Modified artificial bee colony algorithm for solving mixed interval-valued fuzzy shortest path problem

Ali Ebrahimnejad¹ · Mohammad Enayattabr² · Homayun Motameni³ · Harish Garg⁴

Received: 31 August 2020 / Accepted: 18 January 2021 / Published online: 18 February 2021
© The Author(s) 2021

Abstract

In recent years, numerous researchers examined and analyzed several different types of uncertainty in shortest path (SP) problems. However, those SP problems in which the costs of arcs are expressed in terms of mixed interval-valued fuzzy numbers are less addressed. Here, for solving such uncertain SP problems, first a new procedure is extended to approximate the summation of mixed interval-valued fuzzy numbers using alpha cuts. Then, an extended distance function is introduced for comparing the path weights. Finally, we intend to use a modified artificial bee colony (MABC) algorithm to find the interval-valued membership function of SP in such mixed interval-valued fuzzy network. The proposed algorithm is illustrated via two applications of SP problems in wireless sensor networks and then the results are compared with those derived from genetic and particle swarm optimization (PSO) algorithms, based on three indexes convergence iteration, convergence time and run time. The obtained results confirm that the MABC algorithm has less convergence iteration, convergence time and implementation time compared to GA and PSO algorithm.

Keywords Shortest path problem · Mixed interval-valued fuzzy numbers · Artificial bee colony algorithm · Wireless sensor network

Introduction

One of the practical applications of network theory in solving real-world problems is to solve the problem of the shortest path in the network. The aim of shortest path (SP) problem is to determine a path with at least time or cost between two predefined nodes [3]. Many of the real-life optimization problems such as communication and telecommunications systems, transportation system, location issue, computer networks, investment planning and so on can be modeled

as SP problems. Therefore, many scholars have researched on this area.

Conventional SP problems are formulated with crisp weights, which may not always be available in real-world applications. However, in real-world problems, cost and time values are often imprecise or vague in nature as they depend on weather and traffic conditions. The imprecision in the cost and time data of the SP problem can be handled by fuzzy sets. The SP problem is extended to fuzzy SP (FSP) problem in which are weights (time and cost) are represented in terms of fuzzy numbers. Up to now, several approaches have been developed for solving FSP problems. Indeed, directed approach and heuristic approach are two important techniques for finding FSP.

The main idea behind of direct approach is to generalize the classical algorithms for solving FSP problems. Okada and Soper [35] proposed an algorithm for solving FSP problem by determining possibility degree for each arc being on optimal fuzzy path. Nayeem and Pal [34] proposed a new approach that gives a guideline to the decision maker for choosing the best fuzzy SP if it does not find the unique fuzzy SP. Moazeni [33] extended classical Dijkstra algorithm for obtaining all non-dominated paths in a fuzzy network.

✉ Ali Ebrahimnejad
aemarzoun@gmail.com; a.ebrahimnejad@qaemiau.ac.ir

¹ Department of Mathematics, Qaemshahr Branch, Islamic Azad University, Qaemshahr, Iran

² Department of Computer Engineering, Babol Branch, Islamic Azad University, Sari, Iran

³ Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran

⁴ School of Mathematics, Thapar Institute of Engineering and Technology, Deemed University, Patiala, Punjab 147004, India

Chuang and Kung [10] proposed an algorithm that gives the shortest path and the fuzzy shortest weight simultaneously in a discrete FSP problem based on similarity measure. Yu and Wei [47] obtained the FSP based on a linear multi-objective programming problem without using integer variables. Ji et al. [29] formulated three kinds of FSP model based on the credibility measure of fuzzy set theory and proposed a hybrid intelligent algorithm for solving the FSP problem. Mahdavi et al. [32] extended the traditional Floyd–Warshall algorithm to find the FSP from one node to all other nodes. Dou et al. [13] first defined the concepts of the best and the worst ideal paths in a fuzzy network and then determined the FSP by evaluating similarity degrees between all candidate paths and two ideal paths based on similarity measure. The fuzzy reliable SP problem in a mixed fuzzy network has been investigated by Yang et al. [42]. Eshaghezhad et al. [24] proposed an artificial neural network model for solving the FSP problem. Lakdashti et al. [31] presented an algorithm to find the SP in a directed network with triangular vague weights. Dey et al. [12] proposed an algorithm to find the fuzzy minimum spanning tree of an undirected weighted fuzzy graph, in which mixed fuzzy numbers, either triangular or trapezoidal, are used to represent the lengths/costs of the arcs. Ebrahimnead [21] proposed an approach for solving interval shortest path problem based on acceptability index. Abbaszadeh Sori et al. [1] used fuzzy inference system for solving the fuzzy-constrained SP problem with three objectives of cost, time and risk. An overview on solving FSP problem based on different algorithms can be found in Broumi et al. [5].

In recent years, numerous researchers have used heuristic algorithms to solve many complicated optimization problems at the appropriate time [2, 38, 40]. In particular, heuristic approach is an important tool for solving the FSP problem. The genetic algorithm (GA) proposed by Hassanzadeh et al. [28] obtained the SP in a directed graphs with various fuzzy parameters. Particle Swarm Optimization (PSO) algorithm and Artificial Bee Colony (ABC) algorithm proposed by Ebrahimejad et al. [18, 19] had better performance in terms of run time compared to GA.

To the best of our knowledge, there are very few studies in SP literature indicating the uncertainties of arc weights by means of interval-valued fuzzy numbers (IVFNs). Dey et al. [11] extended GA for solving the SP problem with interval-valued trapezoidal fuzzy arc weights. Enayattabar et al. [22] extended Dijkstra algorithm to solve the SP problem in an interval-valued Pythagorean fuzzy network. Enayattabar et al. [23] determined the shortest weights between every pair of nodes in a given interval-valued fuzzy network based on a dynamic approach. Broumi et al. [6] considered the SP problem through Bellman's algorithm for a network using interval-valued neutrosophic numbers. These approaches are not effective in the case that arc weights are represented

in terms of various IVFNs. That is why in this research, the issue of finding such SP with different kinds of interval-valued fuzzy weights is perused. In this contribution, imprecise arc weights have interval-valued trapezoidal and normal fuzzy membership functions. As the SP problem in fuzzy environment is a complex problem in combinatorial optimization, it is time consuming or even difficult to find an exact solution for large samples. Due to efficiency of applying collective intelligence algorithms to solve optimization problems, we intend to use the modified version of the ABC algorithm for solving mixed interval-valued FSP (MIVFSP) in a mixed fuzzy directed graph. It is able to solve the problem under consideration in less time compared to other intelligence algorithms such as GA and PSO. This algorithm is particularly efficient when it requires a quasi-optimal solution in a short time.

As mentioned, one application of SP problems WSNs is investigated in this study. WSNs are comprised of a huge count of low-power devices generally termed as sensor nodes. Sensor devices are considered as the nodes with resource-constraint regarding storage capacity, computational power, communication range energy, and bandwidth. On the basis of application circumstances, the placing of sensor nodes is either accurately made within the area of interest or employed randomly in the field. WSNs are very suitable for monitoring and control in remote or aggressive environments. Each sensor node senses the event as soon as an event occurs in the environment and sends it to the central station with a message. The longer the message is transmitted on the network, the greater the energy consumption of the sensor nodes. Generally, WSNs are responsible for gathering data from its environment and sending it to the sink node. The energy sources in WSNs limit the lifetime of the networks [9, 26]. It is possible to extend the lifetime of the network by saving as much energy as possible. Hence, it is very important to obtain the shortest path for passing data in WSNs to decrease the energy consumption. Finding the SP in such network ensures lower energy consumption by the nodes and increases the lifetime of the WSN. However, the amount of energy consumption in a WSN is imprecise. Thus, in this study, we present the imprecision in data of WSN by means of mixed interval-valued fuzzy data.

On such motivation basis, the main contributions of this study are summarized as follows: (1) to the best of our knowledge, this study is the first attempt for formulating the SP problem in a mixed interval-valued fuzzy environment, (2) a new approach is proposed to compare mixed interval-valued fuzzy numbers, (3) a MABC algorithm is extended for finding the mixed interval-valued fuzzy shortest path, (4) in contrast to GA and PSO algorithm, the MABC algorithm has less convergence iteration, convergence time and implementation time, (5) two simulations on WSNs confirm that the MABC algorithm is quite robust to recognize

the shortest path among all paths of networks in terms of energy consumption.

In Table 1, a comparison study of new incorporations with the previous studies has been illustrated.

The remainder of this paper is organized as follows. The necessary concepts of mixed interval-valued fuzzy numbers (MIVFN), an approximate technique for summing up two

MIVFNs and comparison of two MIVFNs are described in the “Mixed interval-valued fuzzy numbers”. In the “Modified ABC (MABC) algorithm for solving MIVFSP problem”, we present modified ABC algorithm for solving MIVFSP problem in a mixed fuzzy-oriented graph. In the “Experiments and results”, two applications of SP problems in WSNs are solved by the proposed algorithm and the

Table 1 Classification of different approaches for solving FSP problem

Articles	Membership function	Advantages	Disadvantages	Algorithm	
				Classical	Heuristic
Okada and Soper [35]	TFN and TrFN	Providing optimal path	Not providing guidance to DM	Dijkstra	
Chuang and Kung [10]	TFN	Providing both optimal fuzzy cost and fuzzy path	Useful only for discrete FSP		Fuzzy simulation
Ji et al. [29]	TrFN	Taking into account the views of DM	Not providing optimal path		GA
Mahdavi et al. [32]	TFN and TrFN	Providing both optimal fuzzy cost and fuzzy path	The optimal path depends on the type of ranking function	Floyd–Warshall	
Eshaghnezhad et al. [24]	TFN	Providing both optimal fuzzy cost and fuzzy path	High Complexity		Artificial neural network model
Abbaszadeh Sori et al. [1]	TFN	Efficient for multi-objective FSP	High complexity Not providing optimal path		Fuzzy inference system
Hassanzadeh et al. [28]	Mixed fuzzy numbers	Efficient for complex network Having less implementation time than Floyd–Warshall algorithm	Not providing exact optimal path		GA
Ebrahiminejad et al. [18]	Mixed fuzzy numbers	Efficient for complex networks Having less implementation time than GA	Not providing exact optimal path		PSO
Ebrahiminejad et al. [19]	Mixed fuzzy numbers	Efficient for complex networks Having less implementation time than GA and PSO	Not providing exact optimal path		ABC
Enayattabar et al. [23]	Interval-valued Pythagorean fuzzy	Providing all pairs FSP	Not efficient for complex networks	Dynamic approach	
Broumi et al. [6]	Interval-valued neutrosophic numbers	Providing both optimal fuzzy cost and fuzzy path	Not efficient for complex networks	Bellman’s algorithm	
Dey et al. [11]	Interval-valued TrFN	High complexity compared to PSO and ABC	Not providing all pairs SP		GA
Present Study	Mixed interval-valued fuzzy numbers	Having less convergence iteration, convergence time and implementation time compared to GA and PSO algorithm	Not providing all pairs SP		Modified ABC

obtained results are compared with those derived from GA and PSO. Finally, the “Conclusions” concludes the paper.

Mixed interval-valued fuzzy numbers

The aim of this section to review some essential definitions and preliminaries of mixed interval-valued fuzzy numbers [8, 16, 20, 24, 41].

Definition 1 A fuzzy set $\tilde{A}_{\text{TrFN}}$, denoted by $\tilde{A}_{\text{TrFN}} = (a_1, a_2, a_3, a_4; h)$ $0 < h \leq 1$ $0 < h \leq 1$, is called a level h -trapezoidal fuzzy number (TrFN) if its membership function is given as follows:

$$\mu_{\tilde{A}_{\text{TrFN}}}(x) = \begin{cases} h \frac{x-a_1}{a_2-a_1}, & a_1 \leq x \leq a_2, \\ h, & a_2 \leq x \leq a_3, \\ h \frac{a_4-x}{a_4-a_3}, & a_3 \leq x \leq a_4, \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

We denote the set of all level h -TrFNs by $F_{\text{TrFN}}(h)$.

Remark 1 The α -cut for the level h -TrFN $\tilde{A}_{\text{TrFN}} = (a_1, a_2, a_3, a_4; h)$, i.e., $\{x \in \mathbb{R}; \mu_{\tilde{A}_{\text{TrFN}}}(x) \geq \alpha\}$ is given by

$$\begin{aligned} [\tilde{A}_{\text{TrFN}}]_{\alpha} &= \left[[\tilde{A}_{\text{TrFN}}]_{\alpha}^l, [\tilde{A}_{\text{TrFN}}]_{\alpha}^r \right] \\ &= \left[a_1 + (a_2 - a_1) \frac{\alpha}{h}, a_4 - (a_4 - a_3) \frac{\alpha}{h} \right]. \end{aligned} \quad (2)$$

Definition 2 Let $\tilde{A}_{\text{TrFN}}^L \in F_{\text{TrFN}}(h^L)$ and $\tilde{A}_{\text{TrFN}}^U \in F_{\text{TrFN}}(h^U)$. A level (h^L, h^U) -interval-valued

trapezoidal fuzzy number (IVTrFN) $\tilde{\tilde{A}}_{\text{TrFN}}$, denoted by $\tilde{\tilde{A}}_{\text{TrFN}} = [\tilde{A}_{\text{TrFN}}^L, \tilde{A}_{\text{TrFN}}^U] = \langle (a_1^L, a_2^L, a_3^L, a_4^L; h^L), (a_1^U, a_2^U, a_3^U, a_4^U; h^U) \rangle$, is an interval-valued fuzzy set (IVFS) on $\mathbb{R}$. The membership function of the lower TrFN $\tilde{A}_{\text{TrFN}}^L$ is expressed as:

$$\mu_{\tilde{A}_{\text{TrFN}}^L}(x) = \begin{cases} h^L \frac{x-a_1^L}{a_2^L-a_1^L}, & a_1^L \leq x \leq a_2^L, \\ h^L, & a_2^L \leq x \leq a_3^L, \\ h^L \frac{a_4^L-x}{a_4^L-a_3^L}, & a_3^L \leq x \leq a_4^L, \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

The membership function of the upper TrFN $\tilde{A}_{\text{TrFN}}^U$ is expressed as:

$$\mu_{\tilde{A}_{\text{TrFN}}^U}(x) = \begin{cases} h^U \frac{x-a_1^U}{a_2^U-a_1^U}, & a_1^U \leq x \leq a_2^U, \\ h^U, & a_2^U \leq x \leq a_3^U, \\ h^U \frac{a_4^U-x}{a_4^U-a_3^U}, & a_3^U \leq x \leq a_4^U, \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

here, $0 < h^L \leq h^U \leq 1$, $a_1^U \leq a_1^L$ and $a_4^L \leq a_4^U$ (Fig. 1).

We denote the set of (h^L, h^U) -IVTrFNs by $F_{\text{IVTrFN}}(h^L, h^U)$.

Example 1 $\tilde{\tilde{A}}_{\text{TrFN}} = [\tilde{A}_{\text{TrFN}}^L, \tilde{A}_{\text{TrFN}}^U] = \langle (16, 20, 26, 30; 0.7), (14, 18, 28, 32; 0.9) \rangle$ is a level $(0.7, 0.9)$ -interval-valued trapezoidal fuzzy number where the lower and upper membership functions are given as follows:

Fig. 1 The interval-valued trapezoidal fuzzy number $\tilde{\tilde{A}}$

$$\mu_{\tilde{A}_{\text{TrFN}}^L}(x) = \begin{cases} \frac{7}{10} \left(\frac{x-16}{4} \right), & 16 \leq x \leq 20, \\ \frac{7}{10}, & 20 \leq x \leq 26, \\ \frac{7}{10} \left(\frac{30-x}{4} \right), & 26 \leq x \leq 30, \\ 0, & x < 16, x > 30. \end{cases}, \quad \mu_{\tilde{A}_{\text{TrFN}}^U}(x) = \begin{cases} \frac{9}{10} \left(\frac{x-14}{4} \right), & 14 \leq x \leq 18, \\ \frac{9}{10}, & 18 \leq x \leq 28, \\ \frac{9}{10} \left(\frac{32-x}{4} \right), & 28 \leq x \leq 32, \\ 0, & x < 14, x > 32. \end{cases}$$

Definition 3 The summation of two (h^L, h^U) -IVTrFNs $\tilde{A}_{\text{TrFN}} = [\tilde{A}_{\text{TrFN}}^L, \tilde{A}_{\text{TrFN}}^U] = \langle (a_1^L, a_2^L, a_3^L, a_4^L; h^L), (a_1^U, a_2^U, a_3^U, a_4^U; h^U) \rangle$ and $\tilde{B}_{\text{TrFN}} = [\tilde{B}_{\text{TrFN}}^L, \tilde{B}_{\text{TrFN}}^U] = \langle (b_1^L, b_2^L, b_3^L, b_4^L; h^L), (b_1^U, b_2^U, b_3^U, b_4^U; h^U) \rangle$ is defined as follows:

$$(\tilde{A} \oplus \tilde{B})_{\text{TrFN}} = \langle (a_1^L + b_1^L, a_2^L + b_2^L, a_3^L + b_3^L, a_4^L + b_4^L; h^L), (a_1^U + b_1^U, a_2^U + b_2^U, a_3^U + b_3^U, a_4^U + b_4^U; h^U) \rangle. \quad (5)$$

Definition 4 A level-normal fuzzy number (NFN) $\tilde{A}_{\text{NFN}}$, denoted by $\tilde{A}_{\text{NFN}} = (m, \sigma; k)$, $0 < k \leq 1$, is a fuzzy set on $\mathbb{R}$ with the membership function as follows:

$$\mu_{\tilde{A}_{\text{NFN}}}(x) = ke^{-\left(\frac{x-m}{\sigma}\right)^2}, \quad x \in \mathbb{R}. \quad (6)$$

We denote the set of all level k -NFNs by $F_{\text{NFN}}(k)$.

Remark 2 The α -cut for the level k -normal fuzzy number $\tilde{A}_{\text{NFN}} = (m, \sigma; k)$ is given by

$$[\tilde{A}_{\text{NFN}}]_{\alpha} = [\tilde{A}_{\text{NFN}}]_{\alpha}^L, [\tilde{A}_{\text{NFN}}]_{\alpha}^R = \left[m - \sigma \sqrt{-\ln \frac{\alpha}{k}}, m + \sigma \sqrt{-\ln \frac{\alpha}{k}} \right]. \quad (7)$$

Definition 5 Let $\tilde{A}_{\text{NFN}}^L \in F_{\text{NFN}}(k^L)$ and $\tilde{A}_{\text{NFN}}^U \in F_{\text{NFN}}(k^U)$. A level (k^L, k^U) -interval-valued normal fuzzy number (IVNFN) $\tilde{\tilde{A}}_{\text{NFN}}$, denoted by $\tilde{\tilde{A}}_{\text{NFN}} = [\tilde{A}_{\text{NFN}}^L, \tilde{A}_{\text{NFN}}^U] = \langle (m, \sigma^L; k^L), (m, \sigma^U; k^U) \rangle$, is an IVFS on $\mathbb{R}$. In this case, the membership function of the lower NFS $\tilde{A}_{\text{NFN}}^L$ is given as

$$\mu_{\tilde{A}_{\text{NFN}}^L}(x) = k^L e^{-\left(\frac{x-m}{\sigma^L}\right)^2}, \quad x \in \mathbb{R}. \quad (8)$$

The membership function of the upper NFN $\tilde{A}_{\text{NFN}}^U$ is given as

$$\mu_{\tilde{A}_{\text{NFN}}^U}(x) = k^U e^{-\left(\frac{x-m}{\sigma^U}\right)^2}, \quad x \in \mathbb{R}. \quad (9)$$

here, $0 < k^L \leq k^U \leq 1$, $\sigma^L \leq \sigma^U$ and $a_4^L \leq a_4^U$ (Fig. 2).

In a similar way, $F_{\text{IVNFN}}(k^L, k^U)$ denotes the set of all level (k^L, k^U) -IVNFNs.

Example 2 $\tilde{\tilde{A}}_{\text{NFN}} = [\tilde{A}_{\text{NFN}}^L, \tilde{A}_{\text{NFN}}^U] = \langle (5, 7; 0.6), (5, 10; 0.8) \rangle$ is a level $(0.6, 0.8)$ -interval-valued normal fuzzy number where the lower and upper membership functions are given as follows:

$$\mu_{\tilde{A}_{\text{NFN}}^L}(x) = \frac{6}{10} e^{-\left(\frac{x-5}{7}\right)^2}, \quad \mu_{\tilde{A}_{\text{NFN}}^U}(x) = \frac{8}{10} e^{-\left(\frac{x-5}{10}\right)^2}, \quad x \in \mathbb{R}$$

Definition 6 The extended addition on $\tilde{\tilde{A}}_{\text{NFN}} = [\tilde{A}_{\text{NFN}}^L, \tilde{A}_{\text{NFN}}^U] = \langle (m_1, \sigma_1^L; k^L), (m_1, \sigma_1^U; k^U) \rangle$ and $\tilde{\tilde{B}}_{\text{NFN}} = [\tilde{B}_{\text{NFN}}^L, \tilde{B}_{\text{NFN}}^U] = \langle (m_2, \sigma_2^L; k^L), (m_2, \sigma_2^U; k^U) \rangle$ is defined as follows:

$$(\tilde{\tilde{A}} \oplus \tilde{\tilde{B}})_{\text{NFN}} = \langle (m_1 + m_2, \sigma_1^L + \sigma_2^L; k^L), (m_1 + m_2, \sigma_1^U + \sigma_2^U; k^U) \rangle. \quad (10)$$

Remark 3 The α -cut sum of the level h -TrFN $\tilde{A}_{\text{TrFN}} = (a_1, a_2, a_3, a_4; h)$ and the level k -NFN $\tilde{A}_{\text{NFN}} = (m, \sigma; k)$, denoted by $[\tilde{C}]_{\alpha_i} = [\tilde{C}]_{\alpha_i}^L, [\tilde{C}]_{\alpha_i}^R$, is equal to sum of the α -cuts of them:

$$[\tilde{C}]_{\alpha_i} = [\tilde{C}]_{\alpha_i}^L, [\tilde{C}]_{\alpha_i}^R = [\tilde{A}_{\text{TrFN}}]_{\alpha_i} + [\tilde{B}_{\text{NFN}}]_{\alpha_i}. \quad (11)$$

Assuming $[\tilde{A}_{\text{TrFN}}]_{\alpha_i} = [\tilde{A}_{\text{TrFN}}]_{\alpha_i}^L, [\tilde{A}_{\text{TrFN}}]_{\alpha_i}^R$ and $[\tilde{B}_{\text{NFN}}]_{\alpha_i} = [\tilde{B}_{\text{NFN}}]_{\alpha_i}^L, [\tilde{B}_{\text{NFN}}]_{\alpha_i}^R$ are the α -cuts of $\tilde{A}_{\text{TrFN}}$ and $\tilde{B}_{\text{NFN}}$, respectively, formulation (11) is given as follows:

$$[\tilde{C}]_{\alpha_i} = [\tilde{A}_{\text{TrFN}}]_{\alpha_i}^L, [\tilde{A}_{\text{TrFN}}]_{\alpha_i}^R + [\tilde{B}_{\text{NFN}}]_{\alpha_i}^L, [\tilde{B}_{\text{NFN}}]_{\alpha_i}^R = [\tilde{A}_{\text{TrFN}}]_{\alpha_i}^L + [\tilde{B}_{\text{NFN}}]_{\alpha_i}^L, [\tilde{A}_{\text{TrFN}}]_{\alpha_i}^R + [\tilde{B}_{\text{NFN}}]_{\alpha_i}^R. \quad (12)$$

Fig. 2 The interval-valued normal fuzzy number

Regarding Eqs. (2) and (7), formulation (12) is simplified as follows:

$$[\tilde{C}]_{\alpha_i} = \left[[\tilde{C}]_{\alpha_i}^l, [\tilde{C}]_{\alpha_i}^r \right] = \left[(a_2 - a_1) \frac{\alpha_i}{h} + a_1 + m - \sigma \sqrt{-\ln \frac{\alpha_i}{k}}, a_4 - (a_4 - a_3) \frac{\alpha_i}{h} + m + \sigma \sqrt{-\ln \frac{\alpha_i}{k}} \right]. \quad (13)$$

Now, we are in a situation to describe an approximate approach for summing up two IVFNs with different membership functions, i.e., (h^L, h^U) -IVTrFN and (k^L, k^U) -IVNFN. The proposed approach is indeed a direct extension of the method of Tajdin et al. [39] for summing a TrFN and a normal one. In what follows, the summation of the (h^L, h^U) -IVTrFN $\tilde{A}_{\text{TrFN}} = \langle (a_1^L, a_2^L, a_3^L, a_4^L; h^L), (a_1^U, a_2^U, a_3^U, a_4^U; h^U) \rangle$ and the (k^L, k^U) -IVNFN $\tilde{A}_{\text{NFN}} = \langle (m, \sigma^L; k^L), (m, \sigma^U; k^U) \rangle$ is approximated by an interval-valued exponential fuzzy number (IVEFN) using α -cut points.

The upper membership function of the IVEFN $\tilde{C} = [\tilde{C}^L, \tilde{C}^U]$, i.e., $\mu_{\tilde{C}^U}(x)$, is given as

$$\mu_{\tilde{C}^U}(x) = \begin{cases} \mu_{\tilde{C}^U}^l(x) = t^U e^{-\left(\frac{m_1^U - x}{\sigma_1^U}\right)^2}, & x < m_1^U, \\ t^U = \min\{h^U, k^U\}, & m_1^U \leq x \leq m_2^U, \\ \mu_{\tilde{C}^U}^r(x) = t^U e^{-\left(\frac{x - m_2^U}{\sigma_2^U}\right)^2}, & x > m_2^U. \end{cases} \quad (14)$$

Let $x_i^U = [\tilde{C}^U]_{\alpha_i}^l$ and $y_i^U = \mu([\tilde{C}^U]_{\alpha_i}^l)$. To find the parameters (m_1^U, σ_1^U) of the left membership function of (14), we first choose n points (x_i^U, y_i^U) where $\alpha_i = \alpha_{i-1} + \frac{1}{n} \left(\frac{m_1^U - x}{\sigma_1^U} \right)^2, 1, 2, \dots, n$ and then consider the fitting model $y = e^{-\left(\frac{m_1^U - x}{\sigma_1^U}\right)^2}$. To do this we have,

$$\begin{aligned} \ln y_i^U &= -\left(\frac{m_1^U - x_i^U}{\sigma_1^U}\right)^2 \Rightarrow -\ln y_i^U = \left(\frac{m_1^U - x_i^U}{\sigma_1^U}\right)^2 \\ &\Rightarrow \sqrt{-\ln y_i^U} = \left(\frac{m_1^U - x_i^U}{\sigma_1^U}\right) \Rightarrow -\sigma_1^U \sqrt{-\ln y_i^U} + m_1^U = x_i^U. \end{aligned} \quad (15)$$

In this case, the following linear least squares model is defined for the minimization of error:

$$\min E^U = \sum_{i=1}^n \left(-\sigma_1^U \sqrt{-\ln y_i^U} + m_1^U - x_i^U \right)^2. \quad (16)$$

To solve (16), it is needed to have

$$\begin{aligned} \frac{\partial E^U}{\partial \sigma_1^U} &= \sum_{i=1}^n \left[\left(-2\sqrt{-\ln y_i^U} \right) \left(-\sigma_1^U \sqrt{-\ln y_i^U} + m_1^U - x_i^U \right) \right] = 0, \\ \frac{\partial E^U}{\partial m_1^U} &= \sum_{i=1}^n \left[2 \left(-\sigma_1^U \sqrt{-\ln y_i^U} + m_1^U - x_i^U \right) \right] = 0. \end{aligned} \quad (17)$$

Hence, the following normal equations are solved for the unknown parameters m_1^U and σ_1^U :

$$\begin{aligned} -\sigma_1^U \sum_{i=1}^n \ln y_i^U + m_1^U \sum_{i=1}^n \sqrt{-\ln y_i^U} &= \sum_{i=1}^n x_i^U \sum_{i=1}^n \sqrt{-\ln y_i^U}, \\ -\sigma_1^U \sum_{i=1}^n \sqrt{-\ln y_i^U} + n m_1^U &= \sum_{i=1}^n x_i^U. \end{aligned} \quad (18)$$

Therefore, the parameters m_1^U and σ_1^U are given as follows:

$$\begin{aligned} m_1^U &= \frac{\sum_i \ln y_i^U \times \sum_i x_i^U + \sum_i \left(x_i^U \times \sqrt{-\ln y_i^U} \right) \times \sum_i \sqrt{-\ln y_i^U}}{n \sum_i \ln y_i^U + \sum_i \sqrt{-\ln y_i^U} \times \sum_i \sqrt{-\ln y_i^U}}, \\ \sigma_1^U &= \frac{-n \sum_i \left(x_i^U \times \sqrt{-\ln y_i^U} \right) + \sum_i \sqrt{-\ln y_i^U} \times \sum_i x_i^U}{-n \sum_i \ln y_i^U - \sum_i \sqrt{-\ln y_i^U} \times \sum_i \sqrt{-\ln y_i^U}}. \end{aligned} \quad (19)$$

Now assume $\bar{x}_i^U = [\tilde{C}^U]_{\alpha_i}^r$ and $\bar{y}_i^U = \mu([\tilde{C}^U]_{\alpha_i}^r)$. In a similar way, to obtain the parameters (m_2^U, σ_2^U) of the right membership function of (14), we first choose n points $(\bar{x}_i^U, \bar{y}_i^U)$ where $\alpha_i = \alpha_{i-1} + \frac{1}{n} \left(\frac{\bar{x}_i^U - m_2^U}{\sigma_2^U} \right)^2, 1, 2, \dots, n$ and then consider the fitting model as $y = e^{-\left(\frac{\bar{x}_i^U - m_2^U}{\sigma_2^U}\right)^2}$. The same approach gives the following parameters:

$$\begin{aligned} m_2^U &= \frac{\sum_i \ln \bar{y}_i^U \times \sum_i \bar{x}_i^U + \sum_i \left(\bar{x}_i^U \times \sqrt{-\ln \bar{y}_i^U} \right) \times \sum_i \sqrt{-\ln \bar{y}_i^U}}{n \sum_i \ln \bar{y}_i^U + \sum_i \sqrt{-\ln \bar{y}_i^U} \times \sum_i \sqrt{-\ln \bar{y}_i^U}}, \\ \sigma_2^U &= \frac{n \sum_i \left(\bar{x}_i^U \times \sqrt{-\ln \bar{y}_i^U} \right) - \sum_i \sqrt{-\ln \bar{y}_i^U} \times \sum_i \bar{x}_i^U}{-n \sum_i \ln \bar{y}_i^U - \sum_i \sqrt{-\ln \bar{y}_i^U} \times \sum_i \sqrt{-\ln \bar{y}_i^U}}. \end{aligned} \quad (21)$$

Moreover, the lower membership function of the IVTrFN $\tilde{C} = [\tilde{C}^L, \tilde{C}^U]$, i.e., $\mu_{\tilde{C}^L}(x)$, is given as

$$\mu_{\tilde{C}^L}(x) = \begin{cases} \mu_{\tilde{C}^L}^l(x) = t^L e^{-\left(\frac{m_1^L - x}{\sigma_1^L}\right)^2}, & x < m_1^L, \\ t^L = \min\{h^L, k^L\}, & m_1^L \leq x \leq m_2^L, \\ \mu_{\tilde{C}^L}^r(x) = t^L e^{-\left(\frac{x - m_2^L}{\sigma_2^L}\right)^2}, & x > m_2^L. \end{cases} \quad (23)$$

Assume $x_i^L = [\tilde{C}^L]_{\alpha_i}^l$ and $y_i^L = \mu([\tilde{C}^L]_{\alpha_i}^l)$. The same approach gives the parameters (m_1^L, σ_1^L) as follows:

$$m_1^L = \frac{\sum_i \ln y_i^L \times \sum_i x_i^L + \sum_i \left(x_i^L \times \sqrt{-\ln y_i^L} \right) \times \sum_i \sqrt{-\ln y_i^L}}{n \sum_i \sqrt{\ln y_i^L} + \sum_i \sqrt{-\ln y_i^L} \times \sum_i \sqrt{-\ln y_i^L}}, \quad (24)$$

$$\sigma_1^L = \frac{-n \sum_i \left(x_i^L \times \sqrt{-\ln y_i^L} \right) + \sum_i \sqrt{-\ln y_i^L} \times \sum_i x_i^L}{-n \sum_i \ln y_i^L - \sum_i \sqrt{-\ln y_i^L} \times \sum_i \sqrt{-\ln y_i^L}}. \quad (25)$$

Now, let $\bar{x}_i^L = [\tilde{C}^L]_{\alpha_i}^r$ and $\bar{y}_i^L = \mu([\tilde{C}^L]_{\alpha_i}^r)$. The same approach gives the parameters (m_2^L, σ_2^L) as follows:

$$m_2^L = \frac{\sum_i \ln \bar{y}_i^L \times \sum_i \bar{x}_i^L + \sum_i \left(\bar{x}_i^L \times \sqrt{-\ln \bar{y}_i^L} \right) \times \sum_i \sqrt{-\ln \bar{y}_i^L}}{n \sum_i \sqrt{-\ln \bar{y}_i^L} + \sum_i \sqrt{-\ln \bar{y}_i^L} \times \sum_i \sqrt{-\ln \bar{y}_i^L}}, \quad (26)$$

$$\sigma_2^L = \frac{n \sum_i \left(\bar{x}_i^L \times \sqrt{-\ln \bar{y}_i^L} \right) - \sum_i \sqrt{-\ln \bar{y}_i^L} \times \sum_i \bar{x}_i^L}{-n \sum_i \sqrt{-\ln \bar{y}_i^L} - \sum_i \sqrt{-\ln \bar{y}_i^L} \times \sum_i \sqrt{-\ln \bar{y}_i^L}}. \quad (27)$$

Now, we present an approach for comparing each two IVFNs. The $D_{p,q}$ -distance between two fuzzy numbers $\tilde{M}$ and $\tilde{N}$ with corresponding α_i -cuts, is approximately given as [39]:

$$D_{p,q}(\tilde{M}, \tilde{N}) = \left[(1-q) \sum_{i=1}^n \left| [\tilde{M}]_{\alpha_i}^l - [\tilde{N}]_{\alpha_i}^l \right|^p + q \sum_{i=1}^n \left| [\tilde{M}]_{\alpha_i}^r - [\tilde{N}]_{\alpha_i}^r \right|^p \right]^{1/p}. \quad (28)$$

In the case that $p = 2$ and $q = \frac{1}{2}$ the following formula is obtained:

$$D_{2,\frac{1}{2}}(\tilde{M}, \tilde{N}) = \sqrt{\left[\frac{1}{2} \sum_{i=1}^n \left| [\tilde{M}]_{\alpha_i}^l - [\tilde{N}]_{\alpha_i}^l \right|^2 + \frac{1}{2} \sum_{i=1}^n \left| [\tilde{M}]_{\alpha_i}^r - [\tilde{N}]_{\alpha_i}^r \right|^2 \right]}. \quad (29)$$

Hence, for two IVFNs $\tilde{M}$ and $\tilde{N}$, the $D_{p,q}$ -distance with the same parameters is given as follows:

$$D_{2,\frac{1}{2}}(\tilde{M}, \tilde{N}) = D_{2,\frac{1}{2}}(\tilde{M}^L, \tilde{N}^L) D_{2,\frac{1}{2}}(\tilde{M}^U, \tilde{N}^U), \quad (30)$$

where $D_{2,\frac{1}{2}}(\tilde{M}^L, \tilde{N}^L)$ and $D_{2,\frac{1}{2}}(\tilde{M}^U, \tilde{N}^U)$ are given by (29).

Remark 4 The $D_{p,q}$ -distance (30) is used in order of MIVFNs, i.e., it is said to $\tilde{M} \leq \tilde{N}$ if and only if $D_{2,\frac{1}{2}}(\tilde{M}, \tilde{0}) \leq D_{2,\frac{1}{2}}(\tilde{N}, \tilde{0})$.

Remark 5 The fuzzy min operation for p MIVFNs $\tilde{M}_i$ is defined as $\min_{1 \leq i \leq p} \tilde{M}_i = \min \left\{ D_{2,\frac{1}{2}}(\tilde{M}_i, \tilde{0}) \mid i = 1, 2, \dots, p \right\}$.

Modified ABC (MABC) algorithm for solving MIVFSP problem

In the case that arc weights are expressed by MIVFNs, the weight of each path between any two arbitrary nodes will be a MIVFN too. The SP in such network is named as mixed interval-valued FSP (MIVFSP) problem. In this section, we modify ABC algorithm for solving the same problem.

ABC algorithm

Karaboga [30] proposed the ABC algorithm, inspired by honeybees' life style, for solving numerical optimization problems. This is a population-based heuristic algorithm and is inspired by the search for food sources in the colony of honeybees. In this algorithm, each solution is considered as a food source, and the fitness of the solution is determine based on the amount of nectar per source. The algorithm aims to discover the best available source of food by searching the bees in different iterations. Here, three different categories of bees are as follows.

- The employed bees: the bees belonging to this category search around each food source.
- The onlooker bees: the bees belonging to this category are waiting for information from employed bees in dance area and deciding to go to a food source to do more searches.
- Scout bees: the bees belonging to this category in the entire space of the problem are randomly looking for a new food source.

There are three control parameters (SN, MCN, and *limit*) in this algorithm that need to be set correctly in accordance with the problem specification. The SN parameter is the number of food sources (population solutions) and parameter D is equal to the dimensions of the problem. The MCN parameter specifies the number of iterations. The *limit* control parameter indicates when the employed bee becomes a scout bee. That is, if an employed bee cannot locate a suitable source in a local search around a food source in several successive iterations determined by the *limit* variable, it will leave that food source and seek a new food source in the space of the problem. The flowchart of the ABC algorithm is given in Fig. 3.

Initially, the algorithm determines values of control parameters and generates food sources (initial population) randomly in dimensions (SN, D). Then, the process of

Fig. 3 Flowchart of the ABC algorithm

searching each category of bees is repeated in a loop. In each iteration of the algorithm, the search of the employed bees is performed, and then, according to the results of these bees, the operation of the onlooker bees is done, and eventually the operation of the scout bees is performed.

In the initial population, colonies are equally allocated to the employed bees and the onlooker bees. The number of employed bees and the number of onlooker bees are equal to the number of colony solutions. In each iteration, each employed bee performs a local search around the corresponding food source. If this bee finds a better food source, it forgets the previous food source and tries to work on the new

food source. Otherwise, it will divert the new food source and adds one unit to the *limit* value of this source.

The employed bees return to the beehive after the end of the search and share the resulting information, that is, the value of fitness and position of the food source in the dance area for the onlooker bees. Each onlooker bee selects a food source for a local search using a probability function. A food source with greater fitness value has more chance to be selected by the onlooker bees. Then, a local search is done by every onlooker bee like an employed around the chosen food source.

Fig. 4 A network with its vicinity matrix

Table 2 Three random paths

1	4	7	
1	2	5	7
1	3	6	7

After the activity of the onlooker bees, the *limit* value for each food source is compared with the threshold value. In the case that its value is more than the threshold value, the employed bee corresponding to that food source is converted into the scout bee. The scout bee selects a new food source randomly by moving in the search space. The iteration of the algorithm continues until it reaches the termination condition [19].

Some authors have used the ABC algorithm for solving optimization problems [37, 48]. In what follows, we will solve the problem of finding the MIVFSP problem by use of the ABC algorithm and explain each of the steps of the algorithm. Hence, we need to make changes to the initial version of the algorithm to be used to solve discrete MIVFSP problem. We will use the mutation operator introduced by Ebrahimnejad et al. [19] for the search of employed and onlooker bees.

MABC algorithm for finding MIVFSP

The aims of this section are to explain how (1) to produce the initial population, (2) to work the bees and (3) to calculate the fitness of this problem.

MABC colony initialization

For the beginning MABC algorithm for solving MIVFSP problem, it is required to generate the initial colony which consists of some solutions. Each path between the source and destination nodes of the network under consideration is defined as a solution. The path in this network is generated by use of its vicinity matrix. To this end, after finding the vicinity matrix, the initial colony is generated based on Algorithm 1 [28]. In this algorithm, a and n denote the vicinity matrix and destination node, respectively. Figure 4 illustrates a simple graph involving seven nodes and its vicinity matrix. Table 2 gives three solutions (paths).

Algorithm 1: Generating initial colony

Step 1: Assume $G = (N, E)$ is the directed network under consideration. Find the vicinity matrix, assign SN and set $q = 1$.

Step 2: Set $i = 1$, $m = 1$ and $p(m) = 1$.

Step 3: Assume $j \in a(i) = \{j \mid (i, j) \in A, a_{ij} = 1\}$. Let $m = m + 1$ and $p(m) = j$.

Step 4: If $j \neq n$ then let $i = j$ and go to Step 3.

(5) Save the generated path. Let $q = q + 1$.

(6) If $q \leq SN$ then go to Step 2, else stop.

In this algorithm, q denotes the number of generated paths. Since each path is considered as a solution, the number of paths should be equal to the number of worker and onlooker bees (SN). Also, m shows the location of path nodes. As an example, in path 1-4-5-6-7, for node 5 we have $m = 3$.

Fig. 5 Mutation operator on the path

Fig. 6 Generation of the new path

In this algorithm, q denotes the number of generated paths. Since each path is considered as a solution, the number of paths should be equal to the number of worker and onlooker bees (SN). In addition, m shows the location of path nodes. As an example, in path 1–4–5–6–7, for node 5, we have $m = 3$.

The search process of employed bees

In the MABC algorithm, the number of employed bees is equal to the number of colony solutions. The employed bee i is corresponded to the solution i in colony. In each iteration, each employed bee must perform a local search around the corresponding solution. Here, the mutation operator in the MABC is used as local search. In the case, fitness value of the new path is more than that of the current path, the bee uses the new path, ignores the previous one, and sets the value of the *limit* variable of the corresponding path to zero. Otherwise, the bee forgets the new path. In this case, the value of the *limit* variable of the corresponding path is increased by one unit.

The mutation operator in a path is defined as “Random change of the path from the middle of a path to the destination node” [19]. In other words, by random selection of one of the intermediate nodes of a path, we change the direction from that point of the path to the destination. Thus, the

Fig. 7 A wireless sensor network

Table 3 Mixed interval-valued fuzzy energy of WSN

Arc	Lower MF	Upper MF	Arc	Lower MF	Upper MF
(1, 2)	(18, 5; 0.6)	(18, 10; 0.7)	(11, 10)	(74, 78, 84, 88; 0.6)	(72, 76, 86, 90; 0.8)
(1, 3)	(25, 29, 35, 39; 0.4)	(23, 27, 37, 41; 0.8)	(11, 15)	(20, 13; 0.6)	(20, 14; 0.8)
(1, 4)	(22, 26, 32, 36; 0.4)	(20, 24, 34, 38; 0.6)	(11, 16)	(30, 34, 60, 64; 0.6)	(28, 32, 62, 66; 0.8)
(2, 5)	(13, 10; 0.7)	(13, 17; 0.9)	(11, 17)	(18, 22, 28, 32; 0.6)	(16, 20, 30, 34; 0.8)
(2, 6)	(30, 34, 40, 44; 0.7)	(28, 30, 42, 46; 0.8)	(11, 20)	(30, 34, 40, 44; 0.7)	(28, 30, 42, 46; 0.8)
(2, 7)	(38, 15; 0.5)	(38, 20; 0.8)	(12, 16)	(24, 28, 34, 38; 0.6)	(22, 26, 36, 40; 0.9)
(3, 6)	(20, 24, 30, 34; 0.6)	(18, 22, 32, 36; 0.7)	(12, 17)	(11, 11; 0.8)	(11, 22; 0.9)
(3, 7)	(10, 4; 0.5)	(10, 8; 0.8)	(12, 18)	(72, 76, 82, 86; 0.7)	(70, 74, 84, 88; 0.9)
(3, 8)	(28, 32, 38, 42; 0.6)	(26, 30, 40, 44; 0.8)	(12, 20)	(82, 86, 92, 96; 0.7)	(80, 84, 94, 98; 0.8)
(3, 9)	(56, 60, 66, 70; 0.7)	(54, 58, 68, 72; 0.9)	(13, 17)	(66, 70, 76, 80; 0.6)	(64, 68, 78, 82; 0.9)
(4, 8)	(11, 3; 0.7)	(11, 8; 0.9)	(13, 18)	(20, 8; 0.7)	(20, 15; 0.9)
(4, 9)	(18, 22, 28, 32; 0.5)	(16, 20, 30, 38; 0.8)	(14, 17)	(42, 46, 52, 56; 0.6)	(40, 44, 54, 58; 0.9)
(5, 10)	(65, 69, 70, 79; 0.6)	(63, 67, 77, 81; 0.8)	(14, 18)	(34, 38, 44, 48; 0.6)	(32, 36, 56, 60; 0.8)
(5, 11)	(55, 59, 65, 69; 0.7)	(53, 57, 67, 71; 0.9)	(15, 16)	(10, 1; 0.6)	(10, 10; 0.8)
(6, 5)	(15, 7; 0.8)	(15, 15; 0.9)	(15, 19)	(8, 10; 0.5)	(8, 16; 0.8)
(6, 10)	(74, 78, 84, 88; 0.8)	(72, 76, 86, 90; 0.9)	(16, 19)	(12, 6; 0.6)	(12, 12; 0.9)
(6, 11)	(68, 72, 78, 82; 0.6)	(66, 70, 80, 84; 0.8)	(16, 20)	(18, 10; 0.6)	(18, 12; 0.9)
(6, 12)	(32, 36, 42, 46; 0.5)	(30, 34, 44, 48; 0.8)	(16, 22)	(40, 20, 0.8)	(40, 22, 0.9)
(6, 13)	(28, 32, 38, 42; 0.6)	(26, 30, 40, 44; 0.8)	(17, 18)	(48, 52, 58, 62; 0.5)	(46, 50, 60, 68; 0.8)
(7, 11)	(78, 82, 88, 92; 0.7)	(70, 74, 80, 88; 0.8)	(17, 20)	(30, 15; 0.6)	(30, 20; 0.8)
(7, 12)	(62, 66, 72, 76; 0.6)	(60, 64, 70, 78; 0.7)	(17, 21)	(32, 36, 42, 46; 0.5)	(30, 34, 44, 48; 0.8)
(7, 13)	(14, 8; 0.5)	(14, 12; 0.7)	(17, 22)	(72, 76, 82, 86; 0.5)	(70, 74, 84, 88; 0.7)
(7, 14)	(80, 84, 90, 94; 0.8)	(78, 82, 92, 96; 0.9)	(18, 20)	(80, 84, 90, 94; 0.6)	(78, 82, 92, 96; 0.8)
(8, 9)	(12, 10; 0.6)	(12, 16; 0.8)	(18, 21)	(48, 52, 58, 62; 0.5)	(46, 50, 60, 68; 0.8)
(8, 13)	(40, 44, 50, 54; 0.6)	(38, 42, 52, 56; 0.8)	(19, 20)	(16, 10; 0.7)	(16, 20; 0.9)
(8, 14)	(44, 48, 54, 58; 0.7)	(42, 46, 58, 60; 0.9)	(19, 22)	(18, 22, 28, 32; 0.6)	(16, 20, 30, 34; 0.8)
(9, 14)	(34, 38, 44, 48; 0.6)	(32, 36, 46, 50; 0.8)	(20, 21)	(12, 10; 0.6)	(12, 16; 0.8)
(10, 15)	(40, 22; 0.6)	(40, 25; 0.8)	(20, 22)	(65, 69, 70, 79; 0.6)	(63, 67, 77, 81; 0.8)
(10, 16)	(50, 20; 0.5)	(50, 30; 0.7)	(21, 22)	(12, 20; 0.6)	(12, 30; 0.8)
(10, 19)	(16, 20, 26, 30; 0.7)	(14, 18, 28, 32; 0.9)			

new path is identical with the previous one from the first node until the selected random node and is different from the previous one from the selected random node until the destination node. For example, assume that node k is the selected random node. In this case, the path between node 1 and node $k - 1$ remains unchanged and the intermediate nodes belonging to the path between node k and the destination node are replaced with the new nodes determined by Algorithm 1. Figures 5 and 6 illustrate this situation. Figure 5 illustrates the mutation operation on the path 1–2–5–7 and Fig. 6 shows the generation of the new path 1–2–4–6–7. Here, node 2 is the selected random node.

The search process of onlooker bees

In the MABC algorithm, at the end of the search process of the employed bees, the resulting information including the amount of fitness and position of the food source is shared in the dance area for the onlooker bees. Then each onlooker

bee selects one of the paths, with a probability proportional to the fitness value of the paths, such as the roulette wheel. The probability of choosing any of the paths, denoted by p_i , is obtained using the following formulation:

Table 4 Simulation parameters

Iteration number for all algorithms	30
Size colony MABC	20
Limit MABC	8
Size population GA	20
PC GA	0.9
PM GA	0.3
Particles PSO	20
C_1	2
C_2	2
$W_{\min}$	0.2
$W_{\max}$	0.9

Table 5 The results of GA, PSO and MABC

	Convergence iteration			Convergence time			Run time			Shortest path		
	GA	PSO	MABC	GA	PSO	MABC	GA	PSO	MABC	GA	PSO	MABC
1	–	4	1	–	0.32	0.19	0.94	0.92	0.81	1–3–7–12–16–19–22	1–2–5–10–19–22	1–2–5–10–19–22
2	19	1	6	0.55	0.23	0.38	0.98	0.93	0.86	1–2–5–10–19–22	1–2–5–10–19–22	1–2–5–10–19–22
3	4	–	2	0.33	–	0.28	0.95	0.97	0.90	1–2–5–10–19–22	1–3–6–12–17–21–22	1–2–5–10–19–22
4	9	2	1	0.42	0.28	0.21	1.00	0.95	0.82	1–2–5–10–19–22	1–2–5–10–19–22	1–2–5–10–19–22
5	–	–	12	–	–	0.54	0.96	0.91	0.92	1–4–8–13–18–21–22	1–3–6–12–17–21–22	1–2–5–10–19–22
6	11	13	3	0.45	0.66	0.31	0.99	0.94	0.90	1–2–5–10–19–22	1–2–5–10–19–22	1–2–5–10–19–22
7	3	9	7	0.30	0.53	0.41	0.98	0.96	0.89	1–2–5–10–19–22	1–2–5–10–19–22	1–2–5–10–19–22
8	11	7	9	0.43	0.50	0.47	0.97	0.94	0.91	1–2–5–10–19–22	1–2–5–10–19–22	1–2–5–10–19–22
9	5	–	8	0.34	–	0.44	0.96	0.95	0.88	1–2–5–10–19–22	1–3–6–12–17–21–22	1–2–5–10–19–22
10	8	11	1	0.40	0.57	0.25	1.01	0.98	0.89	1–2–5–10–19–22	1–2–5–10–19–22	1–2–5–10–19–22
Min	3	1	1	0.30	0.23	0.19	0.94	0.91	0.81	1–2–5–10–19–22	1–2–5–10–19–22	1–2–5–10–19–22
Avg	–	–	5.0	–	–	0.35	0.97	0.94	0.88	–	–	1–2–5–10–19–22
Max	19	13	12	0.55	0.66	0.54	1.01	0.98	0.92	1–4–8–13–18–21–22	1–3–6–12–17–21–22	1–2–5–10–19–22

Fig. 8 The membership function of MIVFSP weight

$$p_i = \frac{\text{Fit}_i}{\sum_{z=1}^{\text{SN}} \text{Fit}_z}, \quad (31)$$

where Fit_i shows the fitness value of the path i and SN is the number of population solutions. Note that the weight of each path is defined as the fitness value of that path and is computed by summation of weights of arcs on that path.

The path that has more favorable fitness would be more likely to be selected by the onlooker bees. After each onlooker bee selects a path, that bee generates a random number in the interval $[0, 1]$. If such random number is less than the p_i , the onlooker bee is done a local search on this path. If the new path resulting from the mutation has a

Fig. 9 The best, average and worst time of convergence of all algorithms

higher fitness value than the previous path, the algorithm ignores the previous path. In this case, the value of the *limit* variable corresponding to the new path is changed to zero. Otherwise, the algorithm ignores the new path. In such a case, the value of the *limit* variable corresponding to this path increases one unit.

The search process of scout bees

In the MABC algorithm, if a food source is not improved in a predetermined number of successive iterations, that food source food is forgotten. The number of unsuccessful

Fig. 10 The best, average and worst runtime of all algorithms for 30 iterations

Fig. 11 Convergence curve of genetic algorithm, PSO algorithm and MABC algorithm

sequential iterations is defined by the *limit* parameter. Each path whose number of failed searches exceeds the *limit* value, the corresponding employed is converted into a scout bee. The scout bee forgets the previous path and produces a new random path by moving in the search space based on Algorithm 1. In addition, the new path in the population is replaced with the previous path.

In the MABC algorithm, fitness value is determined by summation of the interval-valued fuzzy weights of the arcs in the interval-valued fuzzy path. This value is computed by use of Eqs. (14) and (23).

Experiments and results

In this section, the efficiency of the proposed MABC algorithm in solving MIVFSP problem is demonstrated in WSN. The results of the proposed algorithm are compared with those derived with the GA [28] and PSO [18], which have performed on a network with various type-1 fuzzy numbers.

In the modern decision-making process, a different person may opt for the different evaluation methods like crisp, interval, linguistic or fuzzy and so on to access the object. However, ambiguity is one of the most utmost concern factors keeping in the mind during any process and hence it will make some sort of hesitancy to expose the information in a crisp format. To address it completely, interval-numbers are the appropriate ones. Since trapezoidal numbers handle more uncertainties than the triangular numbers and hence interval-valued trapezoidal fuzzy numbers are utilized in the work. Furthermore, almost all the approaches which are presented in the literature [20, 25] are working under the discrete set of the universe and hence do not tell the clear picture for the fuzzy concept such as “good”, “bad” and so on. Therefore, there is a need to study the behavior of the set for these variables also. Furthermore, in our real-life situations, various social and natural phenomena belong to the normal distribution which has the continuous higher derivative of its membership function. For instance, with respect to the stochastic phenomena, consider the life-length of the electric bulb, the average and the variance of the using lifetime of production are 100 and 2, respectively, which are expressed as an NFN (100, 2), (here, NFN represent normal fuzzy number) sometime we are not 100% sure for this value if we have [80%, 85%] certainty degree and [5%, 10%] negation degree, we can use the interval-valued NFNs (IVNFN) to describe this kind of information, i.e., $\langle (100, 2), [0.8, 0.9], [0.05, 0.10] \rangle$. Therefore, IVNFNs can express the stochastic phenomena better than NFNs by adding membership and non-membership functions. Hence, in the following examples, the arc weights are represented in terms of interval-valued trapezoidal fuzzy numbers and interval-valued normal fuzzy numbers.

Example 3 Consider the WSN shown in Fig. 7 with 22 sensor nodes connected to each other through 59 arcs. The energy data in terms of MIVFNs are given in Table 3. Here, we are going to find the mixed interval-valued fuzzy shortest path between node 1 and node 22. The simulation parameters are given in Table 4.

Since the heuristic algorithms are randomness, then each algorithm is executed ten times independently. Table 5 gives the obtained results. The MABC algorithm was able to detect the mixed interval-valued fuzzy shortest path

Fig. 12 WSN with 36 sensor nodes and 86 arcs

1–2–5–10–19–22 from node 1 to node 22 in every ten implementations. However, the GA only managed to converge to the best solution in eight implementations and the PSO was converged to the best solution in seven implementations.

Regarding Table 4, the iteration number of all algorithms is 30 and the number of members of the population is 20. The convergence time of algorithms, the total runtime, and the iteration number are shown in Table 5. The MABC algorithm was able to find the best solution in the first iteration in three implementations, while the GA determined the mixed interval-valued fuzzy shortest path in its best execution in the third iteration and the PSO founded the best solution only in one implementation. The comparison of the iteration number required for convergence in all algorithms shows that the MABC algorithm has more ability to solve the problem. Moreover, the MABC algorithm had less convergence time compared to GA and PSO. It is worth mentioning that the GA and PSO have not been integrated at all in two and three implementations, respectively. Finally, the implementation time of the MABC algorithm has been less than the time of the implementation of the GA and PSO. Regarding the randomness of the mutation operators and the effect of controlling variables such as *limit* in the MABC and P_c , P_m in the GA, and W_{min} and W_{max} in the PSO, runtimes of these algorithms in each implementation may be slightly different.

The optimal weight of the optimal path 1–2–5–10–19–22 is equal to $[(140.16, 20.55, 157.60, 22.86; 0.60), (134.92,$

$33.05, 167.08, 33.05; 0.70)]$. Figure 8 shows the membership function of this interval-valued fuzzy number.

The convergence time comparison of all three algorithms presented in Table 5 is shown in Fig. 9 with bar chart. The minimum convergence time, mean convergence time, and maximum convergence time for all algorithms are shown in this figure. The red parts display the time of the genetic algorithm and the blue parts shows the time of the PSO algorithm. Moreover, the green parts shows the time of the MABC algorithm. In all the charts, the MABC algorithm had less convergence time compared to GA and PSO algorithm.

Figure 10 shows the best, average and worst run time of all algorithms. In all of these diagrams, the implementation time of the MABC algorithm has been less than the time of the implementation of the genetic and PSO algorithms.

Figure 11 shows the solution convergence diagram of the three algorithms in each iteration. The fitness of the paths has computed after finding the approximate sum of the arc weights. In this figure, the shortest path weights found by the MABC algorithm, GA and PSO are shown with green circles, red stars and blue squares, respectively. In this diagram, which is the result of three independent implementations, the MABC algorithm, GA and PSO have converged in the 6th, 25th and 28th iterations, respectively. This confirms the high ability of the MABC algorithm to solve the MIVFSP problem in a directed network.

Table 6 Data of WSN

Arc	Lower MF	Upper MP	Arc	Lower MF	Upper MP
(1, 2)	(80, 84, 90, 94, 0.8)	(78, 82, 92, 96, 0.9)	(14, 20)	(40, 44, 50, 54, 0.6)	(38, 42, 52, 56, 0.8)
(1, 3)	(12, 10, 0, 0, 0.6)	(12, 16, 0, 0, 0.8)	(14, 25)	(44, 48, 54, 58, 0.7)	(42, 46, 58, 60, 0.9)
(1, 5)	(40, 44, 50, 54, 0.6)	(38, 42, 52, 56, 0.8)	(15, 21)	(34, 38, 44, 48, 0.6)	(32, 36, 46, 50, 0.8)
(2, 4)	(44, 48, 54, 58, 0.7)	(42, 46, 58, 60, 0.9)	(16, 21)	(40, 22, 0.6)	(40, 25, 0.8)
(2, 5)	(34, 38, 44, 48, 0.6)	(32, 36, 46, 50, 0.8)	(16, 22)	(50, 20, 0.5)	(50, 30, 0.7)
(2, 8)	(40, 22, 0.6)	(40, 25, 0.8)	(16, 26)	(16, 20, 26, 30, 0.7)	(14, 18, 28, 32, 0.9)
(3, 5)	(50, 20, 0.5)	(50, 30, 0.7)	(17, 22)	(74, 78, 84, 88, 0.6)	(72, 76, 86, 90, 0.8)
(3, 6)	(16, 20, 26, 30, 0.7)	(14, 18, 28, 32, 0.9)	(17, 23)	(20, 13, 0.6)	(20, 14, 0.8)
(3, 9)	(18, 5, 0.6)	(18, 10, 0.7)	(17, 27)	(30, 34, 60, 64, 0.6)	(28, 32, 62, 66, 0.8)
(4, 7)	(25, 29, 35, 39, 0.4)	(23, 27, 37, 41, 0.8)	(18, 23)	(18, 22, 28, 32, 0.6)	(16, 20, 30, 34, 0.8)
(4, 8)	(22, 26, 32, 36, 0.4)	(20, 24, 34, 38, 0.6)	(18, 24)	(22, 23, 24, 28, 0.2)	(33, 34, 36, 35, 0.7)
(4, 11)	(13, 10, 0.7)	(13, 17, 0.9)	(18, 28)	(20, 15, 0.3)	(20, 22, 0.4)
(5, 8)	(48, 52, 58, 62, 0.5)	(46, 50, 60, 68, 0.8)	(19, 24)	(20, 24, 30, 44, 0.1)	(28, 30, 42, 46, 0.6)
(5, 9)	(16, 10, 0.7)	(16, 20, 0.9)	(19, 25)	(24, 28, 34, 38, 0.6)	(22, 26, 36, 40, 0.9)
(5, 12)	(18, 22, 28, 32, 0.6)	(16, 20, 30, 34, 0.8)	(19, 29)	(11, 11, 0.8)	(11, 22, 0.9)
(6, 9)	(12, 10, 0.6)	(12, 16, 0.8)	(20, 25)	(72, 76, 82, 86, 0.7)	(70, 74, 84, 88, 0.9)
(6, 10)	(65, 69, 70, 79, 0.6)	(63, 67, 77, 81, 0.8)	(21, 26)	(82, 86, 92, 96, 0.7)	(80, 84, 94, 98, 0.8)
(6, 13)	(12, 20, 0.6)	(12, 30, 0.8)	(22, 26)	(66, 70, 76, 80, 0.6)	(64, 68, 78, 82, 0.9)
(7, 35)	(30, 34, 40, 44, 0.7)	(28, 30, 42, 46, 0.8)	(22, 27)	(20, 8, 0.7)	(20, 15, 0.9)
(7, 11)	(38, 15, 0.5)	(38, 20, 0.8)	(22, 30)	(42, 46, 52, 56, 0.6)	(40, 44, 54, 58, 0.9)
(8, 11)	(20, 24, 30, 34, 0.6)	(18, 22, 32, 36, 0.7)	(23, 27)	(34, 38, 44, 48, 0.6)	(32, 36, 56, 60, 0.8)
(8, 12)	(10, 4, 0.5)	(10, 8, 0.8)	(23, 28)	(10, 1, 0.6)	(10, 10, 0.8)
(8, 17)	(28, 32, 38, 42, 0.6)	(26, 30, 40, 44, 0.8)	(23, 31)	(8, 10, 0.5)	(8, 16, 0, 0, 0.8)
(9, 12)	(56, 60, 66, 70, 0.7)	(54, 58, 68, 72, 0.9)	(24, 28)	(12, 6, 0.6)	(12, 12, 0.9)
(9, 13)	(11, 3, 0.7)	(11, 8, 0.9)	(24, 29)	(18, 10, 0.6)	(18, 12, 0.9)
(9, 18)	(18, 22, 28, 32, 0.5)	(16, 20, 30, 38, 0.8)	(24, 32)	(40, 20, 0.8)	(40, 22, 0.9)
(10, 13)	(65, 69, 70, 79, 0.6)	(63, 67, 77, 81, 0.8)	(25, 29)	(48, 52, 58, 62, 0.5)	(46, 50, 60, 68, 0.8)
(10, 14)	(55, 59, 65, 69, 0.7)	(53, 57, 67, 71, 0.9)	(26, 30)	(30, 15, 0.6)	(30, 20, 0.8)
(10, 19)	(18, 5, 0.6)	(18, 10, 0.7)	(27, 30)	(32, 36, 42, 46, 0.5)	(30, 34, 44, 48, 0.9)
(35, 21)	(25, 29, 35, 39, 0.4)	(23, 27, 37, 41, 0.8)	(27, 31)	(72, 76, 82, 86, 0.5)	(70, 74, 84, 88, 0.7)
(35, 16)	(22, 26, 32, 36, 0.4)	(20, 24, 34, 38, 0.6)	(27, 33)	(80, 84, 90, 94, 0.6)	(78, 82, 92, 96, 0.8)
(35, 15)	(13, 10, 0.7)	(13, 17, 0.9)	(28, 31)	(74, 78, 84, 88, 0.8)	(72, 76, 86, 90, 0.9)
(11, 16)	(15, 7, 0.8)	(15, 15, 0.9)	(28, 32)	(48, 52, 58, 62, 0.5)	(46, 50, 60, 68, 0.6)
(11, 17)	(74, 78, 84, 88, 0.8)	(72, 76, 86, 90, 0.9)	(28, 34)	(16, 10, 0, 0, 0.7)	(16, 20, 0, 0, 0.8)
(11, 22)	(68, 72, 78, 82, 0.6)	(66, 70, 80, 84, 0.8)	(29, 32)	(18, 22, 28, 32, 0.6)	(16, 20, 30, 34, 0.9)
(12, 17)	(32, 36, 42, 46, 0.5)	(30, 34, 44, 48, 0.8)	(30, 33)	(12, 10, 0, 0, 0.6)	(12, 16, 0, 0, 0.8)
(12, 18)	(28, 32, 38, 42, 0.6)	(26, 30, 40, 44, 0.8)	(31, 33)	(65, 69, 70, 79, 0.6)	(63, 67, 77, 81, 0.8)
(12, 23)	(78, 82, 88, 92, 0.7)	(70, 74, 80, 88, 0.8)	(31, 34)	(12, 20, 0.4)	(12, 30, 0.7)
(13, 18)	(62, 66, 72, 76, 0.6)	(60, 64, 70, 78, 0.7)	(31, 36)	(34, 38, 44, 48, 0.6)	(32, 36, 46, 50, 0.7)
(13, 19)	(14, 8, 0.5)	(14, 12, 0.7)	(32, 34)	(40, 22, 0, 0, 0.6)	(40, 25, 0, 0, 0.9)
(13, 24)	(80, 84, 90, 94, 0.8)	(78, 82, 92, 96, 0.9)	(33, 36)	(22, 23, 24, 28, 0.2)	(33, 34, 36, 35, 0.7)
(14, 19)	(12, 10, 0.6)	(12, 16, 0.8)	(34, 36)	(20, 15, 0.3)	(20, 22, 0.4)

Example 4 Figure 12 shows another example of a WSN with 36 sensor nodes and 86 arcs. Data are given in Table 6. The aim is to determine the mixed interval-valued fuzzy shortest path between node 1 and node 36. The simulation parameters are given in Table 7. The results of the implementation of the algorithms on Fig. 12 are shown in Table 8. Of the

ten independent runs of each algorithm, MABC has detected the shortest path from the source to the destination node in eight runs, while the genetic algorithm has converged to the best solution in six runs, and the PSO algorithm has found the best path only in five runs. The convergence time of the algorithms, the total implementation time, and the iteration

Table 7 The simulation parameters of Example 4.2

Iteration number for all algorithms	30
Size colony MABC	25
Limit MABC	10
Size population GA	25
PC GA	0.9
PM GA	0.3
Particles PSO	25
C_1	2
C_2	2
$W_{\min}$	0.2
$W_{\max}$	0.9

number that the algorithms have been converged in each run are shown in Table 8. Comparison of the iteration number required for convergence in the three algorithms shows that the MABC algorithm is more capable of finding the optimal solution.

The comparisons of convergence times all three algorithms is illustrated in Fig. 13 with bar chart. In all the charts, the MABC algorithm had less convergence time compared to GA and PSO algorithm. Note that the maximum convergence time bar is related to the MABC algorithm because it converges once in the 27th iteration. It is clear that converging on the final iterations is better than not converging at all. It is worth mentioning that the MABC algorithm, the genetic algorithm and the PSO algorithm have not been integrated at all in two, four and five implementations, respectively. Here, the mean convergence time bar is the mean implementation time in cases where the algorithm converges.

Figure 14 shows the best, average and worst run time of all algorithms. In all of these diagrams, the implementation time of the MABC algorithm has been less than the time of the implementation of the genetic and PSO algorithms.

The interval-valued exponential fuzzy weight of shortest path 1–3–9–18–28–31–36 is equal to [(131.27, 29.98, 174.73, 29.98; 0.50), (126.92, 48.05, 179.44, 50.07; 0.70)]. Figure 15 shows the lower and upper membership functions of this interval-valued fuzzy number.

Finally, Fig. 16 shows the solution convergence diagram of the three algorithms in each iteration. In this diagram, which is the result of three independent implementations, the MABC algorithm and GA have converged in the 7th and 13th iterations, respectively, while the PSO algorithm has not converged at all.

Now we do sensitivity analysis on the results of Example 4. To do this, we change the number of members of the population of all algorithms from 25 to 15 and 40. The obtained results are given in Table 9. As can be seen from Table 9, when the number of members of the population

Table 8 The simulation results

	Convergence iteration			Convergence time			Run time		Shortest path		
	GA	PSO	MABC	GA	PSO	MABC	GA	PSO	GA	PSO	MABC
1	9	4	3	0.48	0.36	0.33	1.18	1.09	1.03	1–3–9–18–28–31–36	1–3–9–18–28–31–36
2	6	–	5	0.41	–	0.39	1.16	1.08	1.04	1–3–9–13–18–28–31–36	1–3–9–18–28–31–36
3	–	5	13	–	0.47	0.59	1.14	1.10	1.02	1–5–8–12–17–27–33–36	1–3–9–18–28–31–36
4	–	–	27	–	–	0.99	1.15	1.07	1.05	1–5–8–12–17–27–33–36	1–3–9–18–28–31–36
5	–	4	19	–	0.35	0.81	1.17	1.08	1.09	1–5–8–12–17–27–33–36	1–3–9–18–28–31–36
6	8	–	–	0.47	–	–	1.15	1.11	1.06	1–3–9–18–28–31–36	1–5–8–12–17–27–33–36
7	–	–	13	–	–	0.65	1.17	1.10	1.08	1–5–8–12–17–27–33–36	1–3–9–18–28–31–36
8	11	10	–	0.58	0.51	–	1.14	1.13	1.09	1–3–9–18–28–31–36	1–5–8–12–17–27–33–36
9	4	–	8	0.37	–	0.48	1.12	1.12	1.05	1–3–9–18–28–31–36	1–3–9–18–28–31–36
10	15	12	17	0.62	0.52	0.74	1.13	1.11	1.07	1–3–9–18–28–31–36	1–3–9–18–28–31–36
Min	4	4	3	0.37	0.35	0.33	1.12	1.07	1.02	1–3–9–18–28–31–36	1–3–9–18–28–31–36
Avg	–	–	–	–	–	–	1.15	1.10	1.06	–	–
Max	15	12	27	0.62	0.52	0.99	1.18	1.13	1.09	1–5–8–12–17–27–33–36	1–5–8–12–17–27–33–36

Fig. 13 The convergence times of algorithms

Fig. 14 The run times of algorithms

of all algorithms is reduced from 25 to 15, there is no difference in the order of the algorithms' superiority, and the MABC algorithm has produced better quality results than the PSO algorithm and genetic algorithm. Moreover, with the increase in the number of members to 40 members, the best results belong to the MABC algorithm, the PSO algorithm and the genetic algorithm, respectively. The results of the algorithms in Table 9 show that by changing the population size there is no change in the ranking of the algorithms.

To sum up with all the above aspects, we can conclude that the simulation results obtained from the MABC algorithm are vigorous regarding three criteria namely, convergence iteration, convergence time and run time. In fact, the results obtained from both simulations confirm that the MABC algorithm has less convergence iteration,

Fig. 15 The membership function of the shortest path weight

Fig. 16 Convergence curve of all algorithms

convergence time and implementation time compared to GA and PSO algorithm. These results confirm that the MABC algorithm has more ability to find the shortest path. Moreover, we can conclude from the both simulations on WSNs that the MABC algorithm is quite robust to recognize the shortest path among all paths of networks in terms of energy consumption.

Conclusions

In this study, we investigated a kind of the SP problem in a network in which arc weights are represented in terms of MIVFNs and proposed a modified version of the ABC algorithm for solving it. To do this, we developed a novel

Table 9 Sensitivity analysis

Size population	Convergence iteration			Convergence time			Run time		
	GA	PSO	MABC	GA	PSO	MABC	GA	PSO	MABC
15									
Min	6	5	5	0.40	0.40	0.32	0.76	0.72	0.69
Max	28	29	26	0.77	0.71	0.70	0.81	0.76	0.73
25									
Min	4	4	3	0.37	0.35	0.33	1.12	1.07	1.02
Max	15	12	27	0.62	0.52	0.99	1.18	1.13	1.09
40									
Min	3	4	2	0.51	0.50	0.44	1.48	1.42	1.33
Max	13	10	9	0.63	0.61	0.48	1.61	1.51	1.43

approach to approximate the summation of two MIVFNs using alpha cuts. Then, we presented an extended distance function for comparing the various interval-valued fuzzy weights of different interval-valued fuzzy paths. The proposed algorithm was illustrated through two applications in WSNs involving 22 and 36 sensor nodes. The results shows that the proposed algorithm has better performance for solving MIVFSP compared to GA and PSO in terms of convergence iteration, convergence time and run time. However, considering other improved ABC algorithms [7, 27, 36] in solving MIVFSP problem is a worthwhile area of research that could be the scope for the future work. Finally, as future works, we can focus on extending the proposed algorithm and method to other systems such as transportation problems [4, 14, 15, 17], new product development projects [43], seru production system [44], manufacturing and supply chain [45], workforce scheduling [46] and so on.

Acknowledgements The authors would like to thank the anonymous reviewers and the associate editor for their insightful comments and suggestions.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Abbaszadeh Sori A, Ebrahimnejad A, Motameni H (2020) The fuzzy inference approach to solve multi-objective constrained shortest path problem. *J Intell Fuzzy Syst* 38(4):4711–4720
2. Abbaszadeh Sori A, Ebrahimnejad A, Motameni H (2020) Elite artificial bees' colony algorithm to solve robot's fuzzy constrained routing problem. *Computat Intell* 36(2):659–681
3. Ahuja RK, Magnanti TL, Orlin JB (1993) Network flows: theory algorithms and applications. Prentice-Hall, Englewood Cliffs
4. Bagheri M, Ebrahimnejad A, Razavyan S et al (2020) Fuzzy arithmetic DEA approach for fuzzy multi-objective transportation problem. *Oper Res Int J*. <https://doi.org/10.1007/s12351-020-00592-4>
5. Broumi S, Talea M, Bakali A et al (2019) Shortest path problem in fuzzy, intuitionistic fuzzy and neutrosophic environment: an overview. *Complex Intell Syst* 5:371–378
6. Broumi S, Dey A, Talea M et al (2019) Shortest path problem using Bellman algorithm under neutrosophic environment. *Complex Intell Syst* 5:409–416
7. Chidambaram C, Marçal MS, Dorini LB, Neto HV, Lopes HS (2012) An improved ABC algorithm approach using SURF for face identification. In: Yin H, Coast JAF, Barreto G (eds) Intelligent data engineering and automated learning—IDEAL 2012. IDEAL 2012. Lecture notes in computer science, vol 7435. Springer, Berlin
8. Chiang J (2001) Fuzzy linear programming based on statistical confidence interval and interval-valued fuzzy set. *Eur J Oper Res* 129:65–86
9. Chen W, Jiang X, Li X, Gao J, Xu X, Ding S (2013) Wireless sensor network nodes correlation method in coal mine tunnel based on Bayesian decision. *Measurement* 46:2335–2340
10. Chuang T-N, Kung J-Y (2006) A new algorithm for the discrete fuzzy shortest path problem in a network. *Appl Math Comput* 174(1):660–668
11. Dey A, Pradhan R, Pal A, Pal T (2018) A Genetic algorithm for solving fuzzy shortest path problems with interval type-2 fuzzy arc lengths. *Malay J Comput Sci* 31(4):255–270
12. Dey A, Mondal S, Pal T (2019) Robust and minimum spanning tree in fuzzy environment. *Int J Comput Sci Math* 10(5):513–524
13. Dou Y, Zhu L, Wang HS (2012) Solving the fuzzy shortest path problem using multi-criteria decision method based on vague similarity measure. *Appl Soft Comput* 12(6):1621–1631
14. Ebrahimnejad A (2016) New method for solving fuzzy transportation problems with LR flat fuzzy numbers. *Inf Sci* 357:108–124

15. Ebrahimnejad A (2016) Fuzzy linear programming approach for solving transportation problems with interval-valued trapezoidal fuzzy numbers. *Sādhanā* 41:299–316. <https://doi.org/10.1007/s12046-016-0464-0>
16. Ebrahimnejad A (2018) A method for solving linear programming with interval-valued trapezoidal fuzzy variables. *RAIRO Oper Res* 52(3):955–979
17. Ebrahimnejad A, Verdegay JL (2018) A new approach for solving fully intuitionistic fuzzy transportation problems. *Fuzzy Optim Decis Making* 17:447–474. <https://doi.org/10.1007/s10700-017-9280-1>
18. Ebrahimnejad A, Karimnejad Z, Alrezaamiri H (2015) Particle swarm optimization algorithm for solving shortest path problems with mixed fuzzy arc weights. *Int J Appl Decis Sci* 8(2):203–222
19. Ebrahimnejad A, Tavana M, Alrezaamiri H (2016) A novel artificial bee colony algorithm for shortest path problems with fuzzy arc weights. *Measurement* 93:48–56
20. Ebrahimnejad A, Verdegay JH, Garg H (2019) Signed distance ranking based approach for solving bounded interval-valued fuzzy numbers linear programming problems. *Int J Intell Syst* 9(34):2055–2076
21. Ebrahimnejad A (2020) An acceptability index based approach for solving shortest path problem on a network with interval weights. *RAIRO Oper Res*. <https://doi.org/10.1051/ro/2020033>
22. Enayattabar M, Ebrahimnejad A, Motameni H (2019) Dijkstra algorithm for shortest path problem under interval-valued Pythagorean fuzzy environment. *Complex Intell Syst* 5(2):93–100
23. Enayattabar M, Ebrahimnejad A, Motameni H, Garg H (2019) A novel approach for solving all-pairs shortest path problem in an interval-valued fuzzy network. *J Intell Fuzzy Syst* 37(5):6865–6877
24. Eshaghnezhad M, Rahbarnia F, Effati S, Mansoori A (2018) An artificial neural network model to solve the fuzzy shortest path problem. *Neural Process Lett*. <https://doi.org/10.1007/s11063-018-9945-y>
25. Farhadinia B (2014) Sensitivity analysis in interval-valued trapezoidal fuzzy number linear programming problems. *Appl Math Model* 38:40–62
26. Funck J, Gühmann C (2014) Comparison of approaches to time-synchronous sampling in wireless sensor networks. *Measurement* 56:203–214
27. Gao R, Wu J, Hu W, Zhang Y (2018) An improved ABC algorithm for energy management of microgrid. *Int J Comput Commun Control* 13(4):477–491
28. Hassanzadeh R, Mahdavi I, Mahdavi-Amiri N, Tajdin A (2013) A genetic algorithm for solving fuzzy shortest path problems with mixed fuzzy arc lengths. *Math Comput Model* 57(1–2):84–99
29. Ji X, Iwamura K, Shao Z (2007) New models for shortest path problem with fuzzy arclengths. *Appl Math Model* 31:259–269
30. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization, Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department
31. Lakdashti A, Sahoo S, Borzooei RA, Pal M, Rashmanlou H, Esfandiari A (2018) Computation of shortest path in a vague network by Euclidean distance. *J Multiple Valued Logic Soft Comput* 30(1):115–123
32. Mahdavi I, Nourifar R, Heidarzade A, Mahdavi-Amiri N (2009) A dynamic programming approach for finding shortest chains in fuzzy network. *Appl Soft Comput* 9(2):503–511
33. Moazeni M (2006) Fuzzy shortest path problem with finite fuzzy quantities. *Appl Math Comput* 183:160–169
34. Nayeem SMA, Pal M (2005) Shortest path problem on a network with imprecise edge weight. *Fuzzy Optim Decis Making* 4:293–312
35. Okada S, Soper T (2000) A shortest path problem on a network with fuzzy arc lengths. *Fuzzy Sets Syst* 109(1):129–140
36. Pian J, Wang G, Li B (2018) An improved ABC algorithm based on initial population and neighborhood search. *IFAC-PapersOn-Line* 51(18):21–256
37. Sasikala E, Rengarajan N (2015) An intelligent technique to detect jamming attack in wireless sensor networks (WSNs). *Int J Fuzzy Syst* 17(1):76–83
38. Sornam M, Prabhakaran M (2019) Logit-based artificial bee colony optimization (LB-ABC) approach for dental caries classification using a back propagation neural Network. In: Krishna A, Srikantaiah K, Naveena C (eds) *Integrated intelligent computing, communication and security. Studies in computational intelligence*, vol 771. Springer, Singapore
39. Tajdin A, Mahdavia I, Mahdavi-Amiri N, Sadeghpour-Gildeh B (2010) Computing a fuzzy shortest path in a network with mixed fuzzy arc lengths using-cuts. *Comput Math Appl* 60(4):989–1002
40. Tam JH, Ong ZC, Ismail Z, Ang BC, Khoo SY (2019) A new hybrid GA-ACO-PSO algorithm for solving various engineering design problems. *Int J Comput Math* 96(5):883–919
41. Wei SH, Chen SM (2009) Fuzzy risk analysis based on interval-valued fuzzy numbers. *Expert Syst Appl* 36:2285–2299
42. Yang Y, Dandan Y, Junhang J (2017) Optimal path selection approach for fuzzy reliable shortest path problem. *J Intell Fuzzy Syst* 32(1):197–205
43. Yilmaz OF, Ozcelik G, Yeni FB (2020) Lean holistic fuzzy methodology employing cross-functional worker teams for new product development projects: a real case study from high-tech industry. *Eur J Oper Res* 282(3):989–1010
44. Yilmaz OF (2020) Operational strategies for seru production system: a bi-objective optimisation model and solution methods. *Int J Prod Res* 58(11):3195–3219
45. Yilmaz OF (2020) Examining additive manufacturing in supply chain context through an optimization model. *Comput Ind Eng* 142:106335
46. Yilmaz OF (2020) Attaining flexibility in seru production system by means of Shojinka: an optimization model and solution approaches. *Comput Oper Res* 119:104917
47. Yu J-R, Wei T-H (2007) Solving the fuzzy shortest path problem by using a linear multiple objective programming. *J Chin Inst Ind Eng* 24(5):360–365
48. Zakeri E, Moezi SA, Eghtesad M (2018) Tracking control of ball on sphere system using tuned fuzzy sliding mode controller based on artificial bee colony algorithm. *Int J Fuzzy Syst* 20(1):295–308

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.