Applied Mathematics and Computational Intelligence
Volume 12, No.1, Apr 2023 [125-145]

An Innovative Approach to Solve Shortest Path Problem Using Dijkstra's Algorithm Based on Interval-Valued Bipolar Neutrosophic Information

Siti Nurul Fitriah Mohamad[1*], Suriana Alias[2], Norarida Abd Rhani[3], Hazwani Hashim[4]

[1,2,3,4] Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA Kampus Machang, 18500 Machang Kelantan

[*] Corresponding author: fitriah@uitm.edu.my

Received: 28 September 2022
Accepted: 24 November 2022

ABSTRACT

The shortest path problem (SPP) is considerably important in several fields such as application in highway networks, the problem of scheduling, road transportation network, etc. The SPP focuses on recommending the path which has a minimum length enclosed by two vertices. The length of the arc represents real world measurements like cost, time, distance, price, or other parameters. A neutrosophic set is a collection of the truth membership, indeterminacy membership, and falsity membership degrees of the elements. In an uncertain environment, neutrosophic numbers can express the arc distance more effectively. In this study, classical Dijkstra's algorithm has been redesigned to handle the case in which most of the parameters of a network are uncertain and given in terms of interval-valued bipolar neutrosophic numbers (IVBNN). The proposed algorithm gives the shortest path length using the score function from sources node to destination node and each of the arc lengths are attributed to an IVBNN. For the validation of the proposed algorithm, a numerical example has been conducted and the objective of this study is to identify the optimal paths to rescue points. Finally, we describe the advantages of the proposed method and give some suggestions to further this study.

Keywords: Dijkstra's algorithm, Neutrosophic numbers, Shortest Path Problem.

1 INTRODUCTION

Graph is an efficient tool to model the real-life problems. By modelling the graph, the objects and their relations are symbolised by nodes and arcs. There exist many different types of information in real-life problems and we need several types of graphs to model those problems such as fuzzy graph, intuitionistic fuzzy graphs and neutrosophic graph theory.

The shortest path problem (SPP) is one of the most fundamental and well-known combinatorial problems that appear in various fields of science and engineering, for example road network application, computer network, communication network, street networks utility, routing in conversation channels, scheduling issues, transportation and many more. In a network, the SPP objectives is to find the path from one source vertex to destination vertex with minimum weight, where some weight is attached to each edge connecting a pair of vertices.

Some effective algorithmic approaches were introduced by Dijkstra and Floyd in between 1950 and 1970. This algorithm refers as classical algorithm. In classical algorithms for SPP, the weights of the edges in a SPP are considered as crisp (real) numbers. In real-world situations, the edges weights of a SPP are used to reflect the cost, distance, time, price or other parameters. Many researchers have studied intensively on the SPP with deterministic edge costs. These SPP are referred to as standard SPP. Decision maker can solve the standard SPP efficiently using several well-known algorithms introduced by some excellent researchers. Although in standard SPP, the costs of the arcs are considered real numbers, most real-life scenarios, however, have many parameters that may not be always precise (i.e., travelling demands, travelling costs, travelling capacities, travelling time etc.). Several types of uncertainty are generally encountered in practical applications of SPP due to imperfect data, maintenance, failure or other reasons.

In conventional SPP, it is assumed that decision maker is certain about the parameters (distance, time etc.) between difference vertices. But in real life situations, there always exist uncertainty about the parameters between different vertices. Also, in practical application of SPP, the edge weights in the direction of a graph have some parameters that are very difficult to find exact capacities, distance, costs, specifications, traffic frequencies, etc. The geographical distance between two cities for example, may be correctly recognized, but due to weather and injuries, the travel cost or travel time may change. So, the edge weights are nondeterministic in such situations and it is impossible to use the classical algorithm to find exact solution of the SPP in such uncertain environment.

The approach of using fuzzy numbers can be used for the world of ambiguity. The crisp number is the number of obtained using the defuzzification function from fuzzy numbers and it is commonly use in methods of optimization. The SPP is not limited to a geometrical distance. Although it is set, the travel time in the cities can be reflected by fuzzy factor. Since the edge weight is unknown in almost all the networks of contact and transportation, it can be formulated into a crisp graph. Okada and Gen [1] first solve fuzzy shortest path problem (FSPP). The most critical consideration of combinatorial optimization is solving the SPP to the directed graph and its preferred format unable to represent the situations where not only the option of each single edge can find the value of the isolated function.

While getting uncertainty in the set of vertices and edges of a graph, then fuzzy graph can be adopted for SPP but if there is indeterminacy exist between the relation vertices and edges then neutrosophic will be preferred concept to deal the real-life problems. Since indeterminacy is also conducted seriously, neutrosphic set (NS) may be able to handle uncertainty in a better way. The model of the NS is an important mechanism to deal with real scientific and engineering as it can deal uncertain, inconsistent and indeterminate information. SPP of the network problem can be determined using neutrosophic set (NS) by considering the edge weight of the graph as neutrosophic numbers and it can be single-valued, interval-valued or bipolar-valued as well.

Smarandache [2] introduced about neutrosophic for the first time in the year 1999 and highlighted an important mathematical mechanism called neutrosophic set theory in order to handle indeterminate, uncertain and imprecise problems which cannot be dealt by fuzzy and its various type. The concept of NS is generalized the concepts of classical sets, fuzzy sets, intuitionistic fuzzy set, interval valued fuzzy sets and interval-valued intuitionistic fuzzy sets by adding an independent indeterminacy membership. Also, NS is a powerful technique to deal with inconsistent, indeterminate, and incomplete information in real world problem. This study has attained further interest from many researchers. The concept of NS is characterized by three independent degrees

namely truth-membership degree (T), indeterminacy-membership degree (I) and falsity-membership degree (F).

Next, Deli et al. [3] proposed bipolar neutrosophic set (BNS) by hybridizing the concept of bipolar fuzzy sets and NS. "Bipolarity refers to the human mind's propensity to reason and make decisions on the basis of positive and negative impacts,"[4]. What is probable, satisfactory, allowed, required, or deemed acceptable is positive knowledge. Negative statements, on the other hand, convey what is unlikely, denied, or prohibited. Negative preferences correspond to limitations because they define which values or objects are to be rejected (i.e., those that do not satisfy the limitations), whereas positive preferences correspond to wishes because they specify which objects are more desirable (i.e., satisfy user wishes) than others without rejecting those that do not fulfil the wishes.

BNS has two fully independent parts, which are positive membership degree and negative membership degree where the positive membership degrees represent truth membership degree, indeterminacy membership degree, and false membership degree, respectively, of an element [3]. The negative membership degrees represent truth membership degree, indeterminacy membership degree, and false membership degree, respectively, of an element to some implicit counter property corresponding to a BNS. Deli et al. defined some operations, namely, score, accuracy, and certainty functions, to compare BNSs and provided some operators to aggregate BNSs.

Based on the idea of Dijkstra's algorithm, SPP is solved for fuzzy and neutrosophic network. To do best of our knowledge, few research papers deal with SPP in neutrosophic environment such as solving SPP in SVNS, IVNS and BVNS. Till now, there is no study in the literature for computing SPP in interval valued bipolar neutrosophic environment. Therefore, there is a need to establish an interval valued bipolar neutrosophic version of Dijkstra's algorithm for neutrosophic shortest path problem (NSPP). The main motivation of this study is to introduce an algorithmic approach for SPP in an uncertain environment which will be simple enough and effective in real-life problem.

2 LITERATURE REVIEW

In a classical network problem, the weight of the edges in a SPP are supposed to be real numbers, but most practical applications have parameters that are not exactly reliable like time, cost, demand etc. In such scenarios, it is very sufficiently suitable to use fuzzy number for modelling the problem which responding to Fuzzy Shortest Path Problem (FSPP). Fuzzy set theory pioneered by Zadeh [5] is an efficient tool to handle the problems of uncertainty.

If the indeterminate, inconsistent and incomplete information has identified, all these kinds of FSPP failed. For this reason, some new approaches have been developed using neutrosophic numbers (NN). Broumi et al. [6] first introduced SPP in single valued neutrosophic number (SVNN). The authors apply Dijkstra's algorithm under neutrosophic setting to solve neutrosophic shortest path problem (NSPP). Later, the same authors used extended version to solve NSPP where the edge weight is characterized by interval valued neutrosophic numbers (IVNN). Next, Hu et al. [7] proposed the convenience of quadratic SPP semi-definite programming and used branch and bound algorithms to solve SPP.

Later, Broumi et al. [8] and Broumi et al. [9] proposed various concepts on NS and analyzed the existing concepts and the proposed NN; thereafter proposed the SPP in an IVNN. Moreover, Kumar

et al. [10] developed an algorithm for solving the SPP in triangular and trapezoidal neutrosophic environments. Broumi et al. [11] presented a study on the NSPP with IVNN on a network. Tan et al. [12] and Broumi et al. [13] proposed the Bellman algorithm for solving the SPP in a neutrosophic graph. Authors in [13] used the original Bellman algorithm to search the shortest path from the start point to the end point, whereas authors in [12] used the improved dynamic programming algorithm for application to the SPP of a trapezoidal fuzzy medium intelligence graph, starting the search from the end point, and the NN was not accurate in the operation process.

After that, Kumar et al. [14] used linear programming approach to solve Gaussian valued NSPP. Saad et al. [15] develop a novel algorithm for finding shortest path in a single valued neutrosophic hesitant fuzzy network (SVNHFN). In addition, the author also introduced the concept of SVNHFN with some related graph theoretical results such as complement, subgraph, degree, and path etc. Biswas [16] developed NSPP in a directed multigraph. The multigraph is a topological generalization of the graph where multiple links (or edges/arcs) may exist between two nodes unlike in graph.

In addition, Prabha et al. [17] investigated a SPP with IVNN using the A* algorithm. This algorithm is extensively applied in path finding and graph traversal. Later, Chakraborty [18] applied the developed score function and accuracy function of the pentagonal NN to the SPP and Yang et al. [19] studied on the shortest path solution method of interval valued neutrosophic graph based on the ant colony algorithm. Furthermore, the author also investigated the convergence processes of the ant colony algorithm with different parameter settings and used different score functions to solve the SPP.

During the same year, the same authors, Yang et al. [20] solved SPP of the neutrosophic graph with an edge distance expressed using trapezoidal fuzzy neutrosophic numbers (TrFNN) and resolve the edge distance according to the score and exact functions based on the TrFNN. Accordingly, the use of a circle-breaking algorithm is proposed to solve the shortest path problem and estimate the shortest distance. Other than that, Liu [21] studied on single-valued neutrosophic graph (SVNG) with application in SPP and consider Bellman–Ford algorithm for SPP using neutrosophic number as arc length. In this study, the authors also discussed the definition of regular SVNG, complete SVNG and strong SVNG.

3 PRELIMINARIES

In this section, some basic concepts related to SVNS, BNS and IVBNS sets are presented.

Definition 1[22]: Let X be a universe of discourse. Then a SVNS is defined as:

$$A=\left\{\left\langle x,T_A(x),I_A(x),F_A(x)\right\rangle : x\in X\right\}, \tag{1}$$

which is characterized by a truth-membership function $T_A(x): X\to[0,1]$, an indeterminacy-membership function $I_A(x): X\to[0,1]$, and a falsity-membership function $F_A(x): X\to[0,1]$. There is no restriction on the sum of $T_A(x), I_A(x)$ and $F_A(x)$ therefore $0\le T_A(x), I_A(x), F_A(x)\le 3$.

Definition 2 [3]: A BNS A in X is defined as an object of the form

$$A=\left\{\left\langle x,T^{+}(x),I^{+}(x),F^{+}(x),T^{-}(x),I^{-}(x),F^{-}(x)\right\rangle : x\in X\right\}, \tag{2}$$

where $T^{+}(x),I^{+}(x),F^{+}(x):X\to[0,1]$ and $T^{-}(x),I^{-}(x),F^{-}(x):X\to[-1,0]$.

Definition 3 [3]: An IVBNS A in X is defined as an object of the form

$$A=\left\langle\left[T_L^{+}(x),T_R^{+}(x)\right],\left[I_L^{+}(x),I_R^{+}(x)\right],\left[F_L^{+}(x),F_R^{+}(x)\right],\left[T_L^{-}(x),T_R^{-}(x)\right],\left[I_L^{-}(x),I_R^{-}(x)\right],\left[F_L^{-}(x),F_R^{-}(x)\right]\right\rangle$$

where T_L^{+}, T_R^{+}, I_L^{+}, I_R^{+}, F_L^{+}, $F_R^{+}:X\to[0,1]$ and T_L^{-}, T_R^{-}, I_L^{-}, I_R^{-}, F_L^{-}, $F_R^{-}:X\to[-1,0]$. (3)

Definition 4 [3]: Let $A=\left\langle\left[T_L^{+},T_R^{+}\right],\left[I_L^{+},I_R^{+}\right],\left[F_L^{+},F_R^{+}\right],\left[T_L^{-},T_R^{-}\right],\left[I_L^{-},I_R^{-}\right],\left[F_L^{-},F_R^{-}\right]\right\rangle$ be an interval-valued bipolar neutrosophic number (IVBNN). Then, the score function $S(A)$ of IVBNN is defined as follows:

$$S(A)=\frac{1}{12}\left(T_L^{+}+T_R^{+}+1-I_L^{+}+1-I_R^{+}+1-F_L^{+}+1-F_R^{+}+1-T_L^{-}+1-T_R^{-}-I_L^{-}-I_R^{-}-F_L^{-}-F_R^{-}\right) \tag{4}$$

4 THE PROPOSED METHODOLOGY

In this section, we slightly modified the single valued neutrosophic Dijkstra's algorithm adopted from Broumi et al. (2016) in order to deal on a network with parameters characterized by an IVBNN. This algorithm finds the shortest path and the shortest distance between a source vertex and any other vertex in the network. The algorithms advance from a vertex *i* to an immediately successive vertex *j* using a neutrosophic labeling process.

Consider a directed graph network, $G(V,E)$ consisting of a finite set of nodes $V=\{1,2,\ldots,n\}$ and a set of *m* directed edges $E\subseteq V\times V$. Each edge is denoted by an ordered pair (i,j) where $i,j\in V$ and $i\neq j$. In this network, we specify two vertices, denoted by *s* and *t*, which are the source vertex and the destination vertex, respectively. We define a path as a sequence

$$P_{ij}=\left\{i=i_1,(i_1,i_2),i_2,\ldots,(i_{\ell-1},i_\ell),i_\ell=j\right\}$$

of alternating nodes and edges. The existence of at least one path P_{si} in $G(V,E)$ is assumed for every $i\in V-\{s\}$. d_{ij} denotes an IVBNN associated with the edge (i,j), corresponding to the length necessary to traverse (i,j) from *i* to *j*. In real problems, the lengths of the edge correspond to distance, cost, time etc. Then, neutrosophic distance along the path *P* is denoted as $d(P)$ is defined as:

$$d(P)=\sum_{(i,j\in P)} d_{ij}$$

Let $\hat{u}_i$ be the shortest distance from vertex 1 to vertex *i* and $s\left(\hat{d}_{ij}\right)\geq 0$ be the length of (i,j) - edge. Then, neutrosophic label for vertex *j* is defined as:

$\left[\hat{u}_j,i\right]=\left[\hat{u}_i\oplus\hat{d}_{ij},i\right]$, where $S\left(\hat{d}_{ij}\right)\geq 0$.

Here, label $\left[\hat{u}_j,i\right]$ means that we are coming from vertex *i* after covering a distance $\hat{u}_j$ from the starting vertex. Dijkstra's algorithm divides the vertices into two subset group which are temporary set (T) and permanent set (P). A temporary neutrosophic label can be replaced with another temporary neutrosophic label, if shortest path to the same neutrosophic vertex is detected. At the point when no better path can be found, the status of temporary label is changed to permanent. The steps of the algorithm are summarized as follows:

Step 1: Assume $\hat{d}_i=\langle[0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0]\rangle$ and assign the first vertex / source vertex as the permanent label $\left(\hat{d}_i=\langle[0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0]\rangle,-\right)$. Making a vertex permanent means that it has been included in shortest path.

Step 2: Set $i=1$.

Step 3: Compute the temporary label $\left[\hat{u}_i\oplus\hat{d}_{ij},i\right]$ for each vertex *j* that incident from *i*, provided *j* is not permanently labeled.

Step 4: Use score function in Definition 4 to rank each vertex *j* incident to *i* and select for which $\hat{d}_{ij}$ minimum.

Step 5: If vertex *j* is already labeled as $\left[\hat{u}_j,k\right]$ through another vertex *k* and if $S\left(\hat{u}_i\oplus\hat{d}_{ij}\right)<S\left(u_j\right)$, replace $\left[\hat{u}_j,k\right]$ with $\left[\hat{u}_i\oplus\hat{d}_{ij},i\right]$.

Step 6: If all the vertices are permanently labeled, the algorithm terminates. Otherwise, choose the label $\left[\hat{u}_r,s\right]$ with shortest distance $\left(\hat{u}_r\right)$ from the list of temporary labels.

Step 7: Set $i=r$ and repeat Step 3.

Step 8: Determine the shortest path between vertex 1 and the destination vertex *j* by tracing backward through the network using the label's information.

Remarks: At each iteration among all temporary vertices, make those vertices permanent which have smallest distance. Note that at any iteration, we cannot move to permanent vertices, however,

reverse is possible. After all, the vertices have permanent labels and only one temporary vertex remains, make it permanent.

To further demonstrate the algorithm, a flowchart is depicted in Figure 1. Figure 2 show the interval-valued bipolar neutrosophic graph.

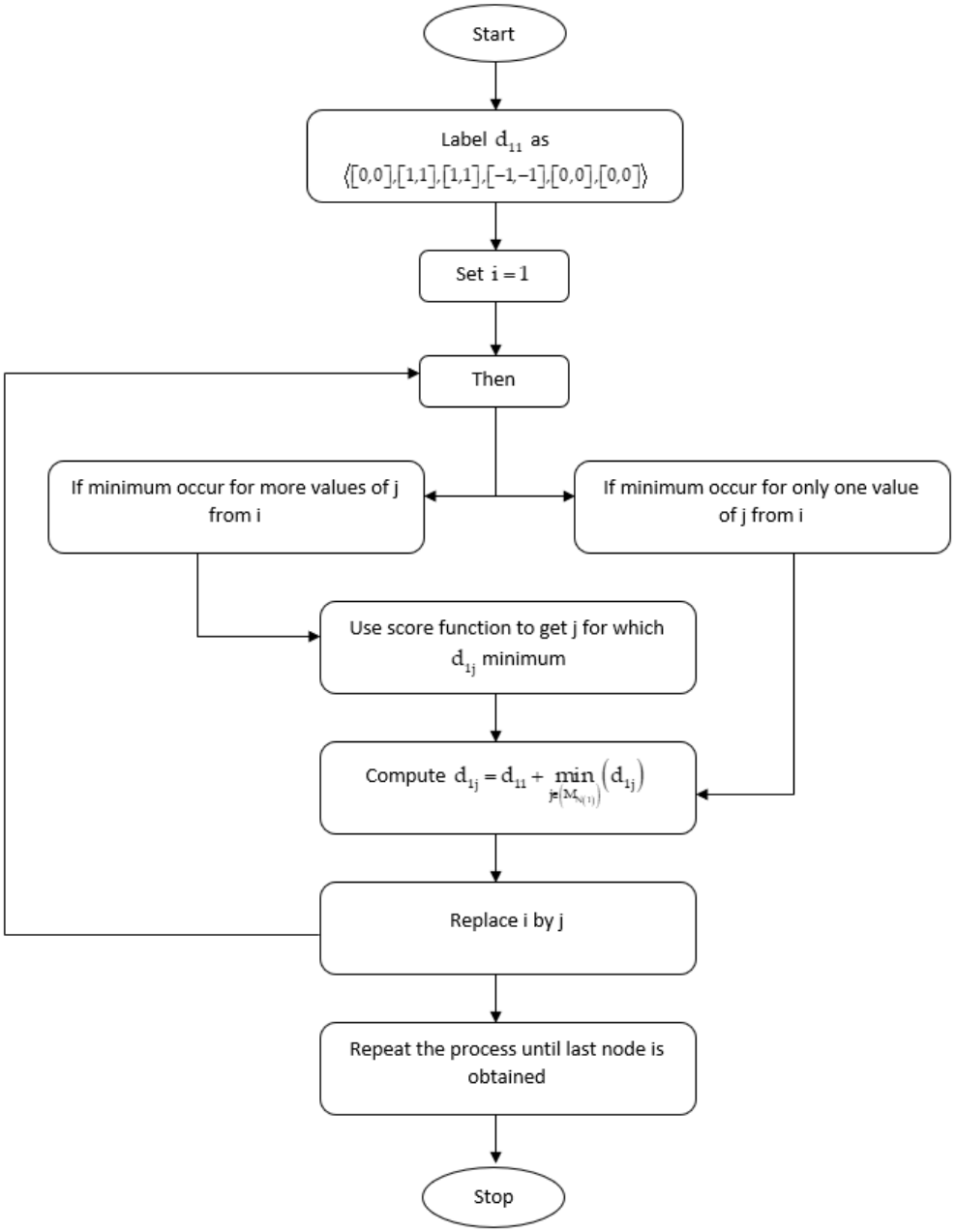

Figure 1: Flowchart of the proposed study

Illustrative Example:

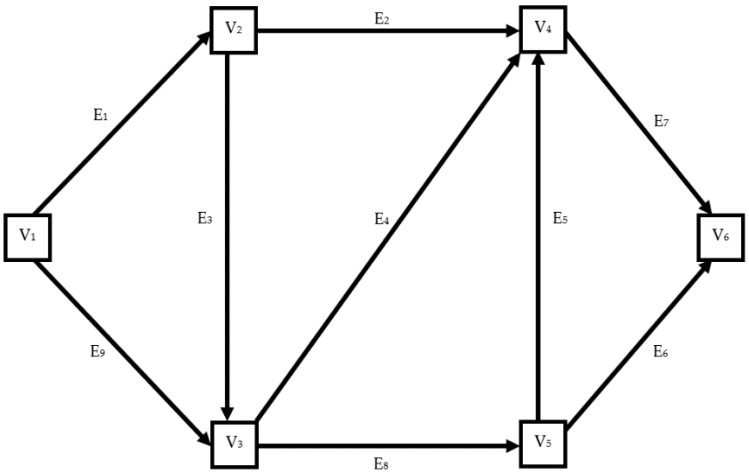

Figure 2: Interval-valued bipolar neutrosophic graph

Table 1: Details of edge information in terms of IVBNN

Edges	End vertices	Weight of The Edges
E_1	V_1V_2	$\left(\begin{matrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3]\end{matrix}\right)$
E_2	V_2V_4	$\left(\begin{matrix}[0.2,0.4],[0.3,0.6],[0.4,0.8],\\ [-0.3,-0.2],[-0.6,-0.4],[-0.4,-0.3]\end{matrix}\right)$
E_3	V_2V_3	$\left(\begin{matrix}[0.4,0.5],[0.2,0.4],[0.3,0.7],\\ [-0.4,-0.3],[-0.6,-0.5],[-0.8,-0.7]\end{matrix}\right)$
E_4	V_3V_4	$\left(\begin{matrix}[0.1,0.3],[0.3,0.4],[0.5,0.7],\\ [-0.5,-0.4],[-0.6,-0.3],[-0.6,-0.5]\end{matrix}\right)$
E_5	V_4V_5	$\left(\begin{matrix}[0.3,0.6],[0.2,0.4],[0.5,0.6],\\ [-0.3,-0.1],[-0.4,-0.3],[-0.7,-0.5]\end{matrix}\right)$
E_6	V_5V_6	$\left(\begin{matrix}[0.3,0.4],[0.2,0.4],[0.4,0.6],\\ [-0.4,-0.3],[-0.7,-0.6],[-0.8,-0.7]\end{matrix}\right)$
E_7	V_4V_6	$\left(\begin{matrix}[0.2,0.3],[0.3,0.5],[0.1,0.3],\\ [-0.5,-0.4],[-0.4,-0.2],[-0.7,-0.6]\end{matrix}\right)$
E_8	V_3V_5	$\left(\begin{matrix}[0.5,0.6],[0.6,0.8],[0.3,0.4],\\ [-0.5,-0.4],[-0.6,-0.5],[-0.9,-0.7]\end{matrix}\right)$
E_9	V_1V_3	$\left(\begin{matrix}[0.3,0.4],[0.2,0.5],[0.1,0.2],\\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6]\end{matrix}\right)$

Iteration 0:

Step 1: Assign the permanent label $\langle[0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0]\rangle$ to vertex 1 (source vertex).

Vertex	Weight	Status
1	$\langle[0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0],-\rangle$	P

Step 2: Set $i=1$. Assume $\hat{d}_1=\langle[0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0],-\rangle$, then the value of $\hat{d}_j; j=2,3,4,5,6$ can be obtained as follows:

Iteration 1:

Step 3: Compute the temporary label for each vertex $j=2,3$ that incident from $i=1$. (Vertex 2 and vertex 3 are incident from (the last permanently labeled) vertex 1). Then, the value of $\hat{d}_2$ and $\hat{d}_3$ as follows:

$$\hat{d}_2=\langle\hat{d}_1\oplus\hat{d}_{12}\rangle$$
$$=\left\langle\begin{pmatrix}[0,0],[1,1],[1,1],\\ [-1,-1],[0,0],[0,0]\end{pmatrix}\oplus\begin{pmatrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3]\end{pmatrix}\right\rangle$$
$$=\langle[0.6,0.8],[0.5,0.7],[0.4,0.6],[-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3]\rangle$$

$$\hat{d}_3=\langle\hat{d}_1\oplus\hat{d}_{13}\rangle$$
$$=\left\langle\begin{pmatrix}[0,0],[1,1],[1,1],\\ [-1,-1],[0,0],[0,0]\end{pmatrix}\oplus\begin{pmatrix}[0.3,0.4],[0.2,0.5],[0.1,0.2],\\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6]\end{pmatrix}\right\rangle$$
$$=\langle[0.3,0.4],[0.2,0.5],[0.1,0.2],[-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6]\rangle$$

Since minimum occurs corresponding to vertex 1, thus, the list of labeled vertices (temporary and permanent) becomes:

Vertices	Weight	Status
1	$\langle[0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0],-\rangle$	P
2	$\left\langle \begin{matrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3],1\end{matrix}\right\rangle$	T
3	$\left\langle \begin{matrix}[0.3,0.4],[0.2,0.5],[0.1,0.2],\\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6],1\end{matrix}\right\rangle$	T

Step 4: In order to compare the weight for each vertex (among temporary labels), we use score function in Definition 6:

$$S_{2,1}\left\langle \begin{matrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3]\end{matrix}\right\rangle = \frac{1}{12}\left(\begin{matrix} T_L^P + T_U^P + 1 - I_L^P + 1 - I_U^P + 1 - F_L^P + 1 - \\ F_U^P + 1 + T_L^N + 1 + T_U^N - I_L^N - I_U^N - F_L^N - F_U^P \end{matrix}\right)$$

$$= \frac{1}{12}\left(\begin{matrix} 0.6+0.8+1-0.5+1-0.7+1-0.4+1-0.6+1+ \\ (-0.2)+1+(-0.1)-(-0.3)-(-0.2)-(-0.4)-(-0.3) \end{matrix}\right) = 0.5083$$

$$S_{3,1}\left\langle \begin{matrix}[0.3,0.4],[0.2,0.5],[0.1,0.2],\\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6]\end{matrix}\right\rangle = \frac{1}{12}\left(\begin{matrix} T_L^P + T_U^P + 1 - I_L^P + 1 - I_U^P + 1 - F_L^P + 1 - \\ F_U^P + 1 + T_L^N + 1 + T_U^N - I_L^N - I_U^N - F_L^N - F_U^P \end{matrix}\right)$$

$$= \frac{1}{12}\left(\begin{matrix} 0.3+0.4+1-0.2+1-0.5+1-0.1+1-0.2+1+ \\ (-0.4)+1+(-0.3)-(-0.6)-(-0.4)-(-0.8)-(-0.6) \end{matrix}\right) = 0.6167$$

Step 5: Since the weight of

$$S_{2,1}\left\langle \begin{matrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3]\end{matrix}\right\rangle < S_{3,1}\left\langle \begin{matrix}[0.3,0.4],[0.2,0.5],[0.1,0.2],\\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6]\end{matrix}\right\rangle,$$

thus, the status of vertex 2 is changed to permanent.

Step 6 and 7: If all vertices are permanently labeled, the algorithm terminates. Otherwise, repeat Step 3.

Iteration 2:

Step 3: Compute the temporary label for each vertex $j=3,4$ that incident from $i=2$. (Vertex 3 and vertex 4 are incident from (the last permanently labeled) vertex 2). Then, the value of $\hat{d}_3$ and $\hat{d}_4$ as follows:

$$\hat{d}_3=\left\langle \hat{d}_2 \oplus \hat{d}_{23} \right\rangle$$
$$=\left\langle \begin{pmatrix} [0.6,0.8],[0.5,0.7],[0.4,0.6], \\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3] \end{pmatrix} \oplus \begin{pmatrix} [0.4,0.5],[0.2,0.4],[0.3,0.7], \\ [-0.4,-0.3],[-0.6,-0.5],[-0.8,-0.7] \end{pmatrix} \right\rangle$$
$$=\left\langle [0.76,0.9],[0.1,0.28],[0.12,0.42],[-0.08,-0.03],[-0.72,-0.6],[-0.88,-0.79] \right\rangle$$

$$\hat{d}_4=\left\langle \hat{d}_2 \oplus \hat{d}_{24} \right\rangle$$
$$=\left\langle \begin{pmatrix} [0.6,0.8],[0.5,0.7],[0.4,0.6], \\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3] \end{pmatrix} \oplus \begin{pmatrix} [0.2,0.4],[0.3,0.6],[0.4,0.8], \\ [-0.3,-0.2],[-0.6,-0.4],[-0.4,-0.3] \end{pmatrix} \right\rangle$$
$$=\left\langle [0.68,0.88],[0.15,0.42],[0.16,0.48],[-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51] \right\rangle$$

Since minimum occurs corresponding to vertex 2, thus, the list of labeled vertices (temporary and permanent) becomes:

Vertices	Weight	Status
1	$\langle [0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0],- \rangle$	P
2	$\left\langle \begin{matrix} [0.6,0.8],[0.5,0.7],[0.4,0.6], \\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3],1 \end{matrix} \right\rangle$	P
3	$\left\langle \begin{matrix} [0.3,0.4],[0.2,0.5],[0.1,0.2], \\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6],1 \end{matrix} \right\rangle$	T
	or	
	$\left\langle \begin{matrix} [0.76,0.9],[0.1,0.28],[0.12,0.42], \\ [-0.08,-0.03],[-0.72,-0.6],[-0.88,-0.79],2 \end{matrix} \right\rangle$	
4	$\left\langle \begin{matrix} [0.68,0.88],[0.15,0.42],[0.16,0.48], \\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51],2 \end{matrix} \right\rangle$	T

Step 4: In order to compare the weight for each vertex (among temporary labels), we use score function in Definition 6:

$$S_{3,2}\left\langle \begin{matrix} [0.76,0.9],[0.1,0.28],[0.12,0.42], \\ [-0.08,-0.03],[-0.72,-0.6],[-0.88,-0.79] \end{matrix} \right\rangle = \frac{1}{12}\left(\begin{matrix} T_L^P + T_U^P + 1 - I_L^P + 1 - I_U^P + 1 - F_L^P + 1 - \\ F_U^P + 1 + T_L^N + 1 + T_U^N - I_L^N - I_U^N - F_L^N - F_U^P \end{matrix} \right)$$

$$= \frac{1}{12}\left(\begin{matrix} 0.76 + 0.9 + 1 - 0.1 + 1 - 0.28 + 1 - 0.12 + 1 - 0.42 + 1 + \\ (-0.08) + 1 + (-0.03) - (-0.72) - (-0.6) - (-0.88) - (-0.79) \end{matrix} \right) = 0.807$$

$$S_{4,2}\left\langle \begin{matrix} [0.68,0.88],[0.15,0.42],[0.16,0.48], \\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51] \end{matrix} \right\rangle = \frac{1}{12}\left(\begin{matrix} T_L^P + T_U^P + 1 - I_L^P + 1 - I_U^P + 1 - F_L^P + 1 - \\ F_U^P + 1 + T_L^N + 1 + T_U^N - I_L^N - I_U^N - F_L^N - F_U^P \end{matrix} \right)$$

$$= \frac{1}{12}\left(\begin{matrix} 0.68 + 0.88 + 1 - 0.15 + 1 - 0.42 + 1 - 0.16 + 1 - 0.48 + 1 + \\ (-0.06) + 1 + (-0.02) - (-0.72) - (-0.52) - (-0.64) - (-0.51) \end{matrix} \right) = 0.7217$$

Step 5: Among the temporary labels $S_{3,1}, S_{3,2}$ and $S_{4,2}$, the weight of $S_{3,1} < S_{3,2}$ and $S_{4,2}$. Therefore, the status of vertex 3 is changed to permanent.

Step 6 and 7: If all vertices are permanently labeled, the algorithm terminates. Otherwise, repeat Step 3.

Iteration 3:

Step 3: Compute the temporary label for each vertex $j = 4,5$ that incident from $i = 3$. (Vertex 4 and vertex 5 are incident from (the last permanently labeled) vertex 3). Then, the value of $\hat{d}_4$ and $\hat{d}_5$ as follows:

$$\hat{d}_4 = \left\langle \hat{d}_3 \oplus \hat{d}_{34} \right\rangle$$

$$= \left\langle \left(\begin{matrix} [0.3,0.4],[0.2,0.5],[0.1,0.2], \\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6] \end{matrix} \right) \oplus \left(\begin{matrix} [0.1,0.3],[0.3,0.4],[0.5,0.7], \\ [-0.5,-0.4],[-0.6,-0.3],[-0.6,-0.5] \end{matrix} \right) \right\rangle$$

$$= \left\langle [0.37,0.58],[0.06,0.2],[0.05,0.14],[-0.2,-0.12],[-0.84,-0.58],[-0.92,-0.8] \right\rangle$$

$$\hat{d}_5 = \left\langle \hat{d}_3 \oplus \hat{d}_{35} \right\rangle$$

$$= \left\langle \left(\begin{matrix} [0.3,0.4],[0.2,0.5],[0.1,0.2], \\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6] \end{matrix} \right) \oplus \left(\begin{matrix} [0.5,0.6],[0.6,0.8],[0.3,0.4], \\ [-0.5,-0.4],[-0.6,-0.5],[-0.9,-0.7] \end{matrix} \right) \right\rangle$$

$$= \left\langle [0.65,0.76],[0.12,0.4],[0.03,0.08],[-0.2,-0.12],[-0.84,-0.7],[-0.98,-0.88] \right\rangle$$

Since minimum occurs corresponding to vertex 3, thus, the list of labeled vertices (temporary and permanent) becomes:

Vertices	Weight	Status
1	$\langle[0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0],-\rangle$	P
2	$\left\langle \begin{matrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3],1\end{matrix}\right\rangle$	P
3	$\left\langle \begin{matrix}[0.3,0.4],[0.2,0.5],[0.1,0.2],\\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6],1\end{matrix}\right\rangle$	P
4	$\left\langle \begin{matrix}[0.68,0.88],[0.15,0.42],[0.16,0.48],\\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51],2\end{matrix}\right\rangle$ or $\left\langle \begin{matrix}[0.37,0.58],[0.06,0.2],[0.05,0.14],\\ [-0.2,-0.12],[-0.84,-0.58],[-0.92,-0.8],3\end{matrix}\right\rangle$	T
5	$\left\langle \begin{matrix}[0.65,0.76],[0.12,0.4],[0.03,0.08],\\ [-0.2,-0.12],[-0.84,-0.7],[-0.98,-0.88],3\end{matrix}\right\rangle$	T

Step 4: In order to compare the weight for each vertex (among temporary labels), we use score function in Definition 6:

$$S_{4,3}\left\langle \begin{matrix}[0.37,0.58],[0.06,0.2],[0.05,0.14],\\ [-0.2,-0.12],[-0.84,-0.58],[-0.92,-0.8]\end{matrix}\right\rangle=\frac{1}{12}\left(\begin{matrix}T_L^P+T_U^P+1-I_L^P+1-I_U^P+1-F_L^P+1-\\ F_U^P+1+T_L^N+1+T_U^N-I_L^N-I_U^N-F_L^N-F_U^P\end{matrix}\right)$$

$$=\frac{1}{12}\left(\begin{matrix}0.37+0.58+1-0.06+1-0.2+1-0.05+1-0.14+1+\\ (-0.2)+1+(-0.12)-(-0.84)-(-0.58)-(-0.92)-(-0.8)\end{matrix}\right)=0.7767$$

$$S_{5,3}\left\langle \begin{matrix}[0.65,0.76],[0.12,0.4],[0.03,0.08],\\ [-0.2,-0.12],[-0.84,-0.7],[-0.98,-0.88]\end{matrix}\right\rangle=\frac{1}{12}\left(\begin{matrix}T_L^P+T_U^P+1-I_L^P+1-I_U^P+1-F_L^P+1-\\ F_U^P+1+T_L^N+1+T_U^N-I_L^N-I_U^N-F_L^N-F_U^P\end{matrix}\right)$$

$$=\frac{1}{12}\left(\begin{matrix}0.65+0.76+1-0.12+1-0.4+1-0.03+1-0.08+1+\\ (-0.2)+1+(-0.12)-(-0.84)-(-0.7)-(-0.98)-(-0.88)\end{matrix}\right)=0.8217$$

Step 5: Among the temporary labels $S_{4,2}, S_{4,3}$ and $S_{5,3}$, the rank of $S_{4,2}<S_{4,3}$ and $S_{5,3}$. Therefore, the status of vertex 4 is changed to permanent.

Step 6 and 7: If all vertices are permanently labeled, the algorithm terminates. Otherwise, repeat Step 3.

Iteration 4:

Step 3: Compute the temporary label for each vertex $j=5,6$ that incident from $i=4$. (Vertex 5 and vertex 6 are incident from (the last permanently labeled) vertex 4). Then, the value of $\hat{d}_5$ and $\hat{d}_6$ as follows:

$$\hat{d}_5 = \left\langle \hat{d}_4 \oplus \hat{d}_{45} \right\rangle$$
$$= \left\langle \begin{pmatrix} [0.68,0.88],[0.15,0.42],[0.16,0.48], \\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51] \end{pmatrix} \oplus \begin{pmatrix} [0.3,0.6],[0.2,0.4],[0.5,0.6], \\ [-0.3,-0.1],[-0.4,-0.3],[-0.7,-0.5] \end{pmatrix} \right\rangle$$
$$= \left\langle \begin{matrix} [0.776,0.952],[0.03,0.168],[0.08,0.288], \\ [-0.018,-0.002],[-0.832,-0.664],[-0.892,-0.755] \end{matrix} \right\rangle$$

$$\hat{d}_6 = \left\langle \hat{d}_4 \oplus \hat{d}_{46} \right\rangle$$
$$= \left\langle \begin{pmatrix} [0.68,0.88],[0.15,0.42],[0.16,0.48], \\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51] \end{pmatrix} \oplus \begin{pmatrix} [0.2,0.3],[0.3,0.5],[0.1,0.3], \\ [-0.5,-0.4],[-0.4,-0.2],[-0.7,-0.6] \end{pmatrix} \right\rangle$$
$$= \left\langle \begin{matrix} [0.744,0.916],[0.045,0.21],[0.016,0.144], \\ [-0.03,-0.008],[-0.832,-0.616],[-0.892,-0.804] \end{matrix} \right\rangle$$

Since minimum occurs corresponding to vertex 4, thus, the list of labeled vertices (temporary and permanent) becomes:

Vertices	Weight	Status
1	$\langle [0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0],- \rangle$	P
2	$\left\langle \begin{matrix} [0.6,0.8],[0.5,0.7],[0.4,0.6], \\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3],1 \end{matrix} \right\rangle$	P
3	$\left\langle \begin{matrix} [0.3,0.4],[0.2,0.5],[0.1,0.2], \\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6],1 \end{matrix} \right\rangle$	P
4	$\left\langle \begin{matrix} [0.68,0.88],[0.15,0.42],[0.16,0.64], \\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51],2 \end{matrix} \right\rangle$	P

5	$\left\langle \begin{matrix} [0.65,0.76],[0.12,0.4],[0.03,0.08], \\ [-0.2,-0.12],[-0.84,-0.7],[-0.98,-0.88],3 \end{matrix} \right\rangle$	T
	or	
	$\left\langle \begin{matrix} [0.776,0.952],[0.03,0.168],[0.08,0.288], \\ [-0.018,-0.002],[-0.832,-0.664],[-0.892,-0.755],4 \end{matrix} \right\rangle$	
6	$\left\langle \begin{matrix} [0.744,0.916],[0.045,0.21],[0.016,0.144], \\ [-0.03,-0.008],[-0.832,-0.616],[-0.892,-0.804],4 \end{matrix} \right\rangle$	T

Step 4: In order to compare the weight for each vertex (among temporary labels), we use score function in Definition 6:

$$S_{5,4}\left\langle \begin{matrix} [0.776,0.952],[0.03,0.168],[0.08,0.288], \\ [-0.018,-0.002],[-0.832,-0.664],[-0.892,-0.755] \end{matrix} \right\rangle = \frac{1}{12}\left(\begin{matrix} T_L^P + T_U^P + 1 - I_L^P + 1 - I_U^P + 1 - F_L^P + 1 - \\ F_U^P + 1 + T_L^N + 1 + T_U^N - I_L^N - I_U^N - F_L^N - F_U^P \end{matrix} \right)$$

$$= \frac{1}{12}\left(\begin{matrix} 0.776 + 0.952 + 1 - 0.03 + 1 - 0.168 + 1 - 0.08 + 1 - \\ 0.288 + 1 + (-0.018) + 1 + (-0.002) - (-0.832) - \\ (-0.664) - (-0.892) - (-0.755) \end{matrix} \right) = 0.8571$$

$$S_{6,4}\left\langle \begin{matrix} [0.744,0.916],[0.045,0.21],[0.016,0.144], \\ [-0.03,-0.008],[-0.832,-0.616],[-0.892,-0.804] \end{matrix} \right\rangle = \frac{1}{12}\left(\begin{matrix} T_L^P + T_U^P + 1 - I_L^P + 1 - I_U^P + 1 - F_L^P + 1 - \\ F_U^P + 1 + T_L^N + 1 + T_U^N - I_L^N - I_U^N - F_L^N - F_U^P \end{matrix} \right)$$

$$= \frac{1}{12}\left(\begin{matrix} 0.744 + 0.916 + 1 - 0.045 + 1 - 0.21 + 1 - 0.016 + 1 - \\ 0.144 + 1 + (-0.03) + 1 + (-0.008) - (-0.832) - \\ (-0.616) - (-0.892) - (-0.804) \end{matrix} \right) = 0.8626$$

Step 5: Among the temporary labels $S_{5,3}, S_{5,4}$ and $S_{6,4}$, the rank of $S_{5,3} < S_{5,4}$ and $S_{6,4}$. Therefore, the status of vertex 5 is changed to permanent.

Step 6 and 7: If all vertices are permanently labeled, the algorithm terminates. Otherwise, repeat Step 3.

Iteration 5:

Step 3: Compute the temporary label for each vertex $j = 6$ that incident from $i = 5$. (Vertex 6 is incident from (the last permanently labeled) vertex 5). Then, the value of $\hat{d}_6$ as follows:

$$\hat{d}_6 = \left\langle \hat{d}_5 \oplus \hat{d}_{56} \right\rangle$$
$$= \left\langle \begin{pmatrix} [0.65,0.76],[0.12,0.4],[0.03,0.08], \\ [-0.2,-0.12],[-0.84,-0.7],[-0.98,-0.88] \end{pmatrix} \oplus \begin{pmatrix} [0.3,0.4],[0.2,0.4],[0.4,0.6], \\ [-0.4,-0.3],[-0.7,-0.6],[-0.8,-0.7] \end{pmatrix} \right\rangle$$
$$= \left\langle \begin{matrix} [0.755,0.856],[0.024,0.16],[0.012,0.048], \\ [-0.08,-0.036],[-0.952,-0.88],[-0.996,-0.964] \end{matrix} \right\rangle$$

Since minimum occurs corresponding to vertex 5, thus, the list of labeled vertices (temporary and permanent) becomes:

Vertices	Weight	Status
1	$\langle [0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0],- \rangle$	P
2	$\left\langle \begin{matrix} [0.6,0.8],[0.5,0.7],[0.4,0.6], \\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3],1 \end{matrix} \right\rangle$	P
3	$\left\langle \begin{matrix} [0.3,0.4],[0.2,0.5],[0.1,0.2], \\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6],1 \end{matrix} \right\rangle$	P
4	$\left\langle \begin{matrix} [0.68,0.88],[0.15,0.42],[0.16,0.64], \\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51],2 \end{matrix} \right\rangle$	P
5	$\left\langle \begin{matrix} [0.65,0.76],[0.12,0.4],[0.03,0.08], \\ [-0.2,-0.12],[-0.84,-0.7],[-0.98,-0.88],3 \end{matrix} \right\rangle$	P
6	$\left\langle \begin{matrix} [0.744,0.916],[0.045,0.21],[0.016,0.192], \\ [-0.03,-0.008],[-0.832,-0.616],[-0.892,-0.804],4 \end{matrix} \right\rangle$	T
	or	
	$\left\langle \begin{matrix} [0.755,0.856],[0.024,0.16],[0.012,0.048], \\ [-0.08,-0.036],[-0.952,-0.88],[-0.996,-0.964],5 \end{matrix} \right\rangle$	

Step 4: In order to compare the weight for each vertex (among temporary labels), we use score function in Definition 6:

$$S_{6,5}\left\langle \begin{matrix} [0.755,0.856],[0.024,0.16],[0.012,0.048], \\ [-0.08,-0.036],[-0.952,-0.88],[-0.996,-0.964] \end{matrix} \right\rangle = \frac{1}{12}\begin{pmatrix} T_L^P + T_U^P + 1 - I_L^P + 1 - I_U^P + 1 - F_L^P + 1 - \\ F_U^P + 1 + T_L^N + 1 + T_U^N - I_L^N - I_U^N - F_L^N - F_U^P \end{pmatrix}$$

$$=\frac{1}{12}\begin{pmatrix}0.755+0.856+1-0.024+1-0.16+1-0.012+1-\\0.048+1+(-0.08)+1+(-0.036)-(-0.952)-\\(-0.88)-(-0.996)-(-0.964)\end{pmatrix}=0.9203$$

Step 5: Among the temporary labels $S_{6,4}$ and $S_{6,5}$, the rank of $S_{6,4} < S_{6,5}$. Therefore, the status of node 6 is changed to permanent.

Vertices	Weight	Status
1	$\langle[0,0],[1,1],[1,1],[-1,-1],[0,0],[0,0],-\rangle$	P
2	$\left\langle \begin{matrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3],1\end{matrix}\right\rangle$	P
3	$\left\langle \begin{matrix}[0.3,0.4],[0.2,0.5],[0.1,0.2],\\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6],1\end{matrix}\right\rangle$	P
4	$\left\langle \begin{matrix}[0.68,0.88],[0.15,0.42],[0.16,0.64],\\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51],2\end{matrix}\right\rangle$	P
5	$\left\langle \begin{matrix}[0.65,0.76],[0.12,0.4],[0.03,0.08],\\ [-0.2,-0.12],[-0.84,-0.7],[-0.98,-0.88],3\end{matrix}\right\rangle$	P
6	$\left\langle \begin{matrix}[0.744,0.916],[0.045,0.21],[0.016,0.192],\\ [-0.03,-0.008],[-0.832,-0.616],[-0.892,-0.804],4\end{matrix}\right\rangle$	P

Step 6 and 7: Since all vertices are permanently labeled, the algorithm terminates.

Step 8: The interval-valued bipolar shortest path between vertex 1 (source vertex) and vertex 6 (destination vertex) is obtained by tracing backward through the network using label's information as follows:

$$(6)\rightarrow\left\langle\begin{matrix}[0.744,0.916],[0.045,0.21],[0.016,0.192],\\ [-0.03,-0.008],[-0.832,-0.616],[-0.892,-0.804],4\end{matrix}\right\rangle\rightarrow(4)\rightarrow$$
$$\left\langle\begin{matrix}[0.68,0.88],[0.15,0.42],[0.16,0.64],\\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51],2\end{matrix}\right\rangle\rightarrow(2)\rightarrow$$
$$\left\langle\begin{matrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3],1\end{matrix}\right\rangle\rightarrow(1)$$

5 RESULT AND DISCUSSION

As shown in Table 2 below, for IVBN-SPP Dijkstra's algorithm, we can see that the required shortest path from the source vertex to the destination vertex is $1\rightarrow2\rightarrow4\rightarrow6$. In addition, labeling of each vertex is shown in the figure 3.

Table 2: IVBN-SPP Dijkstra's algorithm

Vertices	d_i	The shortest path from vertex 1 to j^{th} vertex
2	$\left\langle\begin{matrix}[0.6,0.8],[0.5,0.7],[0.4,0.6],\\ [-0.2,-0.1],[-0.3,-0.2],[-0.4,-0.3]\end{matrix}\right\rangle$	$1\rightarrow2$
3	$\left\langle\begin{matrix}[0.3,0.4],[0.2,0.5],[0.1,0.2],\\ [-0.4,-0.3],[-0.6,-0.4],[-0.8,-0.6]\end{matrix}\right\rangle$	$1\rightarrow3$
4	$\left\langle\begin{matrix}[0.68,0.88],[0.15,0.42],[0.16,0.64],\\ [-0.06,-0.02],[-0.72,-0.52],[-0.64,-0.51]\end{matrix}\right\rangle$	$1\rightarrow2\rightarrow4$
5	$\left\langle\begin{matrix}[0.65,0.76],[0.12,0.4],[0.03,0.08],\\ [-0.2,-0.12],[-0.84,-0.7],[-0.98,-0.88]\end{matrix}\right\rangle$	$1\rightarrow3\rightarrow5$
6	$\left\langle\begin{matrix}[0.744,0.916],[0.045,0.21],[0.016,0.192],\\ [-0.03,-0.008],[-0.832,-0.616],[-0.892,-0.804]\end{matrix}\right\rangle$	$1\rightarrow2\rightarrow4\rightarrow6$

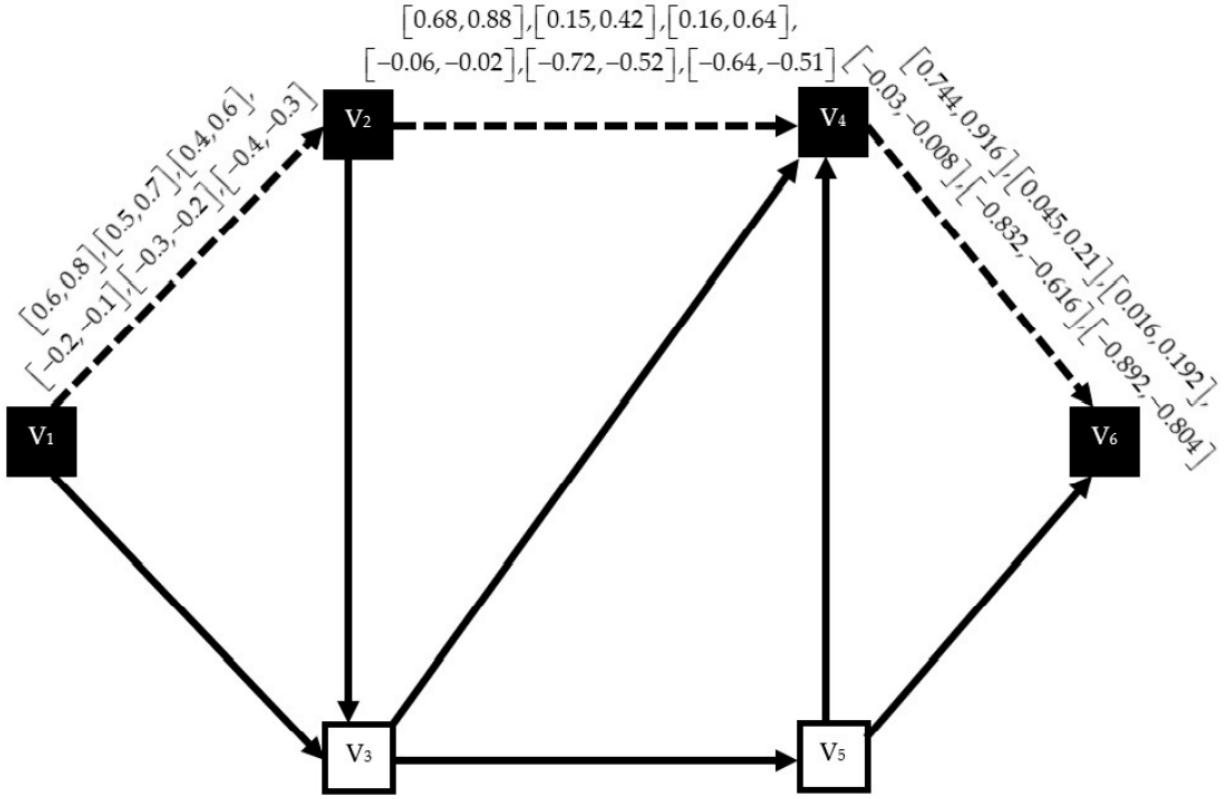

Figure 3: Suggested shortest path using IVBN-SPP Dijkstra's algorithm

6 CONCLUSION

In this paper, we proposed an algorithm for finding shortest path and shortest path length from source node to destination node on a network where the edges weights are assigned by interval valued bipolar neutrosophic number. The procedure of finding shortest path has been well explained and suitably discussed. Furthermore, the implementation of the proposed algorithm is successfully illustrated with the help of an example. The algorithm is easy to understand and can be used for all types of shortest path problems with arc length as triangular neutrosophic, trapezoidal neutrosophic and interval neutrosophic numbers. For future study, the proposed method can be extended using different set theory such as neutrosophic vague sets and Pythagorean neutrosophic set. The findings under different set theory can be compared with proposed method. Besides that, the proposed algorithm could be implemented to the real-time scenarios in supply chain and logistics management in the field of operation research.

REFERENCES

[1] S. Okada, and M. Gen, "Fuzzy shortest path problem", *Computers & Industrial Engineering*, vol 27, pp. 465-468, 1994.

[2] F. Smarandache, "A unifying field in logics: Neutrosophic Logic", *Philosophy: American Research Press*, pp. 1–141, 1999.

[3] I. Deli, M. Ali, and F. Smarandache, "Bipolar neutrosophic sets and their application based on multi-criteria decision making problems", *2015 International conference on advanced mechatronic systems (ICAMechS)*, pp. 249-254, 2015.

[4] P. Bosc, and O. Pivert, "On a fuzzy bipolar relational algebra", *Information Sciences*, vol 219, pp. 1-16, 2013.

[5] L. A. Zadeh, "Fuzzy sets", *Inf. Control,* vol 8*,* pp. 338–353, 1965.

[6] T. M. Broumi, A. Bakali, F. Smarandache, and P. K. Kumar, "Shortest path problem on single valued neutrosophic graphs", *2017 international symposium on networks, computers and communications* (ISNCC), pp. 1–6, 2017.

[7] H. Hu and R. Sotirov, "On solving the quadratic shortest path problem", *INFORMS Journal on Computing*, vol 32, no. 2, pp. 219–233, 2020.

[8] S. Broumi, M. Talea, A. Bakali and F. Smarandache, "Applying Dijkstra algorithm for solving neutrosophic shortest path problem", *Journal of New Theory,* pp. 412–416, 2016a.

[9] S. Broumi, M. Talea, A. Bakali, and F. Smarandache, "On bipolar single valued neutrosophic graphs", *Journal of New Theory,* vol. 11, pp. 84–102, 2016b.

[10] R. Kumar, S. A. Edalatpanah, S. Jha, S. Broumi, R. Singh, and A. Dey, "A multi objective programming approach to solve integer valued neutrosophic shortest path problems," *Neutrosophic Sets and Systems*, vol. 24, pp. 134-149, 2019.

[11] S. Broumi, D. Nagarajan, A. Bakali, M. Talea, F. Smarandache, and M. Lathamaheswari, "The shortest path problem in interval valued trapezoidal and triangular neutrosophic environment," *Complex & Intelligent Systems*, vol. 5, no. 4, pp. 391-402, 2019.

[12] R. Tan, W. Zhang, and S. Chen, "Decision-making method based on grey relation analysis and trapezoidal fuzzy neutrosophic numbers under double incomplete information and its application in typhoon disaster assessment," *IEEE Access*, vol. 8, pp. 3606–3628, 2019.

[13] S. Broumi, A. Dey, M. Talea, A. Bakali, F. Smarandache, D. Nagarajan, and R. Kumar, "Shortest path problem using bellman algorithm under neutrosophic environment," *Complex & Intelligent Systems*, vol. 5, no. 4, pp. 409–416, 2019.

[14] R. Kumar, S. A. Edalatpanah, S. Jha, S. Broumi, R. Singh, and A. Dey, "A multi objective programming approach to solve integer valued neutrosophic shortest path problems," *International Journal of Engineering and Advanced Technology (IJEAT),* vol. 8, no. 3, pp. 347 – 353, 2019.

[15] M. Saad, T. Mahmood, K. Ullah, and N. Jan, "Computing shortest path in a single valued neutrosophic hesitant fuzzy network," *The Nucleus*, vol. 56, no. 3, pp. 123–130, 2020.

[16] S. S. Biswas, "Neutrosophic shortest path problem (NSPP) in a directed multigraph," *Neutrosophic set and system*, vol. 29, pp. 174-185, 2019.

[17] S. K. Prabha, S. Broumi, and F. Smarandache, "Interval Valued Neutrosophic Shortest Path Problem by A* Algorithm," *International Journal of Neutrosophic Science*, vol. 11, no. 1, pp. 53-62, 2020.

[18] A. Chakraborty, "Application of pentagonal neutrosophic number in shortest path problem," *International Journal of Neutrosophic Science*, vol. 3, no. 1, pp. 21–28, 2020.

[19] L. Yang, D. Li, and R. Tan, "Research on the shortest path solution method of interval valued neutrosophic graphs based on the ant colony algorithm," *IEEE Access*, vol. 8, pp. 88717–88728, 2020.

[20] L. Yang, D. Li, and R. Tan, "Shortest path solution of trapezoidal fuzzy neutrosophic graph based on circle-breaking algorithm," *Symmetry*, vol. 12, no. 8, p. 1360, 2020.

[21] R. Liu, "Study on single-valued neutrosophic graph with application in shortest path problem," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 4, pp. 308–313, 2020.

[22] H. Wang, F. Smarandache, Y. Q. Zhang, and R. Sunderraman, "Single valued neutrosophic sets," *Multispace and Multistructure,* vol. 4, pp. 410–413, 2010.