Single-machine scheduling with deteriorating jobs under a series-parallel graph constraint

Ji-Bo Wang[a,b,1] C.T. Ng[b] T.C.E. Cheng[b]

[a]Department of Science, Shenyang Institute of Aeronautical Engineering, Shenyang 110034, People's Republic of China

[b]Department of Logistics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

Abstract

This paper considers single-machine scheduling problems with deteriorating jobs, i.e., jobs whose processing times are an increasing function of their starting times. In addition, the jobs are related by a series-parallel graph. It is shown that for the general linear problem to minimize the makespan, polynomial algorithms exist. It is also shown that for the proportional linear problem to minimize the total weighted completion time, polynomial algorithms exist, too.

Keywords: Scheduling; single machine; deteriorating jobs; series-parallel graph; makespan; total weighted completion time

1 Introduction

In classical scheduling theory it is assumed that the processing time of a job is constant. However, this assumption is invalid for the modelling of many modern industrial processes. There is a growing interest in the literature to study scheduling problems with *deteriorating jobs*, i.e., jobs whose processing times are an increasing function of their starting times. Such deterioration appears, e.g., in scheduling maintenance jobs or cleaning assignments, where any delay in processing a job is penalized and often implies additional time for accomplishing the job. Extensive surveys of different models and problems concerning deteriorating jobs can be found in Alidaee and Womer [1], and Cheng et al. [8].

Most of these studies focus on single-machine settings. Browne and Yechiali [4] considered a single-machine scheduling problem with deteriorating jobs. In this problem, the job processing time is a non-decreasing, start-time dependent linear function. They presented an optimal solution for makespan minimization. Mosheiov [12] considered the problem in which all the jobs are characterized by a common positive basic processing time. Using this basic assumption, Mosheiov

[1] Corresponding author. E-mail addresses: wangjibo75@yahoo.com.cn; lgtctng@polyu.edu.hk; lgtcheng@polyu.edu.hk

proved that the optimal schedule to minimize flowtime is symmetric and has a V-shaped property with respect to the increasing rates of the job processing times. Mosheiov [13] considered the following objective functions: makespan, total flow time, sum of weighted completion times, total lateness, maximum lateness and maximum tardiness, and the number of tardy jobs. When the values of the normal processing time equal zero, all these problems can be solved polynomially. Sundararaghavan and Kunnathur [18] considered the single-machine scheduling problem in which the processing time is a binary function of a common start-time due date. The jobs have processing time penalties for starting after the due date, and the objective is to minimize the sum of weighted completion times. Three special cases of this problem can be solved optimally. Cheng and Ding [7] considered some problems with a decreasing linear model of the job processing times, but with ready time and deadline restrictions. They identified some interesting relationships between the linear models with decreasing and increasing start-time dependent parts. Bachman and Janiak [2] showed that the maximum lateness minimization problem under the linear deterioration assumption is NP-hard, and presented two heuristic algorithms. Bachman et al. [3] considered the problem of minimizing the total weighted completion time introduced by Browne and Yechiali [4]. They proved that the problem is NP-hard. Wu and Lee [20] considered the problem with deteriorating jobs to minimize the makespan on a single machine where the facility has an availability constraint. They showed that the problem can be solved by 0-1 integer programming. Hsu and Lin [10] considered a single-machine problem with deteriorating jobs to minimize the maximum lateness. They designed a branch-and-bound algorithm for deriving exact solutions by incorporating several properties concerning dominance relations and lower bounds.

Chen [6] and Mosheiov [14] considered scheduling linear deteriorating jobs on a group of parallel identical machines. Chen [6] considered minimum flow time and Mosheiov [14] studied makespan minimization. Mosheiov [15] considered makespan minimization and its computational complexity of flow shop, open shop and job shop problems. He introduced a polynomial-time algorithm for the two-machine flow shop and proved its NP-hardness when an arbitrary number of machines (three and above) is assumed. Wang and Xia [19] considered no-wait or no-idle flow shop scheduling problems with job processing times dependent on their starting times. In these problems the job processing time is a simple linear function of a job's starting time and some dominating relationships between machines can be satisfied. They showed that for the problems to minimize makespan or weighted sum of completion times, polynomial algorithms still exist. When the objective is to minimize maximum lateness, the solutions of a classical version may not hold.

In this paper we consider single-machine scheduling problems with deteriorating jobs under a series-parallel precedence constraints. The remaining part of the paper is organized as follows. In next section, a precise formulation of the problem with increasing job processing times is given.

The problems of minimizing the makespan and the total weighted completion time are studied in the third section and the fourth section, respectively. The last section contains some concluding remarks.

2 Basic notation, definition and observation

There are given a single machine and a set $N = \{J_1, J_2, \ldots, J_n\}$ of n independent and non-preemptive jobs. The schedule of the jobs must comply with a series-parallel graph precedence constraint imposed by a given digraph $G = (N, A)$. Each node $J_j \in N$ is identified with a job. Job J_i precedes job J_j if there is a directed path from J_i to J_j in G. Let p_j denote the processing time of job J_j $(j = 1, 2, \ldots, n)$. Throughout the paper, we will consider deteriorating processing times in two forms: general linear $p_j(t) = p_j + \alpha_j t$, and proportional linear $p_j(t) = p_j(a + bt)$, where $p_j > 0, \alpha_j \geq 0, a \geq 0$, $b \geq 0$, and $t \geq 0$ is the job starting-time. For any j, the coefficients of p_j (or ap_j) and α_j (or bp_j) are called the normal processing time and deterioration rate. All the jobs are available for processing at time $t_0 \geq 0$.

For any schedule $\pi = [J_{\pi(1)}, J_{\pi(2)}, \ldots, J_{\pi(n)}]$, where $J_{\pi(j)}$ means the jth job in schedule π, $C_j = C_j(\pi)$ represents the completion time of job J_j. The objective is to find a feasible schedule for which either the makespan $C_{\max}$ is minimized or the total weighted completion time $\sum_{j=1}^{n} w_j C_j$ is minimized. In the remaining part of the paper, all the problems considered will be denoted using the three-field notation scheme $\alpha|\beta|\gamma$ introduced by Graham et al. [9].

3 Makespan problem

First, we need to introduce some notation and terminology; these will be the same as those used by Lawler [11] and Sidney [17] wherever possible.

Definition 1 [11]. The class of *transitive series-parallel graphs* is defined recursively as follows:
1. A graph consisting of a single node, e.g., $G = (\{J_i\}, \emptyset)$, is transitive series-parallel.
2. If $G_1 = (N_1, A_1)$ and $G_2 = (N_2, A_2)$, where $N_1 \cap N_2 = \emptyset$, are transitive series-parallel, then:
(a) The graph

$$G = (N_1 \cup N_2, A_1 \cup A_2 \cup (N_1 \times N_2))$$

is also transitive series-parallel. G is said to be formed by the serial composition of G_1 and G_2.
(b) The graph

$$G = (N_1 \cup N_2, A_1 \cup A_2)$$

is also transitive series-parallel. G is said to be formed by the parallel composition of G_1 and G_2.

A graph G is said to be series-parallel if and only if its transitive closure is transitive series-parallel. Given a series-parallel graph G, it is possible to repeatedly decompose G into series and parallel components, so as to show that the transitive closure of G is obtained by rules 1-2. The result is a rooted binary tree, which Lawer [11] called a decomposition tree, which is a binary tree with the leaves denoting jobs and the internal nodes denoting either a parallel or series composition of the two corresponding subtrees. Parallel and series composition of each internal node are labeled "P" and "S", respectively, where by convention the left son precedes the right son in "S". Figure 2 shows a decomposition tree T for the graph G in Figure 1.

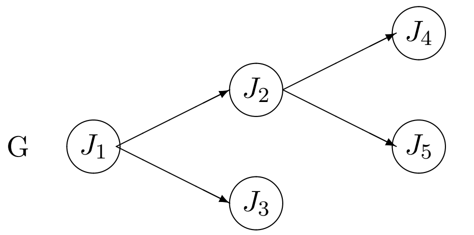

Figure 1 Series-parallel graph.

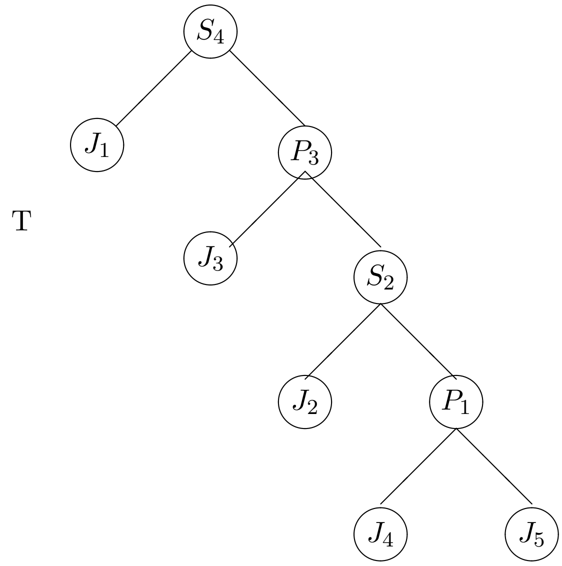

Figure 2. Decomposition tree.

Definition 2 [17]. A non-empty subset $M \subseteq N$ is a (job) module if, for each job $J_j \in N - M$, exactly one of the following three conditions holds:

(a) J_j must precede every job in M,

(b) J_j must follow every job in M,

(c) J_j is not constrained with respect to any job in M.

Definition 3 [17]. Let M be a module. A subset $I \subseteq M$ is an initial set of M, if for each job $J_j \in I$, all the predecessors of J_j in M are also in I.

Suppose that $\pi = [J_{\pi(1)}, J_{\pi(2)}, \ldots, J_{\pi(n)}]$ is any schedule of N and $U = \{J_{\pi(l)}, J_{\pi(l+1)}, \ldots, J_{\pi(m)}\} \subset N$, let

$$\rho(U, \pi) = \frac{\sum_{i=l}^{m}(1 + \alpha_{\pi(i)}) - 1}{\sum_{i=l}^{m} p_i \prod_{j=i+1}^{m}(1 + \alpha_{\pi(j)})}, \tag{1}$$

where $\prod_{j=m+1}^{m}(1+\alpha_{\pi(j)}) := 1$.

Now, we define

$$\rho(U) = \sup_{\pi}\{\rho(U,\pi)\}, \tag{2}$$

the supremum being taken over all feasible schedules of N.

Definition 4 [17]. Let M be a module. An initial set I of M is said to be ρ-maximal for $G = (M, A)$ if $\rho(I) \geq \rho(V)$ for any initial set V in M.

Definition 5 [17]. Let M be a module. An initial set I^* of M is said to be ρ^*-maximal for $G = (M, A)$ if

(a) I^* is ρ-maximal for G;

(b) there is no proper subset $V \subset M$ $(V \neq I^*)$ which is ρ-maximal for G.

Every module M admits at least one ρ-maximal initial set, possibly M itself.

Lemma 1 *[21] For the problem $1|p_j(t) = p_j + \alpha_j t|C_{\max}$, if the sequence is $\pi = [J_1, J_2, \ldots, J_n]$ and the starting time of the first job is $t_0 \geq 0$, then the makespan is*

$$C_{\max}(\pi) = \sum_{i=1}^{n} p_i \prod_{k=i+1}^{n}(1+\alpha_k) + t_0 \prod_{k=1}^{n}(1+\alpha_k),$$

where $\prod_{k=n+1}^{n}(1+\alpha_k) := 1$.

For a given subschedule $\pi = [J_1, J_2, \ldots, J_m]$, where $\{J_1, J_2, \ldots, J_m\}$ is any subset of N, let

$$\rho(\pi) = \rho([J_1, J_2, \ldots, J_m]) = \frac{\sum_{i=1}^{m}(1+\alpha_i) - 1}{\sum_{i=1}^{m} p_i \prod_{j=i+1}^{m}(1+\alpha_j)}.$$

If job J_i must occur before job J_j in every feasible schedule, then we say that job J_i has precedence over job J_j and denote it by $J_i \to J_j$. For the problem $1|chains, p_j + \alpha_j t|C_{\max}$, we consider two chains of the jobs first. One chain, say L_1, consists of the jobs:

$$L_1 : J_1 \to J_2 \to \ldots \to J_k,$$

and the other chain, say chain L_2, consists of the jobs:

$$L_2 : J_{k+1} \to J_{k+2} \to \ldots \to J_n.$$

The next lemma is based on the assumption that if the scheduler decides to start processing jobs of one chain, he has to complete the entire chain before he is allowed to work on the jobs of the other chain.

Lemma 2 *[21] Consider two feasible schedules $\alpha = [U, L_1, L_2, V]$ and $\beta = [U, L_2, L_1, V]$, where U and V are any subsequences. $C_{\max}(\alpha) \leq C_{\max}(\beta)$ if and only if $\rho(L_1) \geq \rho(L_2)$.*

Proof. Let the completion time of the last job in U be t. From Lemma 1, we have

$$C_{\max}([U, L_1, L_2]) = \sum_{i=k+1}^{n} p_i \prod_{l=i+1}^{n} (1+\alpha_l) + \sum_{i=1}^{k} p_i \prod_{l=i+1}^{k} (1+\alpha_l) \prod_{l=k+1}^{n} (1+\alpha_l) + t \prod_{l=1}^{n} (1+\alpha_l),$$

$$C_{\max}([U, L_2, L_1]) = \sum_{i=1}^{k} p_i \prod_{l=i+1}^{k} (1+\alpha_l) + \sum_{i=k+1}^{n} p_i \prod_{l=i+1}^{n} (1+\alpha_l) \prod_{l=1}^{k} (1+\alpha_l) + t \prod_{l=1}^{n} (1+\alpha_l).$$

$$C_{\max}(\alpha) \leq C_{\max}(\beta)$$

if and only if

$$C_{\max}([U, L_1, L_2]) \leq C_{\max}([U, L_2, L_1])$$

if and only if

$$\frac{\sum_{l=1}^{k}(1+\alpha_l) - 1}{\sum_{i=1}^{k} p_i \prod_{l=i+1}^{k}(1+\alpha_l)} \geq \frac{\sum_{l=k+1}^{n}(1+\alpha_l) - 1}{\sum_{i=k+1}^{n} p_i \prod_{l=i+1}^{n}(1+\alpha_l)},$$

i.e.,

$$\rho(L_1) \geq \rho(L_2).$$

This complete the proof. $\square$

Lemma 3 *For any $b > 0, d > 0$ and $k \geq 0$, $\frac{a}{b} > \frac{c}{d}$ if and only if $\frac{a+kc}{b+kd} > \frac{c}{d}$.*

Lemma 4 *For any $b > 0, d > 0$ and $k > 0$, $\frac{a}{b} > \frac{c}{d}$ if and only if $\frac{a}{b} > \frac{kc+a}{kd+b}$.*

Lemma 5 *If $J_1 \longrightarrow J_2 \longrightarrow \ldots \longrightarrow J_u \longrightarrow J_{u+1} \longrightarrow \ldots \longrightarrow J_{l^*}$ and*

$$\rho([J_1, J_2, \ldots, J_{l^*}]) > \rho([J_1, J_2, \ldots, J_u]),$$

then, we can obtain

$$\rho([J_{u+1}, J_{u+2}, \ldots, J_{l^*}]) > \rho([J_1, J_2, \ldots, J_{l^*}]).$$

Proof. From $\rho([J_1, J_2, \ldots, J_{l^*}]) > \rho([J_1, J_2, \ldots, J_u])$, we have

$$\frac{\prod_{i=1}^{l^*}(1+\alpha_i) - 1}{\sum_{j=1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)} > \frac{\prod_{i=1}^{u}(1+\alpha_i) - 1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)}$$

$$\frac{\left(\prod_{i=1}^{u}(1+\alpha_i) - 1\right)\prod_{i=u+1}^{l^*}(1+\alpha_i) + \prod_{i=u+1}^{l^*}(1+\alpha_i) - 1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)\prod_{i=u+1}^{l^*}(1+\alpha_i) + \sum_{j=u+1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)} > \frac{\prod_{i=1}^{u}(1+\alpha_i) - 1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)}$$

From Lemma 3, we have

$$\frac{\prod_{i=u+1}^{l^*}(1+\alpha_i)-1}{\sum_{j=u+1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)} > \frac{\prod_{i=1}^{u}(1+\alpha_i)-1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)}. \tag{3}$$

From (3) and Lemma 4, we have

$$\frac{\prod_{i=u+1}^{l^*}(1+\alpha_i)-1}{\sum_{j=u+1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)} > \frac{(\prod_{i=1}^{u}(1+\alpha_i)-1)\prod_{i=u+1}^{l^*}(1+\alpha_i)+\prod_{i=u+1}^{l^*}(1+\alpha_i)-1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)\prod_{i=u+1}^{l^*}(1+\alpha_i)+\sum_{j=u+1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)},$$

i.e.,

$$\rho([J_{u+1}, J_{u+2}, \ldots, J_{l^*}]) > \rho([J_1, J_2, \ldots, J_{l^*}]).$$

This completes the lemma. □

Lemma 6 *If* $S = [J_1, J_2, \ldots J_u]$, $I^* = [J_{u+1}, J_{u+2}, \ldots, J_v]$, *and*

$$\rho(I^*) > \rho(S \cup I^*),$$

then, we can obtain

$$\rho(I^*) > \rho(S).$$

Proof. From $\rho(I^*) > \rho(S \cup I^*)$, we have

$$\frac{\prod_{i=u+1}^{v}(1+\alpha_i)-1}{\sum_{j=u+1}^{v} p_j \prod_{i=j+1}^{v}(1+\alpha_i)} > \frac{\prod_{i=1}^{v}(1+\alpha_i)-1}{\sum_{j=1}^{v} p_j \prod_{i=j+1}^{v}(1+\alpha_i)}$$

$$\frac{\prod_{i=u+1}^{v}(1+\alpha_i)-1}{\sum_{j=u+1}^{v} p_j \prod_{i=j+1}^{v}(1+\alpha_i)} > \frac{(\prod_{i=1}^{u}(1+\alpha_i)-1)\prod_{i=u+1}^{v}(1+\alpha_i)+\prod_{i=u+1}^{v}(1+\alpha_i)-1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)\prod_{i=u+1}^{v}(1+\alpha_i)+\sum_{j=u+1}^{v} p_j \prod_{i=j+1}^{v}(1+\alpha_i)}$$

From Lemma 4, we have

$$\frac{\prod_{i=u+1}^{v}(1+\alpha_i)-1}{\sum_{j=u+1}^{v} p_j \prod_{i=j+1}^{v}(1+\alpha_i)} > \frac{\prod_{i=1}^{u}(1+\alpha_i)-1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)},$$

i.e.,

$$\rho(I^*) > \rho(S).$$

This completes the lemma. □

An important characteristic of chain L_1 is defined as follows: Let l^* be the smallest integer satisfying

$$\rho^*(L_1) = \frac{\prod_{i=1}^{l^*}(1+\alpha_i)-1}{\sum_{j=1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)} = \max_{1\le s\le k}\left\{\frac{\prod_{i=1}^{s}(1+\alpha_i)-1}{\sum_{j=1}^{s} p_j \prod_{i=j+1}^{s}(1+\alpha_i)}\right\}.$$

The ratio on the left-hand side is called the ρ^*-factor of chain $L_1 : J_1 \to J_2 \to \ldots \to J_k$, which is denoted by $\rho^*(L_1)$. Job J_{l^*} is referred to as the job that determines the ρ^*-factor of the chain (Similar to the concept of ρ^*-factor of a chain in Pinedo [16], page 37). Suppose now that the chain can be interrupted by the jobs of other chains.

Lemma 7 *[21] For the problem $1|chains, p_j+\alpha_j t|C_{\max}$, if job J_{l^*} determines $\rho^*(L_1)$, then there exists an optimal sequence that processes jobs $J_1, J_2, \ldots J_{l^*}$ one after another without any interruption by the jobs of other chains.*

Proof. Similar to the proof of Pinedo [16] (page 37, Lemma 3.1.3). We assume that the theorem is false and show that such an assumption will lead to a contradiction. Here we assume that under the optimal sequence the processing of the subsequence $J_1, J_2, \ldots, J_{l^*}$ is interrupted by a job, say job J_v, from another chain. Let $\pi = [J_1, J_2, \ldots, J_u, J_v, J_{u+1}, \ldots, J_{l^*}]$ be a subsequence of an optimal sequence. It is sufficient to show that either with subsequence $\pi' = [J_v, J_1, J_2, \ldots J_{l^*}]$, or with subsequence $\pi'' = [J_1, J_2, \ldots J_{l^*}, J_v]$, the makespan is less than that with subsequence π. If it is not less than that with the subsequence π', then it has to be less than that with the subsequence π'', and vice versa. From Lemma 2, it follows that if the makespan of π is less than or equal to those with π' and π'', then

$$\frac{\prod_{i=1}^{u}(1+\alpha_i)-1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)} \geq \frac{\alpha_v}{p_v} \geq \frac{\prod_{i=u+1}^{l^*}(1+\alpha_i)-1}{\sum_{j=u+1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)}. \tag{4}$$

Since job J_{l^*} is the job that determines the ρ^*-factor of $I^* : J_1, J_2, \ldots J_{l^*}$, then

$$\frac{\prod_{i=1}^{l^*}(1+\alpha_i)-1}{\sum_{j=1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)} > \frac{\prod_{i=1}^{u}(1+\alpha_i)-1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)}. \tag{5}$$

From (5) and Lemma 5, we can obtain

$$\frac{\prod_{i=u+1}^{l^*}(1+\alpha_i)-1}{\sum_{j=u+1}^{l^*} p_j \prod_{i=j+1}^{l^*}(1+\alpha_i)} > \frac{\prod_{i=1}^{u}(1+\alpha_i)-1}{\sum_{j=1}^{u} p_j \prod_{i=j+1}^{u}(1+\alpha_i)}.$$

It is a contradiction to (4). The same argument can be applied if the interruption of the chain is caused by more than one job. We have proved the theorem. □

Theorem 1 *Let M be a module of $G = (N, A)$ and I^* be a ρ^*-maximal for (M, A), then there exists an optimal schedule for N in which the jobs in I^* precede all the other jobs in M.*

Proof. In order to prove this theorem, let's consider the related network (M, A'), where $A' = A - \{J_i \to J_j | J_i \in I^*, J_j \in M \backslash I^*\}$. Obviously the set of feasible schedules for (M, A') contains the set of feasible schedules for (M, A).

We assume that the theorem is false and show that such an assumption will lead to a contradiction. We assume that $\pi = [S, I^*, T]$ is an optimal schedule for (M, A'), where S and T are disjoint subsets of M with $S \cup T = M \backslash I^*$. Then from Lemma 2, we have

$$\rho(S) \geq \rho(I^*) \geq \rho(T). \tag{6}$$

Obviously, $S \cup I^*$ is initial in (M, A), so from the I^* be a ρ^*-maximal, we have $\rho(I^*) > \rho(S \cup I^*)$. Then from Lemma 6, we have $\rho(I^*) > \rho(S)$. It is a contradiction to (6). This completes the proof. □

Theorem 2 *Let M be a module of $G = (N, A)$ and I^* be a ρ^*-maximal, then I^* is a consecutive subschedule in every optimal schedule for $G = (N, A)$.*

Proof. It is the same as Lemma 7, except that: Here we assume that under the optimal sequence the processing of the subsequence $I^* : J_1, J_2, \ldots, J_{l^*}$ is interrupted by a job, say job J_v, from $M \backslash I^*$, and there is no precedence constraint between J_v and I^*. □

Theorem 3 *Let M be a module of $G = (N, A)$ and σ be an optimal schedule for M. Then there exists an optimal schedule for N that is consistent with σ (i.e., in which the jobs in M appear in the same order as in σ).*

Proof. Similar to the proofs in Lawler [11] and Sidney [17]. □

Hence, from Theorems 1, 2 and 3, we can generalize the method of Lawler [11] and Brucker [5] to the problem $1|sp - graph, p_j + \alpha_j t|C_{\max}$.

To describe the algorithm in more details we need some notations. Let f be an internal node of the decomposition tree, M_f be the union of the two sets M_1 and M_2. Similar to the algorithm of Lawer [11] and Brucker [5], we are going to proceed the algorithm from the bottom of the decomposition tree upward, finding an optimal sequence by using the series composition and parallel composition.

Algorithm 1

1. WHILE there exists an internal node f with two leaves as sons Do
   BEGIN
2. J_i:= leftson(f) J_j:= rightson(f); $M_1 = \{J_i\}, M_2 = \{J_j\}$;
3. IF f has label P THEN
4. $M_f := M_1 \cup M_2$
   Else

5.

5.1 Find $J_i \in M_1$ such that $\rho(J_i) = \min\{\rho(J_k)|J_k \in M_1\}$ and $J_j \in M_2$ such that $\rho(J_j) = \max\{\rho(J_k)|J_k \in M_2\}$. If $\rho(J_i) > \rho(J_j)$, let $M_f = M_1 \cup M_2$ and halt. Otherwise, remove J_i from M_1, J_j from M_2 and form the composite $J_k = (J_i, J_j)$.

5.2

5.2.1 Find $J_i \in M_1$ such that $\rho(J_i) = \min\{\rho(J_k)|J_k \in M_1\}$. If $\rho(J_i) > \rho(J_k)$ ($\rho(J_k)$ is computed by (1)), go to Step 5.3.1.

5.2.2 Remove J_i from M_1 and form the composite job $J_k = (J_i, J_k)$. Return to Step 5.2.1.

5.3

5.3.1 Find $J_j \in M_2$ such that $\rho(J_j) = \max\{\rho(J_k)|J_k \in M_2\}$. If $\rho(J_k) > \rho(J_j)$, let $M_f = M_1 \cup M_2 \cup \{J_k\}$ and halt.

5.3.2 Remove J_j from M_2 and form the composite job $J_k = (J_k, J_j)$. Go to Step 5.2.1.

END {IF}

6. Eliminate J_i and J_j and replace f by a leaf with label M_f.

END {WHILE}

7. Construct π^* by concatenating all the subsequences of the single leaf in non-increasing order of ρ-values.

The following example illustrates the working of Algorithm 1.

Example 1 Consider the problem with a precedence constraint given by graph G in Figure 1, and with normal processing times and deterioration rates as shown in Table 1. $t_0 = 0$.

Table 1. Values of p_j and α_j

jobs	J_1	J_2	J_3	J_4	J_5
p_j	3	4	7	2	5
α_j	0.1	0.2	0.4	0.3	0.5

For P_1, $\rho(J_4) = 3/20 > \rho(J_5) = 1/10$, hence $P_1 : M_1 = \{J_4, J_5\}$. For S_2, $\rho(J_2) = 1/20 < \rho(J_4) = 3/20$, hence J_2 and J_4 form a composite job (J_2, J_4), $\rho(J_2, J_4) = 14/165 < \rho(J_4) = 1/10$, hence, $S_2 : M_2 = \{(J_2, J_4, J_5)\}$. Similarly, $P_3 : M_3 = \{(J_2, J_4, J_5), J_3\}$, $S_4 : M_4 = \{(J_1, J_2, J_4, J_5), J_3\}$. Hence, the optimal sequence is $[J_1, J_2, J_4, J_5, J_3]$ and the optimal value of the makespan is 38.948.

4 Total weighted completion time problem

If $\alpha_j = 0$, the model $p_j(t) = p_j + \alpha_j t$ is the classical model due to Lawler [11]; hence for arbitrary precedence constraints, the problem $1|prec, p_j + \alpha_j t| \sum w_j C_j$ is NP-hard. In this section, we consider a special case, namely $p_j(t) = p_j(a + bt)$.

Similar to Section 3, for the problem $1|sp-graph, p_j(a+bt)|\sum w_jC_j$, we can define the ρ-values as follows:

Suppose that $\pi = [J_{\pi(1)}, J_{\pi(2)}, \ldots, J_{\pi(n)}]$ is any schedule of N and $U = \{J_{\pi(l)}, J_{\pi(l+1)}, \ldots, J_{\pi(m)}\} \subset N$, let

$$\rho(U, \pi) = \frac{\sum_{i=l}^{m} w_{\pi(i)} \prod_{j=l}^{i}(1 + bp_{\pi(j)})}{\prod_{j=l}^{m}(1 + bp_{\pi(j)}) - 1}. \tag{7}$$

Similar to Section 3, the problem $1|sp - graph, p_j(a + bt)|\sum w_jC_j$ can be solved by Algorithm 1 with modified ρ-values given by (7).

The following example illustrates the working of the algorithm for the problem $1|sp-graph, p_j(a+bt)|\sum w_jC_j$.

Example 2 Consider the problem with a precedence constraint given by graph G in Figure 1, and with normal processing times and weights as shown in Table 2. $a = 1, b = 0.1, t_0 = 0$.

Table 2. Values of p_j and w_j

jobs	J_1	J_2	J_3	J_4	J_5
p_j	3	5	7	2	10
w_j	6	3	4	3	5

For P_1, $\rho(J_4) = 12 > \rho(J_5) = 10$, hence $P_1 : M_1 = \{J_4, J_5\}$. For S_2, $\rho(J_2) = 9 < \rho(J_4) = 12$, hence J_2 and J_4 form a composite job (J_2, J_4), $\rho(J_2, J_4) = 99/8 > \rho(J_4) = 12$, hence, $S_2 : M_2 = \{(J_2, J_4), J_5\}$. Similarly, $P_3 : M_3 = \{(J_2, J_4), J_5, J_3\}$, $S_4 : M_4 = \{J_1, (J_2, J_4), J_5, J_3\}$. Hence, the optimal sequence is $[J_1, J_2, J_4, J_5, J_3]$ and the optimal value of the total weighted completion time is 548.94.

5 Conclusions

In this paper we considered the problems of scheduling jobs with start-time increasing processing times (deterioration). The two objectives of the scheduling problems are to minimize the makespan and the total weighted completion time, respectively. Under the series-parallel graph precedence constraint assumption, the problems were proved to be polynomially solvable. In addition, we presented algorithms to solve these problems.

Acknowledgements— The authors are grateful to the one anonymous referee for his/her helpful comments on an earlier version of this paper. This research was supported in part by The Hong

Kong Polytechnic University under the grant number G-YX72. The research of the first author was also partially supposed by the foundation of Shenyang Institute of Aeronautical Engineering under grant number: 05YB08.

References

[1] Alidaee B. Womer N.K. Scheduling with time dependent processing processing times: review and extensions. Journal of the Operational Research Society 50 (1999) 711-720.

[2] Bachman A. Janiak A. Minimizing maximum lateness under linear deterioration. European Journal of Operational Research 126 (2000) 557-566.

[3] Bachman A. Janiak A. Kovalyov M.Y. Minimizing the total weighted completion time of deteriorating jobs. Information Processing Letters 81 (2002) 81-84.

[4] Browne S. Yechiali U. Scheduling deteriorating jobs on a single processor. Operations Research 38 (1990) 495-498.

[5] Brucker P. Scheduling algorithms. Third Edition, Springer 2001.

[6] Chen Z-L. Parallel machine scheduling with time dependent processing times. Discrete Applied Mathematics 70 (1996) 81-94.

[7] Cheng T.C.E. Ding Q. The complexity of scheduling starting time dependent task with release dates. Information Processing Letters 65 (1998) 75-79.

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

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

[10] Hsu Y.S. Lin B.M.T. Minimization of maximum lateness under linear deterioration. Omega 31 (2003) 459-469.

[11] Lawler E.L. Sequencing jobs to minimize total weighted completion time subject to precedence constraints. Annals of Discrete Mathematics 2 (1978) 75-90.

[12] Mosheiov G. V-shaped policies for scheduling deteriorating jobs. Operations Research 39 (1991) 979-991.

[13] Mosheiov G. Scheduling jobs under simple linear deterioration. Computers and Operations Research 21 (1994) 653-659.

[14] Mosheiov G. Multi-machine scheduling with linear deterioration. INFOR 36 (1998) 205-214.

[15] Mosheiov G. Complexity analysis of job-shop scheduling with deteriorating jobs. Discrete Applied Mathematics 117 (2002) 195-209.

[16] Pinedo M. Scheduling: Theory, Algorithms, and Systems. Prentice Hall, 2002.

[17] Sidney J.B. Decomposition algorithms for single-machine sequencing with precedence relations and deferral costs. Operations Research 22 (1975) 283-298.

[18] Sundararaghavan P.S. Kunnathur A.S. Single machine scheduling with start time dependent processing times: some solvable cases. European Journal of Operational Research 78 (1994) 394-403.

[19] Wang J.-B. Xia Z.-Q. Flow shop scheduling with deteriorating jobs under dominating machines. Omega 34 (2006) 327-336.

[20] Wu C.-C. Lee W.-C. Scheduling linear deteriorating jobs to minimize makespan with an availability constraint on a single machine. Information Processing Letters 87 (2003) 89-93.

[21] Zhao C. Algorithms for some new classes of scheduling problems. PhD thesis, Northeastern University, China 2003. (in Chinese)