Single Machine Scheduling with a Generalized Job-Dependent Cumulative Effect

Kabir Rustogi and Vitaly A. Strusevich
Department of Mathematical Sciences, University of Greenwich,
Old Royal Naval College, Park Row, Greenwich, London SE10 9LS, U.K.
E-mail: {K.Rustogi,V.Strusevich}@greenwich.ac.uk

Abstract

We consider a single machine scheduling problem with changing processing times. The processing conditions are subject to a general cumulative effect, in which the processing time of a job depends on the sum of certain parameters associated with previously scheduled jobs. In previous papers, these parameters are assumed to be equal to the normal processing times of jobs, which seriously limits the practical application of this model. We further generalize this model by allowing every job to respond differently to these cumulative effects. For the introduced model, we solve the problem of minimizing the makespan, with and without precedence constraints. For the problem without precedence constraints we also consider a situation in which a maintenance activity is included in the schedule, which can improve the processing conditions of the machine, not necessarily to its original state. The resulting problem is reformulated as a variant of a Boolean programming problem with a quadratic objective, known as a half-product, which allows us to develop a fully polynomial-time approximation scheme with the best possible running time.

Keywords: Single machine; Deterioration; Precedence Constraints; Maintenance; Approximation scheme; Half-Product Problem

1 Introduction

Scheduling models, in which the actual processing times of jobs are not constant but are subject to various effects, have recently generated a considerable volume of publications. Traditionally, in the literature on scheduling with changing processing times two opposite effects are studied: deterioration and learning. Under *deterioration*, the later a job starts, the more time is required to process it. A common rationale for deterioration effects is that the processing quality of a machine-tool gets worse. On the other hand, under a *learning* effect the actual processing times for the jobs that are scheduled later appear to be shorter, which can be illustrated by an example of human operators who improve their skills in performing similar activities by gaining experience.

In this paper, we address several versions of a single machine scheduling problem to minimize the maximum completion time, provided that a generalized linear job-dependent cumulative effect is applied. We are given the jobs of set $N = \{1, 2, \ldots, n\}$, to be processed on a single machine. Each job $j \in N$ is associated with an integer p_j that is called its "normal" processing time. This value can be understood as the actual processing duration of job j, provided that the machine is in a perfect condition.

In scheduling literature, the effects that may affect the actual processing time of a job $j \in N$, usually belong to one of the following types (or their combination):

- *Time-Dependent* effect: the actual processing time of job j depends on the start time of the job; see the book by Gawiejnowicz (2008) which gives a detailed exposition of scheduling models with this effect;
- *Positional* effect: the actual processing time of job j depends on p_j and on the position of the job in the sequence; see a focused survey by Rustogi and Strusevich (2012b) and a discussion in Agetis et al. (2014);
- *Cumulative* effect: the actual processing time of job j depends on p_j and on an accumulated value of some parameter, typically, on the sum of normal processing times of all jobs sequenced earlier; see Kuo and Yang (2006a) and Kuo and Yang (2006b), where a similar effect is introduced.

Suppose that the jobs are processed on a single machine in accordance with the sequence $\pi = (\pi(1), \pi(2), \dots, \pi(n))$. Under the most studied cumulative effect, introduced in Kuo and Yang (2006a) and Kuo and Yang (2006b), the actual processing time of job j scheduled in the r-th position of permutation π is defined by

$$p_j(\pi; r) = p_j \left(1 + b \sum_{h=1}^{r-1} p_{\pi(h)}\right)^A, \tag{1}$$

where A is a given constant, and b is either equal to 1 or to -1, in the case of deterioration or of learning, respectively. The extensions and generalizations of this basic model can be found in Yin et al. (2009) and Huang and Wang (2015). A common drawback of papers on scheduling with a cumulative effect is that normally no convincing practical motivation of the model is given. In particular, it is not well-justified why the actual processing time of a job should depend on total normal time of previously scheduled jobs.

In this paper, we study a cumulative effect that arises when a job $j \in N$ is associated not only with the normal processing time p_j but also with two additional parameters, b_j and $q_j > 0$. Here q_j is a quantity, not necessarily equal to the normal processing time, such that its accumulated value affects the actual processing time of later scheduled jobs. Formally, if job j is scheduled in the r-th position of permutation π is defined by

$$p_j(\pi; r) = p_j \left(1 + b_j \sum_{h=1}^{r-1} q_{\pi(h)}\right), \tag{2}$$

where $b_j > 0$ under a deterioration effect and $b_j < 0$ under a learning effect. Unlike (1), the effect (2) is represented not by a polynomial but by a linear function of the accumulated quantities. On the other hand, no explicit dependence on the normal time of previously scheduled jobs is assumed and the values of b_j can be understood as job-dependent rates that reflect how sensitive a particular job is to the previously scheduled jobs.

For illustration of our model, suppose that a floor sanding machine is used to treat floors in several rooms. The normal time p_j is the time requirement for sanding floors in room j, provided that a new sanding belt/disk is used. The value of q_j can be seen as the amount of generated saw dust or an appropriately measured wear of the sanding belt/disk, which does not necessarily depend on the time of treatment. For some rooms the actual treatment time can be seriously affected by the quality of the equipment, for some rooms the effect may be less noticeable, and this job dependency is captured by the rate parameter b_j. It is not difficult to identify a similar cumulative deterioration effect in other activities/industries.

To illustrate the effect (2) in a learning environment, consider the following situation. Suppose a computer programmer is supposed to write n software pieces for a particular project. These pieces can be developed in any order. Typically, a software piece $j \in N$ can be completed in p_j time units by an inexperienced programmer. Assume that after completing a particular software piece j, the technical skill of the programmer increases by q_j appropriately measured units, and that skill might help to speed up the creation of any peice to follow. Thus, the actual time needed to create a particual peice depends on the accumulated skills gained during the development of previously created peices. Formally, the development time of a software peice decreases linearly with the technical skill of the programmer, so that the actual time taken to write a software peice $j = \pi(r)$ is given by $p_j(\pi; r) = p_j - a_j \sum_{h=1}^{r-1} q_{\pi(h)}$, where the quantity a_j defines how sensitive the development time for software peice j is to the gained technical skills. This formulation can be written in terms of the effect (2) with $b_j = -a_j/p_j$.

Adopting standard scheduling notation, we denote the problem of minimizing the makespan $C_{\max}$, i.e., the maximum completion time, under the effect (1) by $1\left|p_j(\pi; r) = p_j(1 + bP_r)^A\right|C_{\max}$, where P_r stands for the sum of the normal processing times of the jobs scheduled prior to job $\pi(r)$. A similar problem under the effect (2) is denoted by $1\left|p_j(\pi; r) = p_j(1 + b_jQ_r)\right|C_{\max}$, where Q_r represents the sum of the q_j values of the jobs scheduled prior to job $\pi(r)$.

Apart from problem $1\left|p_j(\pi; r) = p_j(1 + b_jQ_r)\right|C_{\max}$ in which the jobs of set N are independent, we also study its version in which precedence constraints are imposed on the set of jobs, so that only those permutations of jobs that respect the constraints are feasible. These precedence constrains are given in a form of an acyclic directed graph, with the nodes representing the jobs and the arcs linking immediate successors and predecessors. Provided that the digraph that defines precedence constraints is series-parallel, we denote the problems under effects (1) and (2) by $1\left|p_j(\pi; r) = p_j(1 + bP_r)^A, SP-prec\right|C_{\max}$ and $1\left|p_j(\pi; r) = p_j(1 + b_jQ_r), SP-prec\right|C_{\max}$, respectively. See Gordon et al. (2008) for a range of results on single machine scheduling with series-parallel precedence constraints and various effects (positional, time-dependent and cumulative), including problem $1\left|p_j(\pi; r) = p_j(1 + P_r)^A, SP-prec\right|C_{\max}$. Extending our floor sanding machine example given above, problem $1\left|p_j(\pi; r) = p_j(1 + b_jQ_r), SP-prec\right|C_{\max}$ can arise if precedence constraints occur due to a particular physical layout of the building in which the rooms to be sanded are located.

For scheduling problems with a deterioration effect, the actual processing times grow. In order to prevent the processing times to become unacceptably large, a maintenance period (MP) can be introduced into a schedule. During an MP no processing takes place, and after the MP the processing facility is in better processing conditions. The duration of an MP is either a constant or depends on its start time τ. See Rustogi and Strusevich (2012a, 2014, 2015) for studies of scheduling models with maintenance under positional effects, combined effects and time-dependent effects, respectively.

Rustogi and Strusevich (2013) study problems $1\left|p_j(\pi; r) = p_j(1 + bP_r), MP(0)\right|C_{\max}$ and $1\left|p_j(\pi; r) = p_j(1 + bP_r), MP(\lambda)\right|C_{\max}$ with exactly one MP introduced into a schedule, where $MP(\lambda)$ means that the duration of the MP is a linear function of its start time τ written as $\lambda\tau + \mu$, where μ and λ are given constants; in particular $MP(0)$ corresponds to the MP of constant duration μ. In this paper, we address more general problems $1\left|p_j(\pi;r) = p_j(1+b_jQ_r), MP(0)\right|C_{\max}$ and $1\left|p_j(\pi;r) = p_j(1+b_jQ_r), MP(\lambda)\right|C_{\max}$ with a single maintenance period. Notice that in the models studied in Rustogi and Strusevich (2013) the MP is assumed to fully restore the processing conditions, so that after the maintenance the machine is "as good as new". In this paper, we consider the MP as a rate-modifying activity, as introduced by Lee and Leon (2001), and assume that for a job $j \in N$ scheduled after the MP the normal processing time changes from p_j to σp_j, where $\sigma \geq 1$ is a given constant.

The problems with a single MP are NP-hard, and we focus on the design of fully polynomial-time approximation schemes (FPTAS). Recall that for a problem of minimizing a function $\Phi(\mathbf{x})$, where $\mathbf{x}$ is a collection of decision variables, a polynomial-time algorithm that finds a feasible solution $\mathbf{x}^H$ such that $\Phi(\mathbf{x}^H)$ is at most $\rho \geq 1$ times the optimal value $\Phi(\mathbf{x}^*)$ is called a $\rho-$*approximation* algorithm; the value of ρ is called a *worst-case ratio bound.* A family of $\rho-$approximation algorithms is called a *fully polynomial-time approximation scheme (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 $1/\varepsilon$.

The remainder of this paper is organized as follows. In Section 2, problem $1\left|p_j(\pi;r) = p_j(1+b_jQ_r)\right|C_{\max}$ is reduced to the classical scheduling problem $1||\sum w_jC_j$ to minimize the sum of weighted completion times on a single machine, and is therefore solvable in $O(n\log n)$ time. Using the theory of minimizing priority-generating functions over series-parallel precedence constraints, in Section 3, we show that problem $1\left|p_j(\pi;r) = p_j(1+b_jQ_r), SP-prec\right|C_{\max}$ is also solvable in $O(n\log n)$ time. In Section 4, we present a fast FPTAS for problem $1\left|p_j(\pi;r) = p_j(1+b_jQ_r), MP(\lambda)\right|C_{\max}$ with a single maintenance period. Some concluding remarks can be found in Section 5.

2 Minimization of Makespan

For a scheduling problem to minimize a function Φ over a set of permutations, an optimal solution can be found by applying a priority rule, i.e., by associating each job $j \in N$ with a value $\omega(j)$ and sorting the jobs in non-increasing order of $\omega(j)$'s. The values $\omega(j)$, $j \in N$, are called 1-*priorities.* The most popular 1-priorities are $\omega(j) = p_j$, $j \in N$, and $\omega(j) = 1/p_j$, $j \in N$, which correspond to the well-known LPT and SPT priority rules, respectively.

Problem $1\left|p_j(\pi;r) = p_j(1+bP_r)^A\right|C_{\max}$ is known to be solvable by the SPT rule if $A < 0$ (learning, see Kuo and Yang (2006b)) and if $A > 1$ (fast deterioration, see Gordon et al. (2008)). For the problem with $A = 1$, the objective function is sequence independent, i.e., any permutation is optimal; see Gordon et al. (2008).

Another well-known scheduling priority rule is the WSPT (or Smith's) rule. This rule is based on 1-priorities $\omega(j) = w_j/p_j$, $j \in N$, and finds an optimal permutation for problem $1||\sum w_jC_j$ of minimizing the sum of weighted completion times.

Assume that in problem $1||\sum w_jC_j$ the processing time of a job $j \in N$ is denoted by q_j. Then the value of the objective function for a schedule associated with a permutation $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$ is given by

$$\sum_{r=1}^{n} w_{\pi(r)}C_{\pi(r)} = \sum_{r=1}^{n} w_{\pi(r)} \sum_{h=1}^{r} q_{\pi(h)},$$

and, as proved by Smith (1956), an optimal permutation can be found in $O(n \log n)$ time by sorting the jobs in non-increasing order of the 1-priorities $\omega(j) = w_j / q_j$.

We use this result to solve problem $1 \left| p_j(\pi; r) = p_j(1 + b_j Q_r) \right| C_{\max}$.

Theorem 1 *For problem $1 \left| p_j(\pi; r) = p_j(1 + b_j Q_r) \right| C_{\max}$, an optimal permutation can be found in $O(n \log n)$ time by sorting the jobs in non-increasing order of the ratios $(p_j b_j)/q_j$, $j \in N$.*

Proof: We reduce the problem under consideration to problem $1 | | \sum w_j C_j$, with the processing times equal to q_j and the weights defined by

$$w_j = b_j p_j, \ j \in N. \tag{3}$$

Given a permutation $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$ of jobs, let $C_{\max}(\pi)$ denote the makespan for a schedule in which the jobs are processed according to permutation π. Then for the original problem, we have

$$\begin{aligned} C_{\max}(\pi) &= p_{\pi(1)} + \sum_{r=2}^{n} p_{\pi(r)} \left(1 + b_{\pi(r)} \sum_{h=1}^{r-1} q_{\pi(h)} \right) = \\ &= \sum_{r=1}^{n} p_{\pi(r)} + \sum_{r=2}^{n} b_{\pi(r)} p_{\pi(r)} \sum_{h=1}^{r-1} q_{\pi(h)} \\ &= \sum_{r=1}^{n} p_{\pi(r)} + \sum_{r=1}^{n} b_{\pi(r)} p_{\pi(r)} \sum_{h=1}^{r-1} q_{\pi(h)}, \end{aligned}$$

where the last equality is due to $\sum_{h=1}^{0} q_{\pi(h)} = 0$.

Using (3), we further rewrite

$$\begin{aligned} C_{\max}(\pi) &= \sum_{r=1}^{n} p_{\pi(r)} + \sum_{r=1}^{n} w_{\pi(r)} \sum_{h=1}^{r-1} q_{\pi(h)} \\ &= \sum_{r=1}^{n} p_{\pi(r)} + \sum_{r=1}^{n} w_{\pi(r)} \sum_{h=1}^{r} q_{\pi(h)} - \sum_{r=1}^{n} w_{\pi(r)} q_{\pi(r)} \\ &= \sum_{r=1}^{n} w_{\pi(r)} \sum_{h=1}^{r} q_{\pi(h)} + \sum_{j=1}^{n} (p_j - w_j q_j). \end{aligned}$$

Thus, $C_{\max}(\pi)$ is minimized if the minimum of $\sum_{r=1}^{n} w_{\pi(r)} \sum_{h=1}^{r} q_{\pi(h)}$ is attained. The latter expression is the objective function in problem $1 | | \sum w_j C_j$, so that the optimal permutation can be found by the WSPT rule. In terms of the original problem, an optimal permutation is obtained by sorting the jobs in non-increasing order of the ratios $(p_j b_j)/q_j$. ■

Reformulating Theorem 1, we conclude that for the problem of minimizing the makespan under an effect (2) the 1-priority is $\omega(j) = b_j p_j / q_j$, $j \in N$. Notice that Theorem 1 holds irrespective of the sign of b_j, i.e., for both deterioration and learning effects.

Theorem 1 can be applied to the effect that resembles (1) with $A = 1$.

Corollary 1 *If effect (2) is applied with $q_j = p_j$, for all $j \in N$, then the resulting problem $1\left|p_j(\pi;r) = p_j(1+b_jP_r)\right|C_{\max}$is solvable in $O(n\log n)$ time by sequencing jobs in non-increasing order of b_j. Moreover, if $b_j = 1$, for all $j \in N$, then an arbitrary permutation of jobs results in an optimal solution to the resulting problem $1\left|p_j(\pi;r) = p_j(1+P_r)\right|C_{\max}$.*

Notice that the part of Corollary 1 on problem $1\left|p_j(\pi;r) = p_j(1+P_r)\right|C_{\max}$ is also proved in Gordon et al. (2008).

3 Minimization of Makespan with Precedence Constraints

In this section, we study problem $1\left|p_j(\pi;r) = p_j(1+b_jQ_r), SP-prec\right|C_{\max}$, provided that precedence constraints are imposed on the set of jobs, and the graph that defines these constraints is series-parallel; see Valdes et al. (1982) and Tanaev et al. (1984) for relevant definitions.

Research on scheduling problems under series-parallel precedence constraints was initiated by Lawler (1978), who presented a polynomial-time algorithm for minimizing the weighted sum of the completion times on a single machine subject to series-parallel constraints. Soon after, it has been discovered that many other scheduling problems can be solved by a similar approach, provided that their objective functions possess specific properties, related to an extended notion of a priority function that is defined for subsequences of jobs rather than just for individual jobs. The definition below can be found in Tanaev et al. (1984) and Monma and Sidney (1979).

Definition 1 *Let $\pi^{\alpha\beta} = (\pi'\alpha\beta\pi'')$ and $\pi^{\beta\alpha} = (\pi'\beta\alpha\pi'')$ be two permutations of n jobs that differ only in the order of the subsequences α and β (here subsequences π' and/or π'' can be dummy permutations with no elements). For a function $\Phi(\pi)$ that depends on a permutation, suppose that there exists a function $\omega(\pi)$ such that for any two permutations $\pi^{\alpha\beta}$ and $\pi^{\beta\alpha}$ the inequality $\omega(\alpha) > \omega(\beta)$ implies that $\Phi(\pi^{\alpha\beta}) \le \Phi(\pi^{\beta\alpha})$, while the equality $\omega(\alpha) = \omega(\beta)$ implies that $\Phi(\pi^{\alpha\beta}) = \Phi(\pi^{\beta\alpha})$. In this case, function Φ is called a* priority-generating function*, while function ω is called its* priority function*. For a (partial) permutation π, the value of $\omega(\pi)$ is called the* priority *of π.*

A priority function applied to a single job becomes a 1-priority for that job. Thus, for function $\Phi(\pi)$ to be priority-generating, it is necessary that the problem of minimizing $\Phi(\pi)$ admits 1-priorities. On the other hand, the existence of 1-priorities does not imply that they can be extended to a priority function. Intuitively, a priority function allows us to rank not only individual jobs but also partial permutations. The fastest algorithm known algorithm minimizes a priority-generating function under series-parallel precedence constraints in $O(n\log n)$ time; see, e.g., Monma and Sidney (1979) and Tanaev et al. (1984).

Various single machine scheduling problems with changing times and series-parallel precedence constraints have been studied in Gordon et al. (2008), Wang et al. (2008) and Dolgui et al. (2012). In particular, Gordon et al. (2008) study problem $1\left|p_j(\pi;r) = p_j(1+P_r)^A, SP-prec\right|C_{\max}$ for a positive integer A. For $A = 1$, the objective function is sequence independent (see Corollary 1), and the problem is solvable in $O(n)$ time under arbitrary precedence constraints since any feasible permutation is optimal. For $A = 2$, the objective function is proved to be priority-generating and the problem is therefore solvable in $O(n \log n)$ time. On the other hand, for $A = 3$, the objective function is proved not to generate a priority function.

Below, we use Definition 1 to prove that for problem $1 \left| p_j(\pi; r) = p_j(1 + b_j Q_r), SP - prec \right| C_{\max}$ the objective function is priority-generating.

Theorem 2 *For the single machine problem to minimize the makespan under the cumulative effect (2), the objective function is priority-generating and*

$$\omega(\pi) = \frac{\sum_{j=1}^{|\pi|} p_{\pi(j)} b_{\pi(j)}}{\sum_{j=1}^{|\pi|} q_{\pi(j)}} \tag{4}$$

is its priority function, so that problem $1 \left| p_j(\pi; r) = p_j(1 + b_j Q_r), SP - prec \right| C_{\max}$ *is solvable in* $O(n \log n)$ *time.*

Proof. For a (partial) permutation π we denote the length of π, i.e., the number of jobs in π, by $|\pi|$. For a partial permutation π conisder a schedule S such that π is contained as a subsequence in a full permutation that defines schedule S. Assume that for S the following holds: (i) the first job in π starts at time τ; and (ii) the sum of the q_j values of the jobs that precede the first job in π, i.e., those completed by time τ, is equal to ζ. Under these assumptions, let $C_{\max}(\pi; \tau; \zeta)$ denote the maximum completion time of the jobs in π. By definition, for problem $1 \left| p_j(\pi; r) = p_j(1 + b_j Q_r), SP - prec \right| C_{\max}$ we deduce

$$C_{\max}(\pi; \tau; \zeta) = \tau + C_{\max}(\pi; 0; \zeta) = \tau + \sum_{k=1}^{|\pi|} p_{\pi(k)} \left(1 + b_{\pi(k)} \left(\zeta + \sum_{i=1}^{k-1} q_{\pi(i)}\right)\right).$$

Let $\pi^{\alpha\beta} = (\pi_1 \alpha \beta \pi_2)$ and $\pi^{\beta\alpha} = (\pi_1 \beta \alpha \pi_2)$ be two permutations of all jobs that only differ in the order of the subsequences α (containing u jobs) and β (containing v jobs). Define $\Delta C = C_{\max}(\pi^{\alpha\beta}) - C_{\max}(\pi^{\beta\alpha})$ and let ζ' denote the total sum of the q_j values of the jobs in π_1. Then $\Delta C = C_{\max}(\alpha\beta\pi_2; 0; \zeta') - C_{\max}(\beta\alpha\pi_2; 0; \zeta')$. Furthermore,

$$\begin{aligned}
C_{\max}(\alpha\beta\pi_2; 0; \zeta') &= C_{\max}(\alpha\beta; 0; \zeta') \\
&\quad + \sum_{k=1}^{|\pi_2|} p_{\pi_2(k)} \left(1 + b_{\pi_2(k)} \left(\zeta' + \sum_{i=1}^{u} q_{\alpha(i)} + \sum_{j=1}^{v} q_{\beta(j)} + \sum_{i=1}^{k-1} q_{\pi_2(i)}\right)\right), \\
C_{\max}(\beta\alpha\pi_2; 0; \zeta') &= C_{\max}(\beta\alpha; 0; \zeta') \\
&\quad + \sum_{k=1}^{|\pi_2|} p_{\pi_2(k)} \left(1 + b_{\pi_2(k)} \left(\zeta' + \sum_{j=1}^{v} q_{\beta(j)} + \sum_{i=1}^{u} q_{\alpha(i)} + \sum_{i=1}^{k-1} q_{\pi_2(i)}\right)\right),
\end{aligned}$$

so that $\Delta C = C_{\max}(\alpha\beta; 0; \zeta') - C_{\max}(\beta\alpha; 0; \zeta')$. To prove the theorem, we derive conditions under which $\Delta C \leq 0$.

Further, we deduce

$$\begin{aligned}
C_{\max}(\alpha\beta;0;\zeta') &= C_{\max}(\alpha;0;\zeta') + \sum_{k=1}^{v} p_{\beta(k)}\left(1+b_{\beta(k)}\left(\zeta' + \sum_{i=1}^{u} q_{\alpha(i)} + \sum_{j=1}^{k-1} q_{\beta(j)}\right)\right)\\
&= \sum_{k=1}^{u} p_{\alpha(k)}\left(1+b_{\alpha(k)}\left(\zeta' + \sum_{i=1}^{k-1} q_{\alpha(i)}\right)\right)\\
&\quad + \sum_{k=1}^{v} p_{\beta(k)}\left(1+b_{\beta(k)}\left(\zeta' + \sum_{i=1}^{u} q_{\alpha(i)} + \sum_{j=1}^{k-1} q_{\beta(j)}\right)\right),\\
C_{\max}(\beta\alpha;0;\zeta') &= C_{\max}(\beta;0;\zeta') + \sum_{k=1}^{u} p_{\alpha(k)}\left(1+b_{\alpha(k)}\left(\zeta' + \sum_{j=1}^{v} q_{\beta(j)} + \sum_{i=1}^{k-1} q_{\alpha(i)}\right)\right)\\
&= \sum_{k=1}^{v} p_{\beta(k)}\left(1+b_{\beta(k)}\left(\zeta' + \sum_{j=1}^{k-1} q_{\beta(j)}\right)\right)\\
&\quad + \sum_{k=1}^{u} p_{\alpha(k)}\left(1+b_{\alpha(k)}\left(\zeta' + \sum_{j=1}^{v} q_{\beta(j)} + \sum_{i=1}^{k-1} q_{\alpha(i)}\right)\right),
\end{aligned}$$

so that for ΔC we derive

$$\begin{aligned}
\Delta C &= \sum_{k=1}^{u} p_{\alpha(k)}\left(\left(1+b_{\alpha(k)}\left(\zeta' + \sum_{i=1}^{k-1} q_{\alpha(i)}\right)\right) - \left(1+b_{\alpha(k)}\left(\zeta' + \sum_{j=1}^{v} q_{\beta(j)} + \sum_{i=1}^{k-1} q_{\alpha(i)}\right)\right)\right)\\
&\quad + \sum_{k=1}^{v} p_{\beta(k)}\left(\left(1+b_{\beta(k)}\left(\zeta' + \sum_{i=1}^{u} q_{\alpha(i)} + \sum_{j=1}^{k-1} q_{\beta(j)}\right)\right) - \left(1+b_{\beta(k)}\left(\zeta' + \sum_{j=1}^{k-1} q_{\beta(j)}\right)\right)\right).
\end{aligned}$$

Proceeding further, we obtain

$$\Delta C = -\sum_{k=1}^{u} p_{\alpha(k)} b_{\alpha(k)} \sum_{j=1}^{v} q_{\beta(j)} + \sum_{k=1}^{v} p_{\beta(k)} b_{\beta(k)} \sum_{i=1}^{u} q_{\alpha(i)}.$$

Dividing by $\sum_{k=1}^{u} q_{\alpha(k)} \sum_{i=1}^{v} q_{\beta(i)}$, we deduce that $\Delta C \le 0$, provided that

$$\frac{\sum_{k=1}^{u} p_{\alpha(k)} b_{\alpha(k)}}{\sum_{k=1}^{u} q_{\alpha(k)}} \ge \frac{\sum_{i=1}^{v} p_{\beta(i)} b_{\beta(k)}}{\sum_{i=1}^{v} q_{\beta(i)}}.$$

For an arbitrary (partial) permutation π, define the function $\omega(\pi)$ by (4). It is easily verified that $\omega(\alpha) > \omega(\beta)$ implies $C_{\max}(\pi^{\alpha\beta}) \le C_{\max}(\pi^{\beta\alpha})$, while $\omega(\alpha) = \omega(\beta)$ implies $C_{\max}(\pi^{\alpha\beta}) = C_{\max}(\pi^{\beta\alpha})$, as required by Definition 1. ■

Theorem 2 holds irrespective of the sign of b_j, $j \in N$. Observe that if (4) is applied to a single job j, i.e., to a permutation of length one, then the priority function becomes a 1-priority function $\omega(j) = (p_j b_j)/q_j$, which is consistent with Theorem 1. Besides, if $q_j = p_j$ and $b_j = b$ for all $j \in N$, when $\omega(j)$ becomes constant, i.e., for problem $1\left|p_j(\pi;r) = p_j(1+bP_r)\right|C_{\max}$ any permutation is optimal, which is consistent with Corollary 1 and Gordon et al. (2008).

4 Minimization of Makespan with Machine Maintenance

In this section, we consider the effect (2) in the deterioration form, with $b_j > 0$. A single rate-modifying maintenance activity is introduced into a schedule, which is able to improve the processing quality of the machine.

An instance of problem $1\left|p_j(\pi; r) = p_j(1 + b_j Q_r), MP(\lambda)\right| C_{\max}$ is defined by the arrays of positive numbers p_j, q_j and b_j, $j \in N$, and positive numbers λ, μ and σ. The duration of the maintenance period (MP) is $\lambda\tau + \mu$ time units, provided that the MP starts at time τ; here $\lambda \geq 0$ and $\mu \geq 0$. For a job $j \in N$ scheduled after the MP, the normal processing time changes from p_j to σp_j, where $\sigma \geq 1$.

In a schedule with a single MP the jobs are split into two groups: group 1 consists of the jobs scheduled before the maintenance and group 2 contains all other jobs. Let N_i be the set of jobs in group i and $|N_i| = n_i$, for $i \in \{1, 2\}$. Due to Theorem 1, we may assume that the jobs in each group are sequenced in non-increasing order of the 1-priorities $(p_j b_j)/q_j$. This is why throughout this section the jobs are renumbered so that

$$\frac{p_1 b_1}{q_1} \geq \frac{p_2 b_2}{q_2} \geq \cdots \geq \frac{p_n b_n}{q_n}. \tag{5}$$

Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ denote a vector with $0-1$ components. Problem $1\left|p_j(\pi; r) = p_j(1 + b_j Q_r), MP(\lambda)\right| C_{\max}$ belongs to a range of scheduling problems that can be reduced to minimizing a function of the form

$$F(\mathbf{x}) = H(\mathbf{x}) + K, \tag{6}$$

where

$$H(\mathbf{x}) = \sum_{1 \leq i < j \leq n}^{n} u_i v_j x_i x_j - \sum_{j=1}^{n} h_j x_j, \tag{7}$$

is known as the *Half-product function.* The coefficients u_j and v_j are non-negative integers, while h_j is an integer that can be either negative or positive.

Let a vector that is optimal for the problem of minimizing (7), or equivalently, (6) be denoted by $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_n^*)$. Notice that we are only interested in the instances of the problem of minimizing function (7) for which the optimal value $H(\mathbf{x}^*)$ is strictly negative; otherwise, setting all decision variables to zero solves the problem. On the other hand, below and in fact in most known applications it is assumed that constant K is such that $F(\mathbf{x}^*) > 0$.

To proceed, we need to refine the definition of an FPTAS for the problem of minimizing a function $H(\mathbf{x})$ which takes both positive and negative values. For such a problem an FPTAS delivers a solution vector $\mathbf{x}^\varepsilon$ such that $H(\mathbf{x}^\varepsilon) - H(\mathbf{x}^*) \leq \varepsilon\,|H(\mathbf{x}^*)|$. For the problem of minimizing function (6) with $F(\mathbf{x}^*) > 0$, an FPTAS outputs a solution vector $\mathbf{x}^\varepsilon$ such that $F(\mathbf{x}^\varepsilon) \leq (1 + \varepsilon) F(\mathbf{x}^*)$.

Badics and Boros (1998) prove that the problem of minimizing function (7) is NP-hard. The first FPTAS for minimizing a function of the form (7) in strongly polynomial time is due to Erel and Ghosh (2008), with the running time of $O\left(n^2/\varepsilon\right)$. This running time should be seen as the best possible, since just computing the value of the objective function for a given vector $\mathbf{x}$ takes $O\left(n^2\right)$ time. However, it is known that an FPTAS for minimizing the function $H(\mathbf{x})$ does not necessarily behave as an FPTAS for minimizing the function $F(\mathbf{x})$ of the form (6) with an additive constant. This is due to the fact the optimal value of $H(\mathbf{x})$ is negative and K can be positive; see Erel and Ghosh (2008), Kellerer and Strusevich (2012) and Kellerer and Strusevich (2015) for discussion and examples.

For the problem of minimizing (6), Erel and Ghosh (2008) outline a procedure, which may behave as an FPTAS.

Theorem 3 *For the problem of minimizing function (6), denote the lower and upper bounds on the value of $F(\mathbf{x}^*)$ by LB and UB, respectively, i.e., $LB \le F(\mathbf{x}^*) \le UB$. If the ratio UB/LB is bounded from above by some positive γ, then there exists an algorithm that delivers a solution $\mathbf{x}^0$ such that $F(\mathbf{x}^0) - LB \le \varepsilon LB$ in $O(\gamma n^2/\varepsilon)$ time.*

Theorem 3 is proved in Erel and Ghosh (2008). If the value of γ is bounded from above by a polynomial of the length of the input of the problem, then the algorithm from Theorem 3 designed by Erel and Ghosh (2008) behaves as an FPTAS. Moreover, if γ is a constant, then such an FPTAS requires the best possible running time of $O\left(n^2/\varepsilon\right)$. In what follows, we refer to the algorithm from Theorem 3 as the γ-FPTAS.

Given problem $1\left|p_j\left(\pi;r\right)=p_j\left(1+b_jQ_r\right), MP(\lambda)\right|C_{\max}$, introduce a Boolean variable x_j in such a way that

$$x_j = \begin{cases} 1, & \text{if job } j \text{ is scheduled in the first group} \\ 0, & \text{otherwise} \end{cases}$$

for each job j, $1 \le j \le n$.

Taking the jobs in order of their numbering given by (5), if job $j \in N$ is scheduled in the first group then it completes at time

$$C_j = p_j x_j \left(1 + b_j \sum_{i=1}^{j-1} q_i x_i\right),$$

so that the MP starts at time $\tau = \sum_{j=1}^{n} p_j x_j \left(1 + b_j \sum_{i=1}^{j-1} q_i x_i\right)$. If job j is scheduled in the second group, then its completion time is given by

$$\begin{aligned} C_j &= \tau + (\lambda\tau + \mu) + \sigma p_j (1 - x_j)\left(1 + b_j \sum_{i=1}^{j-1} q_i (1 - x_i)\right) \\ &= (\lambda + 1)\sum_{j=1}^{n} p_j x_j \left(1 + b_j \sum_{i=1}^{j-1} q_i x_i\right) + \sigma p_j (1 - x_j)\left(1 + b_j \sum_{i=1}^{j-1} q_i (1 - x_i)\right) + \mu. \end{aligned}$$

This implies that in order to solve problem $1\left|p_j\left(\pi;r\right)=p_j\left(1+b_jQ_r\right), MP\left(\lambda\right)\right|C_{\max}$, we need to minimize the function

$$\begin{aligned} Z\left(\mathbf{x}\right) &= (\lambda + 1)\sum_{j=1}^{n} p_j x_j \left(1 + b_j \sum_{i=1}^{j-1} q_i x_i\right) + \sigma \sum_{j=1}^{n} p_j (1 - x_j)\left(1 + b_j \sum_{i=1}^{j-1} q_i (1 - x_i)\right) + \mu \\ &= \sum_{j=1}^{n} \left(\lambda + 1\right) b_j p_j x_j \left(\sum_{i=1}^{j-1} q_i x_i\right) + \sum_{j=1}^{n} \sigma b_j p_j (1 - x_j)\left(\sum_{i=1}^{j-1} q_i (1 - x_i)\right) \\ &\quad + (\lambda + 1)\sum_{j=1}^{n} p_j x_j + \sigma \sum_{j=1}^{n} p_j (1 - x_j) + \mu. \end{aligned}$$

We show that the above function admits a representation of the form (6). As in (3), define $w_j = b_j p_j$, $j \in N$, and rewrite $Z(\mathbf{x})$ as

$$\begin{aligned} Z(\mathbf{x}) &= \sum_{1\le i<j\le n} (\lambda+1)\, q_i w_j x_i x_j + \sum_{1\le i<j\le n} \sigma q_i w_j (1-x_i)(1-x_j) \\ &\quad + (\lambda+1)\sum_{j=1}^{n} p_j x_j + \sigma \sum_{j=1}^{n} p_j (1-x_j) + \mu. \end{aligned} \tag{8}$$

This function is now written in the form that appears as an objective function in the so-called symmetric quadratic knapsack problem, see Kellerer and Strusevich (2012) and Kellerer and Strusevich (2015) for reviews.

Since

$$\begin{aligned} \sum_{1\le i<j\le n} q_i w_j (1-x_i)(1-x_j) &= \sum_{1\le i<j\le n} q_i w_j x_i x_j - \sum_{j=1}^{n} \left(w_j \left(\sum_{i=1}^{j-1} q_i \right) + q_j \left(\sum_{i=j+1}^{n} w_i \right) \right) x_j \\ &\quad + \sum_{1\le i<j\le n} q_i w_j, \end{aligned}$$

and

$$(\lambda+1)\sum_{j=1}^{n} p_j x_j + \sigma \sum_{j=1}^{n} p_j (1-x_j) = (\lambda-\sigma+1)\sum_{j=1}^{n} p_j x_j + \sigma \sum_{j=1}^{n} p_j$$

function $Z(\mathbf{x})$ derived above may be written as

$$\begin{aligned} Z(\mathbf{x}) &= \sum_{1\le i<j\le n} (\lambda+\sigma+1)\, q_i w_j x_i x_j \\ &\quad + \sum_{j=1}^{n} \left((\lambda-\sigma+1)\, p_j - \sigma \left(w_j \left(\sum_{i=1}^{j-1} q_i \right) + q_j \left(\sum_{i=j+1}^{n} w_i \right) \right) \right) x_j \\ &\quad + \left(\mu + \sigma \left(\sum_{1\le i<j\le n} q_i w_j + \sum_{j=1}^{n} p_j \right) \right). \end{aligned} \tag{9}$$

This is clearly a representation of the form (6) with

$$\begin{aligned} u_j &= (\lambda+\sigma+1)\, w_j,\ v_j = q_j,\ h_j = (\lambda-\sigma+1)\, p_j - \sigma \left(w_j \left(\sum_{i=1}^{j-1} q_i \right) + q_j \left(\sum_{i=j+1}^{n} w_i \right) \right),\ j \in N; \\ K &= \mu + \sigma \left(\sum_{1\le i<j\le n} q_i w_j + \sum_{j=1}^{n} p_j \right). \end{aligned}$$

According to Theorem 3, in order to obtain a γ-FPTAS for the problem of minimizing function (6), we are required to find the bounds LB and UB on the value of $Z(\mathbf{x}^*)$ and to prove that the ratio UB/LB is bounded from above by a constant γ. Notice, since we aim at obtaining an FPTAS with the best possible running time of $O\left(n^2/\varepsilon\right)$, we need to find the required lower and upper bounds in no more than $O\left(n^2\right)$ time. This can be done as described below.

Assume that the integrality constraint of the decision variables x_j, is relaxed, i.e., the condition $x_j \in \{0,1\}$ is replaced by $0 \le x_j \le 1$, $j \in N$. If $\mathbf{x}^C = (x_1^C, \ldots, x_n^C)$, $0 \le x_j^C \le 1$, is the corresponding solution vector and $Z(\mathbf{x}^C)$ denotes the optimal value of the function (8) for the continuous relaxation, then clearly $Z(\mathbf{x}^C) \le Z(\mathbf{x}^*)$, i.e., we may set $LB = Z(\mathbf{x}^C)$.

As demonstrated in Kellerer and Strusevich (2010), the relaxation of the problem of minimizing a convex function of the form (6), even with an additional linear knapsack constraint, reduces to finding the minimum cost flow with a convex quadratic cost function in a special network. The latter problem is studied by Tamir (1993) who gives a solution algorithm that in the case under consideration requires $O\left(n^2\right)$ time.

Notice that a function of the form (7) is proved convex, provided that the items are numbered in non-decreasing order of the ratios v_j/u_j, $j \in N$; see Kellerer and Strusevich (2010). In our case, the required numbering is guaranteed by (5), so that the objective function $Z(\mathbf{x})$ as given in (9) is convex and Tamir's algorithm is applicable. Thus, a lower bound $LB = Z(\mathbf{x}^C)$ on the value $Z(\mathbf{x}^*)$ can be found in $O\left(n^2\right)$ time.

To obtain an upper bound, we perform an appropriate rounding of the fractional components of vector $\mathbf{x}^C$. A simple rounding algorithm is described below.

Algorithm Round

Step 1. Given a vector $\mathbf{x}^C = (x_1^C, \ldots, x_n^C)$, $0 \le x_j^C \le 1$, a solution to the continuous relaxation of the problem of minimizing (9), determine the sets $I_1 = \left\{j \in N,\ x_j^C \le \frac{1}{2}\right\}$ and $I_2 = \left\{j \in N,\ x_j^C > \frac{1}{2}\right\}$ and find vector $\mathbf{x}^H = (x_1^H, \ldots, x_n^H)$ with components

$$x_j^H = \begin{cases} 0 & \text{if} \quad j \in I_1 \\ 1 & \text{if} \quad j \in I_2 \end{cases}.$$

Step 2. Output vector $\mathbf{x}^H = (x_1^H, \ldots, x_n^H)$ as a heuristic solution to the problem of minimizing function (9), and therefore function (8).

The running time of Algorithm Round is $O(n)$. Clearly, the inequalities $Z(\mathbf{x}^C) \le Z(\mathbf{x}^*) \le Z(\mathbf{x}^H)$ hold, i.e., we may take $Z(\mathbf{x}^H)$ as an upper bound UB on the optimal value $Z(\mathbf{x}^*)$. We now estimate the ratio $\gamma = UB/LB = Z(\mathbf{x}^H)/Z(\mathbf{x}^C)$.

Theorem 4 *Let $\mathbf{x}^C$ be an optimal solution of the continuous relaxation of the problem of minimizing function $Z(\mathbf{x})$ of the form (9), and $\mathbf{x}^H$ be a vector found by Algorithm Round. Then*

$$\gamma = \frac{Z(\mathbf{x}^H)}{Z(\mathbf{x}^C)} \le 4.$$

Proof: For a vector $\mathbf{x}^C$, let I_1 and I_2 be the index sets found in Step 2 of Algorithm Round.

For a vector $\mathbf{x} = (x_1, \ldots, x_n)$, where $0 \le x_j \le 1$, using the representation (8) define

$$\begin{aligned}
Z_1(\mathbf{x}) &= (\lambda+1) \sum_{\substack{1\le i<j\le n \\ i,j\in I_1}} q_i w_j x_i x_j + \sigma \sum_{\substack{1\le i<j\le n \\ i,j\in I_1}} q_i w_j (1-x_i)(1-x_j); \\
Z_2(\mathbf{x}) &= (\lambda+1) \sum_{\substack{1\le i<j\le n \\ i\in I_1, j\in I_2}} q_i w_j x_i x_j + \sigma \sum_{\substack{1\le i<j\le n \\ i\in I_1, j\in I_2}} q_i w_j (1-x_i)(1-x_j); \\
Z_3(\mathbf{x}) &= (\lambda+1) \sum_{\substack{1\le i<j\le n \\ i\in I_2, j\in I_1}} q_i w_j x_i x_j + \sigma \sum_{\substack{1\le i<j\le n \\ i\in I_2, j\in I_1}} q_i w_j (1-x_i)(1-x_j); \\
Z_4(\mathbf{x}) &= (\lambda+1) \sum_{\substack{1\le i<j\le n \\ i,j\in I_2}} q_i w_j x_i x_j + \sigma \sum_{\substack{1\le i<j\le n \\ i,j\in I_2}} q_i w_j (1-x_i)(1-x_j); \\
Z_5(\mathbf{x}) &= (\lambda+1) \sum_{j\in I_1} p_j x_j + \sigma \sum_{j\in I_1} p_j (1-x_j); \\
Z_6(\mathbf{x}) &= (\lambda+1) \sum_{j\in I_2} p_j x_j + \sigma \sum_{j\in I_2} p_j (1-x_j).
\end{aligned}$$

By the rounding conditions in Step 2 of Algorithm Round, we derive

$$Z_2(\mathbf{x}^H) = Z_3(\mathbf{x}^H) = 0,$$

while

$$\begin{aligned}
Z_1(\mathbf{x}^H) &= \sigma \sum_{\substack{1\le i<j\le n \\ i,j\in I_1}} q_i w_j; & Z_1(\mathbf{x}^C) &\ge \frac{\sigma}{4} \sum_{\substack{1\le i<j\le n \\ i,j\in I_1}} q_i w_j; \\
Z_4(\mathbf{x}^H) &= (\lambda+1) \sum_{\substack{1\le i<j\le n \\ i,j\in I_2}} q_i w_j; & Z_4(\mathbf{x}^C) &\ge \frac{\lambda+1}{4} \sum_{\substack{1\le i<j\le n \\ i,j\in I_2}} q_i w_j; \\
Z_5(\mathbf{x}^H) &= \sigma \sum_{j\in I_1} p_j; & Z_5(\mathbf{x}^C) &\ge \frac{\sigma}{2} \sum_{j\in I_1} p_j; \\
Z_6(\mathbf{x}^H) &= (\lambda+1) \sum_{j\in I_2} p_j; & Z_6(\mathbf{x}^C) &\ge \frac{\lambda+1}{2} \sum_{j\in I_2} p_j.
\end{aligned}$$

Thus, we have that

$$\begin{aligned}
Z(\mathbf{x}^H) &= \sum_{k=1}^{6} Z_k(\mathbf{x}^H) + \mu = Z_1(\mathbf{x}^H) + Z_4(\mathbf{x}^H) + Z_5(\mathbf{x}^H) + Z_6(x^H) + \mu \\
&\le 4Z_1(\mathbf{x}^C) + 4Z_4(\mathbf{x}^C) + 2Z_5(\mathbf{x}^C) + 2Z_6(x^C) + \mu \\
&\le 4\sum_{k=1}^{6} Z_k(\mathbf{x}^C) + 4\mu = 4Z(\mathbf{x}^C),
\end{aligned}$$

as required. ■

It follows immediately from Theorem 4 that for the problem of minimizing function (8) (or, equivalently, function (9)) Theorem 3 is applicable, i.e., the problem admits a γ-FPTAS with $\gamma = 4$. Hence, in terms of the original scheduling problem, we obtain the following statement.

Theorem 5 *Problem* $1\left|p_j\left(\pi;r\right)=p_j\left(1+b_jQ_r\right),MP(\lambda)\right|C_{\max}$ *admits an FPTAS that requires* $O\left(n^2/\varepsilon\right)$ *time.*

Notice that Theorem 5 cannot be improved for problem $1\left|p_j\left(\pi;r\right)=p_j\left(1+b_jQ_r\right),MP(0)\right|C_{\max}$, i.e., for the case of a constant MP duration, since the underlying Boolean programming problem still remains that of minimizing a half-product function.

This is in contrast with the results obtained in Rustogi and Strusevich (2013) for a similar, but simpler problem $1\left|p_j\left(\pi;r\right)=p_j\left(1+bP_r\right),MP(\lambda)\right|C_{\max}$, in which it is additionally assumed that $\sigma=1$, i.e., the MP fully restores the machine back to the default conditions. For problem $1\left|p_j\left(\pi;r\right)=p_j\left(1+bP_r\right),MP(0)\right|C_{\max}$, an FPTAS requires only $O\left(n/\varepsilon\right)$ time, since the underlying Boolean programming problem takes the form of a Subset-Sum problem, with a linear objective function.

For the case of $\lambda>0$, Rustogi and Strusevich (2013) also rely on Theorem 3, but in order to demonstrate that problem $1\left|p_j\left(\pi;r\right)=p_j\left(1+bP_r\right),MP(\lambda)\right|C_{\max}$ with $\sigma=1$ admits a γ-FPTAS, an approximate solution to $1\left|p_j\left(\pi;r\right)=p_j\left(1+bP_r\right),MP(0)\right|C_{\max}$ is used as a lower bound LB, and the ratio UB/LB is bounded by γ that is a linear function of λ. To make Theorem 3 applicable, an additional assumption is made that $\lambda\leq 1$.

The approach described in this paper, based on Algorithm Round and Theorem 4, can also be applied to handling problem $1\left|p_j\left(\pi;r\right)=p_j\left(1+bP_r\right),MP(\lambda)\right|C_{\max}$ with $\lambda>0$ and $\sigma=1$. It will lead to a γ-FPTAS with the running time of $O\left(n^2/\varepsilon\right)$, as in Rustogi and Strusevich (2013), but no additional assumptions regarding the value of λ are needed.

Notice that the results in this section can be extended to handle an enhanced model in which it is assumed that the normal processing time of a job $j\in N$ scheduled after the MP changes from p_j to $\sigma_j p_j$, with a job-dependent factor $\sigma_j>1$, provided that these factors are such that for each pair of jobs i and j the inequality

$$\frac{p_iw_i}{q_i}\leq\frac{p_jw_j}{q_j}$$

implies

$$\frac{\sigma_ip_iw_i}{q_i}\leq\frac{\sigma_jp_jw_j}{q_j}.$$

Similar assumptions are common in the literature on scheduling with rate-modifying maintenance, see. e.g., Lee and Leon (2001) who argue in favour of their practical relevance.

5 Conclusion

The paper introduces a rather general model for scheduling with changing processing times under a cumulative effect. For the problem of minimizing the makespan on a single machine, we adopt Smith's rule to solve the problem in $O\left(n\log n\right)$ time. It follows that the problem with precedence constraints can also be solved in $O\left(n\log n\right)$ time since its objective function is priority-generating. The problem with a rate-modifying maintenance activity, which allows us to (partly) restore the processing conditions of the machine, is linked to a Boolean programming problem with a quadratic objective function, namely the half-product problem. Adapting the results previously known for that problem, we provide an FPTAS that takes

$O\left(n^2/\varepsilon\right)$ time to solve the problem of minimizing the makespan with a single maintenance period.

The next step in studying the models with cumulative deterioration could be a search for approximation algorithms or schemes that would allow us to handle multiple maintenance periods.

References

Agnetis, A., Billaut J.-C., Gawiejnowicz, S., Pacciarelli D., & Soukhal A. (2014). *Multiagent Scheduling. Models and Algorithms. Berlin: Springer.*

Badics, T., & Boros, E. (1998). Minimization of half-products. *Mathematics of Operations Research*, 33, 649–660.

Dolgui, A., Gordon, V. & Strusevich, V. (2012). Single machine scheduling with precedence constraints and positionally dependent processing times. *Computers & Operations Research*, 39 1218–1224.

Erel, E., & Ghosh, J.B. (2008). FPTAS for half-products minimization with scheduling applications. *Discrete Applied Mathematics*, 156, 3046–3056.

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

Gordon, V.S., Potts, C.N., Strusevich, V.A., Whitehead, J.D. (2008). Single machine scheduling models with deterioration and learning: Handling precedence constraints via priority generation. *Journal of Scheduling*, 11, 357–370.

Huang, X., Wang, J.-J. (2015). Machine scheduling problems with a position-dependent deterioration. *Applied Mathematical Modelling*, 39, 2897–2908.

Kellerer, H., & Strusevich, V.A. (2010). Minimizing total weighted earliness-tardiness on a single machine around a small common due date: An FPTAS using quadratic knapsack. *International Journal of Foundations of Computer Science*, 21, 357–383.

Kellerer, H., & Strusevich, V.A. (2012). The symmetric quadratic knapsack problem: approximation and scheduling applications. *4OR*, 10, 111–161.

Kellerer, H., & Strusevich V.A. (2015). Optimizing the Half-Product and related quadratic Boolean functions: Approximation and scheduling applications. Annals of Operations Research, doi:10.1007/s10479-015-2018-y.

Kuo, W.-H., & Yang, D.-L. (2006a). Minimizing the makespan in a single machine scheduling problem with a time-based learning effect. *Information Processing Letters*, 97, 64–67.

Kuo, W.-H., & Yang, D.-L. (2006b). Minimizing the total completion time in a single-machine scheduling problem with a time-dependent learning effect. *European Journal of Operational Research*, 174, 1184-1190.

Lawler, E.L. (1978). Sequencing jobs to minimize total weighted completion time subject to precedence constraints. *Annals of Discrete Mathematics*, 2, 75–90.

Lee, C.-Y., & Leon, V.J. (2001). Machine scheduling with a rate-modifying activity. *European Journal of Operational Research*, 128, 119–128.

Monma, C.L., & Sidney, J.B. (1979). Sequencing with series-parallel precedence constraints. *Mathematics of Operations Research*, 4, 215–234.

Rustogi, K., & Strusevich, V.A. (2012a). Single machine scheduling with general positional deterioration and rate-modifying maintenance. *Omega*, 40, 791-804.

Rustogi, K., & Strusevich, V.A. (2012b). Simple matching vs linear assignment in scheduling models with positional effects: A critical review. *European Journal of Operational Research*, 222, 393–407.

Rustogi, K., & Strusevich, V.A. (2013). Approximation schemes for scheduling on a single machine subject to cumulative deterioration and maintenance. *Journal of Scheduling*, 16, 675–683.

Rustogi, K., & Strusevich, V.A. (2014). Combining time and position dependent effects on a single machine subject to rate-modifying activities. *Omega*, 42, 166–178.

Rustogi, K., & Strusevich, V.A. (2015). Single machine scheduling with time-dependent linear deterioration and rate-modifying maintenance. *Journal of Operational Research Society*, 66, 500–515.

Sahni, S.K. (1976). Algorithms for scheduling independent tasks. *Journal of the Association for Computing Machinery*, 23, 116–127.

Smith, W.E. (1956). Various optimisers for single state production. *Naval Research Logistics Quarterly*, 3, 66–69.

Tamir, A. (1993). A strongly polynomial algorithm for minimum convex separable quadratic cost flow problems on two-terminal series-parallel networks. *Mathematical Programming*, 59, 117–132.

Tanaev, V.S., Gordon, V.S., & Shafransky, Y.M. (1984). *Scheduling Theory. Single-Stage Systems.* Moscow: Nauka (in Russian); translated into English by Kluwer Academic, Dordrecht, 1994.

Valdes, J.R., Tarjan, E. & Lawler, E.L. (1982). The recognition of series-parallel digraphs. *SIAM Journal on Computing*, 11, 361–370.

Wang, J.-B., & Xia, Z.-Q. (2005). Scheduling jobs under decreasing linear deterioration. *Information Processing Letters*, 94, 63–69.

Yin, Y., Xu, D., Sun, K., & Li, H. (2009). Some scheduling problems with general position-dependent and time-dependent learning effects. *Information Sciences*, 179, 2416–2425.