A SINGLE MACHINE SCHEDULING PROBLEM WITH FUZZY TIME DELAYS

Xie Yuan, Xie Jian-ying, Deng Xiao-long

Department of Automation,
Shanghai Jiao Tong University,
Shanghai, China

Abstract: An n+1 jobs single machine problem with fuzzy time delays is considered. Fuzzy time delay means the time between the completion of a job and the beginning of any of its predecessors, which must be within prescribed limits. A special fuzzy delays structure is investigated, which time delay only occur between J_i ($i = 1, 2, \ldots, n$) and J_{n+1}. Optimal solutions for the scheduling model under several cases are developed. *Copyright © 2005 IFAC*

Keywords: Scheduling algorithm, Fuzzy modelling, Time delay, Fuzzy set.

1. INTRODUCTION

Consider the following scheduling problem. There are a single machine and a set of n+1 jobs $J_1, J_2, \ldots, J_n, J_{n+1}$ to be run on that machine without preemption. Wikum, *et al.* (1994) showed a special case that job J_{n+1} couldn't be started until all jobs $J_i, i = 1, 2, \ldots, n$ are completed. There are time delays between the completion of J_i and the beginning of J_{n+1}. And the lower and upper bounds for time delay between J_i and J_{n+1} are given by numbers l_i and u_i respectively. They also showed that the problem could only be solved in polynomial time in the following cases

(1) $l_i = 0$ for $i = 1, 2, \ldots, n$;

(2) $u_i = \infty$ for $i = 1, 2, \ldots, n$;

(3) $l_i = 0$, or $u_i = \infty$ for $i = 1, 2, \ldots, n$.

In real world, input data may be uncertain or imprecise. Recently, Muthusamy *et al.* (2003) considered fuzzy version of problem satisfying (2) and proposed an $O(n^8)$ algorithm. They allowed lower bounds for time delays to be fuzzy numbers, and admitted the presence of additional fuzzy precedences. They left for future research derivation of analogous results for the problems of satisfying (1) and (3).

As we know, fuzzy precedence doesn't often happen in practice because it is difficult to decide the membership function of fuzzy precedence. So, a single machine scheduling problem only with fuzzy time delays will be analysed here. And time delays are fuzzy numbers. From three cases mentioned previously, case (1) and case (2) with fuzzy upper bounds and fuzzy low bounds respectively will be researched to find optimal solution in polynomial time. Furthermore, if both upper bounds are not infinite and low bound are not zero, time delays also can be fuzzified. Upper and low bounds can be considered as parameters of fuzzy time delays. The problem will be resolved by genetic algorithm.

The remainder of this paper is organized as follows. Section 2 presents the problem formulation and introduces three kinds of fuzzy time delays. Then, solution procedure for every kind of fuzzy time delays will be given in section 3. Finally, summary and conclusion are presented in last section.

2. FORMULATION OF PROBLEM

2.1 Single Machine Scheduling with Time Delay.

There are n+1 jobs $J_1, \cdots J_n, J_{n+1}$ to be processed nonpreemptively on a single machine, which can execute at most a job at a time. The machine and jobs are continuously available from time $t = 0$. Each job requires an uninterrupted processing time $p_i, i = 1, \cdots, n, n+1$, where all p_i are positive rational numbers. The schedules are required to satisfy the precedence constraints

$J_i \prec J_{n+1}$ for each $i = 1, 2, \ldots, n$

$J_i \prec J_{n+1}$ means that job J_{n+1} cannot be started until job J_i is completed. The lower and upper bounds for time delay between J_i and J_{n+1} are given by numbers l_i and u_i satisfying $0 \le l_i \le u_i < \infty$ for each $i = 1, 2, \ldots, n$. That is, each job J_i must precede job J_{n+1}, and the time between the completion of J_i and the beginning of job J_{n+1} must be at least l_i but no more than u_i. There are ordinary precedence constraints between jobs from $\mathbf{J} = \{J_1, \cdots J_n\}$. The feasibility of schedules is further constrained by time delays of each job J_i from $\mathbf{J}$, which are different in practical situations. Sometime low bounds of time delays are not very important and regarded as zero. Or upper bounds are always satisfied and considered as infinity. But in most of situations, both low and upper bounds are real numbers in the interval $(0, +\infty)$.

2.2 Fuzzy Time Delay.

The fuzzy time delay is associated with each job and represented by a fuzzy set on $\mathbf{R}^+$ (the positive part of real numbers). For our problem, fuzzy time delay can be represented in three kinds of cases which are different to each other. Firstly, when low bound of time delay is zero, the time delay between the completion of J_i from $\mathbf{J}$ and beginning of J_{n+1} must be less than or equal to a given fuzzy number U_i. That is, U_i is a fuzzy upper bound of the time delay between J_i and J_{n+1}. U_i can be formulated as fuzzy interval $[\tilde{u}_i, \underset{\sim}{u}_i]$, here $\tilde{u}_i$ and $\underset{\sim}{u}_i$ are nonnegative and $\tilde{u}_i < \underset{\sim}{u}_i$. The nonincreasing membership function $f_i(x_i)$ is defined as follows:

$$f_i(x_i) = \begin{cases} 1 & \text{if} \quad x_i \le \tilde{u}_i \\ 1 - \dfrac{x_i - \tilde{u}_i}{\underset{\sim}{u}_i - \tilde{u}_i} & \text{if} \quad \tilde{u}_i < x_i < \underset{\sim}{u}_i \\ 0 & \text{if} \quad x_i \ge \underset{\sim}{u}_i \end{cases} \tag{1}$$

Obviously, the crisp time delays u_i can be interpreted as the limit case for $\tilde{u}_i = \underset{\sim}{u}_i = u_i$. Time delay x_i between J_i and J_{n+1} is considered as infeasible whenever $x_i \ge \underset{\sim}{u}_i$; otherwise x_i is feasible, and the degree of satisfaction with the time delay is given by the formulation (1) (see Fig. 1).

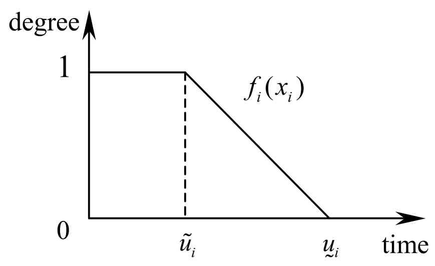

Fig. 1. Membership function with fuzzy upper bound

Secondly, if upper bound of time delay is infinite, the time delay between the completion of J_i from $\mathbf{J}$ and beginning of J_{n+1} must be greater than or equal to the fuzzy low bound L_i (see Fig. 2). $\tilde{l}_i$ and $\underset{\sim}{l}_i$ with

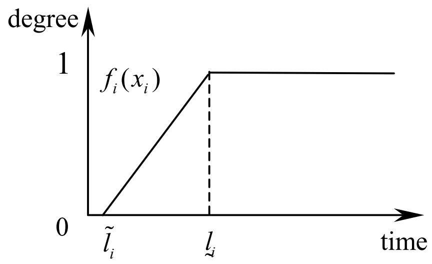

Fig. 2. Membership function with fuzzy low bound

$\tilde{l}_i < \underset{\sim}{l}_i$ are nonnegative and associated with each job J_i. Each L_i is a fuzzy number whose membership function $f_i(x_i): \mathbf{R} \to [0,1]$ is defined by

$$f_i(x_i) = \begin{cases} 0 & \text{if} \quad x_i \le \tilde{l}_i \\ \dfrac{x_i - \tilde{l}_i}{\underset{\sim}{l}_i - \tilde{l}_i} & \text{if} \quad \tilde{l}_i < x_i < \underset{\sim}{l}_i \\ 1 & \text{if} \quad x_i \ge \underset{\sim}{l}_i \end{cases} \tag{2}$$

Thirdly, in most practical situation, upper bounds of time delays are not infinite, and low bounds do not equal to zero, too. A general fuzzy time delay in Fig. 3 may be more appropriate for representing the grade of satisfaction of a decision maker than the linear on Fig. 1 and Fig. 2. The trapezoid membership function in Fig. 3 can be written as

$$f_i(x_i) = \begin{cases} 0 & \text{if} \quad x_i \le \tilde{l}_i \\ \dfrac{x_i - \tilde{l}_i}{\underset{\sim}{l}_i - \tilde{l}_i} & \text{if} \quad \tilde{l}_i < x_i < \underset{\sim}{l}_i \\ 1 & \text{if} \quad \underset{\sim}{l}_i < x_i < \tilde{u}_i \\ 1 - \dfrac{x_i - \tilde{u}_i}{\underset{\sim}{u}_i - \tilde{u}_i} & \text{if} \quad \tilde{u}_i < x_i < \underset{\sim}{u}_i \\ 0 & \text{if} \quad x_i \ge \underset{\sim}{u}_i \end{cases} \tag{3}$$

Where $\tilde{l}_i$ is the earliest low bound of time delay and $\underset{\sim}{l}_i$ the latest low bound. $\tilde{u}_i$ and $\underset{\sim}{u}_i$ can be viewed as the earliest upper bound and latest upper bound of time delay, respectively.

Since the fuzzy time delay of each job represents the satisfaction grade of decision maker, the shape of its membership function should be chosen according to the preference of the decision maker. Different

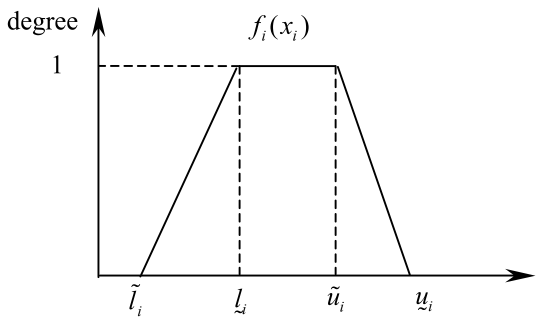

Fig. 3. Membership function of general fuzzy time delay

membership functions may be appropriate for different problems and different jobs. The first and second kind of fuzzy time delays can be considered as fuzzy version of the case (1) and (2) studied by Wikum, *et al.* (1994). The third kind of fuzzy time delay above is a more generalized case than previous two kinds of situations.

2.3 Single Machine Scheduling Problem with Three Kinds of Fuzzy Time Delays

For the single machine scheduling problem with above fuzzy time delays, there are two scheduling criteria. One is to minimize the makespan of schedules. It is defined as the maximum of completion times of all jobs, that is

$$\text{Minimize } C_{\max} = \max\{C_i \mid i=1,2,\ \dots\ ,n+1\} \qquad (4)$$

Obviously job J_{n+1} is final job, and then makespan of a feasible schedule π is the completion time of job J_{n+1}. Thus the first criterion can be rewritten as

$$\text{Minimize } C_{\max}(\pi) = C_{n+1}(\pi) \qquad (5)$$

Another criterion is to maximize the minimum grade of satisfaction of time delays and written as

$$\text{Maximize } f_{\min}=\min\{f_i(\mathrm{x}_i) | i=1,2,\ \dots\ ,n\} \qquad (6)$$

As mentioned in the first part of this section, the practical time delays is the time between the completion of J_i and the beginning of job J_{n+1}. Then for a feasible schedule π, the time delay of job J_i can be calculated as

$$x_i(\pi) = C_{n+1}(\pi) - p_{n+1} - C_i(\pi) \qquad (7)$$

So the problem discussed is to find the optimal schedule σ of n+1 jobs that minimizes the completion time $C_{n+1}(\sigma)$ of J_{n+1} and maximizes the minimum grade of satisfaction $f_{\min}(\sigma)$ with respect to fuzzy time delay in the same time. It can be named problem **P**:

P: Minimize $C_{n+1}(\sigma)$ and maximize $f_{\min}(\sigma)$

Where optimal schedule σ subjects to $\sigma \in \Pi$ that Π is the set of all permutation schedules.

Although the above bi-criteria are common performance of three kinds of fuzzy time delays mentioned before, there are different solution procedures and methods for different kind of fuzzy time delays. For the first kind of fuzzy time delay, which is called fuzzy upper bound, the membership of grade of satisfaction is nonincreasing, that is, the less time delay is, the better grade of satisfaction. From equation (7), the completion time of J_{n+1} will be decreased if time delay is shortened. In other words, C_{n+1} and $f_{\min}$ with problem **P** are consistent for fuzzy upper bound, and one can be optimal if another is optimized. But for fuzzy low bound of time delay, the two criteria are contradictive because the membership of fuzzy time delay is nondecreasing. It means minimum grade of satisfaction $f_{\min}$ will decrease while makespan increase. Therefore, we cannot optimize the two criteria in the same time. For the third kinds of fuzzy time delay, it is more complicated to obtain the optimal sequence. In next section, the solution procedure for problem **P** will be described in detailed for three kinds cases.

3. PROBLEM SOLUTION

3.1 Fuzzy Upper Bound

In this case, the low bound of time delay is not critical and can be zero for each job J_i from $\mathbf{J} = \{J_1, J_2, \dots, J_n\}$. The upper bound of time delay is a fuzzy number and its membership $f_i(x_i)$ is formulated by equation (1) showed in Fig. 1. Time delay x_i between J_i and J_{n+1} is infeasible whenever $x_i > \underset{\sim}{u}_i$, otherwise x_i is feasible. If time delay is less than time $\tilde{u}_i$, there is maximum degree of satisfaction that is one. When time delays beyond $\underset{\sim}{u}_i$, level of satisfaction will decrease to zero. Generally, x_i is in the interval [$\tilde{u}_i, \underset{\sim}{u}_i$]. And $f_i(x_i)$, degree of satisfaction with respect to x_i belongs to [0,1]. In order to maximizing $f_{\min}(\mathrm{x}_i)$, last job J_{n+1} must be processed immediately after completion of all jobs from set $\mathbf{J}$ because it make time delay closer to zero and grade of satisfaction closer to one.

We turn our attention to another criteria, completion C_{n+1} of last job J_{n+1}. Minimizing C_{n+1} is consistent with maximizing the $f_{\min}(\mathrm{x}_i)$. Because earlier the last job is completed, smaller time delay is for every job J_i from **J** according to equation (7); hence greater $f_i(x_i)$ with J_i must be for it is nonincreasing, and bigger $f_{\min}(\mathrm{x}_i)$ will be. So, only one criterion, minimum grade of satisfaction of time delay need to be optimal. And problem **P** is rewritten as **P1**:

P1: maximize $f_{\min}(\sigma)$

Where σ is the optimal schedule for single machine scheduling with fuzzy upper bound of time delay. There are precedence constraints between first *n* jobs. The problem is called 1|prec|$f_{\min}$ that is similar to well-known crisp problem 1|prec|$f_{\max}$. The difference is they have opposite cost function that $f_{\min}$ is to maximize the minimum grade of satisfaction of time delay and $f_{\max}$ to minimize nondecreasing function. Lawler's algorithm (Lawler, 1973) for the problem 1|prec|$f_{\max}$ can be applied directly to the fuzzy case because the membership of fuzzy time delay is nonincreasing. Later on, the algorithm 1 for problem **P1** will be formulated and its validity will be proved.

Obviously job J_{n+1} is final job and there are no idle times during the procedure of jobs processing. Therefore, job J_{n+1} finishes at time

$$C_{n+1}=\sum_{i=1}^{n+1}p_i \tag{8}$$

which have nothing with the precedence of all jobs. Let *I* denote jobs set in which jobs have been scheduled. *I'*, the compensation of *I*, is the set of unscheduled jobs. For those jobs that can be processed at the beginning of sequence with respect to ordinary precedence relation, we denote them by *U*. Here we give the optimal algorithm for problem 1|prec|$f_{\min}$.

Algorithm 1

Step 1. Let *I*=Φ, *I'*={J_1, J_2, ..., J_n}, Let *S*=Φ be the optimal schedule.

Step 2. Let $p=\sum_{J_j\in I}p_j$. Find J^* from set *U* such that

$$f_{J^*}(x)=\max_{J_i\in U}f_{J_i}(x_i) \tag{9}$$

Here, $x_i=C_{n+1}-p_{n+1}-p-p_i$ ($J_i\in U$). Then put J^* into set *I* and delete J^* from *I'* and *U*. Put J^* at the end of *S*, i.e. $S:=S\bigcup J^*$.

Step 3. If *I'*=Φ, go to step 4; Else go to step 2.

Step 4. Put J_{n+1} at the end of *S*

This algorithm is a variant of Lawler algorithm. Even if we employ a straightforward implementation of the Lawler algorithm, the time complexity of the algorithm is $O(n^2)$. We will proof the validity of the algorithm.

Theorem 1. Algorithm 1 for 1|prec|$f_{\min}$ constructs an optimal sequence.

Proof: Enumerate the jobs in such a way that $1,2,\cdots,n$ is the sequence constructed by the algorithm 1. Let $\sigma:\sigma(1),\ldots,\sigma(n)$ be an optimal sequence with $\sigma(i)=i$ for $i=1,2,\ldots,r$ and $\sigma(r+1)\neq r+1$ where r is maximal. We have the following situation

$\sigma:[1,\ldots,r,k,\ldots,j,r+1,\ldots,n]$

It is possible to schedule *r*+1 immediately after *r*. Therefore, *r*+1 and *k* have no predecessor in the set $\{r+1,\ldots,n\}$. This implies $f_{r+1}(x_{r+1})\geq f_k(x_k)$ because $1,2,\cdots,n$ was constructed by the algorithm 1 where

$$x_{r+1}=C_{n+1}-p_{n+1}-p-p_{r+1}\ \ (p=\sum_{i=1}^{r}p_i),$$

$$x_k=C_{n+1}-p_{n+1}-p-p_k\ \ (p=\sum_{i=1}^{r}p_i)$$

Thus, the schedule we get by shifting the block of jobs between *r* and *r*+1 in σ with an amount of p_{r+1} units to the right and processing *r*+1 immediately after *r* is again optimal. This contradicts the maximality of *r*.

3.2 Fuzzy Low Bound

While upper bound of time delay is infinity, the low bound of time delay is fuzzy number with nondecreasing membership function showed in Fig. 2. This kind of case was discussed by Muthusamy *et al.* (2003). They admitted the presence of additional fuzzy precedence that is not considered here. As mentioned in part 2.3, the problem has following two criteria: minimize C_{n+1} and maximize $f_{\min}$. It's called problem **P2**. Since fuzzy low bound of time delay is nondecreasing, the two criteria are contradictive. If completion of last job is minimal, in the meantime, the minimum grade of satisfaction of time delay cannot be maximal. Muthusamy *et al.* (2003) analysed this case with fuzzy precedence appearing. If fuzzy precedence is removed from their problem, the algorithm of Muthusamy *et al.* is same to Tada's algorithm (Tada, 1994) for single machine scheduling with fuzzy due dates. In other words, the problem **P2** can be resolved by Tada's algorithm.

In order to finding the optimal schedules of the problem **P2**, several observations proposed by Muthusamy *et al.* will be repeated in next part.

Observation 1. For every feasible schedule π,

$$C_{n+1}(\pi)=\max\{C_i(\pi)+p_{n+1}+\tilde{l}_i+f_\pi(x_i)(\underline{l}_i-\tilde{l}_i)\} \tag{10}$$

Where $i=1,\ldots n$.

Observation 2. For each optimal schedule, there is no idle time between jobs from $\mathbf{J}=\{J_1,J_2,\ldots,J_n\}$, i.e., all jobs from **J** are processed in the interval $[0,p_1+p_2+\cdots+p_n]$.

For $i=1,\ldots n$, let $\lambda_i:[0,1]\to\mathbf{R}$ be a function defined as follows:

$\lambda_i(\gamma)=\tilde{l}_i+\gamma(\underline{l}_i-\tilde{l}_i)$

From the definition of λ_i that $\lambda_i(\gamma_{ij})=\lambda_i(\gamma_{ij})$ implies

$$\gamma_{ij}=\frac{\tilde{l}_i-\tilde{l}_j}{(\underline{l}_j-\tilde{l}_j)-(\underline{l}_i-\tilde{l}_i)}.$$

Let $\gamma_0=0$ and $\gamma_m=1$, and rank all the quantities $\gamma_{ij}(0<\gamma_{ij}<1)$ in increasing order and create the interval $[\gamma_k,\gamma_{k+1}]$ ($k=0,\ldots,m-1$). All of these intervals follow such observation.

Observation 3. For any γ in the interval $[\gamma_k,\gamma_{k+1}]$, problem **P2** has the same optimal schedule π. If γ passes from interval $[\gamma_k,\gamma_{k+1}]$ to interval $[\gamma_{k+1},\gamma_{k+2}]$, the optimal schedule must be different.

For a permutation π : $\{i=1,\ldots n\}\to\{i=1,\ldots n\}$ and a real number $\gamma\in[0,1]$, the schedule π^γ defined by

I Jobs are processing from time $t=0$,

II There is no idle time between jobs from **J**.

III $C_{n+1}(\pi) = \max\{C_i(\pi) + p_{n+1} + \lambda_i(\gamma)\}$.

Notice that π^{γ} is a feasible schedule such that the processing order of jobs from $\mathbf{J}$ in π^{γ} is π, and the degree $f_{\min}(\pi^{\gamma})$ of satisfaction of fuzzy time delay constraints is γ. Let *I* denote jobs set in which jobs have been scheduled. *I'* is the set of unscheduled jobs. For those jobs that can be processed at the end of sequence with respect to ordinary precedence relation, we denote them by *U*. Now the algorithm 2 for finding the optimal schedule can be described as follows:

Algorithm 2

Step 1. For $1 \le i, j \le n$ with $i \ne j$, Find all $0 \le \gamma \le 1$ such that

$$\gamma_{ij} = \frac{\tilde{l}_i - \tilde{l}_j}{(\underset{\sim}{l}_j - \tilde{l}_j) - (\underset{\sim}{l}_i - \tilde{l}_i)} .$$

Step 2. Arrange all γ_{ij} in increasing order and rename the resulting γ different values so that $0 < \gamma_1 < \gamma_2 < \cdots < \gamma_{m-1} < 1$. Let $\gamma_0 = 0$ and $\gamma_m = 1$. Choose the initial $\gamma = 0$. Let *S* be the set of optimal schedules.

Step 3. Let *I*=Φ, *I'*={*J*$_1$, *J*$_2$,..., *J*$_n$}, $\pi = \Phi$ be the optimal schedule.

Step 4. Find J^* from set *U* such that $\lambda_{J^*}(\gamma) = \min_{i \in U} \lambda_i(\gamma)$ Then put J^* into set *I* and delete J^* from *I'* and *U*. Put J^* at the beginning of π, i.e. $\pi := J^* \cup \pi$.

Step 5. If *I'*=Φ, go to step 6; Else go to step 4.

Step 6. If current $\gamma \ne 1$, add π into set *S* after put J_{n+1} at the end of π. Choose next γ, and go to step 3. Else stop.

Muthusamy *et al.* showed this algorithm is a variant of algorithm designed by Lawler (1973) that can ensure the correctness of it. Same to Lawler's algorithm, the complexity of this one is $o(n^2)$.

3.3 *General Fuzzy Time Delay*

In fact, a more general fuzzy time delay showed in Fig. 3 can more explicitly express the degree of satisfaction of decision maker about actual time delay. In this case, the full satisfaction will be attained in the interval $[\underset{\sim}{l}_i, \tilde{u}_i]$. Earlier or later time delay that leaves this interval will reduce the satisfaction of decision maker about schedules. This kind of fuzzy time delay is the combination of fuzzy upper bound and fuzzy low bound. It is more difficult to deal with than previous kinds of fuzzy times. One cannot get the optimal schedule through just mixing the respective algorithm for fuzzy upper and low bound of time delay. Because the algorithm 1 for fuzzy upper bound queues jobs except J_{n+1} from first one to last, and the algorithm 2 for fuzzy low bound from job last to first. The procedures of two algorithms are opposite and cannot be well done together. So a new method will appear for single machine scheduling problem with general fuzzy time delay. Since it is NP-hard problem when $l_i \ne 0$ or $u_i \ne \infty$ in crisp situation, the fuzzy logic alternative cannot be solved in polynomial time, too. We apply genetic algorithm for this single machine scheduling problem with general fuzzy time delay.

There are two criteria, maximum completion time C_{n+1} and minimum degree of satisfaction with time delay $f_{\min}$, need to be optimal. Hence, the Niched Pareto GA proposed by Horn *et al.* (1994), which is a good multiobjective optimization method, will be employed for this two objective optimization. In the Niched Pareto GA, the optimal solution is called nondominated solution, or Pareto optimal solution. Next the definition of nondominated solution will be given. For any feasible schedule π, schedule vector v^{π} consists two elements, $f^{\pi}_{\min}$ and $C^{\pi}_{\max}$. We denote it as $v^{\pi} = (f^{\pi}_{\min}, C^{\pi}_{\max})$. For two vectors $v^1 = (f^1_{\min}, C^1_{\max})$ and $v^2 = (f^2_{\min}, C^2_{\max})$, we say v^1 dominates v^2 and denote it by $v^1 > v^2$ when $f^1_{\min} \ge f^2_{\min}$, $C^1_{\max} \le C^2_{\max}$ and $v^1 \ne v^2$. If $v^{\pi_1} > v^{\pi_2}$ for two schedules π_1 and π_2, we say π_1 dominates π_2 . A feasible schedule π_1 is called to be nondominated if and only if there exist no feasible schedule that dominates π_1. In order to finding the nondominated solution, the Niched Pareto GA implements a sampling scheme as follows. Two candidates for selection are picked at random from the population. A comparison set of individuals is also picked randomly from the population. Each of the candidates is then compared against each individual in the comparison set. There are two kinds of results:

1. If one candidate is dominated by the comparison set, and the other is not, the latter is selected for reproduction.
2. If both or neither is dominated by the comparison set, then sharing can choose a winner.

Fitness sharing was introduced by Goldberg and Richardson (1987), analyzed in detail by Deb (1989). Sharing calls for the degradation of an individual's objective fitness f_i by a niche count m_i calculated for that individual. But Niched Pareto GA do not implement any form of fitness degradation according to the niche count. Instead, the "best fit" candidate is determined to be that candidate who has the smallest niche count. This type of sharing is called equivalence class sharing. The niche count m_i is an estimate of how crowded is the neighbourhood (niche) of individual *i*. It is calculated over all individuals in the current population:

$$m_i = \sum_{j=1}^{\text{pop\_size}} \text{sh}(d_{ij}) \qquad (11)$$

Where d_{ij} is the distance between individuals *i* and *j* and sh(•) is the sharing function:

$$\mathrm{sh}(d_{ij}) = \begin{cases} 1-\left(\dfrac{d_{ij}}{\sigma_{share}}\right)^{\alpha}, & if \quad d_{ij} < \sigma_{share} \\ 0, & other \end{cases} \tag{12}$$

Here α is a constant and σ_{share} is the niche radius, fixed by the user at some estimate of the minimal separation desired or expected between the goal solutions. So the total procedure of the problem is demonstrated as:

Algorithm 3

Step 1 (Initialization): Randomly generate an initial population of N_{pop} solutions.
Step 2 (Evaluation): Calculate the values of the two objectives for each solution in the current population, and then update the tentative set of nondominated solutions.
Step 3 (Selection): Select a pair of solutions according to Niched Pareto GA method. Repeat this step to produce N_{pop} offspring by crossover operation in step 4.
Step 4 (Crossover): For each selected pair, apply a crossover operation to generate an offspring with the crossover probability P_c.
Step 5 (Mutation): For each solution generated by crossover operation, apply operation with a prespecified mutation probability P_m.
Step 6 (Elitist strategy): Randomly remove N_{elite} solutions from the N_{pop} solutions generated by the above operations, and add the same number of strings from a tentative set of Pareto optimal solution to the current population.
Step 7 (termination test): If a prespecified stepping condition is not satisfied, return to step 2.

The Niched Pareto GA maintains Pareto diversity and shows the final set of Pareto optimal solution to the decision maker. A single solution (i.e. the final solution) is selected by decision maker's preference. Here, in order to cutting the length of the paper, test example will not be provided. But it does not reduce ability that Niched Pareto GA finds nondominated solutions of this problem.

4.CONCLUDING REMARKS

A single machine scheduling problem with fuzzy time delay was discussed here. All feasible schedules must satisfy a given precedence relation with a special structure, in which last job J_{n+1} cannot start until all other jobs J_1, J_2, $\cdots$, J_n are completed. Moreover, the fuzzy delays between the completion time of J_i and the starting time of J_{n+1} are categorized into three kinds: fuzzy upper bound of time delay, fuzzy low bound of time delay and general fuzzy time delay. The objective of scheduling is to maximize minimum degree of satisfaction with fuzzy time delays and minimize maximum completion time of jobs. For different kind of fuzzy time delays, there are different solution procedures. When fuzzy time delay contains upper and low bound, it is general fuzzy time delay. Because the crisp counterpart of problem with general fuzzy time delay is already NP-hard, the fuzzy version is computationally intractable in the sense that it is strongly NP-hard, too. Therefore, some kinds of heuristic search methods are needed for this NP-hard problem. Here, a kind of genetic algorithm, Niched Pareto GA, was applied and it can efficiently provide the set of nondominated solutions for two scheduling objectives. As a matter of fact, the genetic algorithm also fits for fuzzy upper and low bounds of time delay. But it consumes a great deal of time and is not as efficient as modified Lawler algorithm which was discussed in this paper.

In this paper, a kind of simplest single machine scheduling problem with fuzzy time delay was considered. Fuzzy time delay only appeared between last job and each one of other jobs. However, more general situation in which fuzzy time delay occurs between every pair of jobs deserves to research. Otherwise, more than one machine scheduling problems with fuzzy time delay aren't yet analysed seriously. They are good challenges for future work. While their membership functions of three kinds of fuzzy time delays are linear, all of them may be nonlinear. But, fuzzy upper bound must be nonincreasing and fuzzy low bound must be nondecreasing. And for general fuzzy time delay, there are not significantly affected by their shape since genetic algorithm does not restrict the type of fuzzy number. In the future research, other performance functions will be taken into account. And multiobjective scheduling problems more than two are more interesting and intractable. Approximating algorithms such as genetic algorithm will be good solution method for that multiobjective optimization.

REFERENCES

Deb, K. (1989). Genetic algorithms in multimodal function optimization. *MS thesis, TCGA Report No. 89002.* University of Alabama.

Goldberg, D.E. and Richardson, J. J.(1987). Genetic algorithms with sharing for multimodal function optimization. In: *Genetic Algorithms and Their Applications: Proceedings of the Second ICGA* (Grefenstette, J.), 41-49. Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Horn, J., N. Nafpliotis and D. Goldberg (1994). A niched Pareto genetic algorithm for multiobjective optimization. In: *Proceeding of the first IEEE conference on evolutionary computation* (Fogel, D.), 82-87. IEEE Press, New Jersey.

Lawler, E.L. (1973). Optimal sequencing of a single machine subject to precedence constraints. *Management Science*, **19 (5)**, 544-546.

Muthusamy, K., Sung, S.C., Vlach, M., Ishii, H. (2003). Scheduling with fuzzy delays and fuzzy precedences. *Fuzzy Sets and Systems*, **134 (3)**, 387-395.

Wilum, E.D., Llewellyn, D.C., Nemhauser, G.L. (1994), One-machine generalized precedence constrained scheduling problems. Operations Research Letters, 16 (2), 87-99.