International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:2, 2008

Fuzzy Shortest Paths Approximation for Solving the Fuzzy Steiner Tree Problem in Graphs

Miloš Šeda

***Abstract*—**In this paper, we deal with the Steiner tree problem (STP) on a graph in which a fuzzy number, instead of a real number, is assigned to each edge. We propose a modification of the shortest paths approximation based on the fuzzy shortest paths (FSP) evaluations. Since a fuzzy min operation using the extension principle leads to nondominated solutions, we propose another approach to solving the FSP using Cheng's centroid point fuzzy ranking method.

***Keywords*—**Steiner tree, single shortest path problem, fuzzy ranking, binary heap, priority queue.

I. Introduction

THE *Steiner tree problem in graphs* (or *Network Steiner tree problem*, NSTP) [12], [17] is concerned with connecting a subset of vertices at a minimal cost. More precisely, given an undirected connected graph $G=(V,E)$ with vertex set V, edge set E, nonnegative weights associated with the edges, and a subset B of V (called *terminals* or *customer vertices*), the problem is to find a subgraph T that connects the vertices in B so that the sum of the weights of the edges in T is minimised. It is obvious that the solution is always a tree and it is called a *Steiner minimum tree* for B in G.

Applications of the NSTP are frequently found in the layout of connection structures in networks and circuit design [6], [11], [13]. Their common feature is that of connecting together a set of terminals (communications sites or circuits components) by a network of the minimal total length.

If $|B|=2$, then the problem reduces to the *shortest path problem* and can be solved by Dijkstra's algorithm. In the case of $B=V$, the NSTP reduces to the *minimum spanning tree* (MST) *problem* and can be solved by Jarník's (Prim's), Borůvka's and Kruskal's algorithms. All these algorithms are polynomial. However, in the general case, the NSTP is NP-complete [18] and therefore cannot be solved exactly for larger instances so that heuristic or approximation methods must be used. Normally, a Steiner minimum tree is not a mere minimum spanning tree, it can also span some nonterminals, called *Steiner vertices*.

In the graphical representation, vertices and edges can correspond to locations and connections between locations, respectively. The weights (lengths or costs) of edges can express geographical distances of the corresponding vertices or transportation costs expended (or times spent) to move between their end vertices. While geographical distances can be stated deterministically, costs or times can fluctuate with traffic conditions, payload and so on. In the last two cases, deterministic values for representing the edge weights cannot be used. A typical way of expressing these uncertainties in the edge weights is to utilize fuzzy numbers based on fuzzy set theory. In this case, we must define an order relation between fuzzy numbers, because the fuzzy variant of the problem evaluates "fuzzy min" operations. As many approaches for the comparison of fuzzy numbers do not guarantee that fuzzy numbers are totally ordered, they lead to a number of nondominated paths (or Pareto Optimal paths). However, the number of nondominated trees derived from a large network can be too numerous, which makes it difficult for a decision maker to choose a preferable tree.

In this paper, we propose a different approach based on Cheng's fuzzy ranking method [9] that makes it possible to solve the fuzzy Steiner minimum tree (FSMT) in a way similar to that of the crisp version of the problem.

II. Fuzzy Set Basics

Suppose that $X = \{x\}$ is a *universe*, i.e. the set of all possible (feasible, relevant) elements to be considered. Then a *fuzzy subset* (or a *fuzzy set*, for short) A in X is defined as a set of ordered pairs $\{(x, \mu_A(x))\}$, where $x \in X$ and $\mu_A : X \rightarrow [0,1]$ is the *membership function* of A; $\mu_A(x) \in [0,1]$ is the *grade of membership* of x in A, from 0 for full nonbelongingness to 1 for full belongingness through all intermediate values [5].

It is convenient to denote a fuzzy set defined in a finite universe, say A in $X = \{x_1, \ldots, x_n\}$ as $A = \mu_A(x_1)/x_1 + \ldots + \mu_A(x_n)/x_n$ where "$\mu_A(x_i)/x_i$" (called a singleton) is a "grade of membership/element" pair and "+" is used in the set-theoretical sense.

The *α-cut* of a fuzzy set A in X is defined as an ordinary set $A_\alpha \subseteq X$ such that

$$A_\alpha = \{x \in X \mid \mu_A(x) \geq \alpha\}, \forall \alpha \in [0,1]. \quad (1)$$

Similarly, the *α-level cut* of a fuzzy set A in X, denoted by A^α, is the crisp subset of X that contains all of the elements of X with exactly the given degree of membership α:

Manuscript received July 28, 2006. This work was supported in part by the Ministry of Education, Youth and Sports of the Czech Republic under research plan MSM 0021630518 "Simulation Modelling of Mechatronic Systems".

Miloš Šeda works in the Institute of Automation and Computer Science, Faculty of Mechanical Engineering, Brno University of Technology, Technická 2896/2, CZ 616 69 Brno, Czech Republic (phone: +420-54114 3332; fax: +420-54114 2330; e-mail: seda@fme.vutbr.cz).

$$A^{\alpha} = \{x \in X \mid \mu_A(x) = \alpha\}, \forall \alpha \in [0,1]. \quad (2)$$

The *level set of A*, denoted L_A, is a subset of [0,1] containing the values α that determine distinct α-cuts:

$$L_A = \{\alpha \in [0,1] \mid \mu_A(x) = \alpha \text{ for some } x \in X\} \quad (3)$$

III. Level Graphs

Let G be a graph with known vertices and edges, but unknown weights (or distances) on the edges. Let each weight w_i be a fuzzy convex number. Thus each α-level cut w_i^{α} will consist of two elements $w_i^{\alpha-} = \min w_i^{\alpha}$ and $w_i^{\alpha+} = \max w_i^{\alpha}$ (not necessarily distinct, if α=1). Now let G^{α} be the set of crisp graphs with edge i having the weight $w_i^{\alpha-}$ or $w_i^{\alpha+}$. Hence, G^{α} consists of at most $2^{|E|}$ crisp graphs. For α=0 we have $w_i^{\alpha+} = \sup(\text{supp } w_i)$ and $w_i^{\alpha-} = \inf(\text{supp } w_i)$.

Let Π_{ab} be the set of crisp paths from vertex a to vertex b. Define

$$\Sigma^{\alpha} = \{P \in \Pi \mid P \text{ is a shortest path of some graph in } G^{\alpha}\} \quad (4)$$

and the *fuzzy set of shortest paths* for the fuzzy graph G to be the fuzzy set Σ on Π with membership function

$$\eta_{\Sigma}(P) = \max_{\alpha \in (0,1]} \left\{\alpha \mid P \in \Sigma^{\alpha}\right\} \quad (5)$$

By [3] then this fuzzy set of shortest paths can be collapsed into a fuzzy shortest path, where each edge e_i has a membership in the fuzzy set Σ':

$$\mu_{\Sigma'}(i) = \max_{e_i \in P, P \in \Pi} \left\{\eta_{\Sigma}(P)\right\} \text{ for } i = 1, \ldots, |E| \quad (6)$$

Unfortunately, due to $2^{|E|}$ crisp graphs, this approach is not efficient.

IV. Fuzzy Ranking

Let us assume that the weights of the edges be given by *linear trapezoidal* or *triangular fuzzy numbers*.

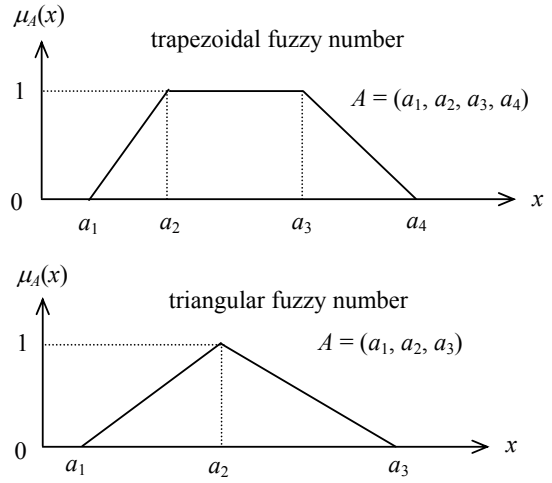

Fig. 1 Fuzzy numbers

Mathematically, a linear trapezoidal fuzzy number A can be represented by a quadruple (a_1, a_2, a_3, a_4) and its membership function μ_A is given by

$$\mu_A(x) = \begin{cases} 0 & , \text{ if } 0 < x \le a_1 \\ \dfrac{x - a_1}{a_2 - a_1} & , \text{ if } a_1 < x \le a_2 \\ 1 & , \text{ if } a_2 < x \le a_3 \\ \dfrac{a_4 - x}{a_4 - a_3} & , \text{ if } a_3 < x \le a_4 \\ 0 & , \text{ if } x > a_4 \end{cases} \quad (7)$$

Similarly, a linear triangular fuzzy number A can be represented by a triple (a_1, a_2, a_3) and its membership function μ_A is given by

$$\mu_A(x) = \begin{cases} 0 & , \text{ if } 0 \le x \le a_1 \\ \dfrac{x - a_1}{a_2 - a_1} & , \text{ if } a_1 \le x \le a_2 \\ 1 & , \text{ if } x = a_2 \\ \dfrac{x - a_3}{a_2 - a_3} & , \text{ if } a_2 \le x \le a_3 \\ 0 & , \text{ if } x \ge a_3 \end{cases} \quad (8)$$

In the sequel we can restrict our considerations to triangular fuzzy numbers without loss of generality.

The *addition* of these fuzzy numbers can be derived using Zadeh's extension principle and it is determined as follows:

$$A \oplus B = (a_1, a_2, a_3) \oplus (b_1, b_2, b_3) = \\ = (a_1 + b_1, a_2 + b_2, a_3 + b_3) \quad (9)$$

This operation always results in a triangular fuzzy number. In the shortest paths computations we must also evaluate minimum operations. This means that it is necessary to have a method of ranking or comparing fuzzy numbers.

An *ordering relation* $\le$ of fuzzy numbers can be defined, e.g., as follows:

$$A \le B \Leftrightarrow (a_1 \le b_1) \wedge (a_2 \le b_2) \wedge (a_3 \le b_3) \quad (10)$$

However, this relation is not a complete ordering relation, as fuzzy numbers A, B satisfying $(\exists\, i,j \in \{1,2,3\})$: $(a_i < b_i) \wedge (a_j > b_j)$ are not comparable by $\le$.

Let us consider a fuzzy min operation defined like the fuzzy addition in the following way:

$$\min(A,B) = (\min(a_1, b_1), \min(a_2, b_2), \min(a_3, b_3)) \quad (11)$$

It is evident that, for noncomparable fuzzy numbers A, B, this fuzzy min operation results in a fuzzy number different from both of them. For example, for A=(4,9,12) and B=(6,8,13), we get from equation (11) a fuzzy min (4,8,12) which differs from A and B.

The previous remarks demonstrate the difficulties with comparisons of fuzzy numbers. For this reason, the ranking or ordering methods of fuzzy quantities have been proposed by many authors. Most of them were summarized in [8], [24], [25]. Zadeh [27] shows that fuzzy graphs may be viewed as a generalisation of the calculi of crisp graphs. Blue *et al*. [3] give a taxonomy of graph fuzziness that distinguishes five basic types combining fuzzy or crisp vertex sets with fuzzy or crisp edge sets and fuzzy weights and fuzzy connectivity. The paper also introduces an approach to finding the shortest path based on level graphs, see Section III. Further methods can be

found, e.g., in [1], [2], [4], [7], [14], [16], [19], [21], [22], [23], [26]. However, all these approaches are rather theoretical and do not address the implementation point of view that will be in the center of our considerations. Unfortunately, none of these methods is commonly accepted.

In this paper, we use for simplicity the fuzzy ranking method described in [9], modified for the case of triangular fuzzy numbers. This method uses inverse functions $g_A^L:[0,1]\to[a_1,a_2]$ and $g_A^R:[0,1]\to[a_2,a_3]$ derived from functions $f_A^L:[a_1,a_2]\to[0,1]$ and $f_A^R:[a_2,a_3]\to[0,1]$, respectively.

From $y=\frac{x-a_1}{a_2-a_1}$ and $y=\frac{x-a_3}{a_2-a_3}$ we can easily derive that

$$g_A^L=a_1+(a_2-a_1)y,\quad g_A^R=a_3+(a_2-a_3)y \tag{12}$$

The ranking function is defined as the distance between the *centroid point* (x_0, y_0) and the origin, i.e.

$$R(A)=\sqrt{(x_0)^2+(y_0)^2} \tag{13}$$

where

$$x_0=\frac{\int_{\mathrm{Supp}\,A} x\mu_A(x)dx}{\int_{\mathrm{Supp}\,A}\mu_A(x)dx} \tag{14}$$

$$y_0=\frac{\int_0^1 yg_A^L dy+\int_0^1 yg_A^R dy}{\int_0^1 g_A^L dy+\int_0^1 g_A^R dy} \tag{15}$$

and Supp A is the support of A.

Fuzzy numbers A, B are then ranked by their ranking function values $R(A)$ and $R(B)$.

V. Shortest Paths Approximation

The shortest paths approximation [12] for solving the Steiner tree problem in graphs is an analogue of Jarník's algorithm for finding a minimum spanning tree. It can be described in a pseudopascal code as follows:

```
input  : connected undirected graph G=(V,E),
         a positive edge cost weight function,
         set of terminals B⊆V
output : Steiner tree T for B
select arbitrary terminal in B and denote it x1
T:={x1};
k:=1;
while k<|B| do
    begin determine a terminal x(k+1) ∉ T closest to T
          T:=T ∪ {x(k+1)};
          T:=T ∪ {a shortest path joining T and x(k+1);
          k:=k+1
    end;
{T is an approximation of the Steiner minimum tree}
```

Fig. 2 demonstrates this algorithm for a simple graph with 3 terminals (denoted by squares).

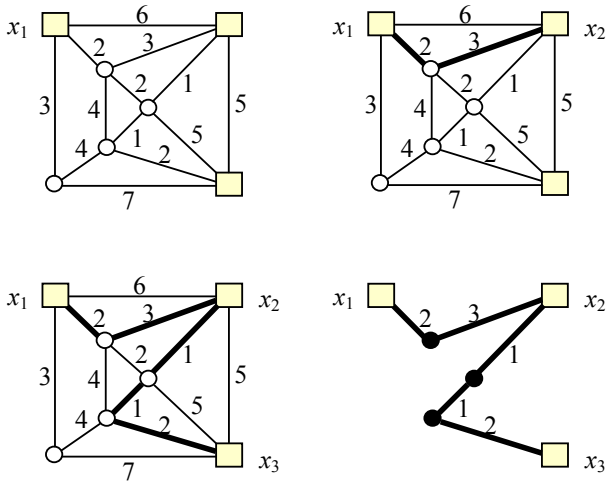

Fig. 2 Shortest paths approximation

In general, the solution can be improved by two additional steps:

1. Determine a fuzzy minimum spanning tree for the subnetwork of G induced by the vertices in T.
2. Delete from this minimum spanning tree nonterminals of degree 1 (one at a time) and edges incident with them. The resulting tree is the (suboptimal) solution.

VI. Fuzzy Shortest Path Problem

Finding the shortest path from a specified *source* (or an *origin*) to a *sink* (or a *destination*) is a fundamental problem in transportation, routing, and communications applications. Alternatively, the problem can be formulated as finding the shortest paths from a source to all other locations (communication sites). The computational problem is called the *single source shortest path problem* (SPP for short).

Before we discuss the algorithm for the fuzzy SPP, we will briefly summarize the data structures used in it.

The most important data structure substantially determining the efficiency is the *priority queue* [20]. The priority queue supports these operations:

Insert(Q,u,*key*) - insert u with the key value *key* in Q; *ExtractMin*(Q) - extract the item with the minimum key value in Q, and *DecreaseKey*(Q,u,*new_key*) - decrease the value of u's key value to *new_key*.

The priority queue can be easily implemented by a *binary heap*. This is a binary tree with vertices numbered by integers and satisfying these conditions:

(1) Each vertex of a binary heap that is not included in the last two levels has two successors.

(2) In the last level, all vertices are placed from the left. This means that, passing vertices in the last but one level from left to right, only some of them (or none) may have two successors. In the latter case, at most one vertex may exist with one successor and all other vertices of this level are

leaves.

(3) The number of each vertex is not higher than the numbers of its successors.

The root of the binary heap is numbered by 1, other vertices at lower levels from left to right are assigned consecutive integers starting from 2.

The binary tree can be represented by a one-dimensional array. It can be proved that, for the defined numbering of the binary heap vertices, the j-th element in the i-th level of a binary heap corresponds to position $2^{i-1}+j-1$ of the array; left and right successors of vertex i have positions $2i$ and $2i+1$, respectively, and its predecessor has position $\lfloor i/2 \rfloor$ (that is position i **div** 2 in the Pascal notation.

We will skip detailed descriptions of Insert, ExtractMin and DecreaseKey operations because they can be found in all books dealing with algorithms and data structures (e.g. [10],[15], [18]) only noting that the time complexity of all of them is $O(\log n)$ where n is the number of vertices.

Dijkstra's algorithm for the "deterministic" SPP finds shortest paths from a source s to all other vertices. Now we will generalise it for the case of fuzzy edge lengths.

Let $d[v]$ be the fuzzy number corresponding to the length of the shortest path from s to v. Initially, $d[s]=(0,0,0)$ and all other $d[v]$ values are set to (∞,∞,∞). The algorithm is based on gradual improvements of the shortest paths distance from s to the other vertices. Let us consider an edge (u,v) whose weight is $w(u,v)$ and suppose that we have already computed current estimates of $d[u]$, $d[v]$. If $R(d[u]\oplus w(u,v))<R(d[v])$, then $d[u]\oplus w(u,v)$ becomes a new estimate of $d[v]$. The process by which an estimate is updated is called *relaxation*. The shortest way back to the source is through u by updating the predecessor pointer. If we repeat the relaxation for all edges then $d[v]$ values converge to the shortest paths of vertices v to the source. Let S be a set of vertices for which we know the final value $d[v]$. Initially, S is empty. The question is, how do we know which vertex of $V-S$ to add next to S? The algorithm uses a greedy strategy, i.e. it takes the vertex for which $d[u]$ is a minimum, in other words, it takes the unprocessed vertex that is closest (by our estimate) to s.

In order to perform this selection efficiently, we store the vertices of $V-S$ in a priority queue (binary heap), where the key value of each vertex u is $d[u]$. Information on which vertices are in S (they have final value $d[v]$) is stored in the *determined* array of Boolean variables.

Here is the algorithm.

```
Input:  connected weighted graph G=(V,E)
          with fuzzy edge lengths w(e), e∈E,
          s - source (root);
output: d[u];
auxiliary variables:
        Adj[u] - set of neighbours of vertex u.
for ∀u∈V do
  begin  d[u]:=(0, ∞,∞);
         determined[u]:=False;
        pred[u] := nil
  end;
d[s] := (0,0,0);
Q := priority_queue(V);   { push all vertices into Q
                            ordered by d[u] }
while NonEmpty(Q) do
  begin  u := ExtractMin(Q);
         for ∀v∈Adj[u] do
           if R(d[u]⊕w(u,v))<R(d[v])
             then begin  d[v] := d[u]⊕w(u,v);
                         DecreaseKey(Q,v, d[v]);
                         pred[v] := u
                  end;
         determined[u]:=True;
  end; { Fuzzy Dijkstra SPP algorithm }
 { pointers in the pred array define the inverted tree of
   the shortest paths pointing back to s }
```

VII. Analysis of the Algorithm

In this section, we analyse the time complexity of the proposed algorithm for the fuzzy network Steiner tree problem.

Theorem 1 *The proposed algorithm runs in $O(|B|\,|E|\log|V|)$ time.*

Proof. Since the time complexity of the algorithm is given by $|B|\times$(time complexity of the fuzzy SPP algorithm) we can focus on the FSSP. Let $O(T_R)$ be the time of the centroid point evaluation. To analyse the algorithm, we account for the time spent on each vertex as it is extracted from the priority queue. It takes $O(\log|V|)$ to extract this vertex from the queue. For each incident edge, we spend potentially $O(\log|V|)$ time decreasing the key of the neighbouring vertex. Thus the time is $O(\log|V|+T_R\ \deg(u)\log|V|)$. The other steps of the update take constant time. So the overall running time is

$$T(V,E)=\sum_{u\in V}(\log|V|+T_R\deg(u)\log|V|)=$$
$$=\sum_{u\in V}(1+T_R\deg(u))\log|V|=\log|V|\sum_{u\in V}(1+T_R\deg(u))=$$
$$=\log|V|(|V|+T_R.2|E|)=O((|V|+|E|)\log|V|)$$

Since G is connected, $|V|$ is asymptotically no greater than $|E|$, so this is $O(|E|\log|V|)$.

•

VIII. Conclusion and Future Work

In this paper, the problem of finding the network Steiner minimum tree was studied. In contrast to traditional approaches, we assumed that the weights of edges were given by linear triangular fuzzy numbers and proposed a fuzzy modification of the shortest paths approximation with an emphasis on efficient implementation of the shortest path subroutine using the priority queue. For comparisons of fuzzy numbers, Cheng's centroid point fuzzy ranking method was used.

In the future, we intend to investigate a different approach based on a reformulation of the problem where the Steiner tree

problem in graphs is to find $S \subseteq V$ so that the spanning tree of the subgraph of G induced by $B \cup S$ has a minimum total weight.

The solutions in the search space are then restricted to binary strings where a 1 or 0 corresponds to whether or not a vertex from $V-B$ is included in the Steiner tree. Because of the exponentially growing size of the search space, we will use metaheuristics (such as genetic algorithms and simulated annealing) for finding an approximation of the solution.

REFERENCES

[1] M.R. Berthold and K.-P. Huber, "Constructing Fuzzy Graphs from Examples," *Intelligent Data Analysis*, vol. 3, pp. 37-53, 1999.
[2] K.R. Bhutani and A. Rosenfeld, "Fuzzy End Nodes in Fuzzy Graphs," *Information Sciences*, vol. 152, pp. 323-326, 2003.
[3] M. Blue, B. Bush and J. Puckett, "Unified Approach to Fuzzy Graph Problems," *Fuzzy Sets and Systems*, vol. 125, pp. 355-368, 2002.
[4] A. Boulmakoul, "Generalized Path-Finding Algorithms on Semirings and the Fuzzy Shortest Path Problem," *Journal of Computational and Applied Mathematics*, vol. 162, pp. 263-272, 2004.
[5] P. Bosc and J. Kacprzyk (eds.), *Fuzziness in Database Management Systems*. Heidelberg: Physica-Verlag, 1995.
[6] D. Cavendish, A. Fei, M. Gerla and R. Rom, "On the Construction of Low Cost Multicast Trees with Bandwidth Reservation," in *Proc. of the Int. Conf. on High-Performance Computing and Networking HPCN*, Amsterdam, 1998, 29 pp.
[7] S. Chanas and P. Zielinski, "Ranking Fuzzy Interval Numbers in the Setting of Random Sets - Further Results," *Information Sciences*, vol. 117, pp. 191-2000, 1999.
[8] P.-T. Chang, and E.S. Lee, "Fuzzy Arithmetics and Comparison of Fuzzy Numbers", in M. Delgado, J. Kacprzyk, J.-L. Verdegay and M.A. Vila (eds.), *Fuzzy Optimization*. Heidelberg: Physica-Verlag, pp. 69-82, 1994.
[9] C.-H. Cheng, "A New Approach for Ranking Fuzzy Numbers by Distance Method," *Fuzzy Sets and Systems*, vol. 95, pp. 307-317, 1998.
[10] T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, *Introduction to Algorithms*. Massachusetts: MIT Press, 2001.
[11] D.W. Corne, M.J. Oates and G.D. Smith (eds.), *Telecommunications Optimization: Heuristic and Adaptive Techniques*. New York: John Wiley & Sons, 2000.
[12] D.-Z. Du, J.M. Smith, and J.H. Rubinstein, *Advances in Steiner Trees*. Dordrecht: Kluwer Academic Publishers, 2000.
[13] A. Fink, G. Schneidereit, G. and S. Voss, "Solving General Ring Network Design Problems by Meta-Heuristics," in M. Laguna and J.L. González Velarde (eds.), *Computing Tools for Modeling, Optimization and Simulation Interfaces in Computer Science and Operations Research*. Boston: Kluwer Academic Publishers, pp. 91-113, 1999.
[14] P. Fortemps and M. Roubens, "Ranking and Defuzzification Methods Based on Area Compensation," *Fuzzy Sets and Systems*, vol. 82, pp. 319-330, 1996.
[15] J. Gross and J. Yellen, *Graph Theory and Its Applications*. Boca Raton: CRC Press, 1999.
[16] P. Grzegorzewski, "Metrics and Orders in Space of Fuzzy Numbers," *Fuzzy Sets and Systems*, vol. 97, pp. 83-94, 1998.
[17] F.K. Hwang, D.S. Richards and P. Winter, *The Steiner Tree Problem*. Amsterdam: North-Holland, 1992.
[18] S. Khuller, "Design and Analysis of Algorithms," Lecture Notes, University of Maryland, Department of Computer Science, 1994, 112 pp.
[19] M. Modarres and S. Sadi-Nezhad, "Ranking Fuzzy Numbers by Preference Ratio," *Fuzzy Sets and Systems*, vol. 118, pp. 429-436, 2001.
[20] D.M. Mount, "Design and Analysis of Computer Algorithms," Lecture Notes, University of Maryland, College Park, 1999, 131 pp.
[21] T. Savšek, M. Vezjak and N. Pavešić, "Fuzzy Trees in Decision Support Systems," *European Journal of Operational Research*, 2006, in print.
[22] A. Sengupta and T.K. Pal, "On Comparing Interval Numbers," *European Journal of Operational Research*, vol. 127, pp. 28-43, 2000.
[23] S. Tan, Y. Yu and P.-Z. Wang,, "Building Fuzzy Graphs form Samples of Nonlinear Functions," *Fuzzy Sets and Systems*, vol. 93, pp. 337-352, 1998.
[24] X. Wang and E.E. Kerre, "Reasonable Properties for Ordering of Fuzzy Quantities (I)," *Fuzzy Sets and Systems*, vol. 118, pp. 375-385, 2001.
[25] X. Wang and E.E. Kerre, "Reasonable Properties for Ordering of Fuzzy Quantities (II)," *Fuzzy Sets and Systems*, vol. 118, pp. 387-405, 2001.
[26] J.-S. Yao and K. Wu, "Ranking Fuzzy Numbers Based on Decomposition Principle and Signed Distance," *Fuzzy Sets and Systems*, vol. 116, pp. 275-288, 2000.
[27] L.A. Zadeh, "Fuzzy Logic and the Calculi of Fuzzy Rules, Fuzzy Graphs, Fuzzy Probabilities," *Computers & Mathematics with Applications*, vol. 37, p. 35, 1999.