INFORMATICA, 2006, Vol. 17, No. 1, 13–24

Job Sequencing with Exponential Functions of Processing Times

Adam JANIAK
Institute of Computer Engineering, Control and Robotics, Wroclaw University of Technology
Janiszewskiego 11/17, 50-372 Wroclaw, Poland
e-mail: adam.janiak@pwr.wroc.pl

Mikhail Y. KOVALYOV
Faculty of Economics, Belarus State University, and
United Institute of Informatics Problems, National Academy of Sciences of Belarus
Skorini 4, 220050 Minsk, Belarus
e-mail: kovalyovmy@bsu.by

Received: March 2005

Abstract. We study single machine scheduling problems, where processing times of the jobs are exponential functions of their start times. For increasing functions, we prove strong NP-hardness of the makespan minimization problem with arbitrary job release times. For decreasing functions, maximum lateness minimization problem is proved to be strongly NP-hard and total weighted completion time minimization problem is proved to be ordinary NP-hard. Heuristic algorithms are presented and computationally tested for these problems.

Key words: scheduling, single machine, start time dependent processing times, computational complexity, heuristics.

1. Introduction

We study single machine scheduling problems, in which job processing times are functions of their start times. If these functions are non-decreasing, then the jobs are called *deteriorating* in the literature, see for example Browne and Yechiali (Browne and Yechiali, 1990). In manufacturing processes, deteriorating jobs correspond to perishable products. Several practical applications for scheduling with start time dependent job processing times were described in the literature. Mosheiov (Mosheiov, 1994) gave an example where an operator sequentially serves customers at some stations and the number of customers at each station is increasing with time as well as their total serving time. Ho, Leung and Wei (Ho *et al.*, 1993) presented a military application, where a task consists of destroying an aerial threat and its execution time decreases with time, as the threat gets closer. Kunnathur and Gupta (Kunnathur and Gupta, 1990) described the problem of scheduling fire fighting tasks. They assumed that the time and effort required to control a fire increases if there is a delay in the start of the fire fighting operation.

Alidaee and Womer (Alidaee and Womer, 1999) provided a recent review of the models, available computational complexity and algorithmic results for scheduling with start time dependent processing times. Most of the dependency functions considered in the literature include linear and piecewise linear functions. However, for many real-life situations, this dependency can be better described by a more complicated function such as a convex or concave function. In this paper, we assume that the dependency functions are exponential. Such functions are adequate to model processing times of tasks related to processes which dynamics is exponentially dependent on time. Examples of such processes are spreading of fire or an epidemic disease, where an effort to localize a part of this process increases exponentially as time goes. Another example is an execution of tasks in an area contaminated with radio-active or chemical materials. Here the task execution time is related to the degradation of the harming materials and it can be described as an exponentially decreasing function of its start time. To our knowledge, scheduling problems with exponential functions of job processing times were not addressed in the literature.

In this paper, we establish computational complexity and present heuristic algorithms for scheduling problems with exponential functions of job processing times. In the next section, we formulate problems considered in this paper. Section 3 contains NP-hardness proofs for the makespan, the maximum lateness and the total weighted completion time minimization problems. In Section 4, heuristic algorithms are presented for these three problems and computational experiments are described. Concluding remarks and suggestions for future research are given Section 5.

2. Formulation of the Problems

Problems studied in this paper can be formulated as follows. There are n independent and non-preemptive jobs to be scheduled for processing on a single machine. Job i is available for processing at its *release time* r_i and may have a *due date* d_i and a *weight* w_i indicating its relative importance. The processing time $p_i(t)$ of job i is an increasing or decreasing exponential function dependent on the starting time t of its execution:

$$p_i(t) = a_i c^{b_i(t-r_i)} \quad \text{or} \quad p_i(t) = a_i c^{-b_i(t-r_i)},$$

where a_i is a *normal* processing time and b_i is an *increasing or decreasing rate* of job i, respectively, and $c > 1$ is the base of the exponential function.

Given a schedule, let C_i denote the completion time of job i. The objective is to find a schedule, for which the value of a cost function depending on job completion times is minimized. In this paper, we consider the following three traditional scheduling cost functions: the makespan $C_{\max} = \max\{C_i\}$, the maximum lateness $L_{\max} = \max\{L_i\} = \max\{C_i - d_i\}$ and the total weighted completion time $\sum w_i C_i$. Here and below we assume that each maximum and summation is taken over all jobs unless it is stated otherwise. All numerical parameters are assumed to be non-negative real numbers.

Adapting the three-field notation $\alpha|\beta|\gamma$ for scheduling problems of Graham *et al.* (Graham *et al.*, 1979), we denote problems studied in this paper as $1|r_i,\ p_i(t) = a_i c^{b_i(t-r_i)}|C_{\max}$, $1|p_i(t) = a_i c^{-b_i t}|L_{\max}$ and $1|p_i(t) = a_i c^{-b_i t}|\sum w_i C_i$.

Notice that for each of the problems considered, a schedule is completely characterized by the job sequence. According to such a sequence, a job i is started immediately after the previous job has been completed if this event happened after time r_i, or at time r_i.

3. NP-hardness Proofs

In this section, we establish strong NP-hardness of problems $1|r_i,\ p_i(t) = a_i 2^{b_i(t-r_i)}|C_{\max}$ and $1|p_i(t) = a_i 2^{-b_i t}|L_{\max}$ and ordinary NP-hardness of problem $1|p_i(t) = a_i 2^{-b_i t}|\sum w_i C_i$. In our proofs, we use reductions from NP-complete problem PARTITION and NP-complete in the strong sense problem 3-PARTITION, see Garey and Johnson (Garey and Johnson, 1979).

PARTITION. Given $m+1$ positive integers $h_1, \ldots, h_m$ and H such that $\sum_{i=1}^{m} h_i = 2H$, is there a set $X \subseteq \{1, \ldots, m\}$ such that $\sum_{i \in X} h_i = H$?

3-PARTITION. Given $3m+1$ positive integers $h_1, \ldots, h_{3m}$ and H such that $\sum_{i=1}^{3m} h_i = mH$ and $H/4 < h_i < H/2$ for $i = 1, \ldots, 3m$, is there a partition of the set $\{1, \ldots, 3m\}$ into m disjoint subsets $X_1, \ldots, X_m$ such that $\sum_{i \in X_j} h_i = H$ for $j = 1, \ldots, m$?

In the proofs given below, it is convenient to assume that there is an artificial job v_0 which is always scheduled at time zero and has zero processing time.

Theorem 1. *Problem $1|r_i, p_i(t) = a_i 2^{b_i(t-r_i)}|C_{\max}$ is NP-hard in the strong sense.*

Proof. Given an instance of 3-PARTITION, we construct the following instance of the scheduling problem. There are $n = 4m$ jobs. Among them there are $3m$ *partition* jobs $1, \ldots, 3m$, and m *enforcer* jobs $v_1, \ldots, v_m$ with the following parameters:

$$\begin{array}{llll} r_i = 0; & a_i = h_i; & b_i = 0; & \text{for } i = 1, \ldots, 3m, \\ r_{v_i} = iH + (i-1); & a_{v_i} = 1; & b_{v_i} = H; & \text{for } i = 1, \ldots, m. \end{array}$$

We show that 3-PARTITION has a solution **if and only if** there exists a solution for the constructed instance of problem $1|r_i, p_i(t) = a_i 2^{b_i(t-r_i)}|C_{\max}$ with makespan value $C_{\max} \leqslant mH + m$.

"Only if". Assume that sets $X_1, \ldots, X_m$ represent a solution of 3-PARTITION. We construct a schedule, in which enforcer job v_i starts at its release time r_{v_i}, and therefore has a unit processing time, $i = 1, \ldots, m$. Partition jobs of the set X_j are scheduled between the jobs v_{j-1} and v_j, $j = 1, \ldots, m$. The makespan value for the constructed schedule is equal to $mH + m$.

"If". Assume now that there exists a schedule with a makespan value not greater than $mH + m$. Observe that for any schedule the total job processing time is equal to $mH + m$ if each enforcer job starts at its release time. If some enforcer job will start at least one

time unit after its release time, then the total job processing time will be at least $mH + m + 2^H > mH + m$. Denote the set of partition jobs scheduled between the jobs v_{j-1} and v_j as X_j, $j = 1, \ldots, m$. We must have $\sum_{i \in X_j} p_i(t) = \sum_{i \in X_j} a_i = \sum_{i \in X_j} h_i \leqslant H$ for $j = 1, \ldots, m$. Then from $\sum_{i=1}^{3m} h_i = mH$ we deduce that $\sum_{i \in X_j} h_i = H$ for $j = 1, \ldots, m$. Thus, 3-PARTITION has a solution.

Theorem 2. *Problem* $1|r_i \in \{0, R\}, p_i(t) = a_i 2^{b_i(t-r_i)}|C_{\max}$ *is NP-hard.*

Proof. A reduction from problem PARTITION similar to the reduction in the previous theorem can be used to prove this theorem.

Theorem 3. *Problem* $1|p_i(t) = a_i 2^{-b_i t}|L_{\max}$ *is NP-hard in the strong sense.*

Proof. Given an instance of 3-PARTITION, we construct the following instance of the scheduling problem. There are $n = 4m$ jobs. Among them there are $3m$ *partition* jobs $1, \ldots, 3m$, and m *enforcer* jobs $v_1, \ldots, v_m$ with the following parameters:

$$\begin{aligned} &d_i = d = mH + \tfrac{m-1}{2}; \quad a_i = h_i; \quad b_i = 0; && \text{for } i = 1, \ldots, 3m, \\ &d_{v_i} = iH + \tfrac{i}{2}; && \text{for } i = 1, \ldots, m. \end{aligned} \qquad a_{v_i} = 1; \quad b_{v_i} = \left[iH + \tfrac{i-1}{2}\right]^{-1};$$

We show that 3-PARTITION has a solution **if and only if** there exists a solution for the constructed instance of problem $1|p_i(t) = a_i 2^{-b_i t}|L_{\max}$ with value $L_{\max} \leqslant 0$.

"Only if". Assume that sets $X_1, \ldots, X_m$ represent a solution to 3-PARTITION. We construct a schedule consisting of m blocks of jobs $B_1, \ldots, B_m$ sequenced in this order. Block B_j contains partition jobs from the set X_j sequenced arbitrarily and followed by the enforcer job v_j. The value of the maximum lateness for the constructed schedule can be calculated as follows. Define $S_1 = \sum_{i \in X_1} h_i$ and $S_j = S_{j-1} + p_{v_{j-1}}(S_{j-1}) + \sum_{i \in X_j} h_i$ for $j = 2, \ldots, m$. Calculate

$$\begin{aligned} L_{\max} &= \max\Big\{ \max_{k=1,\ldots,m} \{L_{v_k}\}, \max_{i=1,\ldots,3m} \{L_i\} \Big\} \\ &= \max\Bigg\{ \max_{k=1,\ldots,m} \Bigg\{ \sum_{j=1}^{k} \sum_{i \in X_j} h_i + a_{v_1} 2^{-b_{v_1} S_1} + \ldots + a_{v_k} 2^{-b_{v_k} S_k} - d_{v_k} \Bigg\}, \\ &\qquad \sum_{j=1}^{m} \sum_{i \in X_j} h_i + a_{v_1} 2^{-b_{v_1} S_1} + \ldots + a_{v_{m-1}} 2^{-b_{v_{m-1}} S_{v_{m-1}}} - d \Bigg\}. \end{aligned}$$

Since $\sum_{i \in X_j} h_i = H$ for $j = 1, \ldots, m$, we have

$$p_{v_j}(S_j) = 1/2, \quad S_1 = H, \quad S_2 = 2H + 1/2, \quad \ldots, \quad S_m = mH + (m-1)/2.$$

Therefore, enforcer job v_j completes at time $C_{v_j} = jH + j/2$ for $j = 1, \ldots, m$. The completion time of the latest partition job is equal to d. Therefore, $L_{\max} = 0$, as required.

"If". Consider a schedule with value $L_{\max} \leqslant 0$. Observe that at least one partition job must be scheduled after $d_{v_{m-1}}$. The minimum L_i value for the latest partition job is achieved when the processing times of the preceding enforcer jobs are minimized, that is, the enforcer jobs start as late as possible. However, each enforcer job v_j must complete by its due date d_{v_j}. If all the enforcer jobs complete at their due dates, then $L_{\max} = 0$. Otherwise, $L_{\max} > 0$. Denote the set of partition jobs scheduled between the jobs v_{j-1} and v_j as X_j, $j = 1, \ldots, m$. We have $\sum_{j \in X_j} h_j = H$ for $j = 1, \ldots, m$.

Theorem 4. *Problem* $1|p_i(t) = a_i 2^{-b_i t}, d_i \in \{d, D\}|L_{\max}$ *is NP-hard.*

Proof. A transformation from problem PARTITION similar to the one in the previous theorem can be used to prove this theorem.

Theorem 5. *Problem* $1|p_i(t) = a_i 2^{-b_i t}|\sum w_i C_i$ *is NP-hard.*

Proof. Given an instance of problem PARTITION, we construct the following instance of the scheduling problem. There are $m+1$ jobs. Parameters of *partition* jobs $1, \ldots, m$ and parameters of a single *enforcer* job v are given as follows

$$\begin{array}{llll} w_i = h_i; & a_i = h_i; & b_i = 0; & \text{for } i = 1, \ldots, m, \\ w_v = H; & a_v = 2H(2\ln 2 + 1)^{-1}; & b_v = H^{-1}. & \end{array}$$

We show that PARTITION has a solution **if and only if** there exists a solution to the constructed instance of the scheduling problem with value

$$\sum w_i C_i \leqslant y := \frac{1}{2}\sum_{i=1}^{m} h_i^2 + H^2\Big(3 + \frac{2}{2\ln 2 + 1}\Big).$$

Denote $M = \{1, \ldots, m\}$. Consider an arbitrary schedule. In this schedule, let X and $M \setminus X$ denote the sets of partition jobs sequenced before and after the enforcer job v, respectively. Let $V = \sum_{j \in X} h_j$ and $U = 2H - V$. Calculate a contribution of the jobs from set X and the enforcer job to the objective function:

$$\sum w_j C_j\big(X \cup \{v\}\big) = \sum_{j \in X} h_j^2 + \sum_{i<j, i,j \in X} h_i h_j + (V + a_v 2^{-b_v V})(w_v + U).$$

Since $2 \cdot \sum_{i<j, i,j \in X} h_i h_j = V^2 - \sum_{j \in X} h_j^2$, we obtain

$$\sum w_j C_j\big(X \cup \{v\}\big) = \frac{1}{2}\sum_{j \in X} h_j^2 + \frac{1}{2}V^2 + (V + a_v 2^{-b_v V})(w_v + U).$$

Similarly, calculate a contribution of the jobs from set $M \setminus X$ to the objective function:

$$\sum w_j C_j(M \setminus X) = \frac{1}{2}\sum_{j \in M \setminus X} h_j^2 + \frac{1}{2}U^2.$$

Let us sum up the contributions of all the jobs and represent it as a function of V :

$$\sum w_jC_j = F(V) = \frac{1}{2}\sum_{j=1}^{m} h_j^2 + 2H^2 + w_vV + a_v2^{-b_vV}(w_v + 2H - V).$$

Let us analyze function $F(V)$. Calculate its derivative

$$F'(V) = w_v - b_va_v2^{-b_vV}\ln 2(w_v + 2H - V) - a_v2^{-b_vV}.$$

The derivative $F'(V)$ is negative for $0 \leqslant V < H$, $F'(V) = 0$ for $V = H$ and it is positive for $H < V \leqslant 2H$. Then we deduce that, for $0 \leqslant V \leqslant 2H$, function $F(V)$ has the only minimum in the point $V = H$ with value

$$F(H) = \frac{1}{2}\sum_{i=1}^{m} h_i^2 + H^2\Big(3 + \frac{2}{2\ln 2 + 1}\Big) = y.$$

"Only if". Assume that set $X \subseteq M$ is a solution to PARTITION, that is, $\sum_{i\in X} h_i = H$. Construct a schedule such that the jobs from set X precede the enforcer job v and the jobs from set $M\backslash X$ follow the enforcer job. From $F(H) = y$, we obtain that for such a schedule $\sum w_jC_j = y$.

"If". Assume that there exists a schedule with value $\sum w_jC_j \leqslant y$. Denote by X the set of partition jobs sequenced before the enforcer job. Since $F(H) = y$ and $F(V) > y$, for $V \neq H$, $V \in \{0, 1, \ldots, 2H\}$, we have $\sum_{\in X} h_i = H$, as required.

Notice that we proved strong NP-hardness of problem $1|r_i, p_i(t) = a_i2^{b_i(t-r_i)}|C_{\max}$ for integer valued job parameters, strong NP-hardness of problem $1|p_i(t) = a_i2^{-b_it}|L_{\max}$ for rational valued job parameters and ordinary NP-hardness of problem $1|p_i(t) = a_i2^{-b_it}|\sum w_iC_i$ for real valued job parameters.

4. Heuristic Algorithms

Since problems $1|r_i, p_i(t) = a_i2^{b_i(t-r_i)}|C_{\max}$, $1|p_i(t) = a_i2^{-b_it}|L_{\max}$ and $1|p_i(t) = a_i2^{-b_it}|\sum w_iC_i$ are NP-hard, we constructed and experimentally compared three heuristic algorithms for each of them.

The algorithms for $C_{\max}$, $L_{\max}$ and $\sum w_jC_j$ minimization problems are denoted, respectively, by Ci, Li, and Si, where $i = 1, 2, 3$. Algorithms C1, L1 and S1 are well known approaches for solving classical versions of the corresponding problems. In algorithm C1, a solution is obtained by scheduling jobs in the non-decreasing order of their release times. In algorithm L1, a solution is obtained by sequencing the jobs in the non-decreasing order of d_j (the *Earliest Due Date* rule of Jackson (Jackson, 1955)). In algorithm S1, a solution is obtained by sequencing the jobs in the non-decreasing order of the ratio p_i/w_i (the *Shortest Weighted Processing Time* rule of Smith (Smith, 1956)).

In algorithm C2, two job subsequences are constructed and a final complete sequence is obtained by their concatenation. From the set of unscheduled jobs, a job with the smallest value of r_j is chosen and it is assigned to the end of the first (earlier) subsequence of jobs. From the set of the remaining jobs, a job with the smallest value of b_j is chosen and it is assigned to the beginning of the second (later) subsequence of jobs. This operation is repeated until there is a job to be assigned. Thus, jobs in the first subsequence appear in the non-decreasing order of r_j and jobs in the second subsequence appear in the non-increasing order of b_j. A short notation for the sequence of jobs constructed by algorithm C2 can be described as C2($r_j \nearrow, b_j \searrow$).

Algorithms C3, L3, S2 and S3 work in the same way, however, using parameters different from those in algorithm C2. The final sequences constructed by C3, L3, S2 and S3 can be described as C3($r_j \nearrow, a_j \nearrow$), L3($b_j \nearrow, d_j \nearrow$), S2($w_j \searrow, b_j \nearrow$) and S3($a_j/w_j \nearrow, b_j \nearrow$), respectively.

Algorithm L2 constructs a solution by sequencing the jobs in the non-decreasing order of the ratio $a_j/(b_j + d_j)$.

The computational complexity of all the algorithms presented above is equal to $\mathrm{O}(n \log n)$.

The quality of solutions delivered by the presented algorithms was tested experimentally. For each problem, 3 different tests with 100 instances were performed for each value of $n \in \{10, 50, 100, 500\}$. Problem parameters were randomly generated according to the uniform distribution.

For the problem of $C_{\max}$ minimization, values of b_j and r_j were generated from intervals $(0, 0.001]$ and $(0, 100]$, respectively. Values of a_j were generated as follows. At first, the jobs were renumbered in the non-decreasing order of r_j. After that, the following three values were calculated: $\Delta_{min} = \min_{1 \leqslant j \leqslant n-1}\{r_{j+1} - r_j\}$, $\Delta_{max} = \max_{1 \leqslant j \leqslant n-1}\{r_{j+1} - r_j\}$ and $\Delta_{avg} = \frac{1}{n-1}\sum_{j=1}^{n-1}(r_{j+1} - r_j)$. Finally, the values of a_j were generated from the intervals $(\Delta_{min}, \Delta_{avg}]$, $(\Delta_{avg}, \Delta_{max}]$ and $(\Delta_{min}, \Delta_{max}]$. These three intervals were used to obtain instances in which optimal solutions in average have large, medium and small number of machine idle times.

For $L_{\max}$ minimization problem, the values of b_j and a_j were generated from intervals $(0, 1]$ and $(0, 10]$, respectively. The values of d_j were generated from intervals $(A/2, A]$, $(A/4, 3A/4]$ and $(0, A]$, where $A = \sum a_j$. These three intervals were used to obtain instances in which optimal solutions in average have small, medium and large number of late jobs.

For $\sum w_j C_j$ minimization problem, the values of b_j and a_j were generated from intervals $(0, 1]$ and $(0, 10]$, respectively. The values of w_j were generated from intervals $(0, 1]$, $(0, 10]$ and $(0, 100]$.

Given a problem instance, let $c1$–$c3$, $l1$–$l3$ and $s1$–$s3$ denote objective function values delivered by algorithms C1–C3, L1–L3 and S1–S3, respectively. In order to avoid non-positive values of the maximum lateness, we calculated $l1$–$l3$ as values of the function $L_{\max} + \max\{d_j\}$.

For each problem and $n = 10$, the optimal objective function value OPT was found by explicit enumeration. For each generated instance of $C_{\max}$, $L_{\max}$ or $\sum w_j C_j$ mini-

Table 1
Average performances of the algorithms

n	$c1$	$c2$	$c3$	$l1$	$l2$	$l3$	$s1$	$s2$	$s3$
10	1.000	1.095	1.055	1.030	1.033	1.095	1.162	1.730	1.476
50	1.001	1.098	1.080	1.000	1.002	1.068	1.000	1.490	1.323
100	1.001	1.014	1.033	1.000	1.002	1.049	1.000	1.475	1.327
500	1.001	1.121	1.095	1.000	1.001	1.026	1.000	1.441	1.325

Table 2
Experimental results for $C_{\max}$ criterion and $n = 10$

Interval for p_j	$c1/OPT$	$c2/OPT$	$c3/OPT$
$(\Delta_{min}, \Delta_{avg}]$	1.000	1.145	1.098
$(\Delta_{avg}, \Delta_{max}]$	1.000	1.032	1.025
$(\Delta_{min}, \Delta_{max}]$	1.000	1.108	1.041

Table 3
Experimental results for $L_{\max}$ criterion and $n = 10$

Interval for d_j	$l1/OPT$	$l2/OPT$	$l3/OPT$
(A/2,A]	1.026	1.026	1.086
(A/4,3A/4]	1.034	1.037	1.115
(0,A]	1.030	1.037	1.083

Table 4
Experimental results for $\sum w_j C_j$ criterion and $n = 10$

Interval for w_j	$s1/OPT$	$s2/OPT$	$s3/OPT$
(0,1]	1.250	1.792	1.541
(0,10]	1.119	1.702	1.447
(0,100]	1.116	1.697	1.441

mization problems, algorithms were evaluated by the performance ratio xi/OPT, where $x \in \{c, l, s\}$, $i \in \{1, 2, 3\}$.

For $n = 50$, $n = 100$ and $n = 500$, performances of the algorithms were evaluated by the relative performance ratio xi/x^*, where $x^* = \min\{xi | i = 1, 2, 3\}$. Results of our experiments are given in Tables 2–7. Tables 2–4 contain the results obtained for each problem and $n = 10$. The results obtained for $n = 50$, $n = 100$ and $n = 500$ are given in Tables 5–7. The average performance ratios calculated for each scheduling problem and given n are summarized in Table 1.

Table 5
Experimental results for C_{max} criterion and $n = 50$, $n = 100$ and $n = 500$

Interval for p_j	$n = 50$			$n = 100$			$n = 500$		
	c1/c*	c2/c*	c3/c*	c1/c*	c2/c*	c3/c*	c1/c*	c2/c*	c3/c*
$(\Delta_{min}, \Delta_{avg}]$	1.000	1.288	1.231	1.000	1.041	1.034	1.000	1.240	1.187
$(\Delta_{avg}, \Delta_{max}]$	1.001	1.001	1.001	1.001	1.001	1.001	1.001	1.001	1.001
$(\Delta_{min}, \Delta_{max}]$	1.001	1.004	1.001	1.001	1.001	1.001	1.001	1.124	1.098

Table 6
Experimental results for L_{max} criterion and $n = 50$, $n = 100$ and $n = 500$

Interval for d_j	$n = 50$			$n = 100$			$n = 500$		
	c1/c*	c2/c*	c3/c*	c1/c*	c2/c*	c3/c*	c1/c*	c2/c*	c3/c*
(A/2,A]	1.000	1.000	1.061	1.000	1.000	1.043	1.000	1.000	1.023
(A/4,3A/4]	1.000	1.000	1.082	1.000	1.000	1.060	1.000	1.000	1.031
(0,A]	1.000	1.007	1.061	1.000	1.005	1.045	1.000	1.002	1.023

Table 7
Experimental results for $\sum w_j C_j$ criterion and $n = 50$, $n = 100$ and $n = 500$

Interval for w_j	$n = 50$			$n = 100$			$n = 500$		
	c1/c*	c2/c*	c3/c*	c1/c*	c2/c*	c3/c*	c1/c*	c2/c*	c3/c*
(0,1]	1.000	1.465	1.312	1.000	1.464	1.325	1.000	1.439	1.324
(0,10]	1.000	1.503	1.327	1.000	1.480	1.328	1.000	1.443	1.326
(0,100]	1.000	1.503	1.329	1.000	1.480	1.328	1.000	1.441	1.324

Our experiments demonstrated that the classical approaches are the best in average. Only for $L_{\max}$ minimization problem, the performance of algorithm L2 was very close to that for the classical approach.

5. Conclusions

In this paper, we established computational complexity of three scheduling problems with exponentially dependent job processing times. We considered minimization of the following cost functions: the makespan, the maximum lateness and the total weighted completion time. In the NP-hardness proofs, we used reductions from problems PARTITION and 3-PARTITION. For solving each of the considered scheduling problems, we constructed and experimentally compared three heuristic algorithms. It follows from the results obtained in our experiments that the well known scheduling heuristics are efficient to solve the problems with exponentially start time dependent job processing times.

Further research can be undertaken to apply proposed heuristics, some other heuristics, local search techniques and enumerative methods for solving real-life scheduling problems with exponential functions of job processing times.

Acknowledgments

This research was supported in part by Polish Scientific Committee KBN. M.Y. Kovalyov was in addition supported by INTAS under grant number 03-51-5501.

References

Alidaee, B., and N.K. Womer (1999). Scheduling with time dependent processing times: Review and extensions. *Journal of the Operational Research Society*, **50**, 711–720.

Browne, S., and U. Yechiali (1990). Scheduling deteriorating jobs on a single processor. *Operations Research*, **38**(3), 495–498.

Garey, M.R., and D.S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-completeness.* Freeman, San Francisco.

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

Ho, K.I-J., J.Y-T. Leung and W-D. Wei (1993). Complexity of scheduling tasks with time-dependent execution times. *Information Processing Letters*, **48**, 315–320.

Jackson, J.R. (1955). *Scheduling a Production Line to Minimize Maximum Tardiness.* Research Report 43, Management Science Research Project, University of California, Los Angeles.

Kunnathur, A.S., and S.K. Gupta (1990). Minimizing the makespan with late start penalties added to processing times in a single facility scheduling problem. *European Journal of Operational Research*, **47**, 56–64.

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

Smith, W.E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, **3**, 59–66.

A. Janiak is a full professor in computer science and operations research of industrial engineering areas in Institute of Engineering Cybernetics at Wroclaw University of Technology, Poland, being a head of Department of Artificial Intelligence and Algorithms Design. He has authored two books and more than 100 papers in edited books, international journals, and conference proceedings. His research interests include sequencing and scheduling problems with classical and generalized models of operations in computer and manufacturing systems, resource allocation problems, complexity theory, theory of algorithms (physical design automation of VLSI). He is a member of the Computer Science Committee of the Polish Academy of Science and a member of Computer Methods in Science Section of Polish Research Council. He was invited as a visiting professor to universities in Australia, Canada, Germany, Hong Kong, Israel, New Zealand, Thailand and France. Prof. Janiak has served on the program committees for several international conferences on operations research and is a regular reviewer for a number of prestigious journals and conferences.

M.Y. Kovalyov is a professor in the Faculty of Economics, Belarus State University. He received the candidate of sciences degree and the doctor of sciences degree from the Institute of Mathematics, Belarus Academy of Sciences. His main research interests are in discrete optimization, scheduling and logistics. He published over 60 papers in international journals. He is an associate editor for Asia-Pacific Journal of Operational Research and a member of the editorial board of European Journal of Operational Research.

Darbų eiliškumas su eksponentinėmis atlikimo trukmių funkcijomis

Adam JANIAK, Mikhail Y. KOVALYOV

Šiame straipsnyje nagrinėjami vieno įrenginio tvarkaraščio sudarymo uždaviniai, kuriuose darbų atlikimo trukmės eksponentiškai priklauso nuo darbų pradžios laiko. Didejančioms funkcijoms įrodoma, kad atlikimo tarpsnio minimizavimo uždavinys priklauso stipraus NP-sudėtingumo klasei. Mažėjančioms funkcijoms įrodoma, kad didžiausio vėlavimo minimizavimo uždavinys priklauso stipraus NP-sudėtingumo klasei, o svorinių pabaigos laikų sumos minimizavimo uždavinys priklauso NP-sudėtingumo klasei. Euristiniai algoritmai yra pristatyti ir eksperimentiškai patikrinti šiems uždaviniams.