A GENETIC ALGORITHM FOR SOLVING FUZZY SHORTEST PATH PROBLEMS WITH INTERVAL TYPE-2 FUZZY ARC LENGTHS

Arindam Dey[1], Rangaballav Pradhan[2], Anita Pal[3], Tandra Pal[2]

[1]Department of Computer science and Engineering,
Soraj Mohan Institute of Technology, Guptipara, West Bengal, India
E-mail: arindam84nit@gmail.com

[2]Department of Computer science and Engineering
National Institute of Technology, Durgapur, West Bengal, India
Email: rangaballav.pradhan@gmail.com, tandra.pal@gmail.com

[3]Department of Mathematics
National Institute of Technology, Durgapur, West Bengal, India
Email: anita.buie@gmail.com

DOI: https://doi.org/10.22452/mjcs.vol31no4.2

ABSTRACT

Shortest path problem is one of the most fundamental and well-known optimization problems in graph theory due to its various real-world applications. Fuzzy set can manage the uncertainty, associated with the information of a problem, where conventional mathematical models may fail to reveal satisfactory result. In most cases, shortest path problem in fuzzy graph, called fuzzy shortest path problem, uses type-1 fuzzy set as arc length. The uncertainty associated with the linguistic description of information is not represented properly by type-1 fuzzy set due to inexactness of human perception in the evaluation of membership degrees having crisp values. An interval type-2 fuzzy set is able to tackle this type of uncertainty. In this paper, we have proposed an algorithmic approach based on genetic algorithm for finding shortest path from a source node to a destination node in a fuzzy graph with interval type-2 fuzzy arc lengths. We have designed a new crossover operator which does not need mutation operation. The purpose of mutation operation has been taken care by the proposed crossover operation. We have compared our algorithm with two other existing genetic algorithms for the fuzzy shortest path problem, where superiority of the proposed algorithm is shown. To the best of our knowledge, no algorithm based on genetic algorithm exists in the literature for fuzzy shortest path problem with interval type-2 fuzzy arc lengths. A numerical example is used to illustrate the effectiveness of the proposed approach.

Keywords: Fuzzy shortest path problem, Fuzzy graph, Type-1 fuzzy set, Interval type-2 fuzzy set, Genetic algorithm.

1.0 INTRODUCTION

The problem of finding shortest path from a specified source node to the destination node in a graph is a fundamental and well-known combinatorial optimization problem in graph theory. It has many applications, including transportation, routing, communications, supply chain management or models involving agents. The classical shortest path problem (SPP) [1], [2], [3], [4] seeks to select a path with minimum length from a finite set of paths in a graph.

In real world problems, arc lengths may represent time, cost, distance or other variables. It is very hard for decision makers [5], [7], [8], [9] to find exact value of each arc length for real life applications. As an example, if we consider transmission time as arc length, the time to transmit data from a specific node to another node, even using the same modem, may not always be the same due to several reasons. So, the arc length does not have specific value, i.e., uncertain. Uncertainty plays an important role to maximize the usefulness of the system model. In type-1 fuzzy set (T1FS), we rely on human perception in evaluation of membership degrees. In this study, interval type-2 fuzzy set (IT2FS) is used to represent the arc length for solving SPP; it considers the uncertainty or inexactness of human perception. Thus, IT2FS can handle more uncertainty than T1FS, i.e., minimizes the effects of uncertainties which ultimately increases the credibility of the resulting system.

A graph is a convenient way of representing information involving relationship between objects. The objects are represented by vertices and relations by edges. When uncertainty is there in the description of the objects or in its relationships or in both, we design fuzzy graph model. In this study, we propose a genetic algorithm to find shortest path in a fuzzy graph, whose arc length is represented by an IT2FS. In the literature, no work has been done on FSPP using IT2FS based on genetic algorithm (GA).

Fuzzy shortest path problem (FSPP), where the arc lengths have fuzzy values, is one of the most studied problems in fuzzy sets and systems area. An algorithmic approach for FSPP has been proposed first by Dubois and Prade [10]. Their idea is based on the extensions of the classical Floyd and Ford-Moore-Bellman (FMB) algorithms. The fuzzy number $D_{i,j}$ is used to represent the length of the shortest path from the node i to node j. Based on the possibility theory, they have proposed the concept of criticality of a path. The criticality value for each path p is described as the possibility of the path p to be the shortest path. This criticality value is used as the membership value of path p in the fuzzy set of the shortest path from the node i and node j. $D_{i,j}$ is considered as a possibility distribution of the shortest distance from node i to node j. But the main weakness of the algorithm [10] is that it computes the distance of the shortest path without identifying an existing path. Then many researchers have done a lot of work on the FSPP using T1FSs. Chanas and Kamburowski [11] have introduced the concept of fuzzy strict preference relation and used it in their proposed algorithm to select the shortest path. The membership value for the shortest path between nodes i and j is not used in their algorithm as discussed in [10]. In Klein's algorithm [12], the cost of each arc in a network is denoted by an integer value between 1 and certain fixed upper bound. The dynamic programming recursion is used in their proposed algorithm to determine a path or paths corresponding to the threshold of membership value, decided by the decision maker. Yager [13] has introduced an algorithm for FSPP using the concept of possibility production system. The possibility values have been assigned for traversing between two states and the overall possibility of a path between initial state and goal state is calculated using on the concept of T-norms. The proposed algorithm selects the path (initial state to goal state) which has maximum possibility. The author has introduced a heuristic search algorithm to avoid the combinatorial explosion to minimize the time for computing optimal solution. In [14], Okada and Soper have used fuzzy numbers to denote the arcs length of a directed fuzzy network. An order relation between fuzzy numbers, based on fuzzy min and fuzzy max operation, is proposed for the purpose of generating nondominated paths. Their proposed algorithm finds all nondominated paths between source node and other nodes. The algorithm is based on the concept of multiple labeling technique for a multicriteria SPP. However, the number of all nondominated paths may be high for large scale problem. Hence, they have introduced a method to reduce the number of paths according to possibility level. In [15], Blue et al. have proposed an algorithm to compute a cut value to limit the number of analyzed paths. This algorithm has been applied in the Eppstein's k-shortest path algorithm (crisp) to solve the FSPP. They have also defined different type of the fuzziness of a graph. Moazeni [19] has described the SPP from a specified vertex to every other vertices on a graph, in which positive fuzzy quantities with finite supports are used to represent the arc lengths of the graph. The author has defined an order relation between fuzzy quantities with finite supports, based on the extended min of fuzzy sets. Then, an algorithmic approach based on Dijkstra's shortest path algorithm and Hansen's multiple labeling method is introduced to find the set of non-dominated paths with respect to the extension principle. In [20], Hernandes et al. have introduced an iterative algorithm to solve the FSPP where fuzzy parameters (triangular fuzzy numbers) are used to represent the edge weights. Their algorithm is based on the classical Ford-Moore-Bellman algorithm. A generic ranking index is applied in their algorithm to compare the fuzzy edge weight. This algorithm can find negative circuits and also work in a graph having negative parameters. Mahdavi et al. [22] have introduced a dynamic programming technique for FSPP to compute the shortest path between every pair of vertices in an acyclic graph. They have proposed a fuzzy ranking technique which can work for comparison on various types of fuzzy numbers and is also reliable in distinguishing unequal fuzzy numbers. This proposed fuzzy ranking technique can avoid generating the set of Pareto optimal (non-dominated) paths because a large number of Pareto optimal paths are derived from a large network. In that case, it is very hard for decision maker to select a preferable path. Hassanzadeh et al. [32] have introduced an algorithm to find out the shortest path in a graph with various types of fuzzy arc lengths. They have proposed an addition operation of various fuzzy numbers in a path using α cut to compute the path length. In their proposed addition operation, a least square model is designed to compute an approximation of the corresponding membership function for the addition operation. They have proposed a genetic algorithm for the FSPP to solve the complexity of the proposed addition operation for large fuzzy graphs. A fuzzy ranking technique for fuzzy number is used in their algorithm for comparison the distance (fuzzy number) between two different paths. Yajuan Zhang et al. [36] have introduced a biologically-inspired algorithm for FSPP. Their path finding model is inspired by *P. polycephalum* which is designed for classical shortest path problems. They have modified the algorithm with the help of fuzzy set theory. Fuzzy numbers are used to represent the arc lengths of the FSPP. The fuzzy arithmetic and fuzzy distance between fuzzy numbers are adopted to modify the path finding algorithm for FSPP.

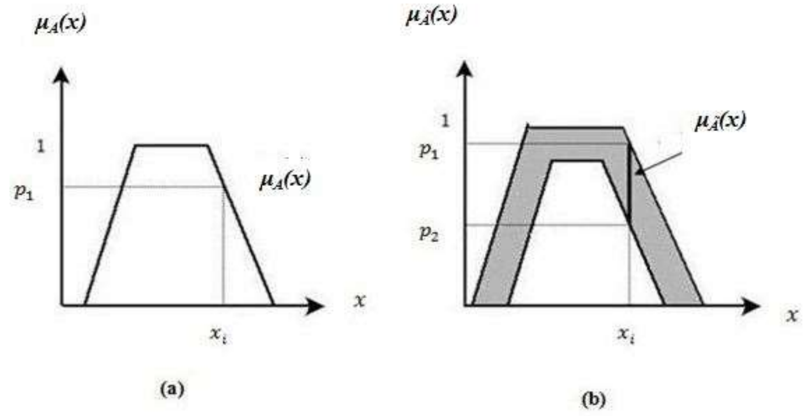


Fig. 1: (a) Crisp grades of membership for T1FS. (b) Fuzzy grades of membership for IT2FS

Most of the researchers have used type-1 fuzzy set (T1FS) to express the imprecise weight of the arc. The membership values of T1FSs are crisp, determined by the corresponding membership function. However, there is also uncertainty in the membership values. For example, if the arc lengths are collected from more than one source which fluctuates regularly, then it cannot be represented by T1FS. This type of problem can be solved using T2FS, where membership functions are fuzzy [6], [40], [41]. T2FSs enhance the number of degrees of freedom to handle inexact (fuzzy) information. Membership functions of both T2FS and IT2FS are three dimensional, but the secondary membership value of the IT2FS is always equal to 1. Several researchers have used IT2FS for solving real world problems [42], [43], [44], [45], [46]. T2FS is computationally expensive, hence computation in IT2FS is more manageable compared to T2FS, making IT2FS quite practical. IT2FS makes the inference more efficient than T1FS with an increasing ability to handle the uncertainty. Let *A* be a T1FS as shown in Fig. 1(a). Then for a specific value of x, say x_i, there is a single membership value p_1 for T1FS. If $\tilde{A}$ is an IT2FS as shown in Fig. 1(b), then for the same value of x_i, there is a set of interval membership values between p_1 and p_2. Due to this reason, IT2FS is able to describe and handle more uncertainty than T1FS. Real life applications suitable for using IT2FS in SPP are described in [47]. V. Anusuya and R. Sathya, in [37], have introduced an algorithm for FSPP between the source and destination nodes on a fuzzy-weighted directed network, in which type-2 fuzzy numbers (T2FNs) are used to represent the arc lengths. Their proposed algorithm computes all possible paths and their corresponding path lengths between source and destination nodes. The authors have added all the complement T2FNs corresponding to the edges present in a path. The ranks of the complement type-2 fuzzy number corresponding to their paths are computed and the path with lowest rank value has been considered as the shortest path. In [39], V. Anusuya and R. Sathya have used a discrete T2FN to each arc as arc length of the graph. All possible paths and their corresponding path lengths from source node to destination node have been computed using the same technique as in [37]. In [47], the authors have used IT2FSs as arc lengths of a fuzzy graph for SPP. The authors have incorporated the uncertainty in the classical Dijkstra's algorithm for SPP using IT2FS as arc length of the fuzzy network. They have also presented a generalized algorithm for the IT2FSPP, based on path algebra.

In this article, we have proposed a genetic algorithm to find the shortest path from a single source node to a single destination node in a fuzzy graph, where the arc lengths are IT2FSs. The motivation behind the work in this study is to use genetic algorithm for IT2FSPP which will be efficient as well as simple and effective in real world scenarios. In our proposed genetic algorithm, we first have discussed how to represent a path from source node to destination node using a chromosome. Then we have proposed a new scheme for generating the initial population. The path length is expressed in the form of IT2FS, which is the summation of all the arcs lengths (IT2FS) present in the path. It is based on the concept of addition operation [24] of IT2FS. Centroid-based ranking [18], [30] is used to find the rank of a path. The rank value is considered as the fitness of the corresponding path. We have designed the crossover operation in such way that GA does not need any mutation operation further. We have compared the proposed algorithm with two existing genetic algorithms for the fuzzy shortest path problem.

The paper is organized as follows. In Section 2, we have presented some basic concepts and definitions on fuzzy graph, type-2 fuzzy set (T2FS), IT2FS and centroid-based ranking of IT2FS. The working principal of genetic algorithm is described in Section 3. In Section 4, we have described our proposed genetic algorithm for FSPP. The results of the proposed approach and its comparisons with two other existing approaches are presented in Section 5. Finally, we have concluded in Section 6.

2.0 PRELIMINARIES

In this section, we discuss fuzzy graph, T2FS, IT2FS and centroid-based ranking of IT2FS to facilitate future discussions

Definition 1. Let *V* be a finite, non-empty set. A fuzzy graph *G* is defined by a pair of functions *σ* and *μ*, G = (σ, μ). Here, *σ* denotes a fuzzy subset of *V* and denotes a symmetric fuzzy relation on *μ*, i.e., σ: *V*→ [0,1] and $\mu : V \times V \to [0,1]$ such that

$$\mu(u,v) \le \left(\sigma(u) \cap \sigma(v)\right), \forall u, \forall v \in V \tag{1}$$

A fuzzy graph is a generalization of crisp graph. In this paper, we consider a graph, whose nodes are crisp but lengths of the arcs are fuzzy, represented by IT2FSs.

Definition 2 [31]. A T2FS, denoted as $\tilde{A}$, in the universe of discourse *X* can be characterized by a type-2 membership function $\mu_{\tilde{A}}(x, u)$ as follows:

$$\tilde{A} = \left\{ \left((x,u), \mu_{\tilde{A}}(x,u) \right) \middle| \forall x \in X, \forall u \in J_x \subseteq [0,1] \right\} \tag{2}$$

Here, *x* is the primary variable, J_x is the primary membership function of *x*, *u* is the secondary variable, and $0 \le \mu_{\tilde{A}}(x,u) \le 1$ is the secondary membership function at *x*. The T2FS $\tilde{A}$ can also be expressed as in (3).

$$\int_{x \in X} \int_{u \in J_x} \mu_{\tilde{A}}(x,u) / (x,u) \tag{3}$$

Here, $J_x \subseteq [0,1]$ and $\int\int$ indicates the union over all admissible *x* and *u*. $\int$ is replaced by $\sum$ for discrete universe of discourse.

Definition 3 [31]. IT2FS is a special from of T2FS. A T2FS which has a uniform secondary membership function is called an IT2FS. Let $\tilde{A}$ be an IT2FS, then

$$\int_{x \in X} \int_{u \in J_x} 1/(x,u) = \int_{x \in X} \left[\int_{u \in J_x} 1/u \right] \Big/ x \tag{4}$$

Here, *x, J_x, u* and $\int_{u \in J_x} 1/u$ are respectively primary variable, primary membership of *x* which is an interval in [0,1], secondary variable and secondary membership function at *x*.

In this study, we describe a method to apply trapezoidal IT2FSs for solving FSPP problems, where the reference points and their corresponding heights of the upper membership function (UMF) and lower membership function (LMF) of trapezoidal IT2FSs are used to characterize trapezoidal IT2FSs. A trapezoidal IT2FS $\tilde{A}_i$ is shown in Fig. 2. The IT2FS $\tilde{A}_i$ is characterized by a UMF, represented by $\overline{\mu}_i(x_i)$ or $\tilde{A}_i^U$ and an LMF, represented by $\underline{\mu}_i(x_i)$ or $\tilde{A}_i^L$. The UMF and LMF of $\tilde{A}_i$ both are trapezoidal T1FSs.

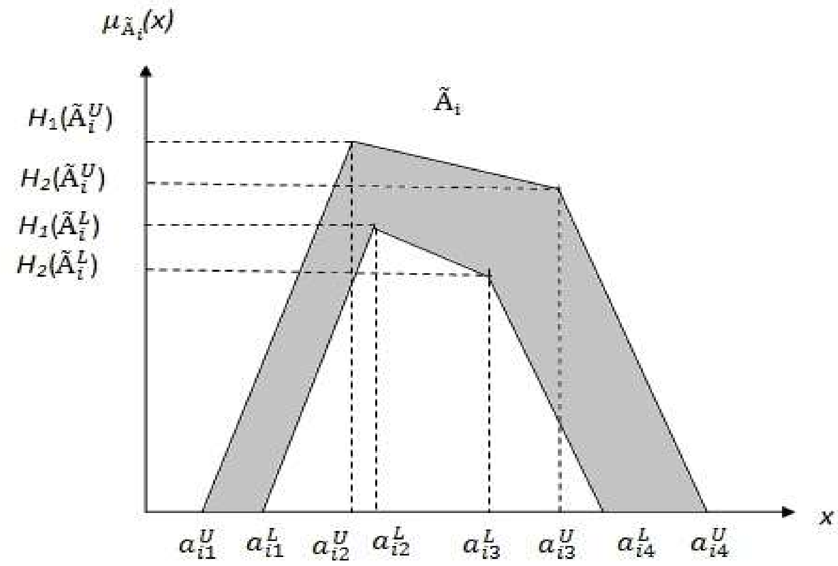


Fig. 2: Reference points with their height to find the FOU of IT2FS $\tilde{A}_i$. The reference points $(a_{i1}^U, a_{i2}^U a_{i3}^U, a_{i4}^U)$ and their corresponding heights $H_1(\tilde{A}_i^U)$ and $H_2(\tilde{A}_i^U)$ find the trapezoidal UMF $\tilde{A}_i^U$ of IT2FS $\tilde{A}_i$ and reference points $(a_{i1}^L, a_{i2}^L a_{i3}^L, a_{i4}^L)$ and their corresponding heights $H_1(\tilde{A}_i^L)$ and $H_2(\tilde{A}_i^L)$ find the trapezoidal LMF $\tilde{A}_i^L$ of IT2FS $\tilde{A}_i$

Let $\tilde{A}_1$and $\tilde{A}_2$ be two trapezoidal IT2FSs where

$$\tilde{A}_1 = \left(\tilde{A}_1^U, \tilde{A}_1^L\right) = \left(\left(a_{11}^U, a_{12}^U, a_{13}^U, a_{14}^U; H_1\left(\tilde{A}_1^U\right), H_2\left(\tilde{A}_1^U\right)\right)\left(a_{11}^L, a_{12}^L, a_{13}^L, a_{14}^L; H_1\left(\tilde{A}_1^L\right), H_2\left(\tilde{A}_1^L\right)\right)\right)$$

$$\tilde{A}_2 = \left(\tilde{A}_2^U, \tilde{A}_2^L\right) = \left(\left(a_{21}^U, a_{22}^U, a_{23}^U, a_{24}^U; H_1\left(\tilde{A}_2^U\right), H_2\left(\tilde{A}_2^U\right)\right)\left(a_{21}^L, a_{22}^L, a_{23}^L, a_{24}^L; H_1\left(\tilde{A}_2^L\right), H_2\left(\tilde{A}_2^L\right)\right)\right)$$

The addition operation [24] between the two trapezoidal IT2FSs $\tilde{A}_1$ and $\tilde{A}_2$ is defined in (5) as follows.

$$\begin{aligned}\tilde{A}_1 \oplus \tilde{A}_2 &= \left(\left(\tilde{A}_1^U, \tilde{A}_1^L\right) \oplus \left(\tilde{A}_2^U, \tilde{A}_2^L\right)\right) = \left(\left(\tilde{A}_1^U \oplus \tilde{A}_2^U\right), \left(\tilde{A}_1^L \oplus \tilde{A}_2^L\right)\right)\\ &= \begin{pmatrix} \left(a_{11}^U + a_{21}^U, a_{12}^U + a_{22}^U, a_{13}^U + a_{23}^U, a_{14}^U + a_{24}^U; \min\left(H_1\left(\tilde{A}_1^U\right), H_1\left(\tilde{A}_2^U\right)\right), \min\left(H_2\left(\tilde{A}_1^U\right), H_2\left(\tilde{A}_2^U\right)\right)\right) \\ \left(a_{11}^L + a_{21}^L, a_{12}^L + a_{22}^L, a_{13}^L + a_{23}^L, a_{14}^L + a_{24}^L; \min\left(H_1\left(\tilde{A}_1^L\right), H_1\left(\tilde{A}_2^L\right)\right), \min\left(H_2\left(\tilde{A}_1^L\right), H_2\left(\tilde{A}_2^L\right)\right)\right) \end{pmatrix}\end{aligned} \tag{5}$$

It shows that the result of addition operation is also a trapezoidal IT2FS. So, it can also be added to another IT2FS $\tilde{A}_3$ resulting again a new IT2FS, say $\tilde{A}_4$. In this way, we can add n number of IT2FSs as defined below in (6).

$$\begin{aligned}\tilde{A}_1 \oplus \tilde{A}_2 \oplus \tilde{A}_3 ..... \oplus \tilde{A}_n &= \left(\left(\tilde{A}_1^U, \tilde{A}_1^L\right) \oplus \left(\tilde{A}_2^U, \tilde{A}_2^L\right) \oplus \left(\tilde{A}_3^U, \tilde{A}_3^L\right) .... \oplus \left(\tilde{A}_n^U, \tilde{A}_n^L\right)\right)\\ &= \left(\left(\tilde{A}_1^U \oplus \tilde{A}_2^U \oplus \tilde{A}_3^U ...... \oplus \tilde{A}_n^U\right), \left(\tilde{A}_1^L \oplus \tilde{A}_2^L \oplus \tilde{A}_3^L ..... \oplus \tilde{A}_n^L\right)\right)\\ &= \begin{pmatrix} \begin{pmatrix} a_{11}^U + a_{21}^U + a_{31}^U .... + a_{n1}^U, a_{12}^U + a_{22}^U + a_{32}^U .... + a_{n2}^U, a_{13}^U + a_{23}^U + a_{33}^U .... + a_{n3}^U, a_{14}^U + a_{24}^U + a_{34}^U .... + a_{n4}^U; \\ \min\left(H_1\left(\tilde{A}_1^U\right), H_1\left(\tilde{A}_2^U\right), H_1\left(\tilde{A}_3^U\right), ..., H_1\left(\tilde{A}_n^U\right)\right), \min\left(H_2\left(\tilde{A}_1^U\right), H_2\left(\tilde{A}_2^U\right), H_2\left(\tilde{A}_3^U\right), ..., H_2\left(\tilde{A}_n^U\right)\right) \end{pmatrix} \\ \begin{pmatrix} a_{11}^L + a_{21}^L + a_{31}^L .... + a_{n1}^L, a_{12}^L + a_{22}^L + a_{32}^L .... + a_{n2}^L, a_{13}^L + a_{23}^L + a_{33}^L .... + a_{n3}^L, a_{14}^L + a_{24}^L + a_{34}^L .... + a_{n4}^L; \\ \min\left(H_1\left(\tilde{A}_1^L\right), H_1\left(\tilde{A}_2^L\right), H_1\left(\tilde{A}_3^L\right), ..., H_1\left(\tilde{A}_n^L\right)\right), \min\left(H_2\left(\tilde{A}_1^L\right), H_2\left(\tilde{A}_2^L\right), H_2\left(\tilde{A}_3^L\right), ..., H_2\left(\tilde{A}_n^L\right)\right) \end{pmatrix} \end{pmatrix}\end{aligned} \tag{6}$$

Here, (6) represents an IT2FS. Thus, we can conclude that the path between two nodes can be represented by an IT2FS, obtained by adding all the IT2FSs corresponding to the arcs, present in the path. It is formally presented in Definition 4.

Definition 4. The length of a path between any two vertices can be expressed by an IT2FS, determined by adding all the trapezoidal IT2FSs corresponding to the edges (edge costs) present in that path.

As an example of the Definition 4, let us consider the graph *G*, shown in Fig. 3. The cost of the edges is expressed by IT2FSs, which are given in Table 1. Let $\tilde{A}_{ij}$ represent an IT2FS, where *i* and *j* are two vertices of graph *G* associated with the edge *(i, j)* directly. We express the path between the node 1 and node 20, through the nodes 3, 9, 15 and 20 applying the law of association on addition operation of two trapezoidal IT2FSs as follows:

$$\begin{aligned}&\left(\left(\tilde{A}_{13} \oplus \tilde{A}_{39}\right) \oplus \tilde{A}_{915}\right) \oplus \tilde{A}_{1520}\\ &= \left(\tilde{A}_{139} \oplus \tilde{A}_{915}\right) \oplus \tilde{A}_{1520}\\ &= \tilde{A}_{13915} \oplus \tilde{A}_{1520}\\ &= \tilde{A}_{1391520}\end{aligned}$$

Here, $\tilde{A}_{139}$ and $\tilde{A}_{13915}$ are two trapezoidal IT2FSs denoting two paths, first one is between the vertex 1 and vertex 9 through the vertex 3 and another is between the vertex 1 and vertex 15 through the vertices 3 and 9. In other words, we can say that each path from source node to destination node can be represented by an arc and can be denoted by an IT2FS.

Definition 5 [25]. The centroid $C(\tilde{A})$ of an IT2FS $\tilde{A}$ is the union of the centroids of all its embedded T1FSs A_e as follows.

$$C\left(\tilde{A}\right)=\bigcup_{\forall A_e} c\left(A_e\right)=\left[c_l\left(\tilde{A}\right),c_r\left(\tilde{A}\right)\right] \tag{7}$$

Here, $\bigcup$ is the union operation and

$$c_l\left(\tilde{A}\right)=\min_{\forall A_e}\left(c\left(A_e\right)\right) \tag{8}$$

$$c_r\left(\tilde{A}\right)=\min_{\forall A_e}\left(c\left(A_e\right)\right) \tag{9}$$

$$c\left(A_e\right)=\frac{\sum_{i=1}^{N} x_i \mu_{A_e}\left(x_i\right)}{\sum_{i=1}^{N} \mu_{A_e}\left(x_i\right)} \tag{10}$$

$c_l(\tilde{A})$ and $c_r(\tilde{A})$ can be expressed [20, 23, 24] as follows.

$$c_l\left(\tilde{A}\right)=\frac{\sum_{i=1}^{L} x_i \bar{\mu}_{\tilde{A}}\left(x_i\right)+\sum_{i=L+1}^{N} x_i \underline{\mu}_{\tilde{A}}\left(x_i\right)}{\sum_{i=1}^{L} \bar{\mu}_{\tilde{A}}\left(x_i\right)+\sum_{i=L+1}^{N} \underline{\mu}_{\tilde{A}}\left(x_i\right)} \tag{11}$$

$$c_r\left(\tilde{A}\right)=\frac{\sum_{i=1}^{R} x_i \underline{\mu}_{\tilde{A}}\left(x_i\right)+\sum_{i=R+1}^{N} x_i \bar{\mu}_{\tilde{A}}\left(x_i\right)}{\sum_{i=1}^{R} \underline{\mu}_{\tilde{A}}\left(x_i\right)+\sum_{i=R+1}^{N} \bar{\mu}_{\tilde{A}}\left(x_i\right)} \tag{12}$$

L and R are switch points. First we compute the centroids of IT2FS $\tilde{A}$. Then we find the average centroid.

$$c\left(\tilde{A}\right)=\frac{c_l\left(\tilde{A}\right)+c_r\left(\tilde{A}\right)}{2} \tag{13}$$

The value of $c(\tilde{A})$ is known as centroid-based ranking value [30] of IT2FS $\tilde{A}$. This ranking method is the generalization of Yager's first ranking method for T1FSs [27] to IT2FSs.

In this paper, we have used centroid-based ranking method [48] to find the ranks of IT2FSs, which represent the path lengths from source node to destination node. The larger is the centroid value $c(\tilde{A})$, the greater is the arc length of the corresponding IT2FS $\tilde{A}$.

3.0 WORKING PRINCIPAL OF GA

Genetic Algorithm [35] is a probabilistic heuristic search process, based on the concepts of natural genetics. It is inspired by Darwin's law of survival of fittest [35] and mimics the process of natural evolution. GA is population-based search. A set of solutions, called chromosomes, defines the population of a generation. GA recombines structural information to locate new points in the search space with expected improved performance. Recombinations are done by three genetic operators, selection, crossover and mutation. The population evolves as more fit solutions / chromosomes replace less fit solutions. This way, GA gradually nears to optimal solutions. In a SPP, an individual / solution represents a path from the source to the destination node. The fitness value of each individual in a population is evaluated by a fitness function. This fitness function is determined by the optimization of the problem to be solved. More fit individuals are selected from the current population through a selection operator. Recombination among the individuals, known as crossover, takes place followed by random mutation to form a new generation. The candidate solutions of the generation are then used to produce solution of next iteration. The algorithm terminates when the termination criteria is reached which may be either a maximum number of generations or a satisfactory fitness value of the solution.

4.0 PROPOSED ALGORITHM

Given an undirected graph and its arc length, proposed genetic algorithm finds the shortest path between two specific nodes, source and destination nodes of the graph. In this section, we have described the algorithm, chromosome encoding and its different functions and operations. The proposed genetic algorithm for IT2FSPP is presented in Algorithm 2. To the best of our knowledge, there exists no algorithm based on GA for FSPP with interval type-2 fuzzy arc lengths.

4.1 Chromosome Encoding

A chromosome represents a possible solution, i.e., a path from source node to destination node, and it may not be optimal. It contains the sequence of nodes in the same order as present in the path. Each node in the graph is assigned a unique integer number.

4.2 Initial Population Generation

Let n be the number of nodes in the undirected graph and *adj* is the corresponding adjacent matrix that stores the adjacent nodes of every node in the graph. We have proposed a method for initial population generation, shown in Algorithm 1. In Algorithm 1, *pop[i]* represents ith chromosome, where a chromosome represents a path between source and destination nodes and contains a sequence of nodes in that path from source node to destination node. So, the first node (node in first position of *pop[i]*) *pop[i][1]* is always the source node. We use a list of rejected nodes. If all the adjacent nodes of a particular node are already present in the chromosome (path) then that node is included in the rejected list. For generating a path required to encode a chromosome, the source node is considered first to generate the next node of the corresponding path. Let us consider the current node, say k, is in the j^{th} position of the ith chromosome, i.e., *pop[i][j]*. We are interested to find its next node in the path of the corresponding chromosome. For this purpose, a node, say p, is randomly chosen from the adjacent nodes of k. If the node p is not already present in *pop[i]*, p is considered as the next node of k in the chromosome corresponding to *pop[i]*. If the node p is already present in *pop[i]* but k is not in the rejected list, another node is selected randomly from the remaining nodes adjacent to k which are not present in the chromosome *pop[i]*. If all the adjacent nodes of k are already present in *pop[i]*, node k is added to the rejected list and the node previous to k is considered as the current node to find the next node. The process of adding the next node in the chromosome is repeated until it reaches the destination node. In similar way, all the chromosomes are encoded for generating the initial population. Population size is assumed as *pop_size.*

Algorithm 1 Algorithm for initial population generation

Input: Adjacent matrix of the graph
Output: Initial population for the proposed GA, containing a set of possible paths between the source and destination nodes

```
Begin
set $i = 1$
set $pop[i][1] = source$, $k = source$, $j = 1$
while $pop[i][j] \neq destination$ do
    choose $p$, a random node from $adj(k)$
    while $p$ is present in $pop[i]$ or $p$ is present in the rejected list do
        if all the nodes of $adj(k)$ are not present in $pop[i]$ then
            choose $p$, a random node from $adj(k)$
        else
```

```
10:             put $k$ in rejected list, decrement $j$ by 1, $k = pop[i][j]$
11:             choose $p$, a random node from $adj(k)$
12:         end if
13:     end while
14:     if $p$ is not present in $pop[i]$ then
15:         increment $j$ by 1, $pop[i][j] = p$, $k = pop[i][j]$
16:     end if
17: end while
18: increment $i$ by 1
19: if $i \leq pop\_size$ then
20:     goto step2
21: end if
22: End
```

4.3 Fitness Function

The fitness of a chromosome indicates how good the corresponding solution is. In our algorithm, the objective is to find the least path between the source and destination nodes. From Definition 4, we know that a path from source node to destination node can be represented by an IT2FS. We have used centroid-based ranking [30] method to find the rank of a path, which is the defuzzified value of the corresponding IT2FS. SPP is a minimization problem, i.e., to find a path having least rank. So, we have considered the fitness function as follows in (14) to make it a maximization problem. Fitness has been scaled to 100 to make the differences among fitness values appreciable. The value of rank is never zero as the arc length is always positive. So, there is no chance of fitness value to be infinity.

$$fitness = \frac{1}{rank} * 100 \tag{14}$$

4.4 Selection

Selection process involves selecting the individuals / chromosomes for more number of times when having higher fitness values, and for less number of times when having lower fitness values. In this study, we have used binary tournament selection, where we have compared the fitness values of two, randomly-selected individuals and select the best one for the next mating pool.

4.5 Crossover

Crossover operator is used to recombine two chromosomes, called parents, to get two new chromosomes, called children. Thus, properties of parent chromosomes are being inherited to their children which represent solutions different from their parent. In this work, we have proposed a single point crossover operator. We have described the proposed crossover below.

Step 1. Choose two parent chromosomes $p_1$and p_2 from the current mating pool.

Step 2. Find a random position p on chromosome for crossover.

Step 3. Let the nodes of chromosomes $p_1$and p_2 at position p be respectively $n_{p_1}$and n_{p_2}.

Step 4. For child1:

4.1. Copy the sequence of nodes of p_1 from source node to $n_{p_1}$for the first part of child1.

4.2. Generate a path from n_{p_1} to n_{p_2}using the method same as initial population generation. Append it to child1 after the first part. If there exists no path from n_{p_1} to n_{p_2}then reset child1 to p_1 and go to step 5.

4.3. Copy the sequence of nodes of p_2 from $n_{p_2}$to destination node at the end of child1.

4.4. If there is any loop in child1 then eliminate it by merging the duplicate nodes.

Step 5. For child2:

5.1. Copy the sequence of nodes of p_2 from source node to $n_{p_2}$to child2 to form its first part.

5.2. Generate a path from n_{p_2} to n_{p_1}using the method same as initial population generation and append it with child2 after its first part. If there exists no path from n_{p_2} to n_{p_1}then reset child2 to p_2 and go to step 6.

5.3. Copy the sequence of nodes of p_1 from $n_{p_1}$to destination node at the end of child2.

5.4. If there is any loop in child2 then eliminate it by merging the duplicate nodes.

Step 6. If child1 is not equal to p_1 and has fitness greater than or equal to that of p_1 or p_2, then replace the worst fit parent among p_1 and p_2 in the mating pool by child1, otherwise discard child1.

Step 7. If child2 is not equal to p_2 and has a fitness greater than or equal to the remaining of p_1 and p_2 or both (when no parent is replaced in step 6), then replace the corresponding parent or worst fit parent among p_1 and p_2 in the mating pool by child2, otherwise discard child2.

Step 8. If both child1 and child2 are discarded then go to step 1.

The proposed crossover has been demonstrated in Fig. 4 with an example for a graph, shown in Fig. 3. Crossover operator generates new paths between n_{p_1} and n_{p_2} , if possible, and add it to the parents to generate children. Thus, the children not only contain the nodes of the parents but also some new nodes. Addition of new nodes takes care of the purpose of mutation operation. So, no further mutation operation is required in our genetic algorithm. We set the termination criteria as the maximum number of generations, denoted as *maxgen.*

Algorithm 2 Proposed Genetic Algorithm for IT2FSPP.

Input: A weighted fuzzy graph with interval type-2 fuzzy arc lengths as the weights, and the source and destination nodes for which the shortest path is to be computed.

Output: Shortest path, i.e., sequence of nodes between the specified source and destination nodes and the corresponding path length in the form of IT2FS as well as crisp value (real number).

Step 1: INITIALIZATION
Generate the initial population of size pop_size as described in Section 4.2.
Set *gen = 0.*

Step 2: FITNESS EVALUATION
Compute the fitness value of each chromosome of the current population using (14).

Step 3: SELECTION
Apply binary tournament selection as described in Section 4.4 for selecting a chromosome from the current population to generate the mating pool. The number of chromosomes in the mating pool is same as *pop_size*.

Step 4: CROSSOVER
Apply the proposed crossover operator as described in Section 4.5 to the mating pool to generate children. Let us call it *children_population*. The number of the children generated by crossover

```
        depends on the crossover probability. If the crossover probability is less than 1, some
        chromosomes are selected randomly from the parent_population and added to the
        children_population such that the size of children_population is also pop_size.

    Step 5: TERMINATION
        If (gen = = maxgen) terminates the algorithm and the chromosome of the current population
        corresponding to highest fitness value is selected as the optimal solution, which represents the
        shortest path. The sequence of nodes in the chromosome describes the path. Addition of IT2FSs
        corresponding to the edges of the shortest path, as defined in Definition 4, provides the length of
        the shortest path in the form of IT2FS and its centroid value. Using (13) provides the length of
        the shortest path in crisp value.
        else
          Set current_population=children_population
          gen = gen +1.
          go to Step 2.
```

5.0 RESULTS

We have tested our algorithm on a graph [20] with 20 nodes and 49 edges. Nodes are numbered arbitrarily from 1 to 20, shown in Fig. 3. Let the source node be 1 and the destination node 20. The arc lengths of the graph are shown in Table 1 in the form of trapezoidal IT2FSs. We did not find any graph in the literature whose arc lengths are given in the form of IT2FSs. For the graph shown in Fig. 3, we have taken nine possible values of interval type-2 fuzzy arc lengths from [30]. They are indexed from one to nine as shown in Table 1.

We have considered the average values of the path lengths in different generation from 10 runs of the proposed algorithm. The crossover probability is considered as 0.9. The average path length and number of generations are plotted in Fig. 5 for different population sizes (15, 20, 25, 35 and 40). For each of the cases, the shortest path length is found as 2.209 for which the corresponding path is 1 - 4 - 9 - 14 - 20. The curve shows that the bigger the size of the population, the smaller the required number of generation for convergence.

Table 1: The arc lengths of the fuzzy graph, represented as IT2FSs

Index	IT2FSs
1	((0, 0, 0.14, 1.97; 1, 1) (0, 0, 0.05, 0.66; 1, 1))
2	((0, 0, 0.14, 1.97; 1, 1) (0, 0, 0.01, 0.63; 1, 1))
3	((0, 0, 0.26, 2.63; 1, 1) (0, 0, 0.05, 0.63; 1, 1))
4	((0, 0, 0.36, 2.63; 1, 1) (0, 0, 0.05, 0.63; 1, 1))
5	((0, 0, 0.64, 2.47; 1, 1) (0, 0, 0.10, 1.16; 1, 1))
6	((0, 0, 0.64, 2.63; 1, 1) (0, 0, 0.09, 0.99; 1, 1))
7	((0.59, 1.50, 2.00, 3.41; 1, 1) (0.79 1.68 1.68 2.21 0.74 .74))
8	((0.38, 1.50, 2.50, 4.62; 1, 1) (1.09, 1.83, 1.83, 2.21; 0.53, .53))
9	((0.09, 1.25, 2.50, 4.62; 1, 1) (1.67, 1.92, 1.92, 2.21; 0.30, .30))

There exists no algorithm that is based on GA for FSPP with interval type-2 fuzzy arc lengths. So, it is not possible for us to compare the performance of our algorithm with any other algorithm existing in the literature.

In this study, we introduce a new crossover method for the proposed genetic algorithm. Crossover operator plays an important role in the proposed GA and improves the performance of the algorithm. To show the superiority of the proposed crossover/algorithm, it has been compared with two existing crossover methods.

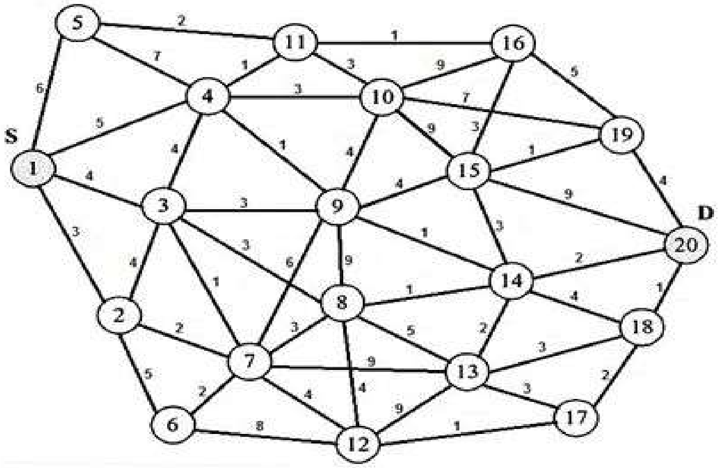


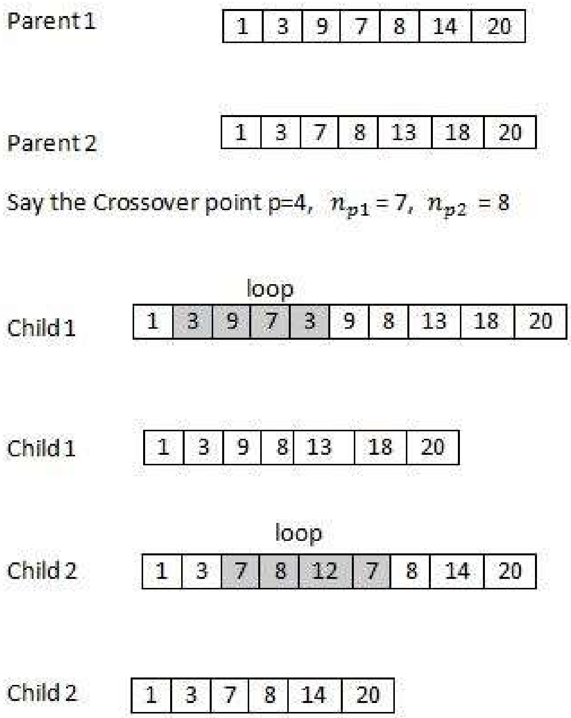
Fig. 3: A graph with 20 nodes and 49 edges

For comparison purpose, we have used following two methods:

Method 1. Crossover operator used in N. Selvanathan and W. J. Tee [33] and one point swapping mutation [33].

Method 2. Crossover operator used in F. T. Lin [34] and one point swapping mutation [33].

Note that, for fair comparison, we use our proposed algorithm, i.e., Algorithm 1, for generation of initial population for both the methods and also use mutation operator described in [33] for those methods. Here, it is to be mentioned that no mutation is required for the proposed crossover.



Parent1 | 1 | 3 | 9 | 7 | 8 | 14 | 20 |

Parent2 | 1 | 3 | 7 | 8 | 13 | 18 | 20 |

Say the Crossover point p=4, $n_{p1}=7$, $n_{p2}=8$

loop

Child1 | 1 | 3 | 9 | 7 | 3 | 9 | 8 | 13 | 18 | 20 |

Child1 | 1 | 3 | 9 | 8 | 13 | 18 | 20 |

loop

Child 2 | 1 | 3 | 7 | 8 | 12 | 7 | 8 | 14 | 20 |

Child 2 | 1 | 3 | 7 | 8 | 14 | 20 |

Fig. 4: Illustration of crossover operator

Fig. 6 shows that the proposed GA outperforms the other two methods. Our algorithm converges faster than the other two methods. The proposed initial population generation uses a list of rejected nodes to keep track of the nodes all of whose neighbors are already visited. It eliminates the effort for regenerating the subpaths which are already generated. There is no mutation operator as the proposed crossover method itself introduces new nodes in the existing paths. The crossover operator eliminates the loop, if found in the path, very efficiently. It ignores the loop and considers the repeated node only once. These features make the proposed GA fast without compromising the quality of solution, i.e., values of optimal path length.

Despite of the promising result of our algorithm, it is to be mentioned that the proposed algorithm will take more time for the defuzzification process, i.e., centroid-based ranking using IT2FS compared to T1FS, which is compromised by the quality of the solution of the proposed algorithm. Other defuzzification techniques existing in the literature [44], [45] can also be used in the proposed algorithm.

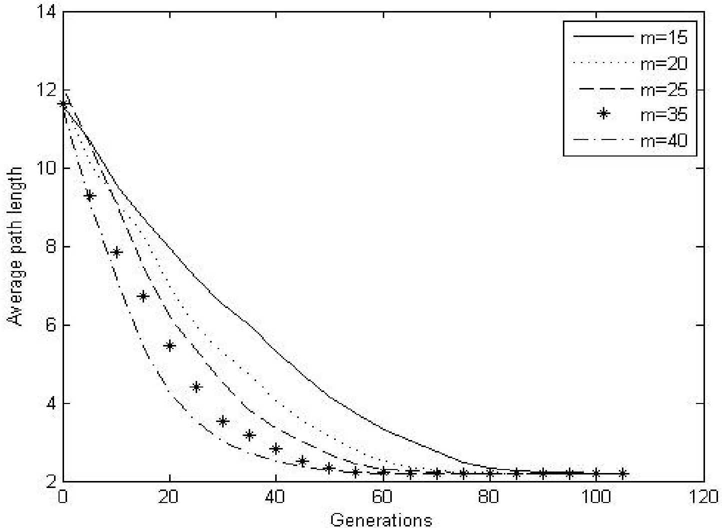


Fig. 5: Average path length with number of generations for different population size

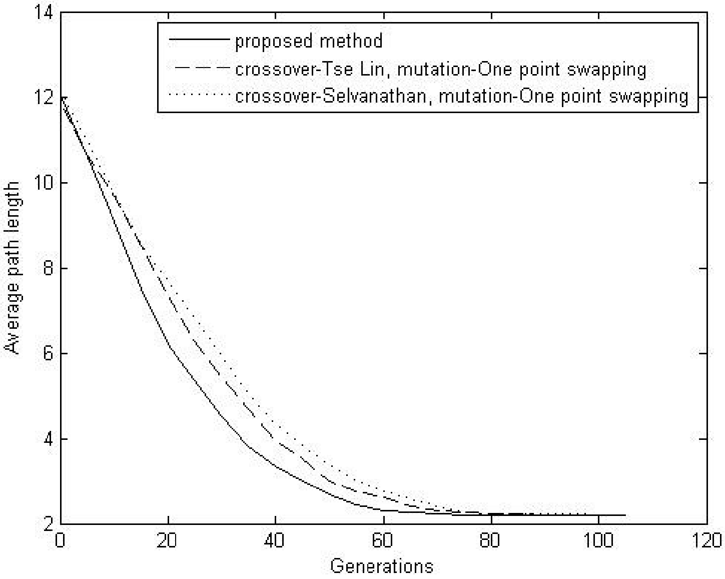


Fig. 6: Comparison different crossover methods

6.0 CONCLUSION

FSPP is one of the most popular problems since it has a wide range of applications in different areas. In this study, we have proposed an approach, based on genetic algorithm (GA), by incorporating the uncertainty using IT2FSs for SPP from a single source node to single destination node. IT2FS can capture more uncertainty than T1FS. We have used centroid-based rank value as fitness measure of the corresponding path. An algorithm for initial population generation is also proposed, which eliminates the effort for regenerating the subpaths that have already been considered. We have introduced a new crossover which makes our GA fast without using any mutation operator. The crossover operator has also avoided the loops, if found in the path, very efficiently. The effectiveness of the proposed algorithm is demonstrated by an example graph. The proposed model is simple and fast and has the flexibility to consider interval type 2 fuzzy arc lengths in a fuzzy graph.

References

[1] E. Bellman, “On a routing problem”, *Quarterly of Applied Mathematics* Vol. 16 No. 1, 1958, pp. 87- 90.

[2] E.W. Dijkstra, “A note on two problems in connection with graphs”, *Numerical Mathematics*, Vol. 1 No. 1, 1959, pp. 269–271.

[3] R.W. Floyd, “Algorithm-97-shortest path”, *Communications of the ACM*, Vol. 5 No. 6, 1962, pp. 345.

[4] S. Dreyfus, “An appraisal of some shortest path algorithms”, *Operations Research,* Vol. 17 No. 3, 1969, pp. 395–412.

[5] H. Frank, “Shortest paths in probability graphs”, *Operations Research*, Vol. 17 No. 4, 1969, pp. 583–599.

[6] G. Mujtaba, L. Shuib, R.G. Raj, R. Rajandram, K. Shaikh, M.A. Al-Garadi, "Automatic ICD-10 multi-class classification of cause of death from plaintext autopsy reports through expert-driven feature selection", PLoS ONE, Vol 12(2): e0170242, 2017. https://doi.org/10.1371/journal.pone.0170242.

[7] R. Hall, “The fastest path through a network with random time-dependent travel time”, *Transportation Science*, Vol. 20 No.3, 1986, pp. 182–188.

[8] P. Loui, “Optimal paths in graphs with stochastic or multidimensional weights”, *Communications of the ACM*, Vol. 26 No. 9, 1983 pp. 670–676.

[9] P.B. Mirchandani, “Shortest distance and reliability of probabilistic networks”, *Computers Operations Research*, Vol. 3 No. 4, 1976, pp. 347–355.

[10] D. Dubois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, New York, 1980.

[11] S. Chanas and J. Kamburowski, “The fuzzy shortest route problem” *in: J.Albrycht, H. Wisniewski (Eds.), Interval and Fuzzy Mathematics, Proc. Polish Symp.,* Technical University of Poznan, Poznan, 1983, pp.35-41.

[12] C. M. Klein, “Fuzzy shortest paths”, *Fuzzy Sets and Systems*, Vol. 39, 1991, pp. 27–41.

[13] R. Yager, “Paths of least resistance on possibilistic production systems”, *Fuzzy Sets and Systems*, Vol. 19, 1986, pp. 121–132.

[14] S. Okada and T. Soper, “A shortest path problem on a network with fuzzy arc lengths”, *Fuzzy Sets and Systems*, Vol. 109, 2000, pp. 129-140.

[15] M.Blue, B. Bush and J. Puckett, “Unified approach to fuzzy graph problems”, *Fuzzy Sets and Systems*, Vol. 125, 2002, pp. 355-368.

[16] S. Okada, “Fuzzy shortest path problems incorporating interactivity among paths”, *Fuzzy Sets and Systems*,Vol. 142 No. 3, 2004, pp. 335-357.

[17] S. M. A. Nayeem and M. Pal, “Shortest path problem on a network with imprecise edge weight”, *Fuzzy Optimization Decision Making*, vol. 4, 2005, pp. 293-312.

[18] K. Islam, and R.G. Raj, "Real-Time (Vision-Based) Road Sign Recognition Using an Artificial Neural Network", Sensors, Vol. 17(4), 2017, pp. 853. MDPI AG. http://dx.doi.org/10.3390/s17040853.

[19] S. Moazeni, “Fuzzy shortest path problem with finite fuzzy quantities”, *Applied Mathematics and Computation*, Vol.183, 2006, pp. 160-169.

[20] F. Hernandes, M. T. Lamata, J. L. Verdegay, and A. Yamakami, “The shortest path problem on networks with fuzzy parameters”, *Fuzzy Sets and Systems*, Vol. 158, 2007, pp. 1561-1570.

[21] J. R Yu and T. H. Wei, “Solving the fuzzy shortest path problem by using a linear multiple objective programming”, *Journal of the Chinese Institute of Industrial Engineers*, Vol. 24 No. 5, 2007, pp. 360 –365.

[22] I. Mahdavi, R Nourifar, A. Heidarzade, and N. M. Amiri, “A dynamic programming approach for finding shortest chains in fuzzy network”, *Applied Soft Computing*, Vol. 9 No. 2, 2009, pp. 503 –511.

[23] T. Hasuike, “Robust shortest path problem based on a confidence interval in fuzzy bicriteria decision making”, *Information Sciences*, Vol. 221, 2013, pp. 520-533.

[24] L. W. Lee and S .M. Chen, “A new method for fuzzy multiple attributes group decision-making based on the arithmetic operations of interval type-2 fuzzy sets”, in *Proc International Conference on Machine Learning and Cybernetics,* New York, Vol. 17, 2008, pp. 3084-3089.

[25] N. Karnik and J. M. Mendel, “Centroid of a type-2 fuzzy set”, *Information Sciences,* Vol. 132, 2001, pp. 195-220.

[26] J.M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New irections*, Prentice-Hall,Upper Saddle River, NJ, 2001.

[27] R. R. Yager, “Ranking fuzzy subsets over the unit interval”, in *Proceedings of the IEEE conference on Decision and Control*, Vol. 17, 1978, pp. 1435-1437.

[28] J. M. Mendel and F. Liu, “Super-exponential convergence of the Karnik-Mendel algorithms for computing the centroid of an interval type-2 fuzzy set”, *IEEE Transactions on Fuzzy Systems*, Vol. 15, 2007, pp. 309-320.

[29] J. M. Mendel, “On centroid calculations for type-2 fuzzy sets”, *Appl. Comput. Math*, Vol. 10, 2011, pp. 88-96.

[30] D. Wu and J. M. Mendel, “A comparative study of ranking methods, similarity measures and uncertaintymeasures for interval type-2 fuzzy sets”, *Information Sciences*, Vol. 179, 2009, pp. 1169-1192.

[31] J. M. Mendel, R. I. John, and F. Liu, “Interval type-2 fuzzy logic systems made simple”, *IEEE Transactionson Fuzzy Systems*, Vol. 14 No. 6, 2006, pp. 808-821.

[32] R. Hassanzadeh, I. Mahdavi, N. M. Amiri, and A. Tajdin, “A genetic algorithm for solving fuzzy shortest path problems with mixed fuzzy arc lengths”, *Mathematical and Computer Modelling*, Vol. 57, 2013, pp. 84–99.

[33] N. Selvanathan and W. J. Tee, “A genetic algorithm solution to solve the shortest path problem in OSPF and MPLS”, *Malaysian Journal of Computer Science*, Vol. 16 No.1, June 2003, pp. 58–67.

[34] F-T. Lin, "Solving the transportation problem with fuzzy coefficients using genetic algorithms, *FUZZ-IEEE*, 2009", pp. 1468–1473.

[35] D. E. Goldberg, *Genetic Algorithm in Search, Optimization, and Machine Learning. Reading*, MA: Addision-Wesley, 1989.

[36] Y. Zhang, Z. Zhang, Y. Deng, and S. Mahadevan, "A biologically inspired solution for fuzzy shortest path problems", *Applied Soft Computing*, Vol. 13 No. 5, 2013, pp. 2356-2363.

[37] V. Anusuya and R. Sathyaa, "Shortest path with complement of type-2 fuzzy number", *Malaya J. Mat.*, Vol. 1, 2013, pp. 71-76.

[38] Y. Gao, "Shortest path problem with uncertain arc lengths", *Computers and Mathematics with Applications*, Vol. 62, 2011, pp. 2591–2600.

[39] V. Anusuya and R. Sathyaa, "Type-2 fuzzy shortest path on similarity measure", *Bull. Math. Stat. Res.*, Vol. 2, 2014, pp. 418- 422.

[40] J. M. Mendel and R. I. B. John, "Type-2 Fuzzy Sets Made Simple", *IEEE Transactions on Fuzzy Systems*, Vol. 10 No. 2, 2002 , pp. 117-127.

[41] L. A. Zadeh, "Fuzzy logic = computing with words", *IEEE Transactions on Fuzzy Systems*, Vol. 4, 1996, pp. 103-111.

[42] E. Hisdal, "The IF THEN ELSE statement and interval-valued fuzzy sets if higher Type", *Int. J. Man-Machine Studies*, Vol. 15, 1981, pp. 385-455.

[43] T. Dereliand and K. Altun, "Technology evaluation through the use of interval type-2 fuzzy sets and systems", *Computers and Industrial Engineering*, Vol. 65 No. 4, 2013, pp. 624-633.

[44] S. M. Chen and L. W. Lee, "Fuzzy multiple attributes group decision making based on the interval type-2 TOPSIS method", *Expert Systems with Applications*, Vol. 37 No. 4, 2010, pp. 2790–2798.

[45] W. Wang, X. Liu and Y. Qin, "Multi-attribute group decision making models under interval type-2 fuzzy environment", *Knowledge-Based Systems*, Vol. 30, 2012, pp. 121-128.

[46] T. Chen, "A linear assignment method for multiple-criteria decision analysis with interval type-2 fuzzy sets", *Applied Soft Computing Journal*, Vol. 13 No. 5, 2013, pp. 2735-2748.

[47] A. Dey, A. Pal and T. Pal, "Interval type 2 fuzzy set in fuzzy shortest path problem", *Mathematics*, Vol. 4 No. 4, 2016, pp. 1-19.