, by using the GrA for each $x,\ x \in N$, an optimal job sequence of the type described in Lemma 1 for some x with the optimal value $F_n(0)$ will be found. The time complexity of such a search is $O(n^2F^*)$ time, since $\frac{1}{2}UB \leq F^* \leq UB$.

□

3 Advantages of the Graphical Algorithm to solve $1|d_j = d|\sum w_jT_j$

In fact, in each step $j = 1, 2, \ldots, n$ of the GrA, we do not consider all points $t \in [0, d],\ t \in Z$, but only points from the interval in which the optimal partial solution changes or where the resulting functional equation of the objective function changes. So, the main difference is that we operate **not with independent values F in each of the points t, but with functions which are transformed in each step analytically, according to their analytical form,** which can have obvious advantages. For example, let us minimize a function $\Psi(t) + F(\pi, t)$, where the function $F(\pi, t)$ corresponds to a function $F(\pi)$ when the jobs are processed not from time 0, but from time t. If the computed function $F(t)$ is presented analytically (not in a tabular form (t, F)) and the function $\Psi(t)$ is presented analytically as well, then the search for the minimum of $\Psi(t) + F(\pi, t)$ will be made in shorter time.

Moreover, such an approach has the following advantages when compared with Algorithm 1 (DPA):

1. The GrA can solve instances, where (some of) the parameters $p_j,\ w_j,\ j = 1, 2, \ldots, n$ or/and d are not in Z.
2. The running time of the GrA for two instances with the parameters $\{p_j, w_j, d\}$ and $\{p_j \cdot 10^{const} \pm 1, w_j \cdot 10^{const} \pm 1, d \cdot 10^{const} \pm 1\}, const > 1$, is the same while the running time of the DPA will be 10^{const} times larger in the second case. Thus, one can usually solve considerably larger instances with the GrA.
3. Properties of an optimal solution can be taken into account, and sometimes the GrA has even a polynomial time complexity, or we can at least essentially reduce the complexity of the standard *DPA* (see the experimental results in Section 6).
4. Unlike DPA, it is possible to construct a FPTAS based on GrA easily. The FPTAS is presented in the next section.

Let us consider another type of a DPA. This algorithm generates iteratively some sets of states. In every iteration l, $l = n, n-1, \dots, 1$, a set of states is generated. Each state can be represented by a string of the form (t, F), where t is the completion time of the last known job scheduled in the beginning of a schedule and F is the value of the function, provided that the early jobs start at time 0 and the last known late job completes exactly at time $\sum_{j=1}^{n} p$. This algorithm can be described as follows:

Alternative DPA

```
1. Enumerate the jobs according to their order (1);
2. In the set of states $V_{n+1}$, put a state $(0, 0)$.
3. FOR $l := n$ TO 1 DO
       FOR each state $(t, F)$ from the set of states $V_{l+1}$ DO
           Put a state $[t + p_l, F + w_l \max\{0, t + p_l - d\}]$;
           Put a state $[t, F + w_l \max\{0, \sum_{j=1}^{n} p_j - (\sum_{j=1}^{l-1} p_j - t) - d\}]$;
4. Find $F^* = \min\{F | (t, F) \in V_1\}$.
```

We need to consider only states, where $t \le d$ and $F \le UB$. If in a list V_l, there are two states with the same objective function value (t_1, F') and (t_2, F') and $t_2 > t_1$, then the state (t_2, F') can be removed from consideration. So, the running time of the Alternative DPA is $O(n \min\{d, F^*\})$ which corresponds to the running time of GrA (note that the GrA can be easily modified to consider only points $t \in [0, d]$). However, in the GrA some of possible but unpromising states are not considered and, in contrast to the alternative DPA, the GrA finds all optimal schedules for all $t \in [-\infty, t^{UB}]$ in time $O(nF^*)$. So, the alternative DPA is not effective for the problems of minimizing a complex function $\Psi(t) + F(\pi, t)$.

4 FPTAS for the Problem $1|d_j = d| \sum w_j T_j$

The idea of the FPTAS is as follows. Let $\delta = \frac{\varepsilon UB}{2n}$. To reduce the time complexity of the graphical algorithm, we have to diminish the number of columns considered, which is the number of different objective function values $b_l^1, b_l^2, \dots, b_l^{m_l+1}$. If we consider not the original values b_l^k but the values $\overline{b_l^k}$ which are rounded up or down to the nearest multiple of δ values b_l^k, there are no more than $\frac{UB}{\delta} = \frac{2n}{\varepsilon}$ different values $\overline{b_l^k}$. Then we will be able to convert the table $F_l(t)$ into a similar table with no more than $4\frac{n}{\varepsilon}$ columns. Furthermore, for such a modified table (function) $F'(t)$, we will have $|F(t) - F'(t)| < \delta \le \frac{\varepsilon F(\pi^*)}{n}$. If we do the rounding and modification after each step 3.4. of the graphical algorithm, then the cumulative error will be no more than $n\delta \le \varepsilon F(\pi^*)$ and the total running time of n runs of the graphical algorithm will be $O(\frac{n^3}{\varepsilon})$, i.e., an FPTAS is received.

By transforming the graphical algorithm, we save the approximated functions $F_l(t)$ in the same tabular form but without the last row which describes an optimal partial job sequence π_l^k. The data from the last row are saved in a tabular form described in Step 3.5. (see below). These n saved tables, corresponding to the n last rows, are used to restore an approximate job sequence.

FPTAS (as a modification of the graphical algorithm).

In the modified graphical algorithm, Steps 1, 2, 3.1., 3.2., 3.3. and 3.4. remain the same. Instead of Step 3.5, we use the following steps. Assume that we have obtained Table 6 for a function $F_l(t)$ in Step 3.4 with the different objective function values $b_l^1, b_l^2, \ldots, b_l^{m_l+1}$.
Step 3.5. Save Table 7.
Table 7: Positions of the job l

k	1	2	$\ldots$	m_l+1	m_l+2
interval k	$(-\infty, t_l^1]$	$(t_l^1, t_l^2]$	$\ldots$	$(t_l^{m_l}, t_l^{UB})$	$(t_l^{UB}, +\infty)$
$position_l^k$	$position_l^1$	$position_l^2$	$\ldots$	$position_l^{m_l+1}$	2

where $position_l^k := 1$ if in the partial sequence π_l^k job l is the first job and $position_l^k = 2$ if job l is the last one (there are only two possibilities).
Step 3.6. If $m_l \le 4\frac{n}{\varepsilon}$, then GOTO 3.7. Else do the following. Round all the values b_l^k from Table 6 to the nearest multiple of δ. Let the values $\overline{b_l^1}, \overline{b_l^2}, \ldots, \overline{b_l^{m_l+1}}$ be obtained, where $\overline{b_l^1} \le \overline{b_l^2} \le \cdots \le \overline{b_l^{m_l+1}}$. We modify the table $F_l(t)$ as follows. Assume that, for $k_1 < k_2$, we have $\overline{b_l^{k_1}} < \overline{b_l^{k_1+1}} = \cdots = \overline{b_l^{k_2}} < \overline{b_l^{k_2+1}}$. We substitute the columns which correspond to the values $\overline{b_l^{k_1}}, \ldots, \overline{b_l^{k_2-1}}$ for the two columns presented in Table 8.
Table 8: Substitution of columns

interval k	$\ldots$	$(t_l^{k_1-1}, t_l^{k_1}]$	$(t_l^{k_1}, t_l^{k_2-1}]$	$\ldots$	$(t_l^{UB}, +\infty)$
b_l^k	$\ldots$	$\overline{b_l^{k_1}}$	$\overline{b_l^{k_2}}$	$\ldots$	$+\infty$
u_l^k	$\ldots$	$u = \frac{\overline{b_l^{k_2}} - \overline{b_l^{k_1}}}{t_l^{k_1} - t_l^{k_1-1}}$	0	$\ldots$	0

Step 3.7. If $l = n$, then GOTO 4 else $l := l + 1$ and GOTO 3.
End of the modification of the graphical algorithm.

Let us analyze the substitution proposed in Step 3.6. Let $\overline{F}_l(t)$ be the function, received after Step 3.4 in the modified algorithm. In fact, the function $F_l(t)$ was modified in Step 3.6 in a way shown in Fig. 2. Let $F_l'(t)$ present the modified function.

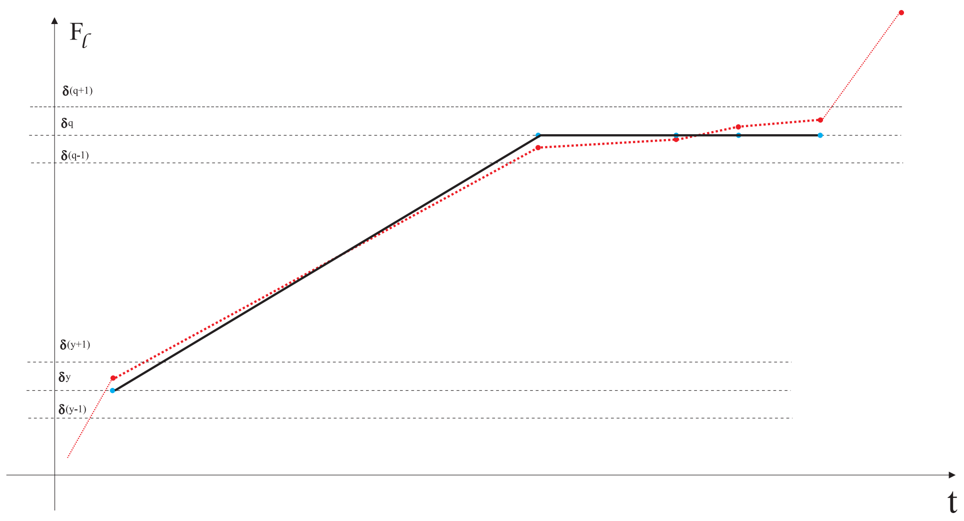

Figure 2: Substitution of columns and modification of $F_l(t)$

Lemma 2 *For all* $t \in (t_l^{k_1-1}, t_l^{k_2-1}]$*, we have* $|F_l'(t) - \overline{F}_l(t)| < \delta/2$.

Proof. It is only necessary to consider the values $|F_l'(t) - \overline{F}_l(t)|$ in the break points $t_l^{k_1-1}, t_l^{k_1}, \ldots, t_l^{k_2-1}$. In these points, the inequality holds and these points are end points of continuous and piecewise linear segments of the functions $\overline{F}_l(t)$ and $F_l'(t)$. $\square$

Assume that functions $F_l(t)$, $l = 1, 2, \ldots, n$, are exact and constructed by the original graphical algorithm. Similarly, $F_l'(t)$, $l = 1, 2, \ldots, n$, are approximated functions constructed by the modified algorithm (the functions after step 3.6. of the modified algorithm).

Lemma 3 *For each* l*,* $l = 1, 2, \ldots, n$*, for all* $t \in (-\infty, +\infty)$*, we have* $|F_l'(t) - F_l(t)| \leq l \cdot \delta/2$.

Proof. The proof is accomplished by induction. The inequality holds for $l = 1$. Assume that it holds for $l-1$, $1 < l < n$, i.e., for $t \in (-\infty, +\infty)$ we have $|F_{l-1}'(t) - F_{l-1}(t)| \leq (l-1)\delta/2$. Let functions $\Phi^1(t)$ and $\Phi^2(t)$ be obtained from function $F_{l-1}(t)$ and functions $\phi^1(t)$ and $\phi^2(t)$ be obtained from function $F_{l-1}'(t)$. Then we have $|\phi^1(t) - \Phi^1(t)| \leq (l-1)\delta/2$ and $|\phi^2(t) - \Phi^2(t)| \leq (l-1)\delta/2$. Thus,

$$|F_l'(t) - F_l(t)| \leq |(\min\{\phi^1(t), \phi^2(t)\} + \delta/2) - \min\{\Phi^1(t), \Phi^2(t)\}| \leq l \cdot \delta/2.$$

So, the Lemma is true. $\square$

An approximate job sequence π' can be restored by backward recursion from the tables saved in Step 3.5. of the modified graphical algorithm as follows. In the table of positions of the job n, we determine the column $(t_n^{k_n}, t_n^{k_n+1}]$, where $t_n^{k_n} < 0 \leq t_n^{k_n+1}$. Then we have an optimal sequence $position_n^{k_n+1} = 1$, job n is the first job in the schedule and for the position of the job $n-1$, we search in the column $(t_{n-1}^{k_{n-1}}, t_n^{k_{n-1}+1}]$ in the table of positions of job $n-1$, where $t_{n-1}^{k_{n-1}} < p_n \leq t_{n-1}^{k_{n-1}+1}$. Otherwise, job n is the last job and for the position of the job $n-1$, we search in the column $(t_{n-1}^{k_{n-1}}, t_{n-1}^{k_{n-1}+1}]$, where $t_{n-1}^{k_{n-1}} < 0 \leq t_{n-1}^{k_{n-1}+1}$. By continuing with such an operation, we restore the job sequence π' in time $O(n \log(n/\varepsilon))$. Let π^* be an optimal job sequence for a chosen straddling job x.

Lemma 4 *Inequality $F(\pi') - F(\pi^*) \leq n \cdot \delta \leq n\frac{2\varepsilon \cdot F(\pi^*)}{2n}$ holds.*

Proof. According to Lemma 2, we prove that $|F(\pi') - F'_n(0)| \leq n \cdot \delta/2$. Furthermore, according to Lemma 3, we have $|F'_n(0) - F_n(0)| \leq n \cdot \delta/2$. From both statements we establish that the lemma is true. $\square$

So, we can conclude the following.

Theorem 3 *By using the modified graphical algorithm for each x, $x \in N$, a job sequence π' of the type described in Lemma 1 will be found in $O(\frac{n^3}{\varepsilon})$ time, where $F(\pi') \leq (1+\varepsilon)F(\pi^*)$.*

The running time $O(\frac{n^3}{\varepsilon})$ is received as follows. The modified graphical algorithm is used n times for each job x, $x \in N$. The running time of the modified graphical algorithm depends on n and the number of columns in the tables which describe the functions $F'(t)$. The number of columns does not exceed $O(\frac{n}{\varepsilon})$.

5 Algorithms for the Special Cases $B-1$, $B-1G$ and for the Problems $1||\sum GT_j$ and $1(no\text{-}idle)||\max\sum w_jT_j$

In this section, modifications of DPA, GrA and FPTAS for two special cases of the problem $1||\sum T_j$ and for the problems $1||\sum GT_j$ and $1(no\text{-}idle)||\max\sum w_jT_j$ are presented.

Lemma 5 *There exists an optimal job sequence π for the special case $B-1G$ that can be represented as a concatenation (G, x, 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 values p_j (longest processing time order, LPT) and all jobs from the set H are processed in non-decreasing order of values p_j (shortest processing time order, SPT).*

The job x is called straddling.

Proof. Let in an optimal job sequence $\pi' = (\pi_1, l, k, \pi_2)$, job k be the first tardy job. Then all jobs from the partial sequence π_2 are tardy and processed in SPT order, since $d_{max} - d_{min} < p_{min}$ (which can be easily proved by contradiction). If $S_k > d_{min}$, then all jobs in the partial sequence (k, π_2) have to be processed in SPT order and the jobs from π_1 can be processed in any order (LPT as well). In this case, l is the straddling job. If $S_k \le d_{min}$, then all jobs in the partial sequence (π_1, l) can be reordered by the LPT rule without loss of optimality. In this case, k is the straddling job. $\square$

Lemma 6 *[19] There exists an optimal job sequence π for the special case $B-1$ 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 LPT order and all jobs from the set H are processed in SPT order.*

Assume that for the case $B-1G$, the jobs are numbered as follows: $p_1 \ge p_2 \ge \dots \ge p_n$.

Lemma 7 *For the special cases $B-1$ and $B-1G$ and a job sequence $\pi_{SPT} = (n, n-1, \dots, 1)$, the following inequality holds: $F(\pi_{SPT}) \le 3F(\pi^*)$.*

Proof. Consider a modified instance, where all due dates are equal to d_{max}. Let $\overline{F}(\pi)$ be the total tardiness value for the modified instance and $F(\pi)$ be the value for the original one. Let in the job sequence $\pi_{SPT} = (n, n-1, \dots, k, \dots, 1)$, job k be the first tardy job for the original instance and $k \ge 2$ (the case where $k = 1$ is trivial). It is obvious that the job sequence π_{SPT} is optimal for the modified instance. The following inequality holds: $\overline{F}(\pi_{SPT}) \le F(\pi^*) \le F(\pi_{SPT})$. Furthermore, $\overline{F}(\pi_{SPT}) \ge F(\pi_{SPT}) - k \cdot p_{min}$ and $\overline{F}(\pi_{SPT}) \ge (k-1) \cdot p_{min}$. Thus, $3\overline{F}(\pi_{SPT}) \ge F(\pi_{SPT})$ and the lemma is true. $\square$

Without loss of generality, we will consider only cases where in a job sequence π_{SPT} at least two jobs are tardy.

Lemma 8 *[17] There exists an optimal job sequence π for the problem $1||\sum GT_j$ that can be represented as a concatenation (G, H), where all jobs $j \in H$ are tardy and $GT_j(\pi) = p_j$. For all jobs $i \in G$, we have $0 \le GT_i(\pi) < p_i$. All jobs from the set G are processed in EDD (early due date) order and all jobs from the set H are processed in LDD (last due date) order.*

Let for the problem $1||\sum GT_j$ the jobs be numbered as follows: $d_1 \le d_2 \le \dots \le d_n$ and let $\pi_{EDD} = (1, 2, \dots, n)$. Denote by T^* the maximal tardiness of a job in the sequence π_{EDD}, i.e., $T^* = \max_{j\in N}\{GT_j(\pi_{EDD})\}$. We remind that $GT_j(\pi) = \min\{\max\{0, C_j(\pi) - d_j\}, p_j\}$ and $F(\pi) = \sum_{j=1}^{n} GT_j(\pi)$.

Lemma 9 *For the problem $1||\sum GT_j$, the following inequality holds: $F(\pi_{EDD}) \leq nF(\pi^*)$.*

Proof. It is easy to show that $F(\pi^*) \geq T^*$. Furthermore, it is obvious that $nT^* \geq F(\pi_{EDD})$. So, the lemma is true. □

Lemma 10 *[16, 18] There exists an optimal job sequence π for the problem $1(no\text{-}idle)||\max\sum w_jT_j$ 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 $\frac{w_j}{p_j}$.*

For the problem $1(no\text{-}idle)||\max\sum w_jT_j$, the value $LB_{maxTT} = \max_{j\in N}(w_j(\sum_{i=1}^{n} p_i - d_j))$ is a lower bound. Then $UB_{maxTT} = nLB_{maxTT}$ is an upper bound on the optimal objective function value.

To solve these problems, Algorithm 1, GrA and FPTAS can be modified as follows.

For the special case $B - 1G$.

- **DPA.** Use the fact described in Lemma 5. In Algorithm 1, we enumerate the jobs according to the order $p_1 \leq p_2 \leq \cdots \leq p_n$. Assume that $F_1(t) := \max\{0, p_1+t-d_1\}$, $\Phi^1(t) := \max\{0, p_l+t-d_l\}+F_{l-1}(t+p_l)$ and $\Phi^2(t) := F_{l-1}(t)+\max\left\{0, \sum\limits_{j=1}^{l} p_j + t - d_l\right\}$. All other lines of the algorithm remain the same. Remember that $w_j = 1$ for all $j \in N$ for the special case of the problem $1||\sum T_j$. The running time of the modified Algorithm 1 is $O(nd_{max})$. Since it is necessary to consider n straddling jobs $x \in N$, an optimal job sequence can be found in $O(n^2 d_{max})$ time by using the modified Algorithm 1;
- **GrA.** The graphical algorithm remains the same. The parameters u_l^k denote the number of tardy jobs which is equal to the total weight of the tardy jobs, since $w_j = 1$ for all $j \in N$. In addition, assume that $UB = F(\pi_{SPT})$. By using the GrA, an optimal schedule can be found in $O(n^2 \min\{d_{max}, F^*\})$ time;
- **FPTAS.** In the FPTAS, assume that $\delta = \frac{\varepsilon F(\pi_{SPT})}{3n}$. Since the GrA is without changes, the time complexity of FPTAS based on the GrA for the special case $B-1G$ has a running time of $O(n^3/\varepsilon)$, which is less than the running time $O(n^7/\varepsilon)$ of the FPTAS for the general case presented in [9].

For the special case $B-1$.

- **DPA.** Use the fact described in Lemma 6. Algorithm 1 is modified as for the special case $B-1G$. The running time of the modified Algorithm 1 is $O(nd_{max})$. Since there is no straddling job, an optimal job sequence can be found in $O(nd_{max})$ time by using the modified Algorithm 1 only once;
- **GrA.** The graphical algorithm remains the same as for the special case $B-1G$. By the GrA, an optimal schedule can be found in $O(n\min\{d_{max},F^*\})$ time;
- **FPTAS.** The FPTAS remains the same as for the special case $B-1G$. Since there is no straddling job, the FPTAS for the special case $B-1$ has a running time of $O(n^2/\varepsilon)$, which is less than the running time of $O(n^3\log n+n^3/\varepsilon)$ of the FPTAS mentioned in [7].

For the problem $1||\sum GT_j$.

- **DPA.** Use the fact described in Lemma 8. In Algorithm 1, we enumerate the jobs according to the order $d_1\geq d_2\geq\cdots\geq d_n$. Assume $F_1(t):=\min\{p_1,\max\{0,p_1+t-d_1\}\}$, $\Phi^1(t):=\min\{p_l,\max\{0,p_l+t-d_l\}\}+F_{l-1}(t+p_l)$ and $\Phi^2(t):=F_{l-1}(t)+\min\left\{p_l,\max\left\{0,\sum\limits_{j=1}^{l}p_j+t-d_l\right\}\right\}$. The running time of the modified Algorithm 1 is $O(nd_{max})$. Since there is no straddling job, an optimal job sequence can be found in $O(nd_{max})$ time by using the modified Algorithm 1 only once;
- **GrA.** The graphical algorithm remains almost the same. In addition to the break points t' and t'' in Steps 3.1 and 3.2, two new break points $\tau'=d_l$ and $\tau''=d_l-\sum\limits_{j=1}^{l-1}p_j$ are considered. The slope u_l^k of the function $F_l(t)$ is changed according to the function $\min\{p_l,\max\{0,p_l+t-d_l\}\}$. By the GrA, an optimal schedule can be found in $O(n\min\{d_{max},nF^*\})$ time, since $UB=F(\pi_{EDD})\leq nF(\pi^*)$ and there are at most $2UB+2$ columns in each table $F_l(t)$ considered in the GrA;
- **FPTAS.** In the FPTAS, we assume $\delta=\frac{\varepsilon F(\pi_{EDD})}{n^2}$. So, the FPTAS has a running time of $O(n^3/\varepsilon)$.

For the problem $1(no\text{-}idle)||\max\sum w_jT_j$.

- **DPA.** We use the fact described in Lemma 10. In Algorithm 1, we enumerate the jobs according to the order $\frac{w_1}{p_1}\leq\frac{w_2}{p_2}\leq\cdots\leq\frac{w_n}{p_n}$. We assume that $F_1(t):=w_1\max\{0,p_1+t-d_1\}$, $\Phi^1(t):=w_l\max\{0,p_l+t-d_l\}+F_{l-1}(t+p_l)$ and $\Phi^2(t):=F_{l-1}(t)+$

$w_l \max \left\{0, \sum_{i=1}^{l} p_i + t - d_l\right\}$. Since total tardiness is maximized, we have $F_l(t) := \max\{\Phi^1(t), \Phi^2(t)\}$. The running time of the modified Algorithm 1 is $O(nd_{max})$. Since there is no straddling job, an optimal job sequence can be found in $O(nd_{max})$ time by using the modified Algorithm 1 only once;

- **GrA.** The graphical algorithm remains the same as for the problem $1|d_j = d|\sum w_jT_j$. In Step 3.3., we have $F_l(t) = \max\{\Phi^1(t), \Phi^2(t)\}$. In [16, 18], it is shown that the functions $F_l(t)$ represent continuous, piecewise-linear and convex functions. By the GrA, an optimal schedule can be found in $O(n \min\{d_{max}, nF^*, \sum w_j\})$ time.
- **FPTAS.** In the FPTAS, assume $\delta = \frac{\varepsilon UB_{maxTT}}{n^2}$. So, the FPTAS has a running time of $O(n^3/\varepsilon)$.

6 Further remarks

Some other graphical algorithms for other single machine problems are presented in [16]. For some of these problems, similar FPTAS can be presented. If for a problem there exists a graphical algorithm with a running time of $O(n^\alpha UB)$, then it is easy to construct a similar FPTAS with a time complexity of $O(n^\gamma + \frac{n^{\alpha+\beta}}{\varepsilon})$, where UB is an upper bound which is no more than $O(n^\beta)$ times greater than the optimal objective function value, computed in $O(n^\gamma)$ time, and α, β, γ are constants. Thus, the running time of such an FPTAS depends on the relative error of the upper bound found, i.e., on β. For the special cases $B-1$, $B-1G$ and for the problem $1|d_j = d|\sum w_jT_j$, we have $\beta = 0$, but for the rest of the problems, we have $\beta = 1$, since $UB/LB = n$.

In the papers [20, 21], a technique is proposed to improve the complexities of approximation algorithms for optimization problems. The technique can be described as follows. If there is an FPTAS for a problem with a running time bounded by a polynomial $P(L, \frac{1}{\varepsilon}, \frac{UB}{LB})$, where L is the problem instance length and UB, LB are known upper and lower bounds, and the value $\frac{UB}{LB}$ is not bounded by a constant, then the technique enables us to find in time $P(L, \log\log\frac{UB}{LB})$ such values UB_0 and LB_0 that $LB_0 \leq F^* \leq UB_0 < 3LB_0$, i.e., $\frac{UB_0}{LB_0}$ is bounded by the constant 3. By using such values UB_0 and LB_0, the running time of the FPTAS will be reduced to $P(L, \frac{1}{\varepsilon})$, where P is the same polynomial.

If we use this technique for an FPTAS for the problems $1(\textit{no-idle})||\max\sum w_jT_j$ and $1||\sum GT_j$, we have a running time of $O(n^2 \log\log n + \frac{n^2}{\varepsilon})$ for the modified FPTAS.

We note that the modified Algorithm 1 (see the end of Section 1) can be considered as a base for an FPTAS as well. The running time of the modification is restricted by UB under the assumption that there is only one interval with the same objective function value $F_l(t)$ although in the proposed FPTAS, there are several intervals with the same objective function value. So, is seems to be more difficult to transform the modification into an FPTAS. The running time of such a modification will be increased by the factor $O(\log \frac{n}{\varepsilon})$ in comparison with the FPTAS based on the GrA.

In the following table, a comparison of Algorithm 1 (classical DPA), GrA and the alternative DPA is presented. Denote $\Theta_l = \{1^{x_1}p_1 + 1^{x_2}p_2 + \dots + 1^{x_l}p_l | x_1, x_2, \dots, x_l \in \{0,1\}\}$. Let $t_n^{UB} \in (-\infty, +\infty)$ be a value, where we have $F_n(t_n^{LB}) = LB$.

Table 9: Comparison of the classical, alternative DPA and the GrA for the problem $1|d_j = d|\sum w_jT_j$

Note	Classical DPA	GrA	Alternative DPA
Can it solve instances with $p_j \notin Z$ and instances with large values p_j	no	yes	yes
states t considered	all $t \in [0,d] \bigcap Z$	only t, where the slope of the function $F_l(t)$ is changed	only t from the set Θ_l
The running time for the initial instance - of the problem $1\|\|\sum GT_j$ is - of the problem $1(no\text{-}idle)\|\|\max \sum w_jT_j$ is	$O(n\min\{d,UB\})$ $O(nd_{max})$ $O(n\min\{d_{max},UB\})$	$O(n\min\{d,UB\})$ $O(n\min\{d_{max},UB\})$ $O(n\min\{d_{max},UB,\sum w_j\})$	$O(n\min\{d,UB\})$ $O(n\min\{d_{max},UB\})$ $O(n\min\{d_{max},UB\})$
It finds all optimal schedules for all starting times $t \in [0,d]$ in time	$O(nd)$	$O(nd)$	-
If finds all optimal schedules for all starting times $t \in (-\infty, UB]$ in time	$O(nUB)$	$O(nUB)$	-
It finds all optimal schedules for all starting times $t \in (-\infty, +\infty)$ in time	$O(nF(\pi', d))$ (see Section 2 for the definition of $F(\pi', d)$	$O(nF(\pi', d))$	-
The running time of the FPTAS is	$O(\frac{n^3}{\varepsilon}\log\frac{n}{\varepsilon}))$	$O(n^3/\varepsilon)^*$	$O(n^3/\varepsilon)^{**}$

* In this time, for all $t \in (-\infty, t_n^{UB}]$ solutions can be found with an absolute error restricted by εLB. For all $t \in [t_n^{LB}, t_n^{UB}]$, $t_n^{LB} \le 0 \le$

t_n^{UB}, solutions can be found with a relative error restricted by ε.
** An approximate solution is only found for the starting time $t = 0$.

In the following table, some GrA and FPTAS are summarized.

Table 10: Time complexity of the GrA and FPTAS based on the GrA

Problem	Time complexity of GrA	Time complexity of FPTAS	Time complexity of classical DPA
$1\|\|\sum w_jU_j$	$O(\min\{2^n, n \cdot \min\{d_{max}, F_{opt}\}\})$ [16]	-	$O(nd_{max})$
$1\|d_j = d'_j + A\|\sum U_j$	$O(n^2)$ [16]	-	$O(n\sum p_j)$
$1\|\|\sum GT_j$	$O(\min\{2^n, n \cdot \{d_{max}, nF^*\}\})$	$O(n^2 \log\log n + \frac{n^2}{\varepsilon})$	$O(nd_{max})$
$1\|\|\sum T_j$ special case $B-1$	$O(\min\{2^n, n \cdot \min\{d_{max}, F^*\}\})$	$O(n^2/\varepsilon)$	$O(nd_{max})$
$1\|\|\sum T_j$ special case $B-1G$	$O(\min\{n^2 \cdot \min\{d_{max}, F^*\}\})$	$O(n^3/\varepsilon)$	$O(n^2d_{max})$
$1\|d_j = d\|\sum w_jT_j$	$O(\min\{n^2 \cdot \min\{d, F^*\}\})$	$O(n^3/\varepsilon)$	$O(n^2d_{max})$
$1(no\text{-}idle)\|\|\max\sum w_jT_j$	$O(\min\{2^n, n \cdot \min\{d_{max}, nF^*, \sum w_j\}\})$ [16]	$O(n^2 \log\log n + \frac{n^2}{\varepsilon})$	$O(nd_{max})$
$1(no\text{-}idle)\|\|\max\sum T_j$	$O(n^2)$ [15]	-	$O(nd_{max})$

In [17], an experimental analysis of the running time of the graphical algorithms for two NP-hard single machine problems was presented. According to the experimental results for a significant part of the instances considered the number of columns (i.e., the running time) in the graphical algorithms does not exceed $O(n^2)$. We can conclude that for such instances the modified graphical algorithm will find exact solutions. So, the modified graphical algorithms remain exact for a significant part of the instances.

7 Conclusion

In this paper, an FPTAS was presented, which can be used with some simple modifications for several single machine problems. The FPTAS is based on a graphical approach from [15, 16]. The idea of such a modification of graphical algorithms enables us to construct an FPTAS easily.

The graphical approach can be applied to problems, where a pseudo-polynomial algorithm exists and Boolean variables are used in the sense that yes/no decisions have to be made (e.g., in the scheduling problems under consideration, a job may be completed on-time or not or for a knapsack problem, an item can be put into the knapsack or not). For the knapsack problem, the graphical algorithm often reduces substantially the number of points to be considered but the time complexity of the algorithm remains pseudo-polynomial. However, for the single machine problem of maximizing total tardiness, the graphical algorithm improved the complexity from $O(n \sum p_j)$ to $O(n^2)$. Thus, the graphical approach has not only a practical but also a theoretical importance.

Acknowledgement

Partially supported by RFBR (Russian Foundation for Basic Research): 11-08-01321, 11-08-13121. The authors are also grateful to Prof. V. Strusevich for his idea to use the GrA as a base for an FPTAS and to Chris Yukna for his help regarding the English presentation.

References

[1] 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

[2] Du J., Leung J. Y.-T. (1990). Minimizing Total Tardiness on One Processor is NP-hard, Math. Oper. Res., 15, 483 – 495.

[3] Lazarev A.A., Gafarov E.R. (2006), Special Case of the Single-Machine Total Tardiness Problem is NP-hard. Journal of Computer and Systems Sciences International, 45(3), 450 – 458.

[4] Lawler E.L. (1977). A Pseudopolynomial Algorithm for Sequencing Jobs to Minimize Total Tardiness. Ann. Discrete Math., 1, 331 – 342.

[5] Szwarc W., Della Croce F., Grosso A. (1999). Solution of the Single Machine Total Tardiness Problem. Journal of Scheduling, 2, 55 – 71.

[6] Potts C.N., Van Wassenhove L.N. (1982). A Decomposition Algorithm for the Single Machine Total Tardiness Problem. Oper. Res. Lett., 1, 363 – 377.

[7] Koulamas C. (2010). The single-machine total tardiness scheduling problem: Review and extensions, European Journal of Operational Research, 202, 1–7.

[8] Hoogeveen H., T'Kindt V. (2010) Minimizing the number of late jobs when the starting time of the machine is variable. Proceedings PMS 2010, 235–238.

[9] Lawler E.L. (1982). A Fully Polynomial Approximation Scheme for the Total Tardiness Problem, Oper. Res. Lett., 1, 207 – 208.

[10] Lazarev A.A., Kvaratskheliya A.G., Gafarov E.R. (2007). Algorithms for Solving the NP-Hard Problem of Minimizing Total Tardiness for a Single Machine, Doklady Mathematics, 75(1), 130-134.

[11] Yuan J. (1992). The NP-hardness of the Single Machine Common Due Date Weighted Tardiness Problem, System Sci. Math. Sci., 5, 328-333.

[12] Fathi Y., Nuttle H.W.L. (1990). Heuristics for the Common Due Date Weighted Tardiness Problem, IEE Trans., 22, 215-225.

[13] Kellerer H., Strusevich V.A. (2006). A Fully Polynomial Approximation Scheme for the Single Machine Weighted Total Tardiness Problem with a Common Due Date, Theoretical Computer Science, 369, 230-238.

[14] Kacem I. (2010), Fully polynomial time approximation scheme for the total weighted tardiness minimization with a common due date, Discrete Applied Mathematics, 158, 1030-1040.

[15] Gafarov E.R., Lazarev A.A. and Werner F. (2012). Transforming a pseudo-polynomial algorithm for the single machine total tardiness maximization problem into a polynomial one. Annals of Operations Research, DOI 10.1007/s10479-011-1055-4, 2012, in print.

[16] Gafarov E.R. , Lazarev A.A. and Werner F. (2010). A Modification of Dynamic Programming Algorithms to Reduce the Time Complexity. Preprint 20/10, FMA, OvGU Magdeburg, 2010, 24 pages.

[17] Gafarov E.R., Lazarev A.A. and Werner. F. (2012). A note on a single machine scheduling problem with generalized total tardiness objective function. Information Processing Letters, 112 (3), 72 – 76.

[18] Gafarov E.R. , Lazarev A.A. and Werner. F. (2010) Classical Combinatorial and Single Machine Scheduling Problems with Opposite Optimality Criteria. Preprint 11/10, FMA, OvGU Magdeburg, 15 pages.

[19] Lazarev A.A. (2007). Solution of the NP-hard total tardiness minimization problem in scheduling theory. Computational Mathematics and Mathematical Physics, 47, 1039–1049.

[20] Chubanov S., Kovalyov M.Y., Pesch E. (2006). An FPTAS for a single-item capacitated economic lot-sizing problem with monotone cost structure, Math. Program., Ser. A 106, 453–466.

[21] Kovalyov M.Y. (1995). Improving the complexities of approximation algorithms for optimization problems, Operations Research Letters, 17, 85-87