Journal of the Operations Research Society of Japan
Vol. 25, No. 2, June 1982

AN EFFICIENT ALGORITHM FOR A CHANCE-CONSTRAINED SCHEDULING PROBLEM

Hiroshi Kise   Akira Shiomi
Minoru Uno   Da-Sheng Chao
Kyoto Institute of Technology

(Received June 29, 1981; Revised October 30, 1981)

Abstract We consider an n-job one machine scheduling problem in which the processing time of each job i is a random variable subject to a normal distribution $N(m_i, v_i^2)$ and the object is to maximize the weighted number of early jobs subject to the constraint that some specified jobs must be early. It is assumed that $m_p < m_q$ implies $v_p^2 \leq v_q^2$, where m_i and v_i^2 are, respectively, a known mean value and a known variance associated with each job i. If such constraint is relaxed, the problem has been shown to be NP-complete, suggesting strongly that there exists no efficient exact algorithm whatever for the problem. Moreover, it is assumed that if $m_p < m_q$ or $v_p^2 < v_q^2$, then $w_p \geq w_q$, where w_i is a known weight associated with each job i. It is well known for the problem with arbitrary weights to be NP-complete even in the deterministic case (i.e., $v_i^2 = 0$).

We show that the problem with the above assumptions can be solved in $O(n^2)$ time and that it has a practical application.

1. Introduction

We consider a chance-constrained scheduling problem defined as follows:

(i) A single machine processes a set of n jobs, $J = \{1,2,\cdots,n\}$, available at time zero.

(ii) Each job i requires a random processing time p_i subject to an independent normal distribution $N(m_i, v_i^2)$, where m_i and v_i^2 are a known mean value and a known variance, respectively. It is assumed that

(1) $\quad m_p < m_q$ implies $v_p^2 \leqq v_q^2$ for $p,q = 1,2,\cdots,n$.

(iii) Each job i has a known due time d_j. It is assumed without loss of generality that

(2) $d_1 \leqq d_2 \leqq \cdots \leqq d_n$.

(iv) A positive real number α ($\alpha < 1$) is given. A job is said *early* if the probability that the job is completed prior to its due time is not smaller than α. Otherwise, the job is said *tardy*.

(v) A subset Q of J is specified such that all the jobs in Q are required to be early. A schedule is called *feasible* if all the jobs in Q are early.

(vi) A weight w_i is associated with each job i, $(i = 1,2,\cdots,n)$. It is assumed that for $p,q \in J-Q$

(3) $w_p \geqq w_q$ if $m_p < m_q$ or $v_p^2 < v_q^2$.

A feasible schedule is *optimal* if it maximizes the weighted number of early jobs (i.e., minimizes the weighted number of tardy jobs) over the feasible schedules. It is desired to find an optimal schedule.

Moore [9] has solved a special case of this problem in which $Q = \phi$, $w_i = 1$ and $v_i^2 = 0$, $i = 1,2,\cdots,n$, are assumed, i.e., the processing time is not a random variable, but a known value. We refer this problem to Moore's problem. Sidney [10] and Lawler [8] have generalized Moore's problem to cases with non-empty Q and with weights satisfying (3), respectively. These are also special cases of our problem. There have been other generalizations for Moore's problem [3,6]. On the other hand, Karp [5] has shown that if arbitrary weights are imposed, Moore's problem becomes NP-complete, suggesting strongly that there exists no efficient exact algorithm whatever for the problem.

Balut [2] has first discussed the chance-constrained scheduling problem in which m_i and v_i^2, $i = 1,2,\cdots,n$, are arbitrarily given. However, it has been quite recently shown [7] that the algorithm proposed by him does not always give an optimal schedule and the problem is NP-complete. Anyway our problem becomes NP-complete if assumption (1) or (3) is relaxed.

In the following we show some properties for the optimality of the problem and propose an $O(n^2)$ time algorithm for the problem, based on these properties. A practical application of the problem is also given.

2. The j-Optimal Set for the Problem

For a schedule $\pi = (i_1,i_2,\cdots,i_n)$, let c_{i_k} be the completion time of job i_k, where i_k denotes the k-th job to be processed. Then

$$c_{i_k} = \Sigma_{h=1}^{k} p_{i_h},\ k = 1,2,\cdots,n.$$

Moreover, for the schedule π, let

(4) $f_{\{i_1,i_2,\cdots,i_k\}} \equiv \Sigma_{h=1}^{k} m_{i_h} + \beta[\ \Sigma_{h=1}^{k} v_{i_h}^2\ ]^{1/2},\ k = 1,2,\cdots,n,$

where $\beta = \Phi^{-1}(\alpha)$ is defined by $\Phi(\cdot)$, the distribution function of the standard normal distribution $N(0,1)$. Then as shown by Balut [2],

$$P[\ c_{i_k} \leqq f_{\{i_1,i_2,\cdots,i_k\}}\ ] = \alpha$$

holds by the reproduction property of normal law. Thus job i_k is early if and only if

$$f_{\{i_1,i_2,\cdots,i_k\}} \leqq d_{i_k},\ k = 1,2,\cdots,n.$$

Here we may assume without loss of generality that

(5) $m_i + \beta v_i \leqq d_i$ for $\forall i \in J$,

since otherwise, job i can not become early in any schedule. A set of jobs E is called *early* if there is a schedule in which all the jobs in E are early. The set $E = \{i_1,i_2,\cdots,i_k\}$ is early if an only if

$$f_{\{i_1,i_2,\cdots,i_h\}} \leqq d_{i_h},\ h = 1,2,\cdots,k,$$

where $i_1 < i_2 < \cdots < i_k$ is assumed (see (2)). This is an extension of Jackson's Lemma [4]. Thus for our object it can be assumed without loss of generality that any schedule takes the form of $\pi = (E,T)$, where E is a set of early jobs ordered according to nondecreasing order of due times and $T = J-E$ is a set of tardy jobs ordered arbitrarily. Therefore, our scheduling problem becomes to find an early set $E \equiv \{i_1,i_2,\cdots,i_k \mid i_1 < i_2 < \cdots < i_k\}$ which

(6) $$\begin{cases} \text{maximizes } \Sigma_{i\in E}\ w_i \\ \text{subject to } E \supseteq Q \\ \text{such that } f_{\{i_1,i_2,\cdots,i_h\}} \leqq d_{i_h},\ h = 1,2,\cdots,k. \end{cases}$$

We define the same problem for each subset $J_j \equiv \{1,2,\cdots,j\}$, $j = 1, 2,\cdots,n$, as (6). A subset X of J_j is called a *feasible subset* of J_j, if it is early and satisfies $X \supseteq Q \cap J_j$. A feasible subset E_j of J_j is called *j-optimal* if it satisfies

(7) $\Sigma_{i\in E_j}\ w_i > \Sigma_{i\in X}\ w_i$

or

(8) $\Sigma_{i\in E_j}\ w_i = \Sigma_{i\in X}\ w_i$ and $f_{E_j} \leqq f_X$

for any other feasible subset X of J_j. Obviously, $E_1 = \{1\}$ by (5) and $E_n = E$ by $J_n = J$.

Here we introduce a list "$\prec$" of jobs in J-Q representing their relative desirability for inclusion in an early set as follows:

$$(9) \qquad p \prec q, \text{ if } \begin{cases} m_p > m_q \ (v_p \geq v_q \text{ by (1)}, \ w_p \leq w_q \text{ by (3)}), \\ m_p = m_q \text{ and } v_q^2 > v_q^2 \ (w_p \leq w_q \text{ by (3)}), \\ m_p = m_q, \ v_p^2 = v_q^2 \text{ and } w_p < w_q \text{ or} \\ m_p = m_q, \ v_p^2 = v_q^2, \ w_p = w_q \text{ and } p < q. \end{cases}$$

Obviously, $\prec$ uniquely orders the jobs in J-Q. It is a generalization of the list used by Lawler [8] for the deterministic case (i.e., $v_i^2 = 0$, $\forall i \in J$).

Lemma 1. For $p, q \in J - Q$, $p \prec q$ implies that

$$m_p \geq m_q, \ v_p^2 \geq v_q^2 \text{ and } w_p \leq w_q.$$

Proof. Obvious from (9). □

Lemma 2. For a subset U of J and two jobs $p, q \in U$, $p \prec q$ implies that

$$\Sigma_{i \in U-\{p\}} w_i \geq \Sigma_{i \in U-\{q\}} w_i \text{ and } f_{U-\{p\}} \leq f_{U-\{q\}}.$$

Proof. By Lemma 1

$$\Sigma_{i \in U-\{p\}} w_i - \Sigma_{i \in U-\{q\}} w_i = w_q - w_p \geq 0.$$

By (4) and Lemma 1

$$\begin{aligned} f_{U-\{p\}} - f_{U-\{q\}} &= \Sigma_{i \in U-\{p\}} m_i - \Sigma_{i \in U-\{q\}} m_i \\ &\quad + \beta[(\Sigma_{i \in U-\{p\}} v_i^2)^{1/2} - (\Sigma_{i \in U-\{q\}} v_i^2)^{1/2}] \\ &= m_q - m_p + \beta[(v_q^2 - v_p^2) / \{(\Sigma_{i \in U-\{p\}} v_i^2)^{1/2} \\ &\quad + (\Sigma_{i \in U-\{q\}} v_i^2)^{1/2}\}] \leq 0. \quad \square \end{aligned}$$

Theorem 1. For each j-optimal set E_j $(j = 1,2,\cdots,n)$, there is a ℓ-optimal set E_ℓ satisfying

$$(10) \qquad E_\ell \subseteq E_j \cup \{j+1, j+2, \cdots, \ell\} \text{ for } \ell = j+1, j+2, \cdots, n.$$

Proof. The theorem obviously holds for $j = 1$ by (5). Thus it is sufficient to show that we have (10) when

$$(11) \qquad E_\ell \subseteq E_{j-1} \cup \{j, j+1, \cdots, \ell\}$$

is assumed. If $E_j = E_{j-1} \cup \{j\}$, we have (10) by (11). Thus we consider another case when E_j is a proper subset of $E_{j-1} \cup \{j\}$, i.e.,

$$D \equiv E_{j-1} \cup \{j\} - E_j \neq \phi.$$

If E_ℓ and D are disjointed, again we have (10) by

$$E_\ell \subseteq E_{j-1}\cup\{j,j+1,\cdots,\ell\} = E_{j-1}\cup\{j\}\cup\{j+1,\cdots,\ell\} = D\cup E_j\cup\{j+1,\cdots,\ell\}.$$

Thus the remaining case to be considered is that

(12) $C \equiv E_\ell \cap D = E_\ell\cap(E_{j-1}\cup\{j\}-E_j) = E_\ell\cap(E_{j-1}\cup\{j\})-E_\ell\cap E_j \neq \phi.$

In Fig.1 we illustrate the relation among sets defined above. Note that $E_j\cup C$ is not a feasible subset of $E_{j-1}\cup\{j\}$ by the optimality of E_j (see (7), (8)), while

$$(E_j\cap E_\ell)\cup C = (E_j\cup C)\cap(E_\ell\cup C) = (E_j\cup C)\cap E_\ell$$

is a feasible one by the feasibility of E_ℓ. Thus

(13) $B \equiv E_j-E_j\cap E_\ell = (E_j\cup C)-(E_j\cup C)\cap E_\ell \neq \phi$ (see Fig.1).

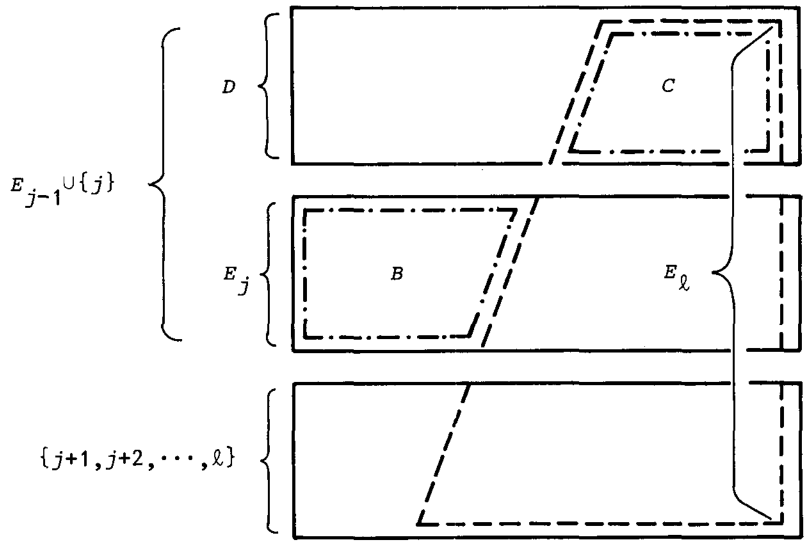

Fig. 1

In short, the subset $E_\ell\cap(E_{j-1}\cup\{j\})$ of E_ℓ is different from the j-optimal set E_j: the former has a non-empty set C, the latter does not have C but another non-empty set B. For (12) and (13), two cases are possible:

(i) For an element $p\in C$, there is an element $q\in B$ such that $m_p = m_q$, $v_p = v_q$ and $w_p = w_q$. Let

(14) $E' \equiv (E_{j-1}\cup\{j\})\cap E_\ell\cup\{q\}-\{p\} = (E_j-B)\cup\{q\}\cup(C-\{p\}),$

i.e., E' is obtained by replacing p with q in $E_\ell \cap (E_{j-1} \cup \{j\})$ and hence has one more common elements with E_j than $E_\ell \cap (E_{j-1} \cup \{j\})$. Obviously it holds that

(15) $f_{E'} = f_{E_\ell \cap (E_{j-1} \cup \{j\})}$ (see (4)).

Moreover, it holds by the feasibility of E_{j-1} (see (11)) and $p \notin Q$ (by the feasibility of E_j) that E' is a feasible subset of $E_{j-1} \cup \{j\}$. For if $j \in E'$, $E'-\{j\}$ is a feasible subset of E_{j-1} by $E'-\{j\} \subseteq E_{j-1}$ and hence E' is a feasible subset of $E_{j-1} \cup \{j\}$ by (15), otherwise $E' (\subseteq E_{j-1})$ is obviously a feasible one. Therefore, we can construct a new ℓ-optimal set from the old one:

(16) $E_\ell \leftarrow \{E_\ell - (E_\ell \cap (E_{j-1} \cup \{j\})\} \cup E'$.

Obviously, the new ℓ-optimal set has one more common elements with E_j than the old one. Thus by repeating (16), we can finally obtain $E_\ell \cap (E_{j-1} \cup \{j\}) = E_j$, i.e., (10) or the next case.

(ii) For an element $p \in C$, any element $q \in B$ is such that $m_p \neq m_q$, $v_p \neq v_q$ or $w_p \neq w_q$. This case does not obviously satisfy (10). However, we show that it contradicts the ℓ-optimality of E_ℓ.

Let p be the minimum element with respect to $\prec$ in C and q the maximum element with respect to $\prec$ in B. Then

(17) $p \prec q$ and $m_p > m_q$, $v_p > v_q$ or $w_p < w_q$.

Since otherwise, it can be easily shown by (9) and Lemma 2 that $E_j \cup \{p\} - \{q\}$ is a feasible subset of $E_{j-1} \cup \{j\}$ and superier to E_j, contradicting the optimality of E_j.

Let

$$\tilde{E} \equiv E_\ell \cup \{q\} - \{p\} = (E_\ell \cap \{j+1,\ j+2, \cdots, \ell\}) \cup E'$$

where E' is obtained by (14). It holds by (17) and Lemma 2 that

$$f_{E'} \leq f_{E_\ell \cap (E_{j-1} \cup \{j\})} \text{ and } \Sigma_{i \in E'} w_i \geq \Sigma_{i \in E_\ell \cap (E_{j-1} \cup \{j\})} w_i,$$

and one of these inequalities strictly holds. Thus it can be shown by the simillar argument as case (i) that E' is a feasible subset of $E_{j-1} \cup \{j\}$. Therefore, we have that $\tilde{E}$ is a feasible subset of $E_{j-1} \cup \{j, j+1, \cdots, \ell\}$ and superier to E_ℓ, contradicting the optimality of E_ℓ. □

Theorem 1 shows that a j-optimal set E_j can be treated as a subset of $E_{j-1} \cup \{j\}$ rather than $J_j = \{1,2,\cdots,j\}$. Utilizing this property, our algorithm sequentially computes E_j, $j = 1,2,\cdots,n$, in this order. Moreover, the above proof suggests that the set $D = E_{j-1} \cup \{j\} - E_j$ may have more than one elements due to the existence of the set Q. We, in fact, see such situation in an example shown in Table 1 (see the next section).

3. Algorithm

Step 1. $j \leftarrow 0$, $E_j \leftarrow \phi$ and $S_j \leftarrow \phi$.

Step 2. If $j = n$, halt. Otherwise, $j \leftarrow j + 1$ and go to Step 3 after letting

$$E_j \leftarrow E_{j-1} \cup \{j\}$$

$$S_j \leftarrow \begin{cases} S_{j-1} & \text{if } j \in Q, \\ S_{j-1} \cup \{j\} & \text{otherwise.} \end{cases}$$

Step 3. If $f_{E_j} \leqq d_j$, return to Step 2. Otherwise, find the minimum element r with respect to $\prec$ in S_j and repeat Step 3 after letting

$$E_j \leftarrow E_j - \{r\} \text{ and } S_j \leftarrow S_j - \{r\}. \quad \square$$

To illustrate the algorithm, we consider an example shown in Table 1.

Table 1 An example with $n = 5$, $\beta = 1$ ($\alpha = 0.84\cdots$) and $Q = \{5\}$

i	1	2	3	4	5
m_i	10	10	8	4	18
v_i^2	5	4	4	1	8
d_i	25	25	25	25	25
w_i	30	30	40	50	10
	1	2	3	4	

Note that it is assumed in this example that job 5 must be early and all the jobs have the same due time 25. Applying the algorithm to this example, the computation proceeds as follows:

1) (Step 1) $j \leftarrow 0$, $E_0 \leftarrow \phi$, $S_0 \leftarrow \phi$.

2) (Step 2) $j \leftarrow 1$, $E_1 \leftarrow \{1\}$, $S_1 \leftarrow \{1\}$.

3) (Step 3) $f_{E_1} \leftarrow 10+\sqrt{5} < 25$.

4) (Step 2) $j \leftarrow 2$, $E_2 \leftarrow \{1,2\}$, $S_2 \leftarrow \{1,2\}$.

5) (Step 3) $f_{E_2} \leftarrow 20+\sqrt{9} \leftarrow 25$.

6) (Step 2) $j \leftarrow 3$, $E_3 \leftarrow \{1,2,3\}$, $S_3 \leftarrow \{1,2,3\}$.

7) (Step 3) $f_{E_3} \leftarrow 28+\sqrt{13} > 25$, $r \leftarrow 1$, $E_3 \leftarrow \{2,3\}$, $S_3 \leftarrow \{2,3\}$.

8) (Step 3) $f_{E_3} \leftarrow 18+\sqrt{8} < 25$.

9) (Step 2) $j \leftarrow 4$, $E_4 \leftarrow \{2,3,4\}$, $S_4 \leftarrow \{2,3,4\}$.

10) (Step 3) $f_{E_4} \leftarrow 22+\sqrt{9} = 25$.

11) (Step 2) $j \leftarrow 5$, $E_5 \leftarrow \{2,3,4,5\}$, $S_5 \leftarrow \{2,3,4\}$.

12) (Step 3) $f_{E_5} \leftarrow 40+\sqrt{17} > 25$, $r \leftarrow 2$, $E_5 \leftarrow \{3,4,5\}$, $S_5 \leftarrow \{3,4\}$.

13) (Step 3) $f_{E_5} \leftarrow 30+\sqrt{9} > 25$, $r \leftarrow 3$, $E_5 \leftarrow \{4,5\}$, $S_5 \leftarrow \{4\}$.

14) (Step 3) $f_{E_5} \leftarrow 22+\sqrt{9} = 25$.

15) (Step 2) $j = 5$. Halt. □

As result, we have an optimal set $E = E_5 = \{4,5\}$ with the objective value $\Sigma_{i\in E} w_i = 60$.

To show the validity of the algorithm, the following two lemmas are necessary.

Lemma 3. Each $E_j (j=1,2,\cdots,n)$ constructed by the algorithm is a feasible subset of $J_j (=\{1,2,\cdots,j\})$.

Proof: E_j is a feasible subset of J_j if E_j is early and $E_j \supseteq Q\cap J_j$. We assume that E_{j-1} constructed by the algorithm is a feasible subset of J_{j-1} $(=\{1,2,\cdots,j-1\})$ and then show that E_j is a feasible subset of J_j.

If $j\in Q$, then j is not included in S_j but E_j in Step 2. Thus j is never excluded from E_j in Step 3. Step 3 repeatedly revises E_j until E_j becomes early. Thus the resultant E_j is a feasible subset of J_j under the above assumption. $E_1 = \{1\}$ is a feasible subset of $J_1 = \{1\}$. Therefore, we have the lemma by induction. □

Lemma 4. E_j $(j=1,2,\cdots,n)$ constructed by the algorithm is j-optimal.

Proof: E_1 is 1-optimal by (5). So we assume that E_{j-1} is $(j-1)$-optimal. Then by Lemma 3 and Theorem 1 it is sufficient to show that

(18) $$\Sigma_{i\in E_j} w_i > \Sigma_{i\in X} w_i$$

or

(19) $$\Sigma_{i\in E_j} w_i = \Sigma_{i\in X} w_i \text{ and } f_{E_j} \leq f_X$$

holds for any other feasible subset X of $E_{j-1} \cup \{j\}$. If $E_j = E_{j-1} \cup \{j\}$, we have (18) or (19). Thus consider another case that $E_{j-1} \cup \{j\}$ is not early. That is,

$$D \equiv E_{j-1} \cup \{j\} - E_j \neq \phi.$$

In this case obviously

$$C \equiv E_{j-1} \cup \{j\} - X \neq \phi.$$

Note that D and C include no element of Q by the feasibility of E_j and X. By Step 3 of the algorithm D has $|D|$ minimal elements with respect to $\prec$ in S_j, where $|\cdot|$ denotes the cardinality of the set therein. Thus let $D \equiv \{r_1, r_2, \cdots, r_{|D|}\}$, where $r_1 \prec r_2 \prec \cdots \prec r_{|D|}$ and $C \equiv \{i_1, i_2, \cdots, i_{|C|}\}$, where $i_1 \prec i_2 \prec \cdots \prec i_{|C|}$, then

(20) $\quad r_k \preceq i_k,\ k = 1,2,\cdots,\ell\ (\ell = \min\{|D|, |C|\}),$

where by $\preceq$ we mean that $r_k \prec i_k$ or $r_k = i_k$. Let

$$Z_k \equiv X \cup \{i_1, i_2, \cdots, i_k\} - \{r_1, r_2, \cdots, r_k\},\ k = 0,1,\cdots,\ell.$$

Note that $Z_0 = X$ and $X \cup \{i_1, i_2, \cdots, i_k\} \supseteq \{r_1, r_2, \cdots, r_k\}$. Then it follows from Lemma 2 and (20) that

$$\Sigma_{i\in X}\, w_i = \Sigma_{i\in Z_0}\, w_i \leqq \Sigma_{i\in Z_1}\, w_i \leqq \cdots \leqq \Sigma_{i\in Z_\ell}\, w_i,$$

$$d_j \geqq f_X = f_{Z_0} \geqq f_{Z_1} \geqq \cdots \geqq f_{Z_\ell}.$$

If $|C| = |D| = \ell$, $Z_\ell = E_j$ and hence we have (18) or (19). If $|C| > |D| = \ell$, then it follows from $X \cup C = E_j \cup D = E_{j-1} \cup \{j\}$ that

$$\begin{aligned} Z_\ell &= X \cup \{i_1, i_2, \cdots, i_\ell\} - D = (E_{j-1} \cup \{j\} - C) \cup \{i_1, i_2, \cdots, i_\ell\} - D \\ &= (E_{j-1} \cup \{j\} - D) - \{i_{\ell+1}, i_{\ell+2}, \cdots, i_{|C|}\} = E_j - \{i_{\ell+1}, i_{\ell+2}, \cdots, i_{|C|}\}. \end{aligned}$$

Thus we have (18), since

$$\Sigma_{h\in E_j}\, w_h > \Sigma_{h\in(E_j - \{i_{\ell+1}, \cdots, i_{|C|}\})}\, w_h \geqq \Sigma_{h\in X}\, w_h.$$

If $\ell = |C| < |D|$, then

$$Z_\ell = X \cup C - \{r_1, r_2, \cdots, r_\ell\} = E_j \cup \{r_{\ell+1}, r_{\ell+2}, \cdots, r_{|D|}\}.$$

Therefore,

$$d_j \geqq f_X \geqq \cdots \geqq f_{Z_\ell} = f_{E_j \cup \{r_{\ell+1}, r_{\ell+2}, \cdots, r_{|D|}\}}.$$

contradicting Step 3 of the algorithm. □

Theorem 2. The proposed algorithm is valid and requires $O(n^2)$ time.

Proof: The validity of the algorithm immediately follows from Lemma 4.

The computational complexity is proved as follows: For each j $(=1,2,\cdots,n)$ Step 2 requires time bounded by a constant value and hence its overall computation takes $O(n)$ time. For each j $(=1,2,\cdots,n)$, let x_j be the number of jobs removed from S_j (or E_j), then Step 3 is repeated x_j times for each j. At each time when a job is removed from S_j, at most $O(n)$ time is required to compute f_{E_j}, the minimum r with respect to $\prec$, E_j and S_j. Thus Step 3 requires at most $O(x_j n)$ time for each j. Its overall computation, therefore, takes $O(\Sigma_{j=1}^{n} x_j n) \leqq O(n^2)$ time by $\Sigma_{j=1}^{n} x_j < n$.

Note that jobs can, if necessary, renumbered according to (2) and the list $\prec$ can be made in $O(n \log n)$ time by using special sorting methods such as heap (e.g. see [1]).

4. An Application

There is a textile factory which performs many manufacturing processes for spinning and weaving. It has a machine shop for the maintenance of machines used in the factory. An important work in the machine shop is to produce various kinds of gears which are frequently requested in manufacturing processes. There is only one machine available for this purpose. Thus it is always busy.

The processing time to make each gear is an uncertain value subject to a normal distribution such that the larger its mean value is, the larger its variance is.

At the beginning of each month, a planner sets up a schedule for making gears the requests of which are in his hand at that time. Some of them must be finished by their due time since they are critical to the overall production schedule for the factory, or have been already demanded in the previous months and not yet finished.

Acknowledgement

The authors are indebted to Associate Prof. T. Ibaraki of Kyoto University for his helpful comments. One of the authors (Chao) would like to appreciate Mr. Wu Bin, the chief engineer of Kweizhou Textile Factory, Kweizhou, The People's Republic of China for his encouragement on this study. The same author also wishes to express his thanks to the Ministry of Education of The People's Republic of China for admitting him to study in Japan.

References

[1] Aho, A., Hopcroft, J. and Ullman, J.: *The Design and Analysis of Computer Algorithm*. Addison-Wesley Company, 1974.

[2] Balut, L. S. J.: Scheduling to Minimize the Number of Late Jobs When Set-Up and Processing Times Are Uncertain. *Management Science*, Vol.19 (1973), 1283-1288.

[3] Ibaraki, T., Kise, H. and Mine, H.: Parallel-Machine Scheduling Problem with Unit Processing Time When Jobs Have Ready and Due Times. *Transaction of IECE of Japan*, Vol.E59 (1976), No.7, 1-6.

[4] Jackson, J. R.: Scheduling a Production Line to Minimize Maximum Tardiness. *Research Report 43, Management Science Research Project, UCLA, Jan. 1955.*

[5] Karp, R. M.: Reducibility among Combinatorial Problem. *Complexity of Computer Computations, R. E. Miller and J. W. Thatcher (eds.)*, Plenum Press, New York, 1972, 85-103.

[6] Kise, H., Ibaraki, T. and Mine, H.: A Solvable Case of the One Machine Scheduling Problem with Ready and Due Times. *Operations Research*, Vol.26 (1978), 121-126.

[7] Kise, H. and Ibaraki, T.: On Balut's Algorithm and NP-Completeness for a Chance-Constrained Scheduling Problem. *Working Paper, Department of Mechanical Engineering, Kyoto Institute of Technology*, 1981, Submitted to Management Science.

[8] Lawler, E. L.: Sequencing to Minimize the Weighted Number of Tardy Jobs. *Revue Française D'Automatique Informatique Recherche Operationalle*, Vol.10 (1976), 27-33.

[9] Moore, J. M.: An n Job, One Machine Sequencing Algorithm for Minimizing the Number of Late Jobs. *Management Science*, Vol.15 (1968), 102-109.

[10] Sidney, J. B.: An Extension of Moore's Due Date Algorithm: *Lecture Notes in Economics and Mathematical Systems, 86 Symposium in the Theory of Scheduling and Its Applications, S. E. Elmaghraby (ed.)*, Springer-Verlag, Berlin, Heiderberg, New York, 1970.

Hiroshi KISE: Department of Mechanical Engineering, Faculty of Engineering, Kyoto Institute of Technology, Matsugasaki, Sakyo-ku, Kyoto, 606, Japan.