Shortest Paths in Fuzzy Weighted Graphs

Chris Cornelis,* Peter De Kesel,† Etienne E. Kerre‡
Department of Applied Mathematics and Computer Science,
Ghent University, Fuzziness and Uncertainty Modelling Research Unit,
Krijgslaan 281 (S9), B-9000 Ghent, Belgium

The task of finding shortest paths in weighted graphs is one of the archetypical problems encountered in the domain of combinatorial optimization and has been studied intensively over the past five decades. More recently, fuzzy weighted graphs, along with generalizations of algorithms for finding optimal paths within them, have emerged as an adequate modeling tool for prohibitively complex and/or inherently imprecise systems. We review and formalize these algorithms, paying special attention to the ranking methods used for path comparison. We show which criteria must be met for algorithm correctness and present an efficient method, based on defuzzification of fuzzy weights, for finding optimal paths. © 2004 Wiley Periodicals, Inc.

1. INTRODUCTION

In many transportation, routing, communications, economical, and other applications, graphs emerge naturally as a mathematical model of the observed real-world system. Indeed, many problems can be reformulated as a quest for a path (between two nodes in a graph) which is optimal in the sense of a number of preset criteria. Very often, these optimality criteria are evaluated in terms of weights, that is, vectors of real numbers, associated with the links of the graph. Numerous algorithms (for a comprehensive introduction, see, e.g., Ref. 1) have been developed to ease this and related quests.

In practice, due to the sheer number of optimization criteria they impose, realistic environments are often so complex that even the most sophisticated implementations become computationally unmanageable. For instance, in the context of computer networking, Arnold et al.[2] report that "in all common shortest path routing strategies only one parameter is used as routing information" (while many more are available, and relevant). To circumvent this limitation, they designed a fuzzy-rule-based system as a flexible way to evaluate and aggregate several parameters of a network link into a single, crisp link weight. In Ref. 3, Aboelela and Douligeris face a similar problem in path optimization for B-ISDN networks

*Author to whom all correspondence should be addressed: e-mail: chris.cornelis@UGent.be.
†e-mail: Peter.De.Kesel@ibsbe.be.
‡e-mail: etienne.kerre@UGent.be.

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 19, 1051–1068 (2004)
 • DOI 10.1002/int.20036

according to various quality of service (QoS) requirements. To obtain good compromise solutions with comparatively little computational effort, they used a fuzzy rule base to evaluate the quality of a number of previously found candidate paths.

The above solutions are both heuristic in a sense that they resort to fuzzy techniques as a trade-off between efficiency and precision: They sacrifice the rigor of the original optimization problem in favor of the flexibility and tolerance for imprecision provided by fuzzy set theory to obtain decent, workable solutions. Nothing prevents us, however, from going one step further and reformulating the shortest path problem itself as a fuzzy optimization problem: Indeed, by allowing that the crisp link weights expressing optimization criteria are replaced by fuzzy numbers (fuzzy weights), and by using the operations of fuzzy arithmetic (addition, multiplication, comparison, etc.), our problem becomes one of finding a fuzzy shortest path. This option has been considered early on and was first formalized by Dubois and Prade[4,5]; it has since been taken up repeatedly in the literature (see, e.g., Refs. 6–9).

In this article, from a formal viewpoint we study the correctness and the efficiency of the fuzzy (multicriteria) shortest path problem. In Section 2, we first recall some basic notions about fuzzy numbers and fuzzy weighted graphs. Section 3 is dedicated to fuzzy ranking methods, which play a key role in our further exposition. Next, Section 4 spells out the details of two generic *labeling* algorithms (inspired by the work of Martins et al.[10]) for solving the fuzzy shortest path problem, and illustrates them by a simple numerical example. In Section 5, we highlight the importance of the ranking method used to compare (fuzzy objective values of) paths, introducing and proving the conditions it should satisfy to maintain algorithm correctness. It turns out that only a limited number of the available ranking methods can be used in practice. Finally, in Section 6, we study the particular case of a fuzzy shortest path problem solved by a ranking method based on defuzzification. We prove that under certain conditions this choice of ranking method leads to a significant efficiency boost. Moreover, we show that the excessive accumulation of uncertainty in the final result that the authors of Ref. 2 considered an argument against fuzzy shortest path implementations and that tempted them to dismiss the whole approach as impractical, in fact, does not inflict our proposal.

2. PRELIMINARY DEFINITIONS

2.1. Fuzzy Quantities, Fuzzy Numbers, and LR-Fuzzy Numbers

A fuzzy quantity is defined as a fuzzy set in the real line $\mathbb{R}$, that is, an $\mathbb{R} \to [0,1]$ mapping A. If A is upper semicontinuous,[a] convex,[b] normal,[c] and has bounded support,[d] then it is called a fuzzy number.

[a] A is upper semicontinuous $\Leftrightarrow (\forall a \in \mathbb{R})(\forall \epsilon > 0)(\exists \delta > 0)(\forall x \in \mathbb{R})(|x - a| < \delta \Rightarrow A(x) < A(a) + \epsilon)$.

[b] A is convex $\Leftrightarrow (\forall x_1, x_2 \in \mathbb{R})(\forall \lambda \in [0,1])(A(\lambda x_1 + (1 - \lambda)x_2) \geq \min(A(x_1), A(x_2)))$

[c] The kernel of A, $Ker(A)$ is defined by $Ker(A) = \{x \in \mathbb{R} | A(x) = 1\}$. A is normal iff its kernel is nonempty.

[d] The support of A, $Supp(A)$, is defined by $Supp(A) = \{x \in \mathbb{R} | A(x) > 0\}$.

If A and B are fuzzy quantities, their sum $A \oplus B$ and product $A \otimes B$ are defined by Zadeh's extension principle. For $z \in \mathbb{R}$,

$$(A \oplus B)(z) = \sup_{x \in \mathbb{R}} \min(A(x), B(z - x))$$

$$(A \otimes B)(z) = \sup_{x \in \mathbb{R} \setminus \{0\}} \min\left(A(x), B\left(\frac{z}{x}\right)\right)$$

The c-level set A_c of a fuzzy quantity A is the subset of the reals whose membership value to A is at least $c \in]0,1]$: $A_c = \{x \in \mathbb{R} | A(x) \geq c\}$.

In practice, it is common to make some a priori assumptions about the shape of the fuzzy numbers that will be used in an application. To this end, it is instructive to consider LR-fuzzy numbers if L and R are two decreasing $[0,+\infty[\to [0,+\infty[$ mappings satisfying $L(0) = R(0) = 1$ and $L(1) = R(1) = 0$ and the real numbers m_l, m_r, α, β satisfy $m_l \leq m_r$ and $\alpha, \beta \geq 0$, then the LR-fuzzy number $(m_l, m_r, \alpha, \beta)_{LR}$ is defined by

$$(m_l, m_r, \alpha, \beta)_{LR}: \mathbb{R} \to [0,1]$$

$$x \mapsto \begin{cases} L\left(\dfrac{m_l - x}{\alpha}\right) & x \leq m_l, \alpha > 0 \\ 1 & x \in [m_l, m_r] \\ R\left(\dfrac{x - m_r}{\beta}\right) & x \geq m_r, \beta > 0 \\ 0 & \text{otherwise} \end{cases}$$

α and β are called the left and right spreads of $(m_l, m_r, \alpha, \beta)_{LR}$, respectively.

When L is defined by

$$L: [0,+\infty[\to [0,+\infty[$$

$$x \mapsto \begin{cases} 1 - x, & x \in [0,1] \\ 0, & x \in]1,+\infty[\end{cases}$$

$(m_l, m_r, \alpha, \beta)_{LL}$ is called a trapezoidal fuzzy number. As a shorthand notation, we write $(m_l, m_r, \alpha, \beta)$.

It can be verified that $(l_a, r_a, \alpha_a, \beta_a)_{LR} \oplus (l_b, r_b, \alpha_b, \beta_b)_{LR} = (l_a + l_b, r_a + r_b, \alpha_a + \alpha_b, \beta_a + \beta_b)_{LR}$. In general, the product of two LR-fuzzy numbers is not an internal operation. This is illustrated in Figure 1 for two trapezoidal fuzzy numbers. Finally, it can be verified that if L and R are strictly decreasing continuous mappings, the c-level set of $A = (m_l, m_r, \alpha, \beta)_{LR}$ is equal to the interval $[m_l - \alpha L^{-1}(c), m_r + \beta R^{-1}(c)]$.

2.2. Fuzzy Weighted Graphs

Due to their transparent semantics in terms of set and relations, graphs have proven rewarding candidates for generalization to a fuzzy framework, and many

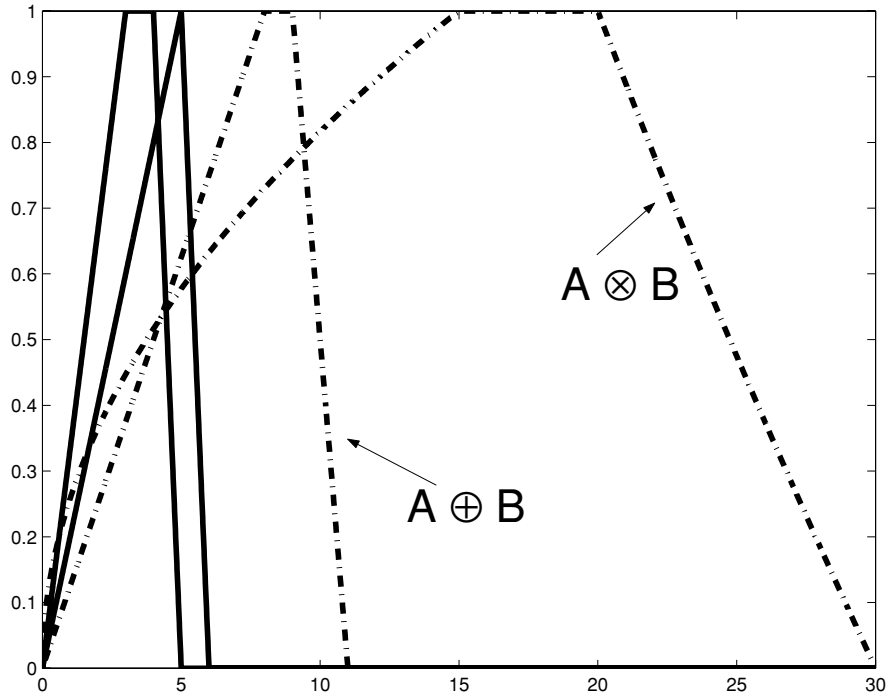

Figure 1. Sum and product of two trapezoidal fuzzy numbers. $A = (3,4,3,1), B = (5,5,5,1)$. $A \oplus B$ is trapezoidal, whereas $A \otimes B$ is obviously not.

variations on the theme of a fuzzy graph exist; see, for example, Ref. 11. In this article, we consider *fuzzy weighted graphs*, that is, vertices (or nodes) and edges (or links) remain crisp, but the edge weights will be fuzzy numbers, as in Ref. 8.

A fuzzy weighted graph $\tilde{G} = (V, E, \tilde{c})$ consists of a set V of vertices or nodes v_i and a binary relation E of edges $e_k = (v_i, v_j) \in V \times V$; we denote $tail(e_k) = v_i$ and $head(e_k) = v_j$. v_i is sometimes called a parent of v_j, whereas v_j is a child of v_i. With each edge (v_i, v_j), a weight or cost $\tilde{c}_{i,j} = \tilde{c}(v_i, v_j) = (\tilde{c}(v_i, v_j)^1, \ldots, \tilde{c}(v_i, v_j)^r)$, a vector of fuzzy numbers with $r \geq 1$, is associated. Each fuzzy number can be seen as the evaluation of a given criterion. For simplicity, and without any loss of generality, in this article we will focus on the case $r = 1$.

An s, t *path* p in $\tilde{G} = (V, E, \tilde{c})$ is an n tuple $p = (e_1, e_2, \ldots, e_n) \in E^n$ such that $head(e_i) = tail(e_{i+1})$ for $i = 1, \ldots, n-1$ and $tail(e_1) = s, head(e_n) = t$. $P(s, t)$ is the set of all s, t paths. An s, t path p is called a cycle if $s = t$; $\tilde{G}$ is called acyclic if it does not contain any cycles. A special path denoted $\varnothing$ is called the empty path; it does not contain any edges. Suppose we have two paths in a graph $\tilde{G}$: $p = (v_1, v_2, \ldots, v_{n_p})$ and $q = (v_{n_p}, v_{n_p+1}, \ldots, v_n)$. We define $p \diamond q = (v_1, v_2, \ldots, v_n)$ and call it the concatenation of p and q in $\tilde{G}$.

To make the notion of "fuzzy shortest path" precise, we need an *objective function* $\tilde{f}$ that associates an objective value (a fuzzy number) with each path

$p \in P(s,t)$; it is assumed that $\tilde{f}(\varnothing) = 0$ and $\tilde{f}(e_i) = \tilde{c}(e_i)$. The objective function determines the *cost criterion*. For instance, if the objective function is such that $\tilde{f}(p \diamond q) = \tilde{f}(p) \oplus \tilde{f}(q)$ for any two paths p and q in $P(s,t)$, we say that we work with the additive cost criterion. In case of the multiplicative criterion, we have: $\tilde{f}(p \diamond q) = \tilde{f}(p) \otimes \tilde{f}(q)$. Other criteria (see, e.g., Ref. 1) are not considered in this article.

For our purposes, paths will be compared based on their objective values; therefore, we need a method $\prec_M$ to rank fuzzy numbers. Then, $\tilde{f}(p) \prec_M \tilde{f}(q)$ means that path p is "shorter," or more optimal than q. We also say that *p dominates q*; for this reason, $\prec_M$ is sometimes also called a dominance checking method. More concisely, we write $p \prec_{D_M} q$. Note that such a ranking method will not necessarily be total, that is, $\tilde{f}(p) \not\prec_M \tilde{f}(q)$ and $\tilde{f}(q) \not\prec_M \tilde{f}(p)$ may occur for distinct p and q; in this case p and q are called *nondominated alternatives*.

A *fuzzy shortest-path algorithm* finds those s,t paths that are not dominated by other ones; so, in general, the solution will not be a single path, but rather a set of nondominated (also called Pareto optimal) alternatives; this situation occurs also in crisp multicriteria shortest path problems. As the next subsection points out, many ranking methods are available. As we will demonstrate, only a few of them are suitable for use in the fuzzy shortest-path algorithm.

3. FUZZY RANKING METHODS

As (vectors of) numbers are often used in real-life problems to describe the performance of alternatives with respect to given criteria, it is important to be able to rank or compare those numerical evaluations. In the set $\mathbb{R}$ of real numbers, this comparison is, of course, straightforward. More generally, a partially ordered set (poset) is a structure $(P, \leq_P)$ consisting of a nonempty set P and a binary relation $\leq_P \subseteq P^2$ such that

P.1	$(\forall x \in P)(x \leq_P x)$	(reflexivity)
P.2	$(\forall x, y \in P)(x \leq_P y \text{ and } y \leq_P x \Rightarrow x = y)$	(antisymmetry)
P.3	$(\forall x, y, z \in P)(x \leq_P y \text{ and } y \leq_P z \Rightarrow x \leq_P z)$	(transitivity)

The partial order $\leq_P$ may or may not be total, that is, $(\forall x, y \in P)(x \leq_P y$ or $y \leq_P x)$ may or may not hold; to any partial order $\leq_P$, a corresponding *strict* order $<_P$ can be associated: $x <_P y \Leftrightarrow x \leq_P y$ and $x \neq y$. Given a subset $A \subseteq P$, a minimal element of A in $(P, \leq_P)$ is $a \in A$ such that $\neg(\exists b \in A)(b <_P a)$.

As an example, consider the set $\mathbb{R}^k$ of real vectors of length k. One of the ways to equip $\mathbb{R}^k$ with a partial order is to define $(p_1,\ldots,p_k) \leq_{\mathbb{R}^k} (q_1,\ldots,q_k) \Leftrightarrow (\forall i \in \{1,\ldots,k\})(p_i \leq q_i)$. A total order on $\mathbb{R}^k$ is $\leq_L$, the lexicographical order, defined by $(p_1,\ldots,p_k) \leq_L (q_1,\ldots,q_k) \Leftrightarrow (p_1,\ldots,p_k) = (q_1,\ldots,q_k)$ or $(\exists i \in \{1,\ldots,k\})(p_i < q_i$ and $(\forall j < i)(p_j = q_j))$. The lexicographical order $\leq_L$ can be generalized to $\leq_{L_\sigma}$ using a permutation σ of $\{1,\ldots,k\}$, defined by $(p_1,\ldots,p_k) \leq_{L_\sigma} (q_1,\ldots,q_k) \Leftrightarrow (p_1,\ldots,p_k) = (q_1,\ldots,q_k)$ or $(\exists i \in \{1,\ldots,k\})(p_{\sigma(i)} < q_{\sigma(i)}$ and $(\forall j < i)(p_{\sigma(j)} = q_{\sigma(j)}))$. Strict versions of all these orders are readily obtained. The following lemma will be useful later on.

LEMMA 1. *If* $(p_1,\ldots,p_k)$ *is a minimal element of* $A \subseteq \mathbb{R}^k$ *in* $(\mathbb{R}^k, \leq_{L_\sigma})$, *then it is also a minimal element of* A *in* $(\mathbb{R}^k, <_{\mathbb{R}^k})$.

In the following two paragraphs, we review some of the solutions proposed in the literature to the ranking of (LR-)fuzzy numbers. Basically, two approaches exist: The first one is faithful to the poset framework (and generates a nontotal partial order), whereas the second one compares fuzzy numbers indirectly using the total order in $\mathbb{R}$; the corresponding ranking methods are not partial orders, however.

3.1. A Partial Order for Fuzzy Numbers

Just like addition and multiplication, the minimum is a binary operation on $\mathbb{R}$ and hence can be generalized to fuzzy quantities using the extension principle. For fuzzy quantities A and B, $\widetilde{\min}(A,B)$ is defined by, for $z \in \mathbb{R}$,

$$\widetilde{\min}(A,B)(z) = \sup_{z=\min(x,y)} \min(A(x),B(y))$$

Okada and Soper[8] used this "fuzzy minimum" to construct a relation $\preccurlyeq_O$, defined for fuzzy numbers A and B by $A \preccurlyeq_O B \Leftrightarrow \widetilde{\min}(A,B) = A$. Ramik and Rimanek[9] proved that $\preccurlyeq_O$ is a partial order on the set of fuzzy numbers, and also that $A \preccurlyeq_O B \Leftrightarrow (\forall c \in]0,1])(\inf A_c \leq \inf B_c$ and $\sup A_c \leq \sup B_c)$. Moreover, if $A = (l_a, r_a, \alpha_a, \beta_a)_{LR}$ and $B = (l_b, r_b, \alpha_b, \beta_b)_{LR}$, they proved that $A \preccurlyeq_O B \Leftrightarrow (l_a, r_a, l_a - \alpha_a, r_a + \beta_a) \leq_{\mathbb{R}^4} (l_b, r_b, l_b - \alpha_b, r_b + \beta_b)$, which allows for easy comparison. Because $\leq_{\mathbb{R}^4}$ is nontotal, so is $\preccurlyeq_O$. A total order $\preccurlyeq_L$ for *LR*-fuzzy numbers can be obtained by replacing $\leq_{\mathbb{R}^4}$ with $\leq_{L_\sigma}$ in the previous definition.

3.2. Ranking Methods for Fuzzy Numbers That Are Not Partial Orders

One drawback of $\preccurlyeq_O$ and its strict version $\prec_O$ defined in the previous section is that, in practice, it generates a lot of incomparabilities, that is, situations in which neither $A \prec_O B$ nor $B \prec_O A$ holds. This can be undesirable, for example, when this ranking method is used to compare paths in a fuzzy weighted graph, and the decision maker is faced with an excessive number of nondominated alternatives to choose from. The phenomenon is confirmed by the experimental results in Ref. 8, which point out that the problem becomes more critical when there are many links with very uncertain weights (i.e., LR-fuzzy numbers with wide left and right spreads). To ease the problem,[e] one may resort to the so-called "degree of optimism" introduced by Tanaka et al.[12]: Given a number $h \in]0,1]$, define $A \preccurlyeq_h B \Leftrightarrow (\forall \alpha \in [h,1])(\inf A_\alpha \leq \inf B_\alpha$ and $\sup A_\alpha \leq \sup B_\alpha)$. For LR-fuzzy numbers with continuous L and R, this yields[9] $(l_a, r_a, \alpha_a, \beta_a)_{LR} \preccurlyeq_h (l_b, r_b, \alpha_b, \beta_b)_{LR} \Leftrightarrow (l_a, r_a, l_a - \alpha_a L^{-1}(h),\ r_a + \beta_a R^{-1}(h)) \leq_{\mathbb{R}^4} (l_b, r_b, l_b - \alpha_b L^{-1}(h),\ r_b + \beta_b R^{-1}(h))$. Although this threshold in effect limits the number of

[e]Of course, one might also consider using $\preccurlyeq_{L_\sigma}$, which by definition is total; however, this method might be debatable because it forces the decision maker to determine, by a strict order, the importance of the characteristics (specifically, left and right spread and left and right kernel values) that are evaluated in comparing *LR*-fuzzy numbers.

incomparable instances, the partial order structure is lost (antisymmetry P.2 does not hold any longer).

Now, when one wants to avoid incomparabilities, another reasonable option is to map fuzzy numbers to the real line (e.g., by computing a defuzzification or utility value) and compare them using the total order $\leq$ in $\mathbb{R}$. Some caution is necessary here: Incomparabilities can still occur in this setting because *different* A and B can be mapped to the same real value! In other words, the antisymmetry condition P.2 is violated, and hence the resulting ranking method is not a partial order. A wide array of ranking methods based on this principle are available. As a simple illustration of this approach, we consider four concrete instances. For a fuzzy number A, define

$$Y_1(A) = \frac{\int_{-\infty}^{+\infty} xA(x)\,dx}{\int_{-\infty}^{+\infty} A(x)\,dx} \tag{1}$$

$$Y_2(A) = \int_0^1 M(A_c)\,dc \tag{2}$$

$$CM^{\lambda}(A) = \int_0^1 (\lambda a_c^+ + (1-\lambda)a_c^-)\,dc \tag{3}$$

$$AD^{\mu}(A) = a_{\mu}^+ \tag{4}$$

where $M(A_c)$ denotes the middle value of (the interval) A_c, $a_c^- = \inf A_c$, $a_c^+ = \sup A_c$, $\lambda \in [0,1]$ is a optimism–pessimism parameter that a decision maker is free to determine, and $\mu \in]0,1]$. Equations 1 and 2 are adapted from Yager's work[13]; adepts of fuzzy-rule-based systems will recognize in Equation 1 the very popular center of gravity or centroid defuzzification method (see, e.g., Ref. 14). Formula 3 is a generalization of Equation 2 and is due to Campos and Muñoz.[15] Finally, Adamo[16] introduced formula 4 that simply characterizes A by the rightmost point of its μ cut. We can now define $A \preccurlyeq_{\mathcal{D}} B \Leftrightarrow \mathcal{D}(A) \leq \mathcal{D}(B)$, where $\mathcal{D} \in \{Y_1, Y_2, CM^{\lambda}, AD^{\mu}\}$. When $\mathcal{D}(A) = \mathcal{D}(B)$, we write $A \sim_{\mathcal{D}} B$, that is, A and B are nondominated alternatives.

In Ref. 17, Wang and Kerre gather known ranking methods based on this principle, call them class I ranking methods, and examine their properties. They furthermore distinguish two more general classes of ranking methods: In class II, a reference set is constructed and all the fuzzy numbers to be ranked are compared with the reference set, whereas in class III a binary fuzzy relation is used to make pairwise comparisons of fuzzy numbers. What sets these methods apart from class I members is that the ranking of fuzzy numbers A and B according to the former is always considered with respect to a set $\mathcal{A}$ of alternatives such that $A, B \subseteq \mathcal{A}$. In particular, for a given method (say, $\prec_M$), it may well occur that $A \prec_M B$ on $\{A, B\}$, but $B \prec_M A$ on $\{A, B, C\}$. Because in the fuzzy shortest path algorithm we often need to compare candidate paths to a node t at a given step without knowing

at that time about all the paths to t, such $\prec_M$ is a priori unfit for our purposes: The ranking of A and B should depend only on A and B, and not on any other fuzzy number C. Because none of the class II methods considered in Ref. 17 satisfies this essential independency criterion, we shall not be concerned with them here. The study of class III methods is more rewarding, so we review them briefly.

Class III ranking methods are derived from a binary fuzzy relation P on $\mathcal{A}$, a set of fuzzy numbers under consideration. Typically, $P(A,B)$ is interpreted as a degree to which A is greater than or equal to B. To serve as a basis for a ranking method, P must satisfy at least the condition of acyclicity,[f] that is, for arbitrary $A_1,\ldots,A_n$ in $\mathcal{A}$

$$\begin{aligned}&(P(A_1,A_2) > P(A_2,A_1),\ldots,P(A_{n-1},A_n) > P(A_n,A_{n-1}))\\ &\quad\Rightarrow P(A_1,A_n) \geq P(A_n,A_1)\end{aligned} \tag{5}$$

By way of illustration, we quote three valid candidate fuzzy relations due to Dubois and Prade.[18] For A and B fuzzy numbers, define

$$PD(A,B) = \sup_{x,\,y\in\mathbb{R},\,x\geq y} \min(A(x),B(y)) \tag{6}$$

$$ND(A,B) = \inf_{x\in\mathbb{R}} \sup_{y\in\mathbb{R},\,x\geq y} \max(1-A(x),B(y)) \tag{7}$$

$$PSD(A,B) = \sup_{x\in\mathbb{R}} \inf_{y\in\mathbb{R},\,y\geq x} \min(A(x),B(y)) \tag{8}$$

Wang[19] then derived a total fuzzy ranking method from acyclic P by defining $\mathcal{H}_1 = \{A_i \in \mathcal{A} | (\forall A_j \in \mathcal{A})(P(A_i,A_j) \geq P(A_j,A_i))\}$. If $\mathcal{B}_1 = \mathcal{A}\backslash\mathcal{H}_1 \neq \emptyset$, he further defined $\mathcal{H}_2 = \{A_i \in \mathcal{B}_1 | (\forall A_j \in \mathcal{B}_1)(P(A_i,A_j) \geq P(A_j,A_i))\}$ and repeated the process until $\mathcal{B}_m = \emptyset$. Then, $A \prec_P B \Leftrightarrow (\exists s,t \in \{1,\ldots,m\})$ ($s < t$ and $A \in \mathcal{H}_s$ and $B \in \mathcal{H}_t$), and $A \sim_P B \Leftrightarrow (\exists s \in \{1,\ldots,m\})(A \in \mathcal{H}_s$ and $B \in \mathcal{H}_s)$.

An important[g] result derived in Refs. 17 and 19 is the following: For any set of fuzzy numbers $\mathcal{A}$ such that $\{A,B\} \subseteq \mathcal{A}$, $A \prec_P B$ on $\mathcal{A} \Leftrightarrow P(A,B) > P(B,A)$ and $A \sim_P B$ on $\mathcal{A} \Leftrightarrow P(A,B) = P(B,A)$ hold as soon as P is a consistent fuzzy relation, that is, for A,B, and C in $\mathcal{A}$

$$(P(A,B) \geq P(B,A) \text{ and } P(B,C) \geq P(C,B)) \Rightarrow P(A,C) \geq P(C,A) \tag{9}$$

Consistency is a stronger requirement than acyclicity. It can be verified that *PD*, *ND*, and *PSD* are not consistent for arbitrary fuzzy numbers. However, *PD* is consistent when it is applied to fuzzy numbers whose kernel is a singleton, whereas *ND* and *PSD* are consistent if applied to fuzzy numbers with strictly monotonous membership functions on both sides of their kernels.

[f]The choice of terminology can be explained in the following way: If we draw a (crisp, unweighted) graph G with vertices $A_1,\ldots,A_n$ and an edge from A_i to A_j, if $P(A_i,A_j) > P(A_j,A_i)$, then G is acyclic iff P is acyclic.

[g]Important in the sense that it assures that the ranking of A and B depends only on A and B.

```
1   'initialize:
2   ∀v ∈ V : L(v,1) ← [∞,(−,−)]_v^1;
3   Q ← ∅;
4   L(s,1) ← [0,(−,−)]_s^1; add(Q, L(s,1));
5   'breadth-first search:
6   while (Q ≠ ∅) do {
7     x ← takeNext(Q);
8     'x contains L(v_i,k) = [f̃(p^k_{s,v_i}),(v_p,k_p)]^k_{v_i};
9     ∀v_j, v_j a child of v_i do :{
10      p^l_{s,v_j} ← p^k_{s,v_i} ◇ (v_i,v_j); (*)
11      f̃(p^l_{s,v_j}) ← f̃(p^k_{s,v_i}) ⊕ c̃_{v_i,v_j};
12      if p^l_{s,v_j} is not dominated in L(v_j) {
13        L(v_j,l) ← [f̃(p^l_{s,v_j}),(v_i,k)]^l_{v_j} ;
14        add L(v_j,l) to list L(v_j);
15        make L(v_j) consistent;
16        add(Q, L(v_j,l));
17      }
18    }
19  }
20  stop
(*) with l an empty position.  If line 15 caused a free position, that one may be reused,
   otherwise a new position is added to the list.
```

Figure 2. Label-correcting algorithm.

4. LABELING ALGORITHMS FOR FUZZY WEIGHTED GRAPHS

4.1. Label-Correcting Algorithm

This section describes the generic label-correcting algorithm (adapted from Martins et al.'s version[10] for the crisp multicriteria shortest path problem) for finding all nondominated s, v_i paths in a connected[h] weighted fuzzy graph $\tilde{G} = (V, E, \tilde{c})$, for all nodes v_i in $\tilde{G}$, where s denotes a fixed initial node of $\tilde{G}$ for all paths. We use the additive cost criterion; the multiplicative criterion is readily obtained by replacing instances of $\oplus$ by $\otimes$. We also require that every cycle c of $\tilde{G}$ has strictly positive weight.[i] This ensures that the algorithm will terminate; all nondominated paths will be finite and have a finite objective value. The pseudo code for the algorithm is shown in Figure 2.

The algorithm uses the following data structures: Q is a first-in–first-out (FIFO) queue, and the linear list $L(v_j)$ contains, at any given moment during algorithm execution, the labels of the s, v_j paths that have been found so far and that are not dominated by other examined paths. Corresponding to the kth nondominated path that is found, we store the kth label in $L(v_j)$ as follows:

$$L(v_j, k) = [\, \tilde{f}(p^k_{s,v_j}), (v_i, k_i)]^k_{v_j}$$

[h]Meaning that all nodes in the graph can be reached.

[i]A fuzzy number A is said to be strictly positive if for all $z \in]-\infty, 0]$, $A(z) = 0$.

in which v_i is the vertex preceding v_j in the path p^k_{s,v_j}, and k_i indicates that for the k_ith s,v_i path $p^{k_i}_{s,v_i}$ in $L(v_i)$ applies: $\tilde{f}(p^k_{s,v_j}) = \tilde{f}(p^{k_i}_{s,v_i}) \oplus \tilde{c}_{i,j}$. When the algorithm terminates, all nondominated paths can be reconstructed by walking reversely through the pointers in all intermediate $L(v_i)$ until the starting vertex s is reached.

The order in which the algorithm walks through the graph is not fixed, but because objective values of s,v_j paths are calculated as the sum (or product in case of a multiplicative criterion) of the objective value of an s,v_i path and $\tilde{c}(v_i,v_j)$, and because $\oplus$ and $\otimes$ are commutative and associative, this does not impact the algorithm's correctness (it may impact the speed of finding the result). Furthermore, it is not necessary to consider all s,v_i paths and select the nondominated ones afterward. The search space can be reduced due to the strong optimality principle: Every subpath of an optimal path is also optimal. In case a journey from Amsterdam to Paris through Brussels seemed better than through Rome, it is useless to consider a stop in Rome before getting to Paris if you are looking for an optimal Amsterdam–Paris–Madrid path. This pruning in the search space is implemented on lines 12 and 15. The ranking method $\preccurlyeq_M$ used in these lines must fulfill certain conditions to be sure that we get a correct result, that is, that the result really contains all nondominated paths. Those conditions are discussed in Section 5.1.

Finally, we focus on a few important lines in the algorithm:

- At algorithm initialization, every node v (except s) is assigned a dummy label $[\infty,(-,-)]^1_v$ [line 2]. For our purposes, ∞ here is just taken to denote a dummy value that is dominated by *any* fuzzy number. s receives the label $[0,(-,-)]^1_s$, where the crisp value 0 represents the objective value of the empty path from s to itself [line 4].
- If a new s,v_j path p_{new} is found, it will not be kept in list $L(v_j)$ if there is already an s,v_j path p_{old} in $L(v_j)$ that dominates p_{new} (i.e., if $p_{old} \prec_{D_M} p_{new}$) [line 12].
- If there is no s,v_j path in $L(v_j)$ that dominates p_{new} in step I, p_{new} is added to the list [line 14] in order to be able to extend this path later on our quest for optimal paths to nodes that can be reached from v_j. In step I, all previously found paths $p_{old,l}$ in $L(v_j)$ that are dominated by p_{new} (i.e., $p_{new} \prec_{D_M} p_{old,l}$) are removed [line 15].
- In line 16, the paths starting from s to all children of v_j in $\tilde{G}$ have to be reconsidered. Suppose that, in this step, for a child k of v_j in $\tilde{G}$ we have an s,k path via parent x that is not dominated by a path via v_j. Now that we have found a new optimal path towards v_j, it is possible that this new path dominates the s,k path via x. For this reason, this algorithm is called *label correcting*: If we are only interested in an optimal path from s to one particular vertex v, we cannot be sure that a constructed s,v path will not be dominated by another alternative until the entire algorithm has finished.

4.2. Label-Setting Algorithm

The efficiency of the label-correcting algorithm shown in Figure 2 can be considerably increased if $\tilde{G}$ does not have negative weights, that is, for all $(v_i,v_j) \in E$ and $z \in]-\infty,0[$, $\tilde{c}_{i,j}(z) = 0$. The idea is as follows: We replace the list Q by a *priority queue* PQ such that on line 7, the label returned by *takeNext*(PQ) refers to a nondominated s,t path p, that is, one such that for all $q \in P(s,t)$, $\tilde{f}(p) \preccurlyeq_M \tilde{f}(q)$.

We can partition the labels of a given node v_i into two classes: $L(v_i)$ contains temporary labels, whereas $\bar{L}(v_i)$ contains permanent labels that correspond to

optimal paths. At creation time, all labels are temporary; only when a label is selected from PQ does it becomes permanent. To check whether a new path is dominated, both $L(v_i)$ and $\bar{L}(v_i)$ have to be examined, but only $L(v_i)$ has to be made consistent in line 15, warranting another significant gain in time. This is called the *label-setting algorithm*, an extension of the well-known Dijkstra algorithm.

To make sure the optimization is correct, we must be able to guarantee that PQ indeed ranks its labels in such a way that the head of the queue always represents a nondominated path. For example, Okada and Soper[8] used LR-fuzzy numbers as link weights and the ranking $\prec_O$ for dominance checking. To rank labels in the priority queue, they used the total order $\preccurlyeq_{L_\sigma}$ with $\sigma(1) = 2$, $\sigma(2) = 3$, $\sigma(3) = 1$, and $\sigma(4) = 4$. By Lemma 1, a label ranked first by PQ corresponds to an optimal path. They also described a variant of the algorithm by replacing $\prec_O$ with $\prec_h$, but they neglected to point out that PQ in that case should not rely on the lexicographic order, but be directly formulated in terms of $\prec_h$. Indeed, for example, consider a PQ containing $(10,12,5,5)_{LR}$ and $(10,13,9,6)_{LR}$; then $(10,12,5,5)_{LR} \prec_1 (10,13,9,6)_{LR}$, still $(10,13,9,6)_{LR} \preccurlyeq_{L_\sigma} (10,12,5,5)_{LR}$, and the algorithm will fail by turning the dominated label $(10,13,9,6)_{LR}$ into a permanent one.

4.3. A Numerical Example

In this subsection, as an illustration we apply the label-setting algorithm to an example fuzzy weighted graph $\tilde{G} = (V, E, \tilde{c})$ shown in Figure 3 and similar to the one in Ref. 8: Starting from a designated initial node s, we search all nondominated s, v_i paths for each node $v_i \in V$. For simplicity, we use trapezoidal fuzzy numbers as weights and we compare them using the ranking method $\prec_{Y_2}$. It can be verified that $Y_2(m_l, m_r, \alpha, \beta) = \frac{1}{2}[m_l + m_r + (\beta - \alpha)/2)]$. In Table I we show the contents of the priority queue PQ, and Table II gives on overview of each node's linear list of labels.

Initialization: $L(s,1) \leftarrow [(0,0,0,0),(-,-)]^1_s$. This label is added to the empty queue PQ and is immediately removed from it by `takeNext(`PQ`)`. Label

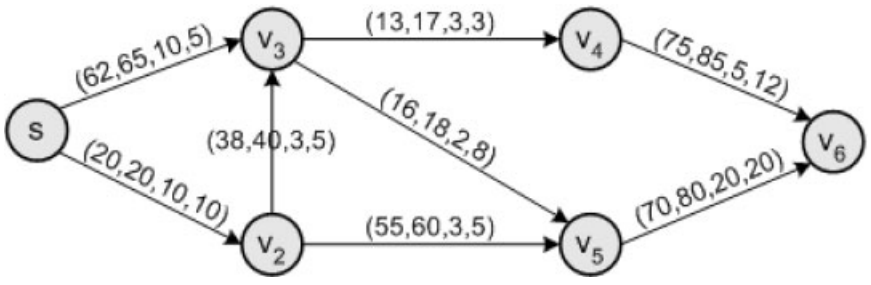

Figure 3. Fuzzy weighted graph $(V, E, \tilde{c})$.

Table I. The labels in the priority queue PQ at the beginning of each step; at each stage, the leftmost label is selected from PQ.

1	$[(0,0,0,0),(-,-)]^1_s$		
2	$[(20,20,10,10),(s,1)]^1_{v_2}$	$[(62,65,10,5),(s,1)]^1_{v_3}$	
3	$[(58,60,13,15),(v_2,1)]^1_{v_3}$	$[(75,80,13,15),(v_2,1)]^1_{v_5}$	
4	$[(71,77,16,18),(v_3,1)]^1_{v_4}$	$[(75,80,13,15),(v_2,1)]^1_{v_5}$	$[(74,78,15,23),(v_3,1)]^2_{v_5}$
5	$[(75,80,13,15),(v_3,1)]^1_{v_5}$	$[(74,78,15,23),(v_3,1)]^2_{v_5}$	$[(146,162,21,30),(v_4,1)]^1_{v_6}$
6	$[(74,78,15,23),(v_3,1)]^2_{v_5}$	$[(145,160,33,35),(v_5,1)]^1_{v_6}$	
7	$[(145,160,33,35),(v_5,1)]^1_{v_6}$	$[(144,158,35,43),(v_5,2)]^2_{v_6}$	

$[(0,0,0,0),(-,-)]^1_s$ becomes permanent and moves to $\bar{L}(s)$. All other nodes v_i are assigned a label $L(v_i,1) = [\infty,(-,-)]^1_{v_i}$ and no permanent ones.

Step 1: Nodes v_2 and v_3 are children of s; suppose that, for example, v_2 is considered first: $p^2_{s,v_2} \leftarrow p^1_{s,s} \Diamond (s,v_2)$ with $\tilde{f}(p^2_{s,v_2}) = \tilde{f}(p^1_{s,s}) \oplus \tilde{c}_{s,v_2} = (0,0,0,0) \oplus (20,20,10,10) = (20,20,10,10)$. This path dominates label $[\infty,(-,-)]^1_{v_2}$ in $L(v_2)$ that will be removed. This free position in the list is used for the new path and its corresponding label: $p^1_{s,v_2} \leftarrow p^2_{s,v_2}$; $L(v_2,1) \leftarrow [(20,20,10,10),(s,1)]^1_{v_2}$. Analogously, after considering v_3, $L(v_3)$ will contain $L(v_3,1) \leftarrow [(62,65,10,5),(s,1)]^1_{v_3}$; this label corresponds to the path $p_{s,v_3} = p^1_{s,s} \Diamond (s,v_3)$. Because $Y_2(20,20,10,10) = 20 < 62.25 = Y_2(62,65,10,5)$, $L(v_2,1)$ is ranked prior to $L(v_3,1)$ in the priority queue PQ.

Step 2: $[(20,20,10,10),(s,1)]^1_{v_2}$ is selected and becomes permanent. Nodes v_3 and v_5 are children of v_2; suppose that, for example, v_3 is considered first: $p^2_{s,v_3} \leftarrow p^1_{s,v_2} \Diamond (v_2,v_3)$ with $\tilde{f}(p^2_{s,v_3}) = \tilde{f}(p^1_{s,v_2}) \oplus \tilde{c}_{v_2,v_3} = (20,20,10,10) \oplus (38,40,3,5) = (58,60,13,15)$. Because $Y_2(\tilde{f}(p^2_{s,v_3})) = 59.5 < Y_2(\tilde{f}(p^1_{s,v_3}))$, this new path dominates label $L(v_3,1)$, which is removed from the list and from the priority queue: $L(v_3) = ([(58,60,13,15),(v_2,1)]^1_{v_3})$. Node v_5 is first reached from

Table II. The labels for each node, ordered by creation time. An underlined label is dominated and is removed at some point during algorithm execution.

v_1	v_2	v_3
$[(0,0,0,0),(-,-)]^1_s$	$[(20,20,10,10),(s,1)]^1_{v_2}$	$\underline{[(62,65,10,5),(s,1)]^1_{v_3}}$ $[(58,60,13,15),(v_2,1)]^1_{v_3}$
v_4	v_5	v_6
$[(71,77,16,18),(v_3,1)]^1_{v_4}$	$[(75,80,13,15),(v_2,1)]^1_{v_5}$ $[(74,78,15,23),(v_3,1)]^2_{v_5}$	$\underline{[(146,162,21,30),(v_4,1)]^1_{v_6}}$ $[(145,160,33,35),(v_5,1)]^1_{v_6}$ $[(144,158,35,43),(v_5,2)]^2_{v_6}$

v_2, so $L(v_5,1) = [\infty,(-,-)]^1_{v_5}$ is replaced by $L(v_5,1) = [(20,20,10,10) \oplus (55,60,3,5),(v_2,1)]^1_{v_5} = [(75,80,13,15),(v_2,1)]^1_{v_5}$. $L(v_3,1)$ and $L(v_5,1)$ are added to PQ.

Step 3: $[(58,60,13,15),(v_2,1)]^1_{v_3}$ is selected and becomes permanent. Nodes v_4 and v_5 are children of v_3. v_4 is reached for the first time, so $L(v_4,1) \leftarrow [(58 + 13, 60 + 17, 13 + 3, 15 + 3),(v_3,1)]^1_{v_4} = [(71,77,16,18),(v_3,1)]^1_{v_4}$. $p^2_{s,v_5} \leftarrow p^2_{s,v_3} \diamond (v_3,v_5)$ with $\tilde{f}(p^2_{s,v_5}) = (58,60,13,15) \oplus (16,18,2,8) = (74,78,15,23)$. Because $Y_2(\tilde{f}(p^2_{s,v_5})) = 78 = Y_2(\tilde{f}(p^1_{s,v_5}))$, the path found in step 2 and this one are nondominated alternatives. Hence $L(v_5,2) = [(74,78,15,23),(v_3,1)]^2_{v_5}$ is added to $L(v_5)$. The labels $L(v_4,1)$ and $L(v_5,2)$ are added to the priority queue.

Step 4: $[(71,77,16,18),(v_3,1)]^1_{v_4}$ becomes permanent, and its only extension, by v_6, is considered. $\tilde{f}(p^1_{s,v_6}) = (71 + 75, 77 + 85, 16 + 5, 18 + 12)$, so the initial $L(v_6,1)$ is replaced by $[(146,162,21,30),(v_4,1)]^1_{v_6}$ in the list of temporary labels and the new label is added to the priority queue.

Step 5: $[(75,80,13,15),(v_2,1)]^1_{v_5}$ becomes permanent. Only v_6 is a child of v_5, so we consider $p^2_{s,v_6} = p^1_{s,v_5} \diamond (v_5,v_6)$. Because this new path with objective value $(75 + 70, 80 + 80, 13 + 20, 15 + 20) = (145,160,33,35)$ dominates the s,v_6 path found in step 4, the latter is removed from $L(v_6)$ as well as from PQ. The new label is added to $L(v_6)$ and to the priority queue.

Step 6: The second temporary label of v_5, $[(74,78,15,23),(v_3,1)]^2_{v_5}$ is selected and becomes permanent. v_6 is a child of v_5, $\tilde{f}(p^2_{s,v_6}) = (74,78,15,23) \oplus (70,80,20,20) = (144,158,35,43)$, with $Y_2(144,158,35,43) = 153 = Y_2(145, 160,33,35)$. In other words, the two nondominated alternative s,v_5 paths cause two nondominated alternative s,v_6 paths. The newly created label for v_6 is added to the list and to PQ.

Step 7: In the priority queue, only labels of v_6 are left. Because this node has no children, the algorithm finishes after turning its labels into permanent ones.

Reconstruction of the nondominated paths: Walking reversily through the pointers in the permanent labels, one finds, for example, one optimal s,v_4 path (s,v_2,v_3,v_4) and two optimal s,v_6 paths: (s,v_2,v_5,v_6) and (s,v_2,v_3,v_5,v_6).

5. SUITABLE RANKING METHODS FOR DOMINANCE CHECKING

5.1. Necessary and Sufficient Conditions for Correctness of the Labeling Algorithms

Given a fuzzy weighted graph $\tilde{G}$, the labeling algorithms from the previous section search all optimal paths using either the additive or the multiplicative cost criterion. The ranking method $\prec_M$ will have to satisfy certain properties to come up with the correct solution. The following theorem spells them out.

THEOREM 1 (NECESSARY AND SUFFICIENT CONDITIONS). *The label-correcting algorithm is correct, that is, obtains a result that contains all nondominated paths, if and only if the following restrictions on $\prec_M$ are met*[j] *($\mathcal{A}$ is taken to be a sufficiently wide collection of possible objective values, e.g., all LR-fuzzy numbers for particular L and R):*

- **(C.1)** *$\prec_M$ is antireflexive, that is,* $(\forall A \in \mathcal{A})(\neg(A \prec_M A))$
- **(C.2)** *$\prec_M$ is transitive, that is,* $(\forall A, B, C \in \mathcal{A})(A \prec_M B \text{ and } B \prec_M C \Rightarrow A \prec_M C)$
- **(C.3)** $(\forall A, B \in \mathcal{A})(A \prec_M B$ *depends only on A and B)*
- **(C.4)** *$\prec_M$ is compatible with $\oplus$, that is,* $(\forall A, B, C \in \mathcal{A})(A \prec_M B \Rightarrow A \oplus C \prec_M B \oplus C)$
- **(C.4′)** *$\prec_M$ is compatible with $\otimes$, that is,* $(\forall A, B, C \in \mathcal{A})(C$ *is strictly positive and* $A \prec_M B \Rightarrow A \otimes C \prec_M B \otimes C)$

The label setting algorithm is correct if and only if C.1–C.4/C.4′ are met, and additionally each label returned by PQ corresponds to a nondominated path.

Proof. Necessity of C.1–C.4/C.4′ can be proved by contraposition; here we only prove C.4. Suppose that in step I, the s, t path p_I is constructed, and in a later step J the s, t path p_J is constructed. In a still later step K, nondominated paths to u, a child of t are searched. In this situation, the algorithm may yield an incorrect result if $\prec_M$ is not compatible with $\oplus$: If $\tilde{f}(p_I) \prec_M \tilde{f}(p_J)$, then p_J will not be added to the list $L(t)$. In step K, $p_I\Diamond(t,u)$ will be considered and $p_J\Diamond(t,u)$ will be omitted, although it is possible in our assumption that

$$\neg((\tilde{f}(p_I) \oplus \tilde{c}(t,u)) \prec_M (\tilde{f}(p_J) \oplus \tilde{c}(t,u)))$$

and hence,

$$p_I\Diamond(t,u) \not\prec_{D_M} p_J\Diamond(t,u)$$

so the algorithm fails by ignoring the path $p_J\Diamond(t,u)$.

It is clear that the above conditions taken together are also sufficient; note in particular that C.4, as well as C.4′, entail the strong optimality principle that justified the pruning steps in the labeling algorithms: Any subpath of a nondominated path will be itself nondominated. ■

5.2. Suitable Ranking Methods

Let us first consider Okada and Soper's implementation[8] of the label-setting algorithm. It is easy to verify that $\prec_O$ is antireflexive, transitive, and compatible with both $\oplus$ and $\otimes$. Clearly, it also satisfies C.3. This also holds for the weakened

[j] C.4 applies when using the additive cost criterion, C.4′ when using the multiplicative cost criterion.

version $\prec_h$. Hence, together with the findings in Subsection 4.2, we conclude their strategy is correct.

All the ranking methods studied in Refs. 17 and 19 satisfy at least the basic properties C.1 and C.2. C.3 is met by all class I methods and by certain class III methods, as mentioned in Subsection 3.2. Properties C.4 and C.4′ turn out to be a great deal more challenging; it is particularly striking that the intuitively appealing and widespread $\prec_{Y_1}$ is incompatible with both $\oplus$ and $\otimes$, and so cannot be used in any implementation. On the other hand, $\prec_{Y_2}$, and more generally $\prec_{CM^\lambda}$, are compatible with $\oplus$; the same is true for $\prec_{AD^\mu}$. C.4 holds for a number of class III methods as well,[19] but, surprisingly, all of them result in the same final ranking as a corresponding class I method[k] and hence they do not add anything new.

Compatibility with $\otimes$ is the privilege of an even more restricted set of ranking methods. Among its members, $\prec_{PD}$, $\prec_{ND}$, and $\prec_{PSD}$ are the most interesting ones.[l] Of all the class I methods, only Adamo's $\prec_{AD^\mu}$ is known to satisfy it.[19]

Concluding, if we want to select a suitable ranking method to implement the fuzzy shortest path algorithms, we are not exactly spoiled for choice. Also, the additive cost criterion will generally involve selecting a different method than the multiplicative cost criterion, unless one is prepared to use Adamo's method. An intuitive argument against the latter method is that in its extreme simplicity it dismisses a lot of information about the evaluated weights, and hence can be expected to generate many incomparabilities. Finally, when using one of $\prec_{PD}$, $\prec_{ND}$, or $\prec_{PSD}$ one should take care that the fuzzy weights satisfy the conditions of Section 3.2 that guarantee C.3.

6. A FAST IMPLEMENTATION

A frequently heard criticism concerning the introduction of imprecision in shortest path problems is that it renders an already complex computational procedure[m] even more intractable by its reliance on costly operations on fuzzy numbers. Let us examine this claim: Although it is true that fuzzy addition $\oplus$ is, in general, a very complex operation that requires the calculation of a supremum over an infinite set of values, in practice, certain efficiency standards can be met, because addition for *LR*-fuzzy numbers is closed and requires only four additions of reals (see Section 2.1). Fuzzy multiplication $\otimes$, admittedly, is a more problematic matter, but when the spreads are not too large, Dubois and Prade[5] suggested ways to arrive at easy-to-compute, satisfactory approximations of the exact result.

[k]Specifically, the ranking methods $\prec_{P_Y}$ (Yuan), $\prec_{K_1}$, and $\prec_{K_2}$ (Kolodziejczyk) and $\prec_{SS}$ (Saade and Schwarzlander) result in the same ranking as $\prec_{Y_2}$, whereas the method $\prec_{N_\lambda}$ (Nakamura) results in the same ranking as $\prec_{CM^\lambda}$.[19]

[l]Again, there are a number of correspondences between methods encountered in the literature that reduce the actual number of different suitable ranking methods: specifically, Wang[19] proves that $\prec_{NSD}$ (Dubois and Prade) and $\prec_{\beta_T}$ and $\prec_{\delta_T}$ (Delgado, Verdegay, and Vila) all result in the same final ranking as *PD*.

[m]It is shown in Ref. 20 that even the crisp bicriterion (i.e., involving just two criteria) shortest path problem cannot be solved in polynomial time.

What appears to harm fuzzy shortest path implementations' efficiency most, however, seems to be the comparisons that need to be made at each step. Assuming we want to address the problem of dominance checking with a certain degree of sophistication by using a fuzzy ranking method that is sufficiently discriminating between alternatives, this soon becomes a serious bottleneck as in each step, for instance, a defuzzification or an evaluation by a consistent fuzzy relation has to be performed.

The aim of this section is to show that for some specific choices of a ranking method, this problem can be overcome. Indeed, consider a class I ranking method $\prec_{DF}$, that is, one that proceeds by assessing a defuzzification value $DF(A)$ that reduces the information contained in a fuzzy number A to a crisp (real) number that is used for comparison. It is clear that we can apply defuzzification at different stages during the search process for optimal paths in $\widetilde{G} = (V, E, \tilde{c})$:

- **Plan I**: defuzzification of weights (before line 1 in Figure 2): The fuzzy weighted graph $\widetilde{G} = (V, E, \tilde{c})$ becomes a crisp graph $G = (V, E, c)$ (with $c = DF(\tilde{c})$) in which all optimal paths can be obtained by using real operations $(+, .)$ to calculate the objective values and $\prec_{\mathbb{R}}$ for dominance checking.
- **Plan II**: maintaining fuzzy weights and using defuzzification for dominance checking (in line 12 and 15): The fuzzy objective value is calculated with the extended operators $\oplus$ and $\otimes$. The dominance relation is determined by $\prec_{DF}$.

Plan I is clearly the faster option: All imprecision in the fuzzy graph disappears before the (now crisp) search algorithm starts. As an extra advantage, plan I allows specialized implementations of the search algorithm (see, e.g., Ref. 21) for crisp graphs to be used for fuzzy weighted graphs without any modification.

The following theorem shows when the result of plan II can be obtained by the faster plan I.

THEOREM 2. *Searching for optimal paths using the additive or multiplicative cost criterion, plan I and plan II are equivalent, correct implementations if the defuzzification operator DF satisfies*

$$(C.a)\ DF(A \oplus B) = DF(A) + DF(B), \forall (A, B) \in \mathcal{A}^2 \text{ (additive cost criterion)}$$
$$(C.m)\ DF(A \otimes B) = DF(A).DF(B), \forall (A, B) \in \mathcal{A}^2 \text{ (multiplicative cost criterion)}$$

where $\mathcal{A}$ denotes the class of fuzzy numbers that can occur as edge weights; in case of the multiplicative cost criterion, $\mathcal{A}$ contains no negative fuzzy numbers.

Proof. Bearing in mind Theorem 1, it can easily be proved that $\prec_{DF}$ meets C.1–C.4 if DF meets C.a and we use the additive cost criterion. We prove, for example, C.4. Because $DF(\mathcal{A}) \subseteq \mathbb{R}$ applies for the set of fuzzy objective values $\mathcal{A}$ we can prove for arbitrary A, B, and C from $\mathcal{A}$:

$$\begin{aligned} A \prec_{DF} B &\Leftrightarrow DF(A) < DF(B) \Leftrightarrow DF(A) + DF(C) < DF(B) + DF(C) \\ &\Leftrightarrow DF(A \oplus C) < DF(B \oplus C) \Leftrightarrow A \oplus C \prec_{DF} B \oplus C \end{aligned}$$

Analogously, from C.m follow C.1–C.4′ using the multiplicative cost criterion. ■

Based on the linearity of the integral, it is easily verified that Y_2 (and, more generally, CM^{λ}) verifies C.a, so plan I can be used. Remark how this optimization affects the complexity of the fuzzy shortest path problem when working with LR-fuzzy numbers. Because computing the defuzzification value can happen in constant time (see, e.g., the numerical example in Section 4.3), *the complexity of plan I is equal to that of the classical (crisp) single-criterion labeling algorithms*. We leave it to the reader to verify the gain in efficiency that this optimization holds in stock for the example in Section 4.3.

On another count, accumulation of uncertainty, that is, a gradual widening of the left and right spreads of the objective values as longer paths are considered, is of no consequence here. This point deserves our closer attention: in Ref. 2, Arnold et al. argued that, when one would attempt to generalize Dijkstra's or any other shortest path algorithm the way we have done, by adding up (or multiplying) fuzzy weights, "one will get a very uncertain conclusion, based on a long chain of uncertainties." Although this is, of course, true, it in no way impacts the actual objective of the algorithms, which is to find one or more optimal paths between two given nodes; moreover, the user will probably never be interested in the uncertainty in the outcome, because its defuzzication value gives him a solid approximation of the expected path weight.

What is even more striking, the strategy that Arnold et al. pursued as an "alternative" to the fuzzy shortest path algorithms can actually be seen as a special instance of the latter. Indeed, for every link in their graph (representing a large telecommunications network), they used a "fuzzy link evaluator" to aggregate several imprecise characteristics of that link into a single fuzzy number, which they then defuzzified and used in a crisp shortest path problem. Clearly this idea is equivalent[n] to our plan I!

7. CONCLUSION

In this article, we have built up a formal framework for the labeling algorithms to find optimal paths in a fuzzy weighted graph; in doing so, we have also spelled out the necessary and sufficient conditions for correctness, and examined which of the commonly used fuzzy ranking methods satisfy them. As a somewhat surprising result, our study has revealed that the method based on the very popular center of gravity defuzzification method has some defects that make it unsuitable for the purposes of the fuzzy shortest path algorithm. Moreover, it has been shown that if a defuzzification method is used as a basis for ranking fuzzy numbers, a significant optimization by defuzzifying all fuzzy weights before algorithm execution is possible under certain conditions on the involved defuzzification operator. What is more, this optimization not only allows us to reduce the complexity of the fuzzy shortest path problem to that of the crisp single-criterion labeling algorithms without sacrificing too much in the way of expressivity, but may also serve to give us more insight into, and hence attribute more credit to, heuristic approaches like the one in Ref. 2 by embedding them in a sound theoretical framework.

[n]It is not mentioned in Ref. 2 which defuzzification operation was used, so we cannot say whether their approach is correct in the sense of our theoretical exposition.

Acknowledgments

Chris Cornelis thanks the Fund for Scientific Research Flanders (FWO) for funding the research reported in this article.

References

1. Cook WJ, Cunningham WH, Pulleyblank WR, Schrijver A. Combinatorial optimization. Interscience Series in Discrete Mathematics and Optimization. New York: Wiley; 1998.
2. Arnold W, Hellendoorn H, Seising R, Thomas C, Weitzel A. Fuzzy routing. Fuzzy Set Syst 1997;85:131–151.
3. Aboelela E, Douligeris C. Fuzzy reasoning approach for QoS routing in B-ISDN. J Intell Fuzzy Syst 2000;9:11–27.
4. Dubois D, Prade H. Algorithmes de plus courts chemins pour traiter des données floues. RAIRO Oper Res 1978;12:213–227.
5. Dubois D, Prade H. Fuzzy sets and systems: Theory and applications. New York: Academic Press; 1980.
6. Klein CM. Fuzzy shortest paths. Fuzzy Set Syst 1991;39:27–41.
7. Lin K, Chen M. The fuzzy shortest path problem and its most vital arcs. Fuzzy Set Syst 1993;58:343–353.
8. Okada S, Soper T. A shortest path problem on a network with fuzzy arc lengths. Fuzzy Set Syst 2000;109:129–140.
9. Ramik J, Rimanek J. Inequality relation between fuzzy numbers and its use in fuzzy optimization. Fuzzy Set Syst 1985;16:123–138.
10. Martins EQV, Pascoal MMB, Dias Rasteiro DML, Santos JLE. The optimal path problem. Investigação Operacional 1999;19:43–60.
11. Rosenfeld A. Fuzzy graphs. In: Zadeh LA, Fu KS, Tanaka K, Shimura M, editors. Fuzzy sets and their applications to cognitive and decision processes. New York: Academic Press; 1975. pp 77–95.
12. Tanaka H, Ichihashi H, Asai K. A formulation of fuzzy linear programming problem based on comparison of fuzzy numbers. Control and Cybernetics 1984;13:185–194.
13. Yager RR. A procedure for ordering fuzzy sets of the unit interval. Inform Sci 1981;24:143–161.
14. Klir J, Yuan B. Fuzzy sets and fuzzy logic—Theory and applications. Upper Saddle River, NJ: Prentice Hall; 1995.
15. Campos L, Muñoz A. A subjective approach for ranking fuzzy numbers. Fuzzy Set Syst 1989;29:145–153.
16. Adamo JM. Fuzzy decision trees. Fuzzy Set Syst 1980;4:207–219.
17. Wang X, Kerre EE. Reasonable properties for the ordering of fuzzy quantities (I–II). Fuzzy Set Syst 2001;118:375–405.
18. Dubois D, Prade H. Ranking fuzzy numbers in the setting of possibility theory. Inform Sci 1986;30:183–224.
19. Wang X. A comparative study of the ranking methods for fuzzy quantities. Ph.D. thesis. Ghent University; 1997.
20. Hansen P. Bicriterion path problems. In: Beckmann M, Kunzi HP, editors. Multiple criteria decision making: Theory and applications. Lecture Notes in Economics and in Mathematical Systems. Berlin: Springer; 1980. pp 109–127.
21. Cherkassky BV, Goldberg AV, Radzik T. Shortest paths algorithms: Theory and experimental evaluation. Stanford University Technical Report STAN-CS-93-1480; 1993.