International Journal of Pure and Applied Mathematics
Volume 13 No. 4 2004, 523-533

SINGLE-MACHINE SCHEDULING TO MINIMIZE MAKESPAN UNDER LINEAR DETERIORATION

Ji-Bo Wang[1 §], Ming-Zheng Wang[2], Zun-Quan Xia[3]
[1,2,3]Department of Applied Mathematics
Dalian University of Technology
Dalian 116024, P.R. CHINA
[1]e-mail: wjb7575@sina.com

Abstract: This paper discusses the scheduling problem under the condition that the job processing time is a linear deterioration function of their start time. The makespan problems on the simple linear deterioration and on the general linear deterioration are considered respectively. The makespan problem on the simple linear deterioration is proved to be polynomial time solvable in this paper. The makespan problem on the general linear deterioration is, generally, not polynomial time solvable. In this paper, it is proved that there are two special cases which are polynomial time solvable.

AMS Subject Classification: 90B35
Key Words: scheduling, single machine, linear deterioration, makespan

1. Introduction

Job processing times of jobs are constant in the classical scheduling theory. In practice, however, we often encounter settings in which processing time increases over time. Such as, scheduling of emergency medical response teams, fire fighting, scheduling of resources to control epidemics, etc. Browne and

Received: April 8, 2004

§Correspondence author

Yechiali [2] consider a scheduling problem in which the processing times of the jobs are not constant over time. n jobs have to be processed on a single machine to minimize the makespan. Job i is characterized by: 1) a "basic" processing time a_i, the length of time required to complete the job if it were scheduled first, i.e., at $t = 0$, and 2) a parameter b_i that jointly with a_i, determines the job's (actual) processing time at $t > 0$, b_i can be interpreted as the growth rate of the processing time of the job i. Assuming linear deterioration, i.e., the processing time of the job increases linearly with its starting time t, the actual processing time is $p_i(t) = a_i + b_i t$. This problem can be solved optimally by scheduling jobs in an increasing order of a_i/b_i. Mosheiov [4] considers the following objective functions: makespan, total flow time, sum of weighted completion times, total lateness, maximum lateness and maxmum tardiness, and the number of tardy jobs. When the values of the normal processing time equal zero, all these problems can be solved polynomially. Zhang [**?**] considers two scheduling problems of minimizing the makespan, one problem is that processing times is $p_i(t) = \begin{cases} b_i t, & t < T, \\ b_i T, & t \geq T, \end{cases}$ for the job i, where T denotes the common critical start time for all jobs in a given schedule. This problem can be solved optimally. The other problem is that processing times is $p_i(t) = \begin{cases} a_i + b_i t, & t < T, \\ a_i + b_i T, & t \geq T, \end{cases}$ for the job i, two special cases of this problem can be solved optimally. Sundararaghavan and Kunnathur [5] consider 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, the objective is to minimize the sum of the weighted completion times. Three special cases of this problem can be solved optimally. Cheng and Ding [3] consider the single machine scheduling problem, in which the processing time is $p_i(t) = \begin{cases} a_i, & t \leq d, \\ a_i + b_i, & t > d, \end{cases}$ for the job i, where d is a common deterioration date for all job, b_i is an extra amount of processing time if the job$i$ scheduled after d. They show that the makespan problem and the flow time problem are NP-complete. An extensive survey of different models and problems concerning start time dependent job processing times can also be found in [1].

In this paper, we consider two scheduling problems with deteriorating jobs to minimize makespan on a single machine. The processing time of each job is given by an increasing linear function of its execution start time. The rest of the paper is organized as follows. In Section 2, we prove that the problem of minimizing the makespan of n simple linear deteriorating jobs can be solved polynomially. In Section 3, we analyze the general linear deterioration case of minimizing the makespan of n deteriorating jobs we conjecture this problem is NP-hard, and leave it as an open problem. For two special cases, we prove this problem can be solved polynomially. In Section 4, some concluding remarks are given.

2. Simple Linear Deterioration

There are a single machine and a set $J = \{1, 2, \ldots, n\}$ of n independent and non-preemptive jobs. So as to minimize makespan, i.e. $C_{\max} = \max\{C_j | j = 1, 2, \ldots, n\}$, where C_j denotes the completion time of the job j. All jobs is available for processing at time $t_0 > 0$. $p_i(t)$ is the piecewise simple linear processing time as defined below:

$$p_i(t) = \begin{cases} b_i T, & t < T, \\ b_i t, & t \geq T, \end{cases} \tag{2.1}$$

where b_i is the growth rate of job i and $t \geq t_0$ is the starting time of job i. T denotes the common critical start time for all jobs in a given schedule, using the three field notation $\alpha|\beta|\gamma$ of scheduling, this problem can be denoted as $1|p_i(t) = b_i T, t < T; p_i(t) = b_i t, t \geq T|C_{\max}$. Let $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(n))$ be a permutation of $(1, 2, \ldots, n)$, where $\sigma(i) = j$ means the job j is the i-th one to be processed. Let $C_{\sigma(k)} = \sum_{i=1}^{k} p_{\sigma(i)}$ denote the completion time of the k-th job in σ. Then one has that

$$\begin{cases} C_{\sigma(1)} = t_0 + b_{\sigma(1)} T, \\ C_{\sigma(2)} = t_0 + b_{\sigma(1)} T + b_{\sigma(2)} T = t_0 + (b_{\sigma(1)} + b_{\sigma(2)}) T, \\ \cdots\cdots \\ C_{\sigma(k)} = t_0 + T \sum_{i=1}^{k} b_{\sigma(i)} \quad (t_0 + T \sum_{i=1}^{k-1} b_{\sigma(i)} < T) \\ C_{\sigma(k+1)} = t_0 + T \sum_{i=1}^{k} b_{\sigma(i)} + b_{\sigma(k+1)} (t_0 + T \sum_{i=1}^{k} b_{\sigma(i)}) \\ \qquad = (t_0 + T \sum_{i=1}^{k} b_{\sigma(i)})(1 + b_{\sigma(k+1)}) \quad (t_0 + T \sum_{i=1}^{k} b_{\sigma(i)} \geq T) \\ \cdots\cdots \\ C_{\sigma(n)} = (t_0 + T \sum_{i=1}^{k} b_{\sigma(i)}) \prod_{i=k+1}^{n} (1 + b_{\sigma(i)}). \end{cases} \tag{2.2}$$

If $C_{\sigma(n)} = t_0 + T \sum_{i=1}^{n} b_{\sigma(i)} \leq T$, then $p_i(t) = b_i T$ is a constant. This is the classical scheduling problem. If $t_0 \geq T$, this is the problem proposed in [4]. So we assume that $t_0 < T$ and $C_{\sigma(n)} = t_0 + T \sum_{i=1}^{n} b_{\sigma(i)} > T$. For the case of convenience, we say that the job $\sigma(k)$ is a critical job, if $C_{\sigma(k-1)} = t_0 + T \sum_{i=1}^{k-1} b_{\sigma(i)} < T$, $C_{\sigma(k)} = t_0 + T \sum_{i=1}^{k} b_{\sigma(i)} \geq T$.

For any permutation $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(n))$, where $\sigma(k)$ is a critical job, we can say from (2) that any permutation of the first $k-1$ jobs cannot change $(k-1)$-th job i.e. $C_{\sigma(k-1)} = t_0 + T\sum_{i=1}^{k-1} b_{\sigma(i)} < T$, so it is not change the completion time of the job $\sigma(n)$. Under the same conditions, any permutation of the last $(n-k)$ jobs cannot change the completion time of the job $\sigma(n)$.

Theorem 2.1. *There is an optimal permutation* $\sigma^* = (\sigma(1), \sigma(2), \ldots, \sigma(n))$*, such that:*

1) $b_{\sigma(1)} \leq b_{\sigma(2)} \leq \ldots \leq b_{\sigma(k-1)}$;

2) $b_{\sigma(k+1)} \leq b_{\sigma(k+2)} \leq \ldots \leq b_{\sigma(n)}$*, where job* $\sigma(k)$ *is a critical job in permutation* σ^*.

From the above analysis we know that the completion time of $\sigma(n)$ can be changed if we interchange the jobs before or after $\sigma(k)$. The below Theorem 2.2 and Theorem 2.3 depict the situation of the completion time by interchanging the job $\sigma(k)$ and job $\sigma(k+1)$, the job $\sigma(k)$ and job $\sigma(k-1)$, respectively.

Theorem 2.2. *Let* $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(n))$ *be a permutation, where* $\sigma(k)$ *is a critical job,* $\tau = (\tau(1), \tau(2), \ldots, \tau(n))$ *is a permutation, which is obtained from the permutation* σ *by interchanging the jobs* $\sigma(k)$ *and* $\sigma(k+1)$*:*

1) If $b_{\sigma(k)} \geq b_{\sigma(k+1)}$*, then* $C_{\sigma(k+1)}(\sigma) \geq C_{\tau(k+1)}(\tau)$*;*

2) If $b_{\sigma(k)} \leq b_{\sigma(k+1)}$*, then* $C_{\sigma(k+1)}(\sigma) \leq C_{\tau(k+1)}(\tau)$*.*

Proof. According to (2.2), one has that

$$C_{\sigma(k+1)}(\sigma) \;=\; (t_0 + T\sum_{i=1}^{k} b_{\sigma(i)})(1 + b_{\sigma(k+1)}) \quad (t_0 + T\sum_{i=1}^{k} b_{\sigma(i)} \geq T)$$

1) Now, we have two cases to prove.

(a) If $t_0 + T\sum_{i=1}^{k} b_{\tau(i)} < T$, then one has that:

$$C_{\tau(k+1)}(\tau) = t_0 + T\sum_{i=1}^{k+1} b_{\tau(i)} = t_0 + T\sum_{i=1}^{k+1} b_{\sigma(i)},$$

$$C_{\sigma(k+1)}(\sigma) - C_{\tau(k+1)}(\tau) = (t_0 + T\sum_{i=1}^{k} b_{\sigma(i)})(1 + b_{\sigma(k+1)}) - (t_0 + T\sum_{i=1}^{k+1} b_{\sigma(i)})$$

$$= b_{\sigma(k+1)}t_0 + Tb_{\sigma(k+1)}(\sum_{i=1}^{k} b_{\sigma(i)} - 1) \geq b_{\sigma(k+1)}t_0 + (-b_{\sigma(k+1)}t_0) = 0.$$

(b) If $t_0 + T\sum_{i=1}^{k} b_{\tau(i)} \geq T$, then one has that

$$C_{\tau(k+1)}(\tau) = (t_0 + T\sum_{i=1}^{k} b_{\tau(i)})(1 + b_{\tau(k+1)})$$

$$= (t_0 + T\sum_{i=1}^{k-1} b_{\sigma(i)} + b_{\sigma(k+1)})(1 + b_{\sigma(k)}),$$

$$C_{\sigma(k+1)}(\sigma) - C_{\tau(k+1)}(\tau) = (t_0 + T\sum_{i=1}^{k} b_{\sigma(i)})(1 + b_{\sigma(k+1)})$$

$$- (t_0 + T\sum_{i=1}^{k-1} b_{\sigma(i)} + b_{\sigma(k+1)})(1 + b_{\sigma(k)})$$

$$= (b_{\sigma(k)} - b_{\sigma(k+1)})(T - T\sum_{i=1}^{k-1} b_{\sigma(i)} - t_0) \geq 0.$$

2) Since $b_{\sigma(k)} \leq b_{\sigma(k+1)}$, hence $t_0 + T\sum_{i=1}^{k} b_{\tau(i)} \geq T$, the last proof is the same as 1). □

Theorem 2.3. *Let $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(n))$ be a permutation, where $\sigma(k)$ is a critical job, $\tau = (\tau(1), \tau(2), \ldots, \tau(n))$ is a permutation, which is obtained from the permutation σ by interchanging the jobs $\sigma(k)$ and $\sigma(k-1)$:*

1) If $b_{\sigma(k)} \leq b_{\sigma(k-1)}$, then $C_{\sigma(k)}(\sigma) = C_{\tau(k)}(\tau)$;

2) If $b_{\sigma(k)} \geq b_{\sigma(k-1)}$, then $C_{\sigma(k)}(\sigma) \leq C_{\tau(k)}(\tau)$.

Proof. According to (2.2), one has that

$$C_{\sigma(k)}(\sigma) = t_0 + T\sum_{i=1}^{k} b_{\sigma(i)}.$$

1) Since $b_{\sigma(k)} \leq b_{\sigma(k-1)}$, one has that $t_0 + T\sum_{i=1}^{k-1} b_{\tau(i)} < T$, according to (2.2), one has that $C_{\tau(k)}(\tau) = t_0 + T\sum_{i=1}^{k} b_{\sigma(i)} = C_{\sigma(k)}(\sigma)$;

2) Now, we have two cases to prove.

(a) If $t_0+T\sum_{i=1}^{k-1} b_{\tau(i)} < T$, it is the same as 1), we have $C_{\sigma(k)}(\sigma) = C_{\tau(k)}(\tau)$.

(b) If $t_0 + T\sum_{i=1}^{k-1} b_{\tau(i)} \geq T$, one has that

$$C_{\tau(k)}(\tau) = (t_0 + T\sum_{i=1}^{k-1} b_{\tau(i)})(1 + b_{\tau(k)}),$$

$$\begin{aligned} C_{\tau(k)}(\tau) - C_{\sigma(k)}(\sigma) &= (t_0 + T\sum_{i=1}^{k-1} b_{\tau(i)})(1 + b_{\tau(k)}) - (t_0 + T\sum_{i=1}^{k} b_{\sigma(i)}) \\ &= Tb_{\tau(k-1)} + b_{\tau(k)}(t_0 + T\sum_{i=1}^{k-1} b_{\tau(i)}) - Tb_{\sigma(k-1)} - Tb_{\sigma(k)} \\ &= b_{\sigma(k-1)}(t_0 + T\sum_{i=1}^{k-1} b_{\tau(i)} - T) \geq 0. \quad \square \end{aligned}$$

From the Theorem 2.2 and Theorem 2.3, the following theorem can be easily obtained.

Theorem 2.4. *The scheduling obtained by the non-decreasing order of* b_i *is optimal for the problem* $1|p_i(t) = b_iT, t < T; p_i(t) = b_it, t \geq T|C_{\max}$.

3. General Linear Deterioration

Schedule a set of n jobs in a single machine. So as to minimize makespan, all jobs is available for processing at time $t_0 \geq 0$, and $p_i(t)$ is the processing time as defined below:

$$p_i(t) = \begin{cases} a_i + b_iT, & t < T, \\ a_i + b_it, & t \geq T, \end{cases} \tag{3.1}$$

where $a_i > 0$ denotes the normal processing time of job i and b_i denotes its growth rate.

Adapting the three field notation $\alpha|\beta|\gamma$ for scheduling problem, we also denote the above problem as $1|p_i(t) = a_i + b_iT, t < T; p_i(t) = a_i + b_it, t \geq T|C_{\max}$. Obviously, if $T = 0$, then it is the model studied in Browne and

Yechiali [2]. If $\sum_{i=1}^{n} a_i + b_i T \leq T$, it is the classical scheduling problem. Let $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(n))$ be a permutation. Let us compute the completion time $C_{\sigma(k)}(\sigma)$ and $C_{\sigma(k+1)}(\sigma)$ of the job $\sigma(k)$ and job $\sigma(k+1)$.

If the completion time $C_{\sigma(k-1)}(\sigma)$ is denoted by t, then the completion time of job $\sigma(k)$ is

$$C_{\sigma(k)}(\sigma) = t + p_{\sigma(k)}(t).$$

If $t < T$, then one has that

$$C_{\sigma(k)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T. \tag{3.2}$$

If $C_{\sigma(k)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T < T$, then one has that

$$C_{\sigma(k+1)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T + a_{\sigma(k+1)} + b_{\sigma(k+1)}T. \tag{3.3}$$

If $C_{\sigma(k)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T \geq T$, then one has that

$$C_{\sigma(k+1)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}t + a_{\sigma(k+1)} + b_{\sigma(k+1)}(t + a_{\sigma(k)} + b_{\sigma(k)}t). \tag{3.4}$$

If $t \geq T$, then one has that

$$C_{\sigma(k)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}t, \tag{3.5}$$

$$C_{\sigma(k+1)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}t + a_{\sigma(k+1)} + b_{\sigma(k+1)}(t + a_{\sigma(k)} + b_{\sigma(k)}t). \tag{3.6}$$

Let two permutations $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(n))$ and $\tau = (\tau(1), \tau(2), \ldots, \tau(n))$ be given. The permutation τ is obtained from the permutation σ by interchanging the jobs from the k-th and $(k+1)$-st positions of σ, i.e., $\sigma(k) = \tau(k+1), \sigma(k+1) = \tau(k), \sigma(i) = \tau(i)$ $(i = 1, 2, \ldots, n; j \neq k, j \neq k+1)$.

The following theorem gives out the relation of $C_{\max}(\sigma)$ and $C_{\max}(\tau)$.

Theorem 3.1. *Let the completion time of $\sigma(k-1)(\tau(k-1))$ in permutation $\sigma(\tau)$ be t.*

(1) If $t \geq T$, then one has that:

$$C_{\max}(\sigma) \leq C_{\max}(\tau) \quad \text{iff} \quad \frac{a_{\sigma(k)}}{b_{\sigma(k)}} \leq \frac{a_{\sigma(k+1)}}{b_{\sigma(k+1)}}.$$

(2) If $t < T$, then one has that:

(a) If $C_{\sigma(k)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T < T, C_{\tau(k)}(\tau) = t + a_{\tau(k)} + b_{\tau(k)}T < T$, then one has that

$$C_{\max}(\sigma) = C_{\max}(\tau).$$

(b) If $C_{\sigma(k)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}t \geq T, C_{\tau(k)}(\tau) = t + a_{\tau(k)} + b_{\tau(k)}t \geq T$, then one has that

$$C_{\max}(\sigma) \leq C_{\max}(\tau),$$

when

$$\frac{a_{\sigma(k)}}{b_{\sigma(k)}} \leq \frac{a_{\sigma(k+1)}}{b_{\sigma(k+1)}} \quad \text{and} \quad b_{\sigma(k)} \leq b_{\sigma(k+1)}.$$

(c) If $C_{\sigma(k)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T \geq T, C_{\tau(k)}(\tau) = t + a_{\tau(k)} + b_{\tau(k)}T < T$, then one has that

$$C_{\max}(\sigma) \geq C_{\max}(\tau).$$

Proof. Clearly, $C_{\max}(\sigma) \leq C_{\max}(\tau) (C_{\max}(\tau) \leq C_{\max}(\sigma))$ is equivalent to $C_{\sigma(k+1)}(\sigma) \leq C_{\tau(k+1)}(\tau)$ $(C_{\tau(k+1)}(\tau) \leq C_{\sigma(k+1)}(\sigma))$. So we need only to prove $C_{\sigma(k+1)}(\sigma) \leq C_{\tau(k+1)}(\tau)$ $(C_{\tau(k+1)}(\tau) \leq C_{\sigma(k+1)}(\sigma))$.

(1) One has from (3.6) that

$$C_{\sigma(k+1)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}t + a_{\sigma(k+1)} + b_{\sigma(k+1)}(t + a_{\sigma(k)} + b_{\sigma(k)}t),$$

$$\begin{aligned} C_{\tau(k+1)}(\tau) &= t + a_{\tau(k)} + b_{\tau(k)}t + a_{\tau(k+1)} + b_{\tau(k+1)}(t + a_{\tau(k)} + b_{\tau(k)}t) \\ &= t + a_{\sigma(k+1)} + b_{\sigma(k+1)}t + a_{\sigma(k)} + b_{\sigma(k)}(t + a_{\sigma(k+1)} + b_{\sigma(k+1)}t), \end{aligned}$$

$$\begin{aligned} C_{\sigma(k+1)}(\sigma) - C_{\tau(k+1)}(\tau) &= b_{\sigma(k+1)}a_{\sigma(k)} - b_{\sigma(k)}a_{\sigma(k+1)} \\ &= b_{\sigma(k)}b_{\sigma(k+1)}\left(\frac{a_{\sigma(k)}}{b_{\sigma(k)}} - \frac{a_{\sigma(k+1)}}{b_{\sigma(k+1)}}\right). \end{aligned}$$

So

$$C_{\max}(\sigma) \leq C_{\max}(\tau) \quad \text{iff} \quad \frac{a_{\sigma(k)}}{b_{\sigma(k)}} \leq \frac{a_{\sigma(k+1)}}{b_{\sigma(k+1)}}.$$

(2) (a) One has from (3.3) that

$$C_{\sigma(k+1)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T + a_{\sigma(k+1)} + b_{\sigma(k+1)}T,$$

$$\begin{aligned} C_{\tau(k+1)}(\tau) &= t + a_{\tau(k)} + b_{\tau(k)}T + a_{\tau(k+1)} + b_{\tau(k+1)}T \\ &= t + a_{\sigma(k+1)} + b_{\sigma(k+1)}T + a_{\sigma(k)} + b_{\sigma(k)}T = C_{\sigma(k+1)}(\sigma). \end{aligned}$$

(b) One has from (3.4) that

$$C_{\sigma(k+1)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T + a_{\sigma(k+1)} + b_{\sigma(k+1)}(t + a_{\sigma(k)} + b_{\sigma(k)}T),$$

$$C_{\tau(k+1)}(\tau) = t + a_{\tau(k)} + b_{\tau(k)}T + a_{\tau(k+1)} + b_{\tau(k+1)}(t + a_{\tau(k)} + b_{\tau(k)}T)$$
$$= t + a_{\sigma(k+1)} + b_{\sigma(k+1)}T + a_{\sigma(k)} + b_{\sigma(k)}(t + a_{\sigma(k+1)} + b_{\sigma(k+1)}T),$$

$$C_{\sigma(k+1)}(\sigma) - C_{\tau(k+1)}(\tau) = (b_{\sigma(k)} - b_{\sigma(k+1)})(T - t) + b_{\sigma(k+1)}a_{\sigma(k)}$$
$$- b_{\sigma(k)}a_{\sigma(k+1)} = (b_{\sigma(k)} - b_{\sigma(k+1)})(T - t)$$
$$+ b_{\sigma(k)}b_{\sigma(k+1)}(\frac{a_{\sigma(k)}}{b_{\sigma(k)}} - \frac{a_{\sigma(k+1)}}{b_{\sigma(k+1)}}).$$

One has that

$$C_{\max}(\sigma) \leq C_{\max}(\tau),$$

when

$$\frac{a_{\sigma(k)}}{b_{\sigma(k)}} \leq \frac{a_{\sigma(k+1)}}{b_{\sigma(k+1)}} \quad \text{and} \quad b_{\sigma(k)} \leq b_{\tau(k)}.$$

(c) In view of (3.3), (3.4), and $C_{\sigma(k)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T \geq T$, one has that

$$C_{\sigma(k+1)}(\sigma) = t + a_{\sigma(k)} + b_{\sigma(k)}T + a_{\sigma(k+1)} + b_{\sigma(k+1)}(t + a_{\sigma(k)} + b_{\sigma(k)}T),$$

$$\begin{aligned} C_{\tau(k+1)}(\tau) &= t + a_{\tau(k)} + b_{\tau(k)}T + a_{\tau(k+1)} + b_{\tau(k+1)}T \\ &= t + a_{\sigma(k+1)} + b_{\sigma(k+1)}T + a_{\sigma(k)} + b_{\sigma(k)}T. \end{aligned}$$

Furthermore, one has that

$$C_{\sigma(k+1)}(\sigma) - C_{\tau(k+1)}(\tau) = b_{\sigma(k+1)}(t + a_{\sigma(k)} + b_{\sigma(k)}T) - b_{\sigma(k+1)}T \geq 0.$$

Hence, one has that $C_{\max}(\sigma) \geq C_{\max}(\tau)$. □

From the Theorem 3.1, the following corollaries can be easily obtained.

Corollary 3.2. *Let $b_i = b, (i = 1, 2, \ldots, n)$, then the schedule obtained by the non-decreasing order of a_i is optimal for the problem $1|p_i(t) = a_i + bT, t < T; p_i(t) = a_i + bt, t \geq T|C_{\max}$.*

Corollary 3.3. *Let $b_i = ka_i, (i = 1, 2, \ldots, n), k > 0$, then the schedule obtained by the non-decreasing order of a_i is optimal for the problem $1|p_i(t) = a_i + ka_iT, t < T; p_i(t) = a_i + ka_it, t \geq T|C_{\max}$.*

The below two examples show that if $a_1 = a_2 = \ldots = a_n = a$, then the schedule obtained by the non-decreasing order of b_i or by the non-increasing order is not optimal for the problem $1|p_i(t) = a + b_iT, t < T; p_i(t) = a + b_it, t \geq T|C_{\max}$.

Example 1. Assume that $n = 2, t_0 = 1, T = 3, a = 5, b_1 = 1, b_2 = 2$. $C_{\max}$ obtained by the sequence (1,2) is 32. The optimal $C_{\max}$ value obtained by the sequence (2,1) is 29.

Example 2. Assume that $n = 2, t_0 = 1, T = 5, a = 5, b_1 = 1, b_2 = 2$. $C_{\max}$ obtained by the sequence (2,1) is 25. The optimal $C_{\max}$ value obtained by the sequence (1,2) is 22.

Based on Theorem 2.4, Corollary 3.1 and Corollary 3.2, the optimal solutions for the problem $1|p_i(t) = b_iT, t < T; p_i(t) = b_it, t \geq T|C_{\max}$, $1|p_i(t) = a_i + bT, t < T; p_i(t) = a_i + bt, t \geq T|C_{\max}$, and $1|p_i(t) = a_i + ka_iT, t < T; p_i(t) = a_i + ka_it, t \geq T|C_{\max}$ can be obtained in $O(n\log n)$ steps by a simple sorting algorithm.

4. Conclusions

In this paper, we investigated the problem of scheduling jobs with start time dependent processing times (deterioration) for the makespan minimization. First the makespan problem on the simple linear deterioration is solved polynomially. Then the makespan problem on the general linear deterioration is considered, we conjecture that this problem is NP-hard, the complexity of this problem remains open, two special cases of this problem are proved to be polynomial time solvable.

Acknowledgements

This research was partially supported by the foundation of Ph.D Units of Chinese Education Ministry (20020141013).

References

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

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

[3] T.C.E. Cheng, Q. Ding, Single machine scheduling with step deterioration processing times, *European Journal of Operational Research,* **134** (2001), 623-630.

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

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

[6] F. Zhang, Scheduling to minimize makespan about increase of processing times, *Appl. Math. J. Chinese Univ. Ser. A.,* **16** (2001), 228-234, In Chinese.