An implementation of the Dijkstra algorithm for fuzzy costs. (Technical Report 2018)

Lissette Valdés Valdés, Sira M. Allende, Alfonso Ariza, Gonzalo Joya

1 Dijkstra algorithm for type V fuzzy graph

In this section, we propose a modified Dijkstra algorithm applied to a type V fuzzy graph (Definition 1.1).

Definition 1.1. ***(Type V fuzzy graph)***
Also known as a crisp graph with fuzzy costs, a type V fuzzy graph has known vertices and edges, but uncertainty is present in the costs on the edges, i.e.:

This algorithm finds the shortest path between the source vertex r and any other vertex on $\tilde{G}$, but dealing with fuzzy weights defined on links. In order to introduce our fuzzy version of Dijkstra algorithm, is important to define the arithmetic operations and a ranking method for fuzzy numbers. The method that we use for the arithmetic operations of triangular fuzzy numbers is introduced in section 1.1, we define the ranking method in section 1.2, and in section 1.3 we give our proposal of a Fuzzy Dijkstra algorithm for a type V fuzzy graph.

1.1 Method for the arithmetic operations of normalized triangular fuzzy numbers

At first, we give the definition of the so-called f_L-f_R positive fuzzy number:

Definition 1.2. ***($\mathbf{f_L - f_R}$ Positive fuzzy number)***
Let $C_{(u,v)} = (a, b, c, d, \omega)$ be the positive fuzzy number representing the cost of an edge $e = \langle u, v\rangle$ with $u, v \in V$. Its membership function can be expressed as:

$$\mu_{C_{(u,v)}}(x) = \begin{cases} f_L(x) & a \leq x \leq b \\ \omega & b \leq x \leq c \\ f_R(x) & c \leq x \leq d \\ 0 & \textit{otherwise} \end{cases} \tag{1}$$

where $f_L : [a, b] \to [0, \omega]$ and $f_R : [c, d] \to [0, \omega]$ are both strictly monotonic (increasing and decreasing, respectively) and continuous functions with $\omega \in [0, 1]$.

We asume f_L and f_R to be linear functions. Thus, according to the values of ω the cost of each edge is either normalized or generalized, and according to a, b, c and d a triangular or trapezoidal fuzzy number.

- $L_{(u,v)}$ normalized triangular fuzzy number.($\omega = 1$, $b = c$)
- $L_{(u,v)}$ normalized trapezoidal fuzzy number.($\omega = 1$, $a \neq b \neq c \neq d$)
- $L_{(u,v)}$ generalized triangular fuzzy number.($\omega \neq 1$, $b = c$)
- $L_{(u,v)}$ generalized trapezoidal fuzzy number. ($\omega \neq 1$, $a \neq b \neq c \neq d$)

For the arithmetic operations of fuzzy numbers, we use the α-cut method. This method is quite simple (in comparison to the extension principle method) since the operation between fuzzy numbers is reduced to the operation between ordinary intervals in $\mathbb{R}$.

First, we state two essential properties for the operations of triangular and trapezoidal fuzzy numbers:

1. The results from addition and subtraction between triangular or trapezoidal fuzzy numbers are also a triangular o trapezoidal fuzzy number, respectively.
2. The results from multiplication or division as in maximum or minimum operations are not a triangular or trapezoidal fuzzy number.

The α-*cut* interval of a normalized triangular fuzzy number $\tilde{A}$, denoted by $\tilde{A}_\alpha$, is an ordinary crisp interval in $\mathbb{R}$ containing all the elements of $\tilde{A}$ whose membership degree is greater than or equal to the specified value $\alpha \in [0, 1]$. Let $\tilde{A} = (a, b, c)$ be a normalized triangular fuzzy number with membership function with shape as in expression 1, where $f_L(x) = \dfrac{x-a}{b-a}$ and $f_R(x) = \dfrac{c-x}{c-b}$, $\tilde{A}_\alpha$ is defined as in equation 2.

$$\boxed{\tilde{A}_\alpha = [(b-a)\alpha + a, -(c-b)\alpha + c] \qquad \alpha \in [0, 1]} \tag{2}$$

The following two properties of fuzzy numbers are based on the representation of a fuzzy set and the definition of the α-cut of a fuzzy set,

(a) Each fuzzy set, and thus also each fuzzy number, can fully and univocally be represented by its α-cut.

(b) α-cuts of each fuzzy number are closed intervals of real numbers for all $\alpha \in [0, 1]$

Definition 1.3 provides the necessary elements to define the α-cut of the fuzzy number $(\tilde{A} * \tilde{B})$ by the α-cuts of the fuzzy numbers $\tilde{A}$ and $\tilde{B}$. Thus, we can use the operation of standard crisp intervals to find $(\tilde{A} * \tilde{B})_\alpha$.

Definition 1.3.
Let $\tilde{A}$ and $\tilde{B}$ denote fuzzy numbers and let $*$ denote any of the four basic arithmetic operations "$+, -, \cdot, \div$". Then for any $\alpha \in (0, 1]$, the fuzzy set on $\mathbb{R}$, $(\tilde{A} * \tilde{B})$, is defined through its α-cut $(\tilde{A} * \tilde{B})_\alpha$:

$$\boxed{(\tilde{A} * \tilde{B})_\alpha = \tilde{A}_\alpha * \tilde{B}_\alpha}$$

When $*$ = "$\div$", clearly, $0 \notin \tilde{B}_\alpha \; \forall \alpha \in (0, 1]$.

According to First Decomposition Theorem, the fuzzy number $(\tilde{A} * \tilde{B})$ is interpreted as the union of special fuzzy numbers ${}_{\alpha}(\tilde{A} * \tilde{B})$, $\forall \alpha \in [0,1]$, i.e.,

$$\boxed{(\tilde{A} * \tilde{B}) = \bigcup_{\alpha \in (0,1]} {}_{\alpha}(\tilde{A} * \tilde{B})}$$

where $\mu_{{}_{\alpha}(\tilde{A}*\tilde{B})} = \alpha \cdot \chi_{(\tilde{A}*\tilde{B})_{\alpha}}$

The membership function of $(\tilde{A} * \tilde{B})$ is computed using the union and decomposition of fuzzy sets, i.e.,

$$\boxed{\mu_{(\tilde{A}*\tilde{B})}(x) = \max_{\alpha \in (0,1]} \left\{ \mu_{{}_{\alpha}(\tilde{A}*\tilde{B})} \right\} = \max_{\alpha \in (0,1]} \left\{ \alpha \cdot \chi_{(\tilde{A}*\tilde{B})_{\alpha}}(x) \right\}}$$

where $\chi_{(\tilde{A}*\tilde{B})_{\alpha}}(x)$ is the charactaristic function of $(\tilde{A} * \tilde{B})_{\alpha}$.

As summary, based on the properties and arithmetic analysis stated above, we can perform the addition and subtraction of normalized triangular fuzzy numbers by using their α-cuts. We obtain in each case the expressions 3-4,

$$\text{Addition: } \tilde{A} + \tilde{B} = (a_1, a_2, a_3) \oplus (b_1, b_2, b_3) = (a_1 + b_1, a_2 + b_2, a_3 + b_3) \tag{3}$$

$$\text{Subtraction: } \tilde{A} - \tilde{B} = (a_1, a_2, a_3) \ominus (b_1, b_2, b_3) = (a_1 - b_1, a_2 - b_2, a_3 - b_3) \tag{4}$$

1.2 Ranking method of triangular fuzzy numbers

For the comparison of the fuzzy costs, we use a ranking criterion proposed by [1]. This method compares the Total Integral of fuzzy numbers depending on a parameter α, also called index of optimism. This index represents the degree of optimism of the decision maker. Let $\tilde{A} = (a, b, c, d)$ be a trapezoidal fuzzy number, for α taking values greater than 0.5 the comparison is made giving considering priority to numbers greater than the central value. The opposite occurs when α is lower than 0.5.

Let $C_{(u,v)} = (a, b, c, d, \omega)$ be a generalized trapezoidal fuzzy number representing the fuzzy cost in a link joining the vertices u and v. For a fixed α, the *Total Integral of* $C_{(u,v)}$ is defined as:

$$S_T^{\alpha}(C_{(u,v)}) = \frac{\omega}{2} \left[(1 - \alpha)(a + b) + \alpha(c + d) - 2X_{\min} \right] \tag{5}$$

where $X_{\min} \leq a$. For simplicity, since every cost in $\tilde{G}$ is assumed to be a positive fuzzy number in our study, we set $X_{\min} = 0$ without affecting the process performance.

By making the proper modifications on a, b, c and d, we can obtain the total integral for a normalized and generalized triangular fuzzy number as for normalized or generalized trapezoidal fuzzy number.

We need to establish a ranking for the fuzzy costs used in our algorithm. To obtain this ranking, we must define an *order relation* on the set of fuzzy numbers. Let C^1 and C^2 be two fuzzy weights (defined either on an edge or a path), for $\alpha \in [0,1]$, each C^i, $i = 1, 2$ has total integral $S_T^{\alpha}(C^i)$, then,

- If $S_T^\alpha(C^1) > (<)\, S_T^\alpha(C^2)$ then C^1 is greater (smaller) than C^2, denoted as $C^1 \succ (\prec)\, C^2$
- If $S_T^\alpha(C^1) = S_T^\alpha(C^2)$ and Me$(C^1) > (<)$ Me(C^2) then $C^1 \succ (\prec)\, C^2$
- If $S_T^\alpha(C^1) = S_T^\alpha(C^2)$ and Me$(C^1) =$ Me(C^2) then $C^1 = C^2$

where Me(C^i) $(i = 1, 2)$ denotes the median of the fuzzy number C^i.

This relation, indeed, is an order relation because it meets the antisymmetry, reflexivity and transitivity properties. On the other hand, since the Total Integral satisfies the linearity property, we can apply this order relation for both the comparison between the cost of two links and the cost of two paths by means of the comparison between their Total Integrals. Therefore, we can apply the classic Dijkstra Algorithm.

1.3 Algorithm

Let $\Psi(r, v)$ denote the set of all paths between r and v ($v, r \in V$ with $v \neq r$), and $P^* \in \Psi(r, v)$ be the shortest path between r (source) and v. The cost of P^* is defined as:

$$\widetilde{\delta}_{(r,v)} = C_{P^*(r,v)} = \begin{cases} \min\limits_{\forall P \in \Psi(r,v)} \{C_{(P(r,v))}\} & \text{if } \Psi(r, v) \neq \emptyset \\ \infty & \text{if } \Psi(r, v) = \emptyset \end{cases}$$

Then, for each vertex $v \in V$, the algorithm defines an attribute $\widetilde{d}$, which is an upper bound on the weight of the shortest path between the source r and v,

$$\widetilde{d}(v) \geq \widetilde{\delta}_{(r,v)}$$

Given a vertex $v \in V$, we will denote by $\Gamma(v)$ the set of neighbors of v, that is, adjacent vertices to v. Moreover, we will denote by $w(v)$ the predecessor of v in P^*. The algorithm considers $w(v)$ to be either a vertex of $\Gamma(v)$ or NIL. Actually, during the execution of the algorithm, $w(v)$ is the predecessor of v in the shortest path to v known so far. Only when the algorithm has ended, we can say that $w(v)$ is the predecessor of v by the shortest path from r to v.

Every vertex has got assigned a *label* which adapts throughout the algorithm execution. At each stage, the label of vertex $v \in V$ contains its predecessor in the (known so far) shortest path from r to v, and the corresponding cost of this path, i.e.,

$$\text{Label}(v) = \left[\widetilde{d}(v), w(v)\right] \quad \text{with} \quad \begin{array}{l} \widetilde{d}(v) = \widetilde{d}(w(v)) \oplus C_{(w(v),v)} \\ w(v) : \text{ (predecessor vertex of } v \text{ in the provisional path } P^*(r, v)) \end{array}$$

Where $C_{(w(v),v)}$ is the cost of link $(w(v), v)$. At the end of the algorithm, for each vertex v, $\widetilde{d}(v)$ will coincide with $\widetilde{\delta}_{(r,v)}$ and $w(v)$ will be its predecessor in $P^*(r, v)$.

The Initialization (algorithm 1) assigns a label to every vertex in $\tilde{G}$. The cost of the shortest path from r to any other vertex in $v \in \tilde{G}$ is inicialized as ∞, except for r, which is set to be equal to 0.

Algorithm 1 Initialization

function INITIALIZE-SINGLE-SOURCE($\tilde{G}, r, \mathfrak{L}$)
    $\widetilde{d}(r) \leftarrow (0, 0, 0)$
    $w(r) \leftarrow$ NIL
    for each vertex $v \in V - \{r\}$ **do**
        $\widetilde{d}(v) \leftarrow \infty$
    end for
end function

The algorithm advances by selecting and extracting vertices from a set, denoted as H, which is initially defined as V. At each iteration, the vertex u with minimum cost is selected in H according to the ranking method described in section 1.2. Then, the neighbors of this vertex are analyzed in algorithm 2 (Relaxation). The label of each neighbor v is updated if the path from r to v through u has a cost less than the current one.

Algorithm 2 Relaxation of v

function RELAXATION($u, v, C_{(u,v)}, \alpha$)
    $\widetilde{d}^{\text{new}}(v) := \widetilde{d}(u) \oplus C_{(u,v)}$
    if $S_T^{\alpha}(\widetilde{d}(v)) > S_T^{\alpha}(\widetilde{d}^{\text{new}}(v))$ **then**
        $\widetilde{d}(v) \leftarrow \widetilde{d}^{\text{new}}(v)$
        $w(v) \leftarrow u$
    end if
end function

The algorithm ends when there are no vertices left in H. At this point, the label of each vertex v contains $\widetilde{\delta}_{(r,v)}$ together with its predecessor $w(v)$ on $P^*(r, v)$. Following a backward procedure, the shortest path between r and each vertex can be found. Algorithm 3 shows the pseudocode of the Dijkstra algorithm proposed.

Algorithm 3 Dijkstra($\tilde{G}, r, \mathfrak{L}$)

Initialize-single-source($\tilde{G}, r, \mathfrak{L}$) ▷ Initialization
$H \leftarrow V$
while $H \neq \emptyset$ **do**
    $u \leftarrow t | \widetilde{d}(t) = \min_{\forall x \in H} \left\{ \widetilde{d}(x) \right\}$
    Update $H \leftarrow H - \{u\}$
    if $\Gamma(u) \cap H \neq \emptyset$ **then**
        for each vertex $v \in \Gamma(u) \cap H$ **do**
            Relaxation($u, v, C_{(u,v)}, \alpha$) ▷ Relaxation of v
        end for
    end if
end while

References

[1] Vincent F. Yu and luu Quoc Dat: An improved ranking method for fuzzy numbers with integral values. Applied Soft Computing 14., 603-608 (2014)