Control and Cybernetics

vol. **38** (2009) No. 3

A new optimal algorithm for a time-dependent scheduling problem*

by

Marek Kubale and Krzysztof M. Ocetkiewicz

Department of Algorithms and System Modeling
Gdańsk University of Technology, Gdańsk
ul. Gabriela Narutowicza 11/12, 80-233 Gdańsk Wrzeszcz, Poland
{kubale, krzysztof.ocetkiewicz}@eti.pg.gda.pl

Abstract: In this article a single machine time-dependent scheduling problem with total completion time criterion is considered. There are n given jobs, $j_1, \ldots, j_n$, and the processing time p_i of the i-th job is given by $p_i = 1 + b_i s_i$, where s_i is the starting time of the i-th job, $i = 1, \ldots n$. If all jobs have different and non-zero deterioration rates and $b_i > b_j \Rightarrow b_i \geq \frac{b_{\min}+1}{b_{\min}} b_j + \frac{1}{b_{\min}}$, where $b_{\min} = \min\{b_i\}$, then an optimal schedule can be found in $O(n \log n)$ time. The conducted computational experiments show that the presented algorithm performs very well even on data not satisfying the assumed constraints.

Keywords: scheduling, single machine, deteriorating jobs, total completion time, algorithms.

1. Introduction

Scheduling time-dependent jobs has received increasing attention in recent years. In such a scheduling problem, the time needed to complete a given job is not constant, but depends on the job starting time. There are two types of functions describing such a change of processing time. The first type is related to non-decreasing functions (deteriorating jobs) and the second type to non-increasing ones. Numerous applications of this kind of scheduling include metallurgy, fire-fighting, problems of military activities, modeling of financial operations, maintenance or cleaning assignments. Most published results concern a single machine and the makespan or the total completion time criteria. There are also known results for identical and dedicated parallel machines. A comprehensive survey of known results can be found in Cheng et al. (2004) and Gawiejnowicz (2008).

*Submitted: April 2008; Accepted: June 2009.

Given a set of n jobs $j_1, \ldots, j_n$ with processing times $p_i = a + b_i s_i$, where $a > 0$, $b_i > 0$, and $s_i \geq 0$ denote job starting time, for $i = 1, \ldots, n$, our aim is to find a schedule of length as small as possible or a schedule with total completion time as small as possible. These two scheduling problems can be described by the three-field notation scheme $\alpha|\beta|\gamma$ (Graham et al., 1979), as follows: $1|p_i = a + b_i s_i|C_{\max}$ and $1|p_i = a + b_i s_i|\sum C_i$. The first problem represents a criterion advantageous to the processor. This is so because $C_{\max}$ reflects the effort required to complete all tasks (manpower, processor time, fuel consumption etc.). The second one realizes the "interest" of tasks. Why? This is so because the total completion time reflects the mean time, taken over all jobs, having elapsed from the moment the job becomes available for execution to its completion. For example, in terms of dealing with forest fires, this means the average time, during wich a fire damages the environment. These two objectives are often in conflict, as shown in the following example. Suppose that we have $n = 4$ fires to contain with processing times $p_i = 1 + b_i s_i$, where $b_1 = 0$ and $b_i = 1$ for $i = 2, 3, 4$. In Fig. 1(a) we show a Gantt chart of the optimal schedule with $C_{\max} = 8$ and $\sum C_i = 19$, while in Fig. 1(b) we show a Gantt chart of the optimal schedule with $C_{\max} = 9$ and $\sum C_i = 17$.

The situation when all jobs have only deterioration rate (i.e. the processing time of the i-th job can be expressed as $p_i = b_i s_i$) is among the easiest. Any order of tasks gives the same value of the schedule length, because its value is equal to the product of all deterioration rates increased by one and the t_0 – the moment at which the first job starts being processed (it should be greater than zero, because otherwise all jobs would require no time to complete). All schedules without idle periods are optimal (Mosheiov, 1994). When the base processing time is present, but is proportional to the job deterioration rate (i.e. when $p_i = b_i + r b_i s_i$, and r is common to all jobs), the problem remains trivial (Bachman and Janiak, 1997). All schedules have the same, and therefore optimal, length.

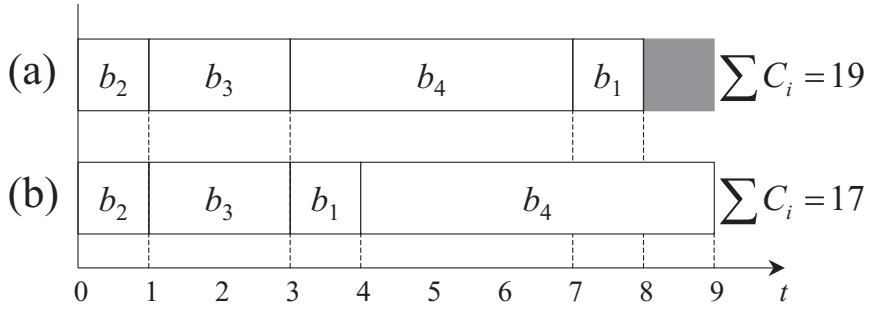

Figure 1: Minimizing $C_{\max}$ does not always result in a schedule with the smallest $\sum C_i$

In general, every job has its own base processing time and deterioration rate, i.e. $p_i = a_i + b_i s_i$. The shortest job sequence can be found by ordering all jobs in nondecreasing order of $\frac{a_i}{b_i}$. This is quite plausible — jobs with short base processing time and large deterioration rate should be processed before the ones that require longer time to complete but deteriorate slower. If there are jobs that have no base processing time ($a_i = 0$), they should be processed before all others (in which case they will take no time, since their execution begins at moment $s_i = 0$). Jobs that do not deteriorate (their deterioration rate is equal to zero) should be processed last, as they may be postponed without any penalty (Gawiejnowicz and Pankowska, 1995). Thus, all we need is sorting the remaining jobs according to $\frac{a_i}{b_i}$, which can be done in $O(n \log n)$ time.

The $\sum C_i$ criterion is more difficult to consider. Our goal is to find a sequence of jobs that gives the smallest possible sum of completion times of all jobs. This criterion is equivalent to minimizing the average completion time of all jobs.

Even when jobs do not have the base processing time, the problem is no longer trivial. However, the optimal solution can be found quickly by means of sorting — the nonincreasing order of deterioration rates gives the best possible schedule (Mosheiov, 1994).

Similarly, in the presence of the base processing time proportional to the job deterioration rate (i.e. when $p_i = b_i + rb_i s_i$, and r is common to all jobs), the problem can also be solved by ordering jobs. The optimal solution has the deterioration rates arranged with respect to nondecreasing order of b_i (Bachman and Janiak, 1997).

The case with common deterioration rate and individual base processing times also can be solved by means of sorting — the best schedule has its jobs ordered in nondecreasing order of base processing times (Cheng and Ding, 2000).

In the paper we focus on the problem with common base processing time and individual deterioration rates ($p_i = a + b_i s_i$), where $a > 0$, $b_i > 0$ Some interesting properties of an optimal schedule are known. One of the most important is that the optimal schedule is V-shaped with respect to deterioration rates (Mosheiov, 1991), i.e.

- the job with the largest deterioration rate is placed first,
- the job with the smallest deterioration rate is scheduled neither second nor last,
- jobs occurring before the job with the smallest b_i are arranged in nonincreasing order of b_i,
- jobs occurring after the job with the smallest b_i are arranged in nondecreasing order of b_i.

The V-shape property of an optimal schedule allows us to significantly reduce the number of possibly optimal schedules.

Another interesting fact is that reversing the order of all jobs except for the first, in any schedule, results in a solution whose total completion time is the same as that of the original schedule (Mosheiov, 1991). As a consequence, there are always at least two optimal solutions.

Only the presence of common base processing time is important. Its value only acts as a scale factor of the goal function, therefore it is often assumed to be 1.

A number of approximation algorithms for solving this problem have been proposed (e.g. Gawiejnowicz et al., 2002, and Mosheiov, 1991), but none of them guarantees that the returned solution is close to the optimal one. It was also shown (Gawiejnowicz, Lai and Chiang, 2000), that if there is only a fixed number of different deterioration rates, this problem can be solved in polynomial time by enumerating all possible schedules and choosing the one with the smallest value of the goal function.

2. The optimal algorithm

Consider the following algorithm. ($[\ldots]$ denotes a list, and $[\ldots]+[\ldots]$ is the operation of list concatenation):

Algorithm P:

Step 1. Order the jobs in such a way that $b_i < b_{i+1}$, for $i = 1, \ldots, n-1$.
Step 2. Set $P = (1 + b_{n-1})$, $\pi_b = [b_n,\ b_{n-1}]$, $R = 0$, $\pi_e = [\,]$.
Step 3. For $i = n-2$ down to 2 do:

Step 3*a*. If $P > R$, put b_i at the beginning of π_e and set $R = (R+1)(b_i+1)$

Step 3*b*. Otherwise, put b_i at the end of π_b and set $P = (P+1)(b_i+1)$.

Step 4. Return $\pi_b + [b_1] + \pi_e$.

THEOREM 1 *If:*
(1) $b_{\min} > 0$,
(2) all b_i *are different,*
(3) $b_i > b_j \Rightarrow b_i \geq \frac{b_{\min}+1}{b_{\min}} b_j + \frac{1}{b_{\min}}$,
where $b_{\min}$ *is the value of the smallest deterioration rate of all jobs, the algorithm* P *solves such instance of data optimally in* $O(n \log n)$ *time.*

Proof. For convenience, let B_i represent the i-th job deterioration rate increased by one, i.e. $B_i = 1 + b_i$ for $i = 1, \ldots, n$. The completion time of this job in a schedule π can be determined by using the following formula: $C_{\pi(i)} = 1 + \sum_{j=2}^{i} \prod_{k=j}^{i} B_{\pi(k)}$ (where $\pi(i)$ is the index of the i-th job in the schedule π), and the sum of completion times of jobs $1, \ldots, n$ is equal to $\sum_{i=1}^{n} C_i = n + \sum_{i=2}^{n} \sum_{j=2}^{i} \prod_{k=j}^{i} B_{\pi(k)}$. When we begin execution at the moment T instead of 0, the total completion time changes to

$$\sum_{i=1}^{n} C_{\pi(i)}(T) = n + \sum_{i=p}^{q} \sum_{j=p}^{i} \prod_{k=j}^{i} B_{\pi(k)} + (T-1) \sum_{i=p}^{q} \prod_{k=p}^{i} B_{\pi(k)}.$$

Therefore, the contribution of a sequence of jobs $j_{\pi(p)}, \ldots, j_{\pi(q)}$ to the total completion time of a schedule is described by a function of the completion time

of the job scheduled before $j_{\pi(p)}$, namely:

$$\sum_{i=p}^{q} C_{\pi(i)} = S_{p,q} + V_{p,q}(C_{\pi(p-1)} - 1), \tag{1}$$

where $S_{p,q} = (|q-p|+1)+\sum_{i=p}^{q}\sum_{j=p}^{i}\prod_{k=j}^{i} B_{\pi(k)}$ and $V_{p,q} = \sum_{i=p}^{q}\prod_{k=p}^{i} B_{\pi(k)}$.

Consider two schedules differing only in the order of a set of consecutive jobs $j_p, \ldots, j_q$, where $p < q$. Such schedules are presented in Fig. 2. The difference between the total completion times of these two schedules can be seen in Fig. 3. Because both these schedules have the same sequences of jobs before the p-th and after the q-th position, the following equalities occur: $S_{1,p-1} = S'_{1,p-1}$, $C_{p-1} = C'_{p-1}$, $S_{q+1,n} = S'_{q+1,n}$ and $V_{q+1,n} = V'_{q+1,n}$. Moreover, $S_{p,q} = S'_{q,p}$, i.e. both these values are the sums of the same products, although they are added in different order. If we consider sequence $j_p, \ldots, j_q$, whose processing starts at the time C_{p-1}, and notice that $C'_p - C_q = V_{p,q} - V'_{q,p}$, then we obtain:

$$TC(\pi') - TC(\pi) = (R_q - P_p)((Y_{pq}+1)(b_{\pi(p)}+1) - (X_{pq}+1)(b_{\pi(q)}+1)) \tag{2}$$

where

$$P_p = \sum_{j=2}^{p-1}\prod_{k=j}^{p-1} B_{\pi(k)} \qquad R_q = \sum_{j=q+1}^{n}\prod_{k=q+1}^{j} B_{\pi(k)}$$

$$X_{pq} = \sum_{j=p+1}^{q-1}\prod_{k=j}^{q-1} B_{\pi(k)} \qquad Y_{pq} = \sum_{j=p+1}^{q-1}\prod_{k=p+1}^{j} B_{\pi(k)}.$$

Note that the first factor of the right-hand side of equation (2) (i.e. $R_q - P_p$) depends only on the jobs at positions $1, \ldots, p-1$ and $q+1, \ldots, n$, while the second one depends on the jobs at positions $p, \ldots, q$. Moreover, since X_{pq} and Y_{pq} are calculated using the same set of jobs, the difference between them can be estimated as follows.

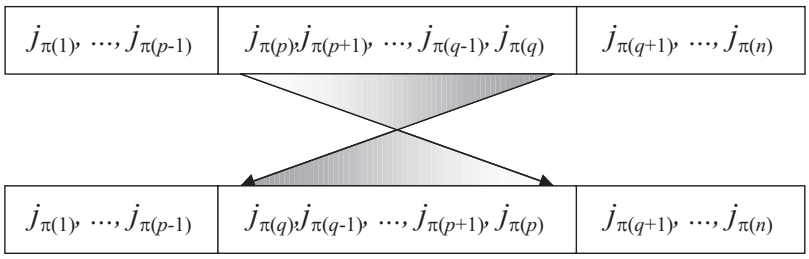

Figure 2: Reversing a set of jobs $j_p, \ldots, j_q$

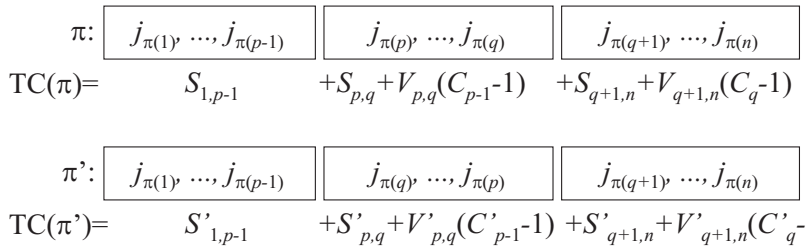

Figure 3: Total completion times of consecutive parts of schedules before (π) and after reversing a set of consecutive jobs (π')

Consider a set of jobs $j_{p+1}, \ldots, j_{q-1}$ and a positive value K that is smaller than the smallest deterioration rate of all jobs, i.e. $b_{\pi(i)} \geq K$ ($i = p+1, \ldots, q-1$). Both X_{pq} and Y_{pq} are greater than the product of deterioration rates increased by one of all the jobs in the considered set, which is one out of $(q-p-1)$ elements of the sums X_{pq} and Y_{pq}. Also, both these sums can be bounded from above — neither of them is greater than the above product multiplied by $\frac{K+1}{K}$. Therefore, the difference between those sums is not greater than the difference between their upper and lower bounds. That means that $|X_{pq} - Y_{pq}| \leq \frac{1}{K} \min(X_{pq}, Y_{pq})$

Now, assume that all b_i are different and that if $b_i > b_j$ then $b_i \geq \frac{K+1}{K} b_j + \frac{1}{K}$. Consider a schedule $\pi = \{j_1, \ldots, j_n\}$ and a pair of jobs j_p and j_q, such that $b_p > b_q$. If job j_p is scheduled sometime before j_q, reversing the sequence of jobs $j_p, \ldots, j_q$ will result in the following value of the second factor of equation (2)

$$(Y_{pq}+1)(b_p+1)-(X_{pq}+1)(b_q+1) \geq (b_q+1)\left(-\frac{1}{K}Y_{pq} + \frac{1}{K}Y_{pq} + \frac{1}{K}\right) > 0. \quad (3)$$

Similarly, if job j_q is scheduled before j_p, we have

$$(Y_{pq}+1)(b_q+1) - (X_{pq}+1)(b_p+1) < 0.$$

The sign of the expression $(Y_{pq}+1)(b_p+1) - (X_{pq}+1)(b_q+1)$ depends only on the position of $\max\{b_p, b_q\}$: if the larger one is next to $(Y_{pq}+1)$ — the value of the expression is positive, if it is next to $(X_{pq}+1)$ — it is negative.

As a consequence, knowing the first part (i.e. $j_1, \ldots, j_{p-1}$) and the last part (i.e. $j_{q+1}, \ldots, j_n$) of the optimal schedule, one can decide where to put the job j_i with the largest deterioration rate among the remaining jobs:

- (i) if $P_p > R_q$ then the job j_i should stand just before j_{q+1} (otherwise it would be scheduled just after j_{p-1} and a schedule with the reversed sequence of jobs $j_p, \ldots, j_q$ would have a smaller total completion time),
- (ii) if $P_p < R_q$ then j_i should stand just after j_{p-1},
- (iii) in the case $P_p = R_q$, j_i can be placed either before j_{q+1} or after j_{p-1} — there are two optimal schedules, having a different ordering of jobs $j_p, \ldots, j_q$.

Having a starting point, one can repeatedly assign the remaining jobs, using the above rules to construct an optimal schedule.

As mentioned before, the job with the largest deterioration rate must be scheduled first. Because the optimal schedule is V-shaped, a job with the second largest deterioration rate has to be placed either second or last. Since the order of all jobs except the first one can be reversed without changing the total completion time, there are always at least two optimal solutions, one of them having the job with the second largest deterioration rate scheduled last, and the other having this job scheduled second. This fact gives us freedom to choose the starting point for the construction of an optimal schedule. We can begin with either an initial chunk consisting of two jobs with the largest deterioration rates and an empty final chunk, or an initial chunk containing the job with the largest value of b_i and a final chunk containing the job with the second largest value of b_i.

The most time-consuming part of this algorithm is the initial ordering of jobs in *Step* 1, requiring $O(n \log n)$ time. The loop in *Step* 3 repeatedly inserts jobs into the schedule, either by appending to an already constructed initial part of a schedule, or inserting it before the last part of a schedule, using rules (i)–(iii). This takes $\Theta(n)$ iterations, each of them consuming $O(1)$ time, giving the total of $\Theta(n)$. Both the initialization of variables in *Step* 2 and the construction of the result in *Step* 4 require $O(1)$ time. Therefore, the complexity of the whole algorithm P is $O(n \log n)$. ∎

It should be noted that the greater the value of K, the more easily the presented requirements are met. Therefore, the greatest value of K that we can use is the value of the smallest deterioration rate of all jobs.

The presented conditions guarantee that the algorithm finds an optimal solution, but this does not mean that the returned schedule cannot be optimal if they are not satisfied. Satisfying the condition $b_{i+1} \geq \frac{b_{\min}+i}{b_{\min}+1} b_i + \frac{i}{b_{\min}+1}$, which is sometimes more easily met than the above conditions, also results in an optimal schedule. The proof of this fact is based on the observation that $|X_{pq} - Y_{pq}| \leq \frac{b_{\min}+q-p-1}{b_{\min}+1} \prod_{j=p+1}^{q-1}(b_j + 1)$, and the arguments presented in this paper.

3. Computational experiments

If none of the presented conditions is satisfied, the presented algorithm can still be used as an approximation algorithm, and it still appears to perform quite well. Table 1 shows a comparison of computational results of different heuristics. $H1$ and $H2$ are heuristics presented in Mosheiov (1991) and algorithm G comes from Gawiejnowicz, Kurc and Pankowska (2002), while our algorithm is denoted by P. Optimal solutions, whose values are presented in the OPT column, were found using an exhaustive-search algorithm. In the last four columns, ΔA denotes the average difference between the optimal solution and the value returned by

Table 1. Results of computational experiments with algorithms $H1$, $H2$, G and P

α	n	OPT	$\Delta H1$	$\Delta H2$	ΔG	ΔP
1.0	10	$2.075 \cdot 10^{02}$	$3.02 \cdot 10^{-01}$	$4.32 \cdot 10^{00}$	$2.43 \cdot 10^{-02}$	$1.05 \cdot 10^{-02}$
1.0	15	$1.546 \cdot 10^{03}$	$2.37 \cdot 10^{00}$	$2.70 \cdot 10^{01}$	$5.80 \cdot 10^{-02}$	$4.66 \cdot 10^{-02}$
1.0	20	$6.794 \cdot 10^{03}$	$7.78 \cdot 10^{00}$	$1.02 \cdot 10^{02}$	$1.53 \cdot 10^{-01}$	$9.30 \cdot 10^{-02}$
1.0	25	$7.631 \cdot 10^{04}$	$4.02 \cdot 10^{01}$	$6.69 \cdot 10^{02}$	$9.14 \cdot 10^{-01}$	$7.73 \cdot 10^{-01}$
1.0	30	$4.320 \cdot 10^{05}$	$1.56 \cdot 10^{02}$	$3.27 \cdot 10^{03}$	$3.42 \cdot 10^{00}$	$1.17 \cdot 10^{00}$
10.0	10	$5.899 \cdot 10^{06}$	$1.14 \cdot 10^{03}$	$5.54 \cdot 10^{04}$	$1.55 \cdot 10^{01}$	$1.05 \cdot 10^{01}$
10.0	15	$2.169 \cdot 10^{10}$	$6.54 \cdot 10^{05}$	$1.55 \cdot 10^{08}$	$3.07 \cdot 10^{04}$	$1.93 \cdot 10^{03}$
10.0	20	$3.205 \cdot 10^{14}$	$6.51 \cdot 10^{09}$	$1.12 \cdot 10^{12}$	$1.28 \cdot 10^{08}$	$2.14 \cdot 10^{06}$
10.0	25	$6.810 \cdot 10^{18}$	$3.81 \cdot 10^{13}$	$1.95 \cdot 10^{16}$	$2.98 \cdot 10^{11}$	$8.54 \cdot 10^{10}$
10.0	30	$6.207 \cdot 10^{21}$	$3.19 \cdot 10^{16}$	$1.40 \cdot 10^{19}$	$1.61 \cdot 10^{15}$	$4.50 \cdot 10^{10}$

algorithm A, for $A = H1, H2, G, P$. All results are averaged over 20 iterations, n denotes the number of jobs in a problem instance and deterioration rates were chosen randomly in the range $\langle 0, \alpha \rangle$. The value of the base processing time was equal to 1.

As Table 1 shows, schedules returned by P have their total completion time closest to the optimum. The difference increases as the size of the problem (n) or the values of deterioration rates tend to infinity.

4. Conclusions

In this paper we have studied a single processor time-dependent scheduling problem of minimizing the total completion time of a set of jobs having a common base processing time and deteriorating at a linear rates. We have presented a polynomial algorithm for a certain set of instances. The algorithm efficiency on other kinds of input data has been studied as well. Future research may focus on expanding the polynomially solvable set of instances by developing tighter bounds for expressions used in the proof. The presented results can also be helpful in limiting the solution space in a branch-and-bound algorithm for the given problem.

References

Bachman, A. and Janiak, A. (1997) Scheduling jobs with special type of start time dependent processing times. Wroclaw University of Technology, Report PRE 34/97.

Cheng, T.C.E. and Ding, Q. (2000) Single machine scheduling with deadlines and increasing rates of processing times. *Acta Informatica* **36** (9-10), 673–692.

Cheng, T.C.E., Ding, Q. and Lin, B.M.T. (2004) A concise survey of scheduling with time-dependent processing times *European Journal of Operational Research* **152**, 1–13.

Gawiejnowicz, S. and Pankowska, L. (1995) Scheduling jobs with varying processing times. *Information Processing Letters* **54**, 175–178.

Gawiejnowicz, S., Lai, T.-C. and Chiang, M.-H. (2000) Polynomially solvable cases of scheduling deteriorating jobs to minimize total completion time. In: *Extended Abstracts of the 7-th Workshop on Project and Management Scheduling*, 131–134.

Gawiejnowicz, S., Kurc, W. and Pankowska, L. (2002) A greedy approach for a time-dependent scheduling problem. **LNCS 2328**, Springer 79–86.

Gawiejnowicz, S. (2008) *Time-Dependent Scheduling.* Springer-Verlag, Berlin.

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

Mosheiov, G. (1991) V-shaped policies for scheduling deteriorating jobs. *Operations Research*, **39** (6), 979–991.

Mosheiov, G. (1994) Scheduling jobs under simple linear deterioration. *Computers and Operations Research* **21** (6), 653–659.