

A Graphical Approach for Solving Single Machine Scheduling Problems Approximately^{*}

Evgeny R. Gafarov^{*} Alexandre Dolgui^{**}
Alexander A. Lazarev^{*} Frank Werner^{***}

^{*} *Institute of Control Sciences of the Russian Academy of Sciences, Profsoyuznaya st. 65, 117997 Moscow, Russia (e-mail: axel73@mail.ru, lazarev@ipu.ru).*

^{**} *Ecole Nationale Supérieure des Mines, FAYOL-EMSE, CNRS:UMR6158, LIMOS, F-42023 Saint-Etienne, France (e-mail: dolgui@emse.fr)*

^{***} *Fakultät für Mathematik, Otto-von-Guericke-Universität Magdeburg, PSF 4120, 39016 Magdeburg, Germany (e-mail: frank.werner@mathematik.uni-magdeburg.de).*

Abstract: For five single machine total tardiness problems a fully polynomial-time approximation scheme (FPTAS) based on a graphical algorithm is presented. The FPTAS has the best running time among the known approximation schemes for these problems.

Keywords: Scheduling algorithms, Dynamic programming, FPTAS

1. INTRODUCTION

We consider several single machine total tardiness problems which can be formulated as follows. We are given a set $N = \{1, 2, \dots, n\}$ 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 $\pi = (j_1, j_2, \dots, j_n)$ 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 $C_j(\pi) > d_j$, 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 the objective function $F(\pi) = \sum_{j=1}^n T_j(\pi)$ and for the total late work problem the objective function $F(\pi) = \sum_{j=1}^n GT_j(\pi)$ have to be minimized. The following special cases of the problems are also considered:

- minimizing weighted total tardiness when all due dates are equal, i.e., $d_j = d$, $j = 1, 2, \dots, n$. This problem 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 $p_1 \geq p_2 \geq \dots \geq p_n$, $d_1 \leq d_2 \leq \dots \leq d_n$ and $d_n - d_1 \leq p_n$;
- the special case $B - 1G$ of the problem $1|\sum T_j$ with $d_{max} - d_{min} \leq p_{min}$, 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 $1(no-idle)|\max \sum w_j T_j$, 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. Such problems with the *maximum* criterion have practical interpretations and applications, but their investigation is also a theoretically significant task.

All the problems and special cases mentioned above are NP-hard in the ordinary sense (Gafarov et al. (2012)). For the special case $B - 1$, an FPTAS with a running time of $O(n^3 \log n + n^3/\epsilon)$ is known, see Koulamas (2010). An FPTAS for the total weighted late work problem with the same running time was presented in Kovalyov et al. (1994). An FPTAS for the problem $1|d_j = d|\sum w_j T_j$ with a running time of $O((n^6 \log \sum w_j)/\epsilon^3)$ was presented in Kellerer et al. (2006). These problems can be considered as particular cases when 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,

^{*} 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.

when the starting time of the machine is variable, was considered in Hoogeveen et al. (2012). 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 $\{\pi_1\} \cup \{\pi_3\} = N_1$ and $\{\pi_2\} = N_2$. The jobs from N_2 have to be scheduled according to one of the functions mentioned above. The graphical and approximation algorithms presented in this paper can be used both for the initial problems and for the 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 $1/\varepsilon$.

For a practical realization of some pseudo-polynomial algorithms with Boolean variables (e.g. a job can be on-time or tardy) one can use a modification called a *graphical algorithm* (GrA) which we present for the problem $1|d_j = d|\sum w_j T_j$ as well as an FPTAS based on this GrA with a running time of $O(n^3/\varepsilon)$. Modifications of these GrA and FPTAS for the cases $B - 1$, $B - 1G$ and the problems $1||\sum GT_j$ and $1(no-idle)||\max \sum w_j T_j$ are also presented.

2. DYNAMIC PROGRAMMING FOR THE PROBLEM $1|D_J = D|\sum W_J T_J$

Lemma 1. There exists an optimal job sequence π for 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 $S_j \geq d$ for all $j \in H$, and all jobs $i \in G$ are on-time. All jobs from set G are processed in non-increasing order of the values $\frac{p_j}{w_j}$ and all jobs from set H are processed in non-decreasing order of the 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: $\frac{p_2}{w_2} \leq \frac{p_3}{w_3} \leq \dots \leq \frac{p_n}{w_n}$, (*) where the job with number 1 is the straddling job.

As a corollary from Lemma 1, there is a straddling job $x \in N$, to which the number 1 will be assigned, such that for each $l \in \{1, 2, \dots, n\}$, there exists an optimal schedule in which all jobs $j \in \{1, 2, \dots, l\}$ are processed from time t one by one, and there is no job $i \in \{l+1, l+2, \dots, n\}$ which is processed between these jobs. Thus, we can present a dynamic programming algorithm (DPA) based on Lemma 1. For each $x \in N$, we number the jobs from the set $N \setminus \{x\}$ according to (*) and perform Algorithm 1. Then we choose a best 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 $\{1, 2, \dots, l\}$

and for each possible starting time t of the first job (which represents a possible state in the DPA). $F_l(t)$ 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. Number the jobs according to order (*);
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
 $\pi^1 := (l, \pi_{l-1}(t + p_l)), \pi^2 := (\pi_{l-1}(t), l);$
 $\Phi^1(t) := w_l \max\{0, p_l + t - d\} + F_{l-1}(t + p_l);$
 $\Phi^2(t) := F_{l-1}(t) + w_l \max\{0, \sum_{j=1}^l p_j + t - d\};$
 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 2. By using Algorithm 1 for each $x \in N$, an optimal job sequence of the type described in Lemma 1 can be found in $O(n^2 \sum p_j)$ time.

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

Algorithm 1 can be modified by considering for each $l = 1, 2, \dots, n$, only the interval $[0, d - p_l]$ instead of the interval $[0, \sum_{i=l+1}^n p_i]$ 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 an optimal schedule can be found in $O(n^2 d)$ time.

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, 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 $1, 2, \dots, l$ and $F_l(t)$ denotes the corresponding value of the monotonic objective function). 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 need to 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 for all $j \in N$ and d are integer, then there are at most $UB + 2$ different values $F_l(t)$.

In addition, if there is a point $t' \in (-\infty, +\infty)$ such that $F_l(t') = 0$ and $F_l(t' + 1) > 0$, then $F_l(t) = 0$ for all $t \leq t'$ and $F_l(t) < F_l(t+1)$ for all $t \geq t' + 1$, since all the functions $F_l(t)$ are monotonic. Thus, we can modify Algorithm 1 as follows. If we will save at each stage l instead of all states $t \in [0, t']$ only one state t' , then the number of saved states will be restricted by UB , since for all saved states t we have $F_l(t) < F_l(t + 1)$. The running time of

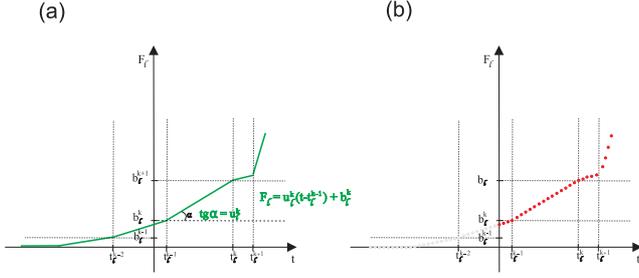


Fig. 1. Function $F_l(t)$ in the GrA and in Algorithm 1

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 $l, l = 1, 2, \dots, n$, 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 using 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 $F_l(t) < F_l(t + 1)$ 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 $1 || \sum GT_j$.

3. GRA FOR PROBLEM $1 | D_j = D | \sum W_j T_j$

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 $F_l(t) = F_l^k(t) = u_l^k \cdot (t - t_l^{k-1}) + b_l^k$. $F_l(t)$ is a continuous piecewise linear function whose parameters can be described in a tabular form.

Namely, 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 total weight of tardy jobs) in a tabular form as given in Table 1.

Table 1: Function $F_l(t)$

k	1	2	...	$m_l + 1$	$m_l + 2$
int.	$(-\infty, t_l^1]$	$(t_l^1, t_l^2]$...	$(t_l^{m_l}, t_l^{m_l+1}]$	$(t_l^{m_l+1}, \infty)$
b_l^k	0	b_l^2	...	$b_l^{m_l+1}$	∞
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 k th 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 k th 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 $1, 2, \dots, l$ 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(a)). We have $F_l(t) = 0$ 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., $t \in (-\infty, +\infty)$. This table describes a function $F_l(t)$ which is a continuous, piecewise-linear function in the interval $(-\infty, t_l^{UB}]$. The points $t_l^1, t_l^2, \dots, t_l^{m_l+1}$ 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 $b_l^1 = b_l^2 < b_l^3 < \dots < b_l^{m_l+1} < UB$ holds, since $t_l^1 < t_l^2 < \dots < t_l^{m_l+1}$.

In the GrA, the functions $F_l(t)$ reflect the same functional equations as in Algorithm 1, i.e., for each $t \in Z \cap [0, \sum_{j=2}^n p_j]$, 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.

The function $\Phi^1(t)$ is obtained from $F_{l-1}(t)$ 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 $F_{l-1}(t)$ 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 $s = m_{l-1} + 1$, we have $(t_{l-1}^{UB} - p_l, t']$ and $(t', +\infty)$). 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 $u_{l-1}^{s+1}, u_{l-1}^{s+2}, \dots, u_{l-1}^{m_{l-1}+1}$ 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.

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 $t_{l-1}^h < t'' < t_{l-1}^{h+1}$, $h \leq m_{l-1}$, 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 $(t_{l-1}^{UB}, t'']$ and $(t'', +\infty)$). 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. We construct a table that corresponds to the function $F_l(t) = \min\{\Phi^1(t), \Phi^2(t)\}$ as follows. We consider functions $\Phi^1(t)$ and $\Phi^2(t)$ on all resulting intervals from both tables and search for intersection points of the diagrams of these functions.

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 $(t_{z-1}, t_z]$ from the table for $\Phi^1(t)$ and in $(t_{y-1}, t_y]$ 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 $(t_i, t_{i+1}]$ 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.

Let m be the number of columns in the resulting table of $F_l(t)$ and for k , $1 \leq k < m_l$, the inequality $b_l^k < UB \leq b_l^{k+1}$ holds. From the equality $UB = (t_l^{UB} - t_l^{k-1})u_l^k + b_l^k$, we compute the value t_l^{UB} . In the column with the 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 than $UB + 2$ columns if all the parameters of the problem are integer.

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 the chosen job x and the optimal function value $F(\pi^*) = b_n^{k+1} + (0 - t_n^k) \cdot u_n^{k+1}$.

Theorem 3. By using the GrA for each $x \in N$, an optimal job sequence of the type described in Lemma 1 can be found in $O(n^2 F^*)$ time, where F^* is the optimal objective function value.

4. ADVANTAGES OF GRA FOR $1|D_J = D|\sum W_J T_J$

In each step $l = 1, 2, \dots, n$ 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 starting from time 0, but from time t . If the 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 $\Psi(t) + F(\pi, t)$ can be made in shorter time.

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

1. GrA can solve instances, where (some of) the parameters p_j , w_j , $j = 1, 2, \dots, n$ 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.
4. Unlike DPA, it is possible to construct a FPTAS based on GrA easily which is presented in Section 5.

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 $\sum_{j=1}^n p_j$. This algorithm can be described as follows:

Alternative DPA

1. Number the jobs according to order (*);
2. Put the state $(0, 0)$ into the set of states V_{n+1} .
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 $t \leq d$ and $F \leq UB$. 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 the GrA (note that the GrA can be easily modified to consider only points $t \in [0, d]$). However, in the GrA some of the possible but unpromising states are not considered and, in contrast to the alternative DPA, the GrA finds all optimal schedules for all $t \in [-\infty, t_n^{UB}]$ in $O(n F^*)$ time. So, the alternative DPA is not effective for the problems of minimizing a complex function $\Psi(t) + F(\pi, t)$.

5. AN FPTAS FOR 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 GrA, we have to diminish the number of columns considered, which is the number of different objective function values $0 = b_l^1 = b_l^2, b_l^3, \dots, b_l^{m_{l+1}} = UB$. If we consider not the original values b_l^k but the values \bar{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 $|F(t) - F'(t)| < \delta \leq \frac{\varepsilon F(\pi^*)}{n}$. If we do the rounding and modification after each stage of the GrA, then the cumulative error will be no more than $n\delta \leq \varepsilon F(\pi^*)$, and the total running time of n runs of the GrA will be $O(\frac{n^3}{\varepsilon})$, i.e., an FPTAS is obtained.

By transforming the GrA, we save the approximated functions $F_l^A(t)$ in the same tabular form but without the last row describing an optimal partial job sequence π_l^k .

FPTAS (as a modification of the GrA).

Assume that we have obtained a table for a function $F_l(t)$ after stage l with the objective function values $b_l^1, b_l^2, \dots, b_l^{m_l+1}$. If $m_l > 4\frac{n}{\varepsilon}$, then do the following. Round all the values b_l^k from Table 1 to the nearest multiple of δ . Let the values $\overline{b}_l^1, \overline{b}_l^2, \dots, \overline{b}_l^{m_l+1}$ be obtained, where $\overline{b}_l^1 \leq \overline{b}_l^2 \leq \dots \leq \overline{b}_l^{m_l+1}$. We modify the table $F_l^A(t)$ as follows. Assume that, for $k_1 < k_2$, we have $\overline{b}_l^{k_1} < \overline{b}_l^{k_1+1} = \dots = \overline{b}_l^{k_2} < \overline{b}_l^{k_2+1}$. We substitute the columns which correspond to the values $\overline{b}_l^{k_1}, \dots, \overline{b}_l^{k_2-1}$ for the two columns presented in Table 2.

Table 2: Substitution of columns

int. k	...	$(t_l^{k_1-1}, t_l^{k_1}]$	$(t_l^{k_1}, t_l^{k_2-1}]$...	(t_l^{UB}, ∞)
b_l^k	...	$b_l^{k_1}$	$b_l^{k_2}$...	∞
u_l^k	...	$u = \frac{b_l^{k_2} - b_l^{k_1}}{t_l^{k_1} - t_l^{k_1-1}}$	0	...	0

In fact, in the modification we substitute some linear fragments which follow one by one and are close (in terms of the absolute error which is $\leq \delta/2$) to the same line by this line. Let $F_l'(t)$ be the function obtained at stage l in the modified algorithm before rounding and let $F_l^A(t)$ describe the rounded function.

Lemma 4. For all $t \in (t_l^{k_1-1}, t_l^{k_2-1}]$, we have $|F_l^A(t) - F_l'(t)| < \delta/2$.

Before the next stage $l+1$, we assign $F_l'(t) := F_l^A(t)$. Assume that functions $F_l(t)$, $l = 1, 2, \dots, n$, are exact and constructed by the original GrA. Similarly, $F_l^A(t)$, $l = 1, 2, \dots, n$, are approximated functions constructed by the modified algorithm.

Lemma 5. For each l , $l = 1, 2, \dots, n$, and all $t \in (-\infty, +\infty)$, we have $|F_l^A(t) - F_l(t)| \leq l \cdot \delta/2$.

Let π^* be an optimal job sequence for the chosen straddling job x .

Lemma 6. Inequality $F(\pi') - F(\pi^*) \leq n \cdot \delta \leq n \frac{2\varepsilon \cdot F(\pi^*)}{2n}$ holds.

Theorem 7. By using the modified GrA for each $x \in N$, 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 obtained as follows. The modified GrA is used n times for each job $x \in N$. The running time of the modified GrA depends on n and the

number of columns in the tables which describe functions $F_l^A(t)$. The number of columns does not exceed $O(\frac{n}{\varepsilon})$.

6. ALGORITHMS FOR THREE SPECIAL CASES

Lemma 8. 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 set G are processed in non-increasing order of the values p_j (LPT order) and all jobs from set H are processed in non-decreasing order of the values p_j (SPT order).

The job x is called straddling.

Lemma 9. (Gafarov et al. (2012)) There exists an optimal job sequence π for the special case $B - 1$ that can be represented as a concatenation (G, H) . 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: $p_1 \geq p_2 \geq \dots \geq p_n$.

Lemma 10. For the special cases $B - 1$ and $B - 1G$ and a job sequence $\pi_{SPT} = (n, n - 1, \dots, 1)$, inequality $F(\pi_{SPT}) \leq 3F(\pi^*)$ holds.

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

Lemma 11. (Gafarov et al. (2012)) 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 $0 \leq GT_i(\pi) < p_i$. All jobs from the set G are processed in EDD (earliest 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: $d_1 \leq d_2 \leq \dots \leq d_n$ and let $\pi_{EDD} = (1, 2, \dots, n)$. Denote by T^* the maximal tardiness of a job in the sequence π_{EDD} , i.e., $T^* = \max_{j \in N} \{GT_j(\pi_{EDD})\}$.

Lemma 12. For problem $1||\sum GT_j$, the following inequality holds: $F(\pi_{EDD}) \leq nF(\pi^*)$.

Lemma 13. (Gafarov et al. (2012)) There exists an optimal job sequence π for the problem $1(no-idle)||\max \sum w_j T_j$ 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 $\frac{w_j}{p_j}$.

For problem $1(no-idle)||\max \sum w_j T_j$, 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, Algorithms 1, GrA and FPTAS can be modified as follows.

For the special case $B - 1G$:

- **DPA.** Use the fact described in Lemma 8. In Algorithm 1, we number the jobs according to the order $p_2 \leq p_3 \leq \dots \leq p_n$. 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 $\Phi^2(t) := F_{l-1}(t) + \max\{0, \sum_{j=1}^l p_j + t -$

$d_l\}$. All other steps of the algorithm remain the same. Remember that $w_j = 1$ for all $j \in N$ for 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 GrA remains the same. The parameters u_i^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 the 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 by Lawler.

For the special case $B - 1$:

- **DPA.** Use the fact described in Lemma 9. 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 GrA 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 by Koulamas.

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

- **DPA.** Use the fact described in Lemma 11. In Algorithm 1, we number the jobs according to the order $d_1 \geq d_2 \geq \dots \geq d_n$. Assume that $F_1(t) := \min\{p_1, \max\{0, p_1 + t - d_1\}\}$, $\Phi^1(t) := \min\{p_l, \max\{0, p_l + t - d_l\}\} + F_{l-1}(t + p_l)$ and $\Phi^2(t) := F_{l-1}(t) + \min\{p_l, \max\{0, \sum_{j=1}^l p_j + t - d_l\}\}$. 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 GrA remains almost the same. In addition to the break points t' and t'' , two new break points $\tau' = d_l$ and $\tau'' = d_l - \sum_{j=1}^{l-1} p_j$ are considered.

The slope u_i^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.** Here, we assume that $\delta = \frac{\varepsilon F(\pi_{EDD})}{n^2}$. So, the FPTAS has a running time of $O(n^3/\varepsilon)$.

For the problem $1(no-idle)||\max \sum w_j T_j$:

- **DPA.** We use the fact described in Lemma 13. In Algorithm 1, we enumerate the jobs according to the order $\frac{w_1}{p_1} \leq \frac{w_2}{p_2} \leq \dots \leq \frac{w_n}{p_n}$. 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 GrA remains the same as for the problem $1|d_j = d||\sum w_j T_j$. We have $F_l(t) = \max\{\Phi^1(t), \Phi^2(t)\}$. It is known (Gafarov et al. (2012)) 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 that $\delta = \frac{\varepsilon UB_{max} TT}{n^2}$. So, the FPTAS has a running time of $O(n^3/\varepsilon)$.

In Gafarov et al. (2012), GrA for 8 scheduling problems and FPTAS for 6 scheduling problems are presented.

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. 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., a job may be completed on-time or not). 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.

REFERENCES

- E.R. Gafarov, A. Dolgui and F. Werner. Dynamic Programming Approach to Design FPTAS for Single Machine Scheduling Problems. *CNRS Report*, 26 pp. (2012).
- C. Koulamas. The Single-Machine Total Tardiness Scheduling Problem: Review and Extensions. *European Journal of Operational Research*, 202, 1 – 7 (2010).
- M. Y. Kovalyov, C. N. Potts and L. N. van Wassenhove. A Fully Polynomial Approximation Scheme for Scheduling a Single Machine to Minimize Total Weighted Late Work. *Mathematics of Operations Research*, Vol. 19, No. 1, 86–93 (1994).
- H. Hoogeveen and V. T'Kindt. Minimizing the Number of Late Jobs When the Starting Time of the Machine is Variable. *Proceedings PMS*, 235 – 238 (2010).
- H. Kellerer and V.A. Strusevich. A Fully Polynomial Approximation Scheme for the Single Machine Weighted Total Tardiness Problem with a Common Due Date. *Theoretical Computer Science*, 369, 230 – 238 (2006).