The Third AFSS(1998), 640-645

A Sequencing Problem with Fuzzy Preference Relation

Kyung-Mi Lee[a], Takeshi Yamakawa[a], Keon-Myung Lee[b] [1]

[a] School of Computer Science and Systems Engineering, Kyushu Institute of Technology Iizuka, Fukuoka, 820 Japan
[b]Dept. of Computer Science, ChungBuk National University CheongJu, ChungBuk, 361-763 Korea, e-mail: kmlee@cbucc.chungbuk.ac.kr

Abstract
A sequencing problem is one to find an ordered sequence of some entities which maximizes (or minimize) some objective function. This paper introduces a new type of sequencing problems, named *a sequencing problem with fuzzy preference relation*, where a fuzzy preference relation is provided for the evaluation of the quality of sequences. It presents how such a problem can be formulated in the point of objective function. In addition, it proposes a genetic algorithm applicable to such a sequencing problem.

Keywords: sequencing problem, fuzzy preference relation, genetic algorithm

1. Introduction

In everyday life, we often confront with the problems that order some number of entities in a sequence and then process them in the ordered sequence. Depending on the ordered sequences of entities, we get different benefits from their sequenced processing. We call such problems, to find an ordered sequence of some entities which maximizes (or minimize) some objective function, *sequencing problems*. As the typical examples of sequencing problems, there exist traveling salesman problem, single machine job scheduling, job shop scheduling, flow shop scheduling, and so on.[1][3] Although these examples have different evaluation criteria and different kinds of entities to be sequenced, they in nature belong to the class of sequencing problems.

This paper has interest in a new class of sequencing problem where a fuzzy preference relation is provided for the evaluation of the quality of sequences. A fuzzy preference relation contains information about how much preferable an entity is sequenced earlier than another one[6][8][9]. According to the application domains, it is determined how to use the fuzzy preference relation in the quality evaluation of sequences. From now on, such a sequencing problem is referred as *a sequencing problem with fuzzy preference relation.*

In this paper, we present in detail the sequencing problem with fuzzy preference relation and how to use fuzzy preference relation in the quality evaluation of sequences. It is known that sequencing problems belong to the NP-complete problem class[1][3]. A sequencing problem with fuzzy preference relation is also an NP-complete problem since the conventional sequencing problems can be regarded as a special case of the sequencing problems with fuzzy preference relation[3]. Thus, in order to solve such sequencing problems with fuzzy preference relation, we propose a method to apply a genetic algorithm to them.

The paper is organized as follows: Section 2 presents the sequencing problem with fuzzy preference relation and Section 3 reviews the fuzzy preference relation. Section 4 briefly introduces genetic algorithms and Section 5 proposes a genetic algorithm for the sequencing problem. Section 6 presents some experiment results to show the applicability of the proposed method. In final, Section 7 draws the conclusions.

2. A Sequencing Problem with Fuzzy Preference Relation

[1] The research was partially supported by ChungBuk National University Research Foundation, Korea.

A sequencing problem with fuzzy preference relation is described as follows: There are n entities to be sequenced and a fuzzy preference relation is given to represent fuzzy preference degrees for each pair of these entities. Depending on the situations, such fuzzy preference degrees may indicate the degree of preference for which an entity is processed before another one, or may indicate the amount of credit for which an entity precedes another one. The objective of the problem is to find an ordered sequence of entities which satisfies maximally the given objective (e.g., gives the highest preference).

For the convenience's sake, let's use the following notations:

$T = \{1, 2, 3, ..., n\}$: the set of given entities

$P = [fp(i,j)]_{n \times n}$: fuzzy preference relation

$fp(i,j) \in [0,1]$: fuzzy preference degree for the situation that entity i precedes entity j

$S = (t_1, t_2, ..., t_n)$: an ordered sequence of entities ($t_i \in T$, $i = 1, ..., n$)

$C(S)$: the fitness of the sequence S as the optimal solution (i.e., objective function)

In this sequencing problem, various kinds of fitness evaluation methods for sequences can be considered. Those evaluation methods can be classified into two approaches: The first approach is to compute the fitness by taking account of the preference degrees of adjacent entities in the sequence. The following shows some fitness evaluation methods based on the approach:

$$C(S) = \sum_{i=1}^{n-1} fp(i, i+1) \tag{1}$$

$$C(S) = \min_{i=1}^{n-1} fp(i, i+1) \tag{2}$$

$$C(S) = \alpha \min_{i=1}^{n-1} fp(i, i+1) + (1-\alpha) \max_{i=1}^{n-1} fp(i, i+1) \tag{3}$$

The second approach is to evaluate the fitness $c(t_i)$ of each individual entity t_i in a sequence S and to aggregate them in order to obtain the overall fitness $C(S)$ of the sequence S. The fitness of an individual entity can be defined as the contribution of the entity to the others in the sequence. We can consider two sorts of contribution of an entity: following credit and preceding credit. The *following credit* $c_f(t_i)$ of an entity t_i denotes the contribution of the entity to its leading entities $\{ t_1, t_2, ..., t_{i-1}\}$ in a sequence $S = (t_1, t_2, ..., t_{i-1}, t_i, ..., t_n)$. The *preceding credit* $c_p(t_i)$ of an entity t_i denotes the contribution of the entity to its subsequent entities $\{ t_{i+1}, t_{i+2}, ..., t_n\}$ in a sequence $S = (t_1, t_2, ..., t_i, t_{i+1}, ..., t_n)$.

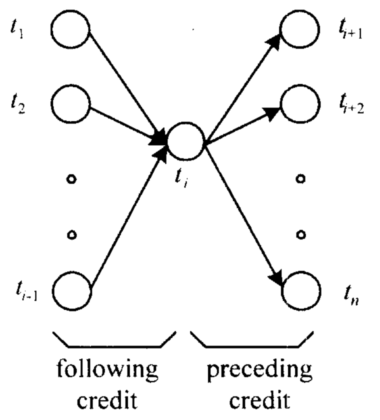

Figure 1. Following credit and preceding credit

The following shows some methods to evaluate the following credit $c_f(t_i)$ of an entity t_i in a sequence $S = (t_1, t_2, ..., t_n)$.

$$c_f(t_i) = \min\{ fp(j,i) \mid j < i\} \tag{4}$$

$$c_f(t_i) = \max\{ fp(j,i) \mid j < i\} \tag{5}$$

$$c_f(t_i) = \alpha\min\{ fp(j,i) \mid j < i\} + (1-\alpha) \max\{ fp(j,i) \mid j < i\} \tag{6}$$

$$c_f(t_i) = \Pi_{j<i} fp(j,i) \tag{7}$$

$$c_f(t_i) = \Sigma_{j=1,i-1} fp(j,i) \tag{8}$$

The next shows some methods to evaluate the preceding credit $c_p(t_i)$ of an entity t_i in a sequence $S = (t_1, t_2, ..., t_n)$.

$$c_p(t_i) = \min\{ fp(i,j) \mid j > i\} \tag{9}$$

$$c_p(t_i) = \max\{ fp(i,j) \mid j > i\} \tag{10}$$

$$c_p(t_i) = \alpha\min\{ fp(i,j) \mid j > i\} + (1-\alpha) \max\{ fp(i,j) \mid j > i\} \tag{11}$$

$$c_p(t_i) = \Pi_{j>i} fp(i,j) \tag{12}$$

$$c_p(t_i) = \Sigma_{j=i+1,n} fp(i,j) \tag{13}$$

The fitness $c(t_i)$ of an individual entity t_i can be defined by using the following credit and the preceding credit as follows:

i) the cases that consider only the following credits

$$c(t_i) = c_f(t_i) \tag{14}$$

ii) the cases that consider only the following credits

$$c(t_i) = c_p(t_i) \tag{15}$$

iii) the cases that consider both

$$c(t_i) = \beta c_f(t_i) + \gamma c_p(t_i) \; (0 \le \beta, \gamma \le 1) \tag{16}$$

The parameters α and γ make it possible to reflect relative importance of both credits.

The following presents some methods to aggregate the fitness $c(t_i)$ of individual entities in order to obtain the overall fitness $C(S)$ of a sequence S.

$$C(S) = \min\{\, c(t_i) \mid i = 1, ..., n\} \quad (17)$$

$$C(S) = \Sigma_i\, c(t_i) \quad (18)$$

$$C(S) = \alpha\max\{c(t_i) \mid i = 1, ..., n\} + (1-\alpha)\min\{c(t_i) \mid i = 1, ..., n\} \quad (19)$$

$$C(S) = \Sigma_i\, w_i c(t_i), \quad w_i\text{: weight factor } (\Sigma_i\, w_i = 1) \quad (20)$$

$$C(S) = OWA(c(t_i), i = 1,.., n; w_j, j = 1, ..., n) \quad (21)$$

OWA: order-weighted aggregation operator[14]

3. Fuzzy Preference Relation

A relation R on a domain $X\times Y$ is defined as a set of ordered pairs $R = \{(x,y) | (x,y) \in X\times Y\}$. A fuzzy relation R on a domain $X\times Y$ is a relation whose element has a membership degree ranged in the domain [0,1], i.e., $R = \{((x,y), \mu_R(x,y)) \mid (x,y) \in X\times Y, \mu_R(x,y) \in [0,1]\}$.[2][14] A preference relation on a set is a relation to represent whether or not one element is completely preferred to another one. On the other hand, a fuzzy preference relation is a fuzzy relation where a preference degree is represented by a value in the interval [0, 1] instead of {0, 1}.

In practical applications, the fuzzy preference relations are determined subjectively by decision-maker or computationally by some criteria or some evaluation measures. Depending on the situations, the fuzzy preference relations may be max-min or max-product transitive[14], i.e., $\mu_R(x,z) \geq \{\mu_R(x,y), \mu_R(x,z)\}$ or $\mu_R(x,z) \geq \mu_R(x,y)\cdot\mu_R(x,z)$, for $x, y, z \in X$. The fuzzy preference relations may be a compensatory to unit relation satisfying the next two conditions: $\mu_R(x,y) + \mu_R(y,x) = 1$, for $x, y \in X$, and $\mu_R(x,x) = 0.5$ for all $x \in X$.[10][11][12][13] Table 1 shows a fuzzy preference relation which is a compensatory to unit relation.

Table 1. A fuzzy preference relation

	1	2	3	4	5	6	7	8	9	10
1	0.5	0.0	0.1	0.6	0.9	0.8	0.5	0.8	0.5	0.0
2	1.0	0.5	0.5	0.0	0.7	0.2	0.3	0.8	0.5	0.8
3	0.9	0.5	0.5	0.3	0.4	0.5	0.6	0.3	0.8	0.7
4	0.4	1.0	0.7	0.5	0.2	0.9	0.4	0.9	0.6	0.7
5	0.1	0.3	0.6	0.8	0.5	0.2	0.7	0.4	0.9	0.6
6	0.2	0.8	0.5	0.1	0.8	0.5	0.9	0.8	0.9	0.0
7	0.5	0.7	0.4	0.6	0.3	0.1	0.5	0.3	0.6	0.1
8	0.2	0.2	0.7	0.1	0.6	0.2	0.7	0.5	0.0	0.9
9	0.5	0.5	0.2	0.4	0.1	0.1	0.4	1.0	0.5	0.4
10	1.0	0.2	0.3	0.3	0.4	1.0	0.9	0.1	0.6	0.5

4. Genetic Algorithms

Genetic Algorithms are search algorithms based on the mechanics of natural selection and natural genetics.[4][5] Motivated by the biological adaptation, they generate a new set of chromosomes (i.e., candidate solutions) from parent chromosomes via stochastic operation. Chromosomes with high fitness values survive and those with low fitness values die off generation to generation. While randomized, genetic algorithms efficiently exploit historical information to speculate on new search points with expected improved performance. The building block hypothesis and the schemata theory provide the theoretical support that genetic algorithm approach can find optimal or near-optimal solutions.[4][5]

In order to use genetic algorithm approach to solve some problem, the following components of genetic algorithm should be developed: The first component is *Encoding Scheme*. Each chromosome corresponds to a candidate solution and should have a format to which genetic operators can be applied. Thus we should develop some scheme to represent candidate solutions in coded strings (chromosomes). The second component is *Genetic Operators*. Genetic operators take charge of creating new chromosomes from the existing chromosomes. Crossover and mutation are the typical genetic operators in the genetic algorithm literature: Crossover operator forms new chromosomes by interchanging substrings of two existing chromosomes. Mutation operator produces a new chromosome by randomly replacing a symbol of an existing chromosome with an alternating symbol. The creation of new chromosomes by genetic operators drives the search for the optimal solution. Therefore we should invent some genetic operators like crossover and mutation appropriate to the given problem domain and the employed encoding scheme. The third one is *Population Initialization*. A well-initialized population helps genetic algorithms to find desirable solutions more easily and faster than does a poorly-initialized one. Thus we need a deliberate method to create an initial population. The last one is *Evaluation Function*. During the operation of genetic algorithms, all chromosomes (i.e., candidate solutions) are evaluated to see how they are proper as the solution of problem. Therefore we have to develop a function to evaluate the fitness of candidate solutions.

In general, genetic algorithms consist of the following steps:

Step 1. Initialize a population of chromosomes.
Step 2. Evaluate the fitness of each chromosome.
Step 3. Create new chromosomes by applying genetic operators to current chromosomes.
Step 4. Delete some members from the population to make room for the new chromosomes.
Step 5. Evaluate the fitness of the new chromosomes and insert them into the population according to the evaluation results.
Step 6. If the termination condition is satisfied, stop and return the best chromosome; otherwise, go to Step 3.

5. A Genetic Algorithm for the Sequencing Problem with Fuzzy Preference Relation

The sequencing problem with fuzzy preference relation is an NP-complete problem. As mentioned in the previous section, genetic algorithms are a good approach for such NP-complete combinatorial problems. Here we propose a genetic algorithm for the sequencing problem. This section presents the four major components of the proposed genetic algorithm: encoding scheme of candidate solutions, population initialization method, genetic operator, and evaluation function.

5.1 Encoding Scheme of Candidate Solution

In sequencing problems, a solution corresponds to an ordered sequence of given entities. Thus to represent a solution, an ordered list representation of entities is used as follows:

$$S = (t_1\ t_2\ .\ .\ .\ t_n),$$

where t_i indicates an entity identifier.

5.2 Population Initialization

All permutations of entities can be a candidate solution for sequencing problems. Hence, the initial population is constructed by random permutations of all entity identifiers.

5.3 Genetic Operator

For sequencing problems, various existing genetic operators can be used to generate new chromosome (i.e., sequence) from existing chromosomes[4]. No existing genetic operator has been designed in consideration of fuzzy precedence relation. Here we propose a new genetic operator named *threshold-based crossover* which performs crossover operation with reference to the given fuzzy precedence relation. The threshold-based crossover operator works as follows:

Step 1. Select randomly an entity t_i in one parent chromosome P1.
Step 2. Mask the entity t_i and some other entities of P1 with reference to the given threshold (θ) in the following way:
When only the following credit is considered in the evaluation of the fitness of individual entity, mask the entities t_j such that $fp(t_j,t_i) \geq \theta$ ($j < i$).
When only the preceding credit is considered in the evaluation of the fitness of individual entity, mask the entities t_j such that $fp(t_i,t_j) \geq \theta$ ($j > i$).
When both the following credit and the preceding credit are considered in the evaluation of the fitness of individual entity, mask the entities t_j such that $fp(t_j,t_i) \geq \theta$ ($j < i$) or $fp(t_i,t_j) \geq \theta$ ($j > i$).
The threshold θ can be provided by a system parameter or by the overall credit of the parent chromosome (sequence) P1.
Step 3. Copy the masked entities to the same locations of the new child chromosome, and then copy the remaining entities into the empty locations of the child chromosome in the order that they appear in the other parent chromosome P2.

The following example shows how the threshold-based crossover operator performs its operation. This example assumes that the threshold value(θ) is set to 0.6 and the fuzzy preference relation is like Table 1. The crossover operator is applied to two parent chromosomes P1 and P2. Suppose that the entity 3 is selected from P1. Now it finds the entities t_j (i.e., 10, 7, and 9) such that $fp(3,t_j) \geq 0.6$ with reference to Table 1 and marks them. By Step 3, a new child chromosome C1 is created as follows:

P1 :	(8	1	$\underline{\mathbf{3}}$	$\underline{\mathbf{10}}$	6	$\underline{\mathbf{7}}$	$\underline{\mathbf{9}}$	4	5	2)
			t_i	X		X	X			
P2 :	($\underline{3}$	$\underline{10}$	5	8	6	4	2	1	$\underline{9}$	$\underline{7}$)
C :	(5	8	3	10	6	7	9	4	2	1)

5.4 Evaluation Function

The fitness of a sequence S is measured by the overall fitness evaluation function $C(S)$. Depending on the application domain, the fitness evaluation function is determined. Some number of possible fitness evaluation functions $C(S)$ are exemplified in Section 2.

Suppose that $c(t_i) = \min\{ fp(t_i, t_j) \mid j > i\}$ is used as the fitness evaluation function of individual entity and $C(S) = \Sigma_i\ c(t_i)$ is used as the overall fitness evaluation function. When this evaluation function is applied to the following sequence S with respect to the fuzzy preference relation shown in Table 1, the overall fitness is evaluated as follows:

$S = (5 \quad 8 \quad 3 \quad 10 \quad 6 \quad 7 \quad 9 \quad 4 \quad 2 \quad 1)$

$c(5) = \min\{0.4, 0.6, 0.6, 0.2, 0.7, 0.9, 0.8, 0.3, 0.1\}$
$= 0.1$

$c(8) = \min\{0.7, 0.9, 0.2, 0.7, 0.0, 0.1, 0.7, 0.2\} = 0.0$

$c(3) = \min\{0.7, 0.5, 0.6, 0.8, 0.3, 0.5, 0.9\} = 0.3$

$c(10) = \min\{1.0, 0.9, 0.6, 0.3, 0.2, 1.0\} = 0.2$

$c(6) = \min\{0.9, 0.9, 0.1, 0.8, 0.1\} = 0.1$

$c(7) = \min\{0.6, 0.6, 0.7, 0.5\} = 0.5$

$c(9) = \min\{0.3, 0.2, 0.5\} = 0.2$

$c(1) = 0.0$

$C(S) = \Sigma_i c(t_i) = 0.1+0.0+0.3+0.2+0.1+0.5+0.2+0.0$
$= 1.4$

6. Experiment Results based on the Proposed Genetic Algorithm

To see the applicability of the proposed genetic algorithm to the sequencing problem with fuzzy preference relation, some experiments have been performed. In the experiment, the proposed genetic algorithm has been applied to some randomly generated data set (i.e., fuzzy preference relations). To compare the efficiency of the proposed genetic algorithm, a genetic algorithm has been applied to the same data set, which uses the order crossover [4] that is a genetic crossover operator applicable to sequencing problems. In the experiment, a fuzzy preference relation with 30x30 size was randomly generated and the following fitness evaluation function was employed:

Individual entity fitness : $c_p(t_i) = \Sigma_{j=i+1,n} fp(i,j)$

Overall fitness : $C(S) = \Sigma_i\ c(t_i)$

Both the proposed genetic algorithm using the threshold-based crossover and the genetic algorithm using the order crossover have been applied to the data set 15 times, respectively. At each run of the genetic algorithms, they were set to generate 300,000 new chromosomes. For the threshold-based crossover operator, the threshold value was 0.35. Figure 2 shows the experiment results, where we can see that the proposed genetic algorithm gives more promising results than the genetic algorithm using the order crossover.

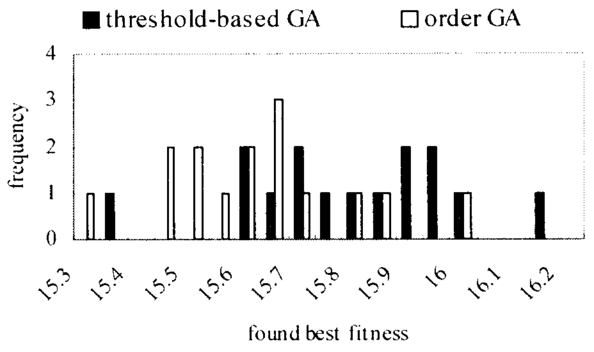

Fig.2 Experiment Results

7. Conclusions

This paper introduced a new problem called a sequencing problem with fuzzy preference relation. In the problem, the fuzzy preference relation is used to evaluate the quality of sequences. Depending on the employed fitness evaluation function, the sequencing problem comes to represent other well-known sequencing problem. For instance, when Eq. (1) is taken as the fitness evaluation function, the sequencing problem comes to a traveling salesman problem. When Eq. (2) is taken, it is similar to the problem to find the path with the maximum flow through the represented path.

To solve the sequencing problem, a genetic algorithm is proposed where a new crossover operator named the threshold-based crossover is developed. The crossover operator works in consideration of the given fuzzy preference relation. In order to show the applicability of the proposed genetic algorithm for the sequencing problem with fuzzy preference relation, we applied the proposed genetic algorithm to some randomly generated test data sets. To compare the efficiency of the proposed genetic operator, another genetic algorithm has been applied to the same data sets, which uses the order crossover that is a genetic operator applicable to the sequencing problem. The performed experiments showed that the proposed genetic algorithm can be one of useful tools for the sequencing problem with fuzzy preference relation.

It is expected that many sequencing problems can be transformed into the sequencing problem with fuzzy preference relation and they can be solved by the methods for the sequencing problem with fuzzy preference relation like the proposed genetic algorithm. In this paper, the characteristics of the sequencing problem with fuzzy preference relation are not thoroughly investigated on the basis of possible fitness evaluation functions. Thus, we yet need some more studies on them.

References

1. K. R. Baker, Introduction to Sequencing and Scheduling, John Wiley & Sons, 1974.
2. D. Dubois, H. Prade, *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, 1980.
3. J. Blazewicz, K. Ecker, G. Schmit, J. Weglarz, *Scheduling in Computer and Manufacturing Systems*, Springer-Verlag, 1993.
4. Z. Michalewicz, *Genetic Algorithms+ Data Structures = Evolution Programs*, Springer-Verlag, 1989.
5. D.E. Goldberg, *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley, 1989.
6. A. Banerjee, Rational choice under fuzzy preference: The Orlovsky choice function, *Fuzzy Sets and Systems*, Vol.53, pp.295-299, 1993.
7. F. Choobineh, H. Li, An index for ordering fuzzy numbers, *Fuzzy Sets and Systems*, Vol.54, pp.287-294, 1993.
8. W. Kolodziejczyk, Orlovsky's concept of decision-making with fuzzy preference relation – further study, *Fuzzy Sets and Systems*, Vol.19, pp.11-20, 1986.
9. K. Nakamura, Preference relations on a set of fuzzy utitlities as a basis for decision making, *Fuzzy Sets and Systems*, Vol.20, pp.147-162, 1986.
10. S. A. Orlovsky, Decision-making with a fuzzy preference relation, *Fuzzy Sets and Systems*, Vol.1, pp.155-167, 1978.
11. P. Perny, B. Roy, The use of fuzzy outranking relations in preference modeling, *Fuzzy Sets and Systems*, Vol.49, pp.33-53, 1992.
12. B. Roy, Partial preference analysis and decision-aid: the fuzzy outranking relation concept, Conflicting Objectives in Decisions (D.E. Bell, R.L. Keeney, H. Raiffa, eds., John Willy & Sons, Chichester), pp.40-75, 1977.
13. K.-M. Lee, C.H. Cho, H. Lee-Kwang, Ranking fuzzy values with satisfaction function, *Fuzzy Sets and Systems*, pp.295-309, 1994.
14. H.-J. Zimmermann, Fuzzy Set Theory and its Applications, Kluwer-Nojhoff publishing, New York, 1985.