International Journal of Computational Intelligence Systems, Vol. 6, No. 6 (November, 2013), 1040-1051

# Diameter Constrained Fuzzy Minimum Spanning Tree Problem

**Sk. Md. Abu Nayeem [1,*], Madhumangal Pal [2]**

[1] *Department of Mathematics, Aliah University,*
*Salt Lake Campus, DN - 41, Sector - V,*
*Kolkata - 700 091, West Bengal, India.*
*E-mail: nayeem.math@aliah.ac.in*

[2] *Department of Applied Mathematics with Oceanology and Computer Programming,*
*Vidyasagar University, Midnapore - 721 102,*
*West Bengal, India.*
*E-mail: mmpalvu@gmail.com*

Received 30 June 2012
Accepted 5 December 2012

**Abstract**

In this paper, we have studied the constrained version of the fuzzy minimum spanning tree problem. Costs of all the edges are considered as fuzzy numbers. Using the $m_\lambda$ measure, a generalization of credibility measure, the problem is formulated as chance-constrained programming problem and dependent-chance programming problem according to different decision criteria. Then the crisp equivalents are derived when the fuzzy costs are characterized by trapezoidal fuzzy numbers. Furthermore, a fuzzy simulation based hybrid genetic algorithm is designed to solve the proposed models using Prüfer like code representation of labeled trees.

*Keywords:* Minimum spanning tree; possibility and necessity measure; chance constrained programming; Prüfer code; genetic algorithm.

## 1. Introduction

The minimum spanning tree (MST) problem is one of the fundamental problems in graph theory. Let $G = (V,E)$ be a finite undirected simple connected graph with a set $V$ of vertices and a set $E$ of edges. A tree $T$ with the same vertex set $V$ is called a spanning tree of $G$. Suppose a positive weight $w_i$ is associated with every edge $e_i$ in $G$. Then, $w(T) = \sum_{e_i \in T} w_i$ is called the weight of the spanning tree $T$. The MST problem is to find a spanning tree $T^*$ from the set $\Gamma$ of all spanning trees of $G$ such that $w(T^*) = \min_{T\in\Gamma} w(T)$. The MST problem has been studied extensively by many researchers and many efficient algorithms have been found by Kruskal, Prim, Dijkstra and many others.[1,2,3,4,5,6]

For every pair of distinct vertices $v_i, v_j \in V$, there exists a unique path $P^T_{ij}$ in $T$, joining $v_i$ and $v_j$. Denote the number of edges in the path $P^T_{ij}$ by $d^T_{ij}$. Also denote by $diam(T) = \max\{d^T_{ij} : v_i, v_j \in V\}$, the diameter of $T$. Given a positive integer $2 \leqslant D \leqslant |V|-1$, the diameter constrained minimum spanning tree (DCMST) is to find a spanning tree $T^*$ from $\Gamma$ such that $w(T^*) = \min_{T\in\Gamma} w(T)$ and $diam(T^*) \leqslant D$.

*Corresponding author.

DCMST has been shown to be NP-hard for $D \geqslant 4$.[7] Gouvieta and Magnanti[8] have modeled the problem as a network design to set up communication between every pair of vertices, meeting or surpassing a given quality requirement. This problem has also been applied to data compression by Bookstien and Klien,[9] and distributed mutual exclusion in parallel computing by Raymond,[10] and Deo and Abdalla.[11]

Different formulation of the DCMST has been found in the literature.[8,11,12] These formulations implicitly use a property of feasible diameter constrained spanning tree, pointed out by Handler.[13] He pointed out that, when $D$ is even, a central vertex $i \in V$ must exist in a feasible tree $T$, such that no other vertex of $T$ is more than $D/2$ edges away from $i$ and when $D$ is odd, a central edge $e = (i, j) \in E$ must exist in $T$, such that no vertex of $T$ is more than $(D-1)/2$ edges away from the closest extremity of $(i, j)$. More recently, dos Santos *et. al*[14] proposed an alternative formulation of the odd $D$ case of DCMST, by introducing an artificial vertex. They have also applied a lifting procedure to strengthen the formulation. But none of them have studied the problem in fuzzy environment.

In many real applications, the problem parameters are found to be vague or imprecise in nature. Then the problem parameters may be considered as fuzzy variables by an expert system. Ito and Ishii[15] formulated an MST problem with fuzzy cost as chance-constrained programming based on necessity measure. Chang and Lee[16,17] defined three means based on overall existence ranking index (OERI) for ranking fuzzy costs of spanning trees. Recently, Liu[18,19] developed the credibility theory including credibility measure, pessimistic value and expected value as fuzzy ranking methods. Gao and Lu[20] proposed the concepts of expected minimum spanning tree (EMST), $\alpha$-pessimistic minimum spanning tree ($\alpha$-PMST) and most minimum spanning tree (MMST) in a fuzzy quadratic minimum spanning tree (FQMST) problem, based on the credibility theory. They also discussed the crisp equivalent problems when the fuzzy costs are characterized by trapezoidal fuzzy numbers and devised genetic algorithm to solve those. Yang and Iwamura[21] introduced the $m_\lambda$ measure as the linear combination of possibility measure and necessity measure and employed that measure to construct the fuzzy chance-constrained programming models.

In this paper, we propose the $\alpha$-PMST and MMST models of the diameter constrained fuzzy minimum spanning tree (DCFMST) problem, based on the $m_\lambda$ measure. Also we discuss the crisp equivalents of the models when the fuzzy costs are taken as trapezoidal fuzzy numbers. A simulation based genetic algorithm is designed to solve the proposed models. This algorithm uses the Prüfer like code given by Deo and Micikevicius[22] to represent labeled trees.

## 2. A brief introduction to $m_\lambda$ measure

In fuzzy optimization theory, the most important fuzzy ranking methods are based on the possibility and necessity measures. The possibility theory was proposed by Zadeh[23] and developed by many researchers such as Dubois and Prade.[24] These possibility and necessity measures are used to describe the chance of fuzzy event. Let $\xi$ be a fuzzy variable with membership function $\mu_\xi(x)$ and $B$ be an arbitrary subset of $\mathbb{R}$. Then the possibility measure of fuzzy event $\{\xi \in B\}$ is defined as $\text{Pos}\{\xi \in B\} = \sup_{x \in B} \mu_\xi(x)$. The necessity of this fuzzy event is defined as the impossibility of the opposite event. That is, $\text{Nec}\{\xi \in B\} = 1 - \text{Pos}\{\xi \in B^c\} = 1 - \sup_{x \in B^c} \mu_\xi(x)$. In general $\text{Pos}\{\xi \in B\} \geqslant \text{Nec}\{\xi \in B\}$. It is obvious that a fuzzy event may fail to occur even though its possibility is 1 and may happen to occur even if its necessity is 0. But if the necessity achieves 1, the fuzzy event must hold. Thus, if the decision-maker is optimistic and does not care about the potential risk, the necessity measure may be considered as a decision making tool. Suppose the decision-maker seeks to find the best solution $x^*$ in order to maximize the chance of occurring the fuzzy event $\{f(x, \xi) \in B\}$, where $x$ be the decision variable and $\xi$ be the fuzzy parameter vector. Then if the possibility measure is employed as a chance measure, a decision $x^*$ will be recognized as the best decision if it satisfies $\text{Pos}\{f(x^*, \xi) \in B\} = 1$. Similarly if the decision-maker is pessimistic, the necessity measure may be chosen as the chance measure. In fact, the solution $x^*$ is not necessarily be the best solution for the necessity measure as the corresponding objective value is less than or equal to 1.

But, in reality, most decision-makers are neither absolutely optimistic, nor absolutely pessimistic. To balance between the optimism and pessimism, a convex combination of the possibility measure and the necessity measure is introduced by Yang and Iwamura.[21] Formally, the $m_\lambda$ measure for the chance of a fuzzy event is defined as,

$$m_\lambda\{\xi \in B\} = \lambda \text{Pos}\{\xi \in B\} + (1-\lambda)\text{Nec}\{\xi \in B\},$$

where, the parameter $\lambda \in [0,1]$ is pre-determined by the decision-maker according to the degree of optimism or pessimism.

It is easy to verify that $m_\lambda$ has the following properties:

1. $0 \leqslant m_\lambda\{\xi \in A\} \leqslant 1$ for any set $A \subset \mathbb{R}$.
2. $\text{Nec}\{\xi \in A\} \leqslant m_\lambda\{\xi \in A\} \leqslant \text{Pos}\{\xi \in A\}$ for all $A \subset \mathbb{R}$.
3. $m_\lambda\{\xi \in A\} \leqslant m_\lambda\{\xi \in B\}$ whenever $A \subset B$.
4. For any $A \subset \mathbb{R}$ and $\lambda \in [0,1], m_\lambda\{\xi \in A\} + m_{1-\lambda}\{\xi \in A^c\} = 1$.
5. If $\lambda_1 \leqslant \lambda_2$, then $m_{\lambda_1}\{\xi \in A\} \leqslant m_{\lambda_2}\{\xi \in A\}$ for any $A \subset \mathbb{R}$.

The credibility measure Cr, introduced by Liu[18,19] is an average of possibility measure and necessity measure. Clearly, when $\lambda = \frac{1}{2}$, the $m_\lambda$ measure reduces to the credibility measure.

Based on the $m_\lambda$ measure, the critical values of a fuzzy variable are defined as follows:

**Definition 1.** Let $\xi$ be a fuzzy variable and $\alpha \in [0,1]$. Then $\xi_{\inf}(\lambda;\alpha) = \inf\{r : m_\lambda\{\xi \leqslant r\} \geqslant \alpha\}$ is called the $(\lambda;\alpha)$-pessimistic value of $\xi$ and $\xi_{\sup}(\lambda;\alpha) = \sup\{r : m_\lambda\{\xi \geqslant r\} \geqslant \alpha\}$ is called the $(\lambda;\alpha)$-optimistic value of $\xi$.

## 3. Diameter constrained fuzzy minimum spanning tree

Let the cost associated with the edge $e_i$ be $\xi_i$, which may represent the construction or running cost, $i = 1,2,\ldots,m$. Let $x$ be a binary decision variable vector whose components are defined by

$$x_i = \begin{cases} 1 & \text{if edge } e_i \text{ is in the spanning tree } x \\ 0 & \text{otherwise,} \end{cases}$$

then the cost of the spanning tree $x = (x_1, x_2, \ldots, x_n)$ is given by $C(x,\xi) = \sum_{i=1}^{m} \xi_i x_i$.

Let $\Gamma$ be the set of all spanning trees corresponding to the graph $G$. then the spanning tree $x^*$ is called a minimum spanning tree if

$$C(x^*,\xi) \leqslant C(x,\xi) \text{ for all } x \in \Gamma.$$

In reality, exact information about the construction or running cost may not be available to the decision-maker and then the costs $\xi_i, i = 1,2,\ldots,m$ may be specified as fuzzy variables according to the expert system. Then the cost function $C(x,\xi)$ also becomes fuzzy. Now the decision maker may set a confidence level $\alpha$ as an appropriate safety margin and the degree of optimism (or pessimism) $\lambda$, and may wish to minimize the $(\lambda;\alpha)$-pessimistic value of $C(x,\xi)$. For this case, the $(\lambda;\alpha)$-pessimistic minimum spanning tree [$(\lambda;\alpha)$-PMST] can be defined as follows:

**Definition 2.** A spanning tree $x^*$ is called the $(\lambda;\alpha)$-pessimistic minimum spanning tree if

$$\min\{r : m_\lambda\{C(x^*,\xi) \leqslant r\} \geqslant \alpha\} \\ \leqslant \min\{r : m_\lambda\{C(x,\xi) \leqslant r\} \geqslant \alpha\},$$

for all spanning tree $x \in \Gamma$, where $\alpha$ is predetermined confidence level; $\lambda$ is predetermined degree of optimism (or pessimism) and $\min\{r : m_\lambda\{C(x^*,\xi) \leqslant r\} \geqslant \alpha\}$ is called the $(\lambda;\alpha)$-pessimistic minimum cost.

Sometimes, the decision-maker may provide a cost supremum $\overline{C}$ and hope that the $m_\lambda$-measure of the costs not exceeding $\overline{C}$ will be maximized, subject to the predetermined value of $\lambda$. For this case, the concept of most minimum spanning tree (MMST) is adopted as follows:

**Definition 3.** A spanning tree $x^*$ is called the most minimum spanning tree if

$$m_\lambda\{C(x^*,\xi)\leqslant\overline{C}\}\geqslant m_\lambda\{C(x,\xi)\leqslant\overline{C}\}$$

for all spanning tree $x\in\Gamma$, where $\overline{C}$ is the predetermined cost supremum, $\lambda$ is the predetermined degree of optimism or pessimism and $m_\lambda\{C(x^*,\xi)\leqslant\overline{C}\}$ is called the most $m_\lambda$-measure for the given cost supremum.

Given a positive integer $2\leqslant D\leqslant|V|-1$, the diameter constrained fuzzy minimum spanning tree (DCFMST) $x^*$ is either $(\lambda;\alpha)$-PMST, or MMST satisfying the diameter constraint $diam(x^*)\leqslant D$.

## 4. Chance-constrained programming models of DCFMST

Chance-constrained programming is a powerful method to model stochastic decision systems and fuzzy decision systems.[25] The main idea of chance constrained programming is to optimize the critical value of the fuzzy objective with certain confidence level subject to some chance constraints. Let $D$ be the given integral upper bound of the spanning trees. If the $(\lambda;\alpha)$-PMST is sought for given $\lambda$ and $\alpha$ values, then the following model works:

**Model - I**

$$\begin{aligned}\min\quad & r\\ \text{subject to}\quad & m_\lambda\left\{\sum_{i=1}^m \xi_i x_i\leqslant r\right\}\geqslant\alpha\\ & diam(x)\leqslant D \qquad (1)\\ & x_i\in\{0,1\}, i=1,2,\ldots,m\\ & x\in\Gamma.\end{aligned}$$

Now if the decision-maker does not stick to the total fulfilment of the crisp inequality (1), then $D$ may be considered as a fuzzy number $\widetilde{D}$ and the inequality (1) may be replaced by another chance constraint with the predetermined confidence level $\alpha_1$ and degree of optimism $\lambda_1$, as $m_{\lambda_1}\left\{diam(x)\leqslant\widetilde{D}\right\}\geqslant\alpha_1$. It may be noted that $\lambda_1$ and $\alpha_1$ may eventually coincide with $\lambda$ and $\alpha$.

Thus we have the revised $(\lambda;\alpha)$-PMST model as below:

**Model - II**

$$\begin{aligned}\min\quad & r\\ \text{subject to}\quad & m_\lambda\left\{\sum_{i=1}^m \xi_i x_i\leqslant r\right\}\geqslant\alpha\\ & m_{\lambda_1}\left\{diam(x)\leqslant\widetilde{D}\right\}\geqslant\alpha_1\\ & x_i\in\{0,1\}, i=1,2,\ldots,m\\ & x\in\Gamma.\end{aligned}$$

Now if the decision maker wishes to maximize the chance function of some events, then the dependent-chance programming concept, introduced by Liu[18] can be applied. Here the main idea is to select the decision with maximal chance to meet the fuzzy event. For the DCFMST problem, suppose the decision maker sets a cost supremum $\overline{C}$ and wishes to find the MMST. Then we have the dependent-chance programming model as follows:

**Model - III**

$$\begin{aligned}\max\quad & m_\lambda\left\{\sum_{i=1}^m \xi_i x_i\leqslant\overline{C}\right\}\\ \text{subject to}\quad & diam(x)\leqslant D\\ & x_i\in\{0,1\}, i=1,2,\ldots,m\\ & x\in\Gamma.\end{aligned}$$

Again considering $D$ as a fuzzy number, as in Model - II, we can have the following revised model for MMST.

**Model - IV**

$$\begin{aligned}\max\quad & m_\lambda\left\{\sum_{i=1}^m \xi_i x_i\leqslant\overline{C}\right\}\\ \text{subject to}\quad & m_{\lambda_1}\left\{diam(x)\leqslant\widetilde{D}\right\}\geqslant\alpha_1\\ & x_i\in\{0,1\}, i=1,2,\ldots,m\\ & x\in\Gamma\end{aligned}$$

where $\lambda_1$ and $\alpha_1$ are pre-specified degree of optimism and confidence level respectively.

## 5. Crisp equivalents of the DCFMST models

In this section, we give the crisp equivalents of the DCFMST models I – IV given in earlier section.

**Lemma 1.** *Let $\xi = (r_1, r_2, r_3, r_4)$ be a trapezoidal fuzzy variable, $\lambda$ be a given degree of optimism and $\alpha$ be a given confidence level. Then the constraint $m_\lambda\{\xi \leqslant r\} \geqslant \alpha$ is equivalent to $f_\alpha \leqslant r$, where*

$$f_\alpha = \begin{cases} \frac{\alpha}{\lambda}(r_2 - r_1) + r_1 & \text{if } \alpha \leqslant \lambda \\ \frac{1-\alpha}{1-\lambda}(r_3 - r_4) + r_4 & \text{if } \alpha > \lambda \end{cases}$$

*and hence $\xi_{\inf}(\lambda;\alpha) = f_\alpha$.*

**Proof.** Since $\xi = (r_1, r_2, r_3, r_4)$ is a trapezoidal fuzzy number, its membership function, defined as

$$\mu_\xi(x) = \begin{cases} \frac{x - r_1}{r_2 - r_1} & \text{if } r_1 \leqslant x \leqslant r_2 \\ 1 & \text{if } r_2 \leqslant x \leqslant r_3 \\ \frac{x - r_4}{r_3 - r_4} & \text{if } r_3 \leqslant x \leqslant r_4 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

is a continuous function in $x$ and it takes every value in $[0,1]$ at least once. Then we have the following two cases.

Case - 1 ($\alpha \leqslant \lambda$): Here $\frac{\alpha}{\lambda}$ being in $[0,1]$, there exists a real number $f$ such that $\mu_\xi(f) = \frac{\alpha}{\lambda}$.

Let $f_\alpha = \inf\left\{f : \mu_\xi(f) = \frac{\alpha}{\lambda}\right\} = \frac{\alpha}{\lambda}(r_2 - r_1) + r_1$.

Then by using the continuity of $\mu_\xi(x)$ and the properties of fuzzy numbers, we have

$$\text{Pos}\{\xi \leqslant f_\alpha\} = \sup_{y \leqslant f_\alpha}\{\mu_\xi(y)\} = \mu_\xi(f_\alpha) = \frac{\alpha}{\lambda}$$

and $\text{Pos}\{\xi > f_\alpha\} = 1$, which implies that $m_\lambda\{\xi \leqslant f_\alpha\} = \alpha$.

We also see that the value of $m_\lambda\{\xi \leqslant f_\alpha\}$ increases if we replace the number $f_\alpha$ with any larger value. Thus the crisp equivalent of $m_\lambda\{\xi \leqslant r\} \geqslant \alpha$ is $f_\alpha \leqslant r$, where $f_\alpha = \frac{\alpha}{\lambda}(r_2 - r_1) + r_1$.

Case - 2 ($\alpha > \lambda$): If $r \geqslant r_3$, we have $\text{Pos}\{\xi \leqslant r\} = 1$ and then $m_\lambda\{\xi \leqslant r\} \geqslant \alpha$ is equivalent to $\text{Nec}\{\xi \leqslant r\} \geqslant \frac{\alpha - \lambda}{1 - \lambda}$. Since possibility measure and necessity measures are dual, $\text{Nec}\{\xi \leqslant r\} \geqslant \frac{\alpha-\lambda}{1-\lambda}$ is equivalent to $\text{Pos}\{\xi > r\} \leqslant \frac{\alpha-\lambda}{1-\lambda}$.

Now $0 \leqslant \alpha \leqslant 1, 0 \leqslant \lambda \leqslant 1$ and $\alpha > \lambda$ gives $\frac{1-\alpha}{1-\lambda} \in (0,1)$. So, there exists a real number $f$ such that $\mu_\xi(f) = \frac{1-\alpha}{1-\lambda}$.

Let $f_\alpha = \sup\left\{f : \mu_\xi(f) = \frac{1-\alpha}{1-\lambda}\right\} = \frac{1-\alpha}{1-\lambda}(r_3 - r_4) + r_4 \geqslant r_3 (\text{as } r_3 - r_4 \leqslant 0)$.

Again by using the continuity of $\mu_\xi(x)$ and the properties of fuzzy numbers, we have

$$\text{Pos}\{\xi > f_\alpha\} = \sup_{y > f_\alpha}\{\mu_\xi(y)\} = \alpha.$$

Noting that the possibility $\text{Pos}\{\xi > f_\alpha\}$ decreases if the number $f_\alpha$ is replaced with any larger value, we have that the crisp equivalent of $\text{Pos}\{\xi > r\} \leqslant \frac{\alpha-\lambda}{1-\lambda}$ is $r \geqslant f_\alpha$, where $f_\alpha = \frac{1-\alpha}{1-\lambda}(r_3 - r_4) + r_4$.

Hence the proof. □

**Theorem 2.** *Let $\xi_i = (r_{i1}, r_{i2}, r_{i3}, r_{i4}), i = 1, 2, \ldots, m$ are independent trapezoidal fuzzy numbers. Then the crisp equivalent of Model – I is given by*

$$\begin{aligned} &\min f \\ &\text{s.t. } diam(x) \leqslant D \\ &x_i \in \{0,1\}, i = 1, 2, \ldots, m \\ &x \in \Gamma \end{aligned}$$

*where,*

$$f = \begin{cases} \left(1 - \frac{\alpha}{\lambda}\right)\sum_{i=1}^{m} r_{i1}x_i + \frac{\alpha}{\lambda}\sum_{i=1}^{m} r_{i2}x_i & \text{if } \alpha \leqslant \lambda \\ \frac{1-\alpha}{1-\lambda}\sum_{i=1}^{m} r_{i3}x_i + \frac{\alpha-\lambda}{1-\lambda}\sum_{i=1}^{m} r_{i4}x_i & \text{if } \alpha > \lambda \end{cases} \tag{3}$$

**Proof.** Since $x_i \geqslant 0$, for all $i = 1, 2, \ldots, m$, and $\xi_i = (r_{i1}, r_{i2}, r_{i3}, r_{i4}), i = 1, 2, \ldots, m$ are trapezoidal fuzzy numbers, $\sum_{i=1}^{m} \xi_i x_i$ is also a trapezoidal fuzzy number given by the quadruple $\left(\sum_{i=1}^{m} r_{i1}x_i, \sum_{i=1}^{m} r_{i2}x_i, \sum_{i=1}^{m} r_{i3}x_i, \sum_{i=1}^{m} r_{i4}x_i\right)$.

Hence, by Lemma 1, the constraint $m_\lambda\left\{\sum_{i=1}^{m}\xi_i x_i \leqslant r\right\} \geqslant \alpha$ is equivalent to $f \leqslant r$, where

$$f = \begin{cases} \frac{\alpha}{\lambda}\left(\sum_{i=1}^{m} r_{i2}x_i - \sum_{i=1}^{m} r_{i1}x_i\right) + \sum_{i=1}^{m} r_{i1}x_i & \text{if } \alpha \leqslant \lambda \\ \frac{1-\alpha}{1-\lambda}\left(\sum_{i=1}^{m} r_{i3}x_i - \sum_{i=1}^{m} r_{i4}x_i\right) + \sum_{i=1}^{m} r_{i4}x_i & \text{if } \alpha > \lambda. \end{cases}$$

Hence the theorem. □

**Lemma 3.** *Let $\xi = (r_1, r_2, r_3, r_4)$ be a trapezoidal fuzzy variable, $\lambda$ be a given degree of optimism and $\alpha$ be a given confidence level. Then the constraint $m_\lambda\{r \leqslant \xi\} \geqslant \alpha$ is equivalent to $r \leqslant g_\alpha$, where*

$$g_\alpha = \begin{cases} \frac{\alpha}{\lambda}(r_3 - r_4) + r_4 & \text{if } \alpha \leqslant \lambda \\ \frac{1-\alpha}{1-\lambda}(r_2 - r_1) + r_1 & \text{if } \alpha > \lambda \end{cases}$$

*and hence $\xi_{\sup}(\lambda; \alpha) = g_\alpha$.*

**Proof.** Similar to Lemma 1. □

**Theorem 4.** *Let $\xi_i = (r_{i1}, r_{i2}, r_{i3}, r_{i4}), i = 1, 2, \ldots, m$ and $\widetilde{D} = (d_1, d_2, d_3, d_4)$ are independent trapezoidal fuzzy numbers. Then the crisp equivalent of Model – II is given by*

$$\begin{aligned} &\min f \\ &\text{s.t. } diam(x) \leqslant \lceil \widehat{D} \rceil \\ &x_i \in \{0,1\}, i = 1, 2, \ldots, m \\ &x \in \Gamma \end{aligned}$$

*where,*

$$\widehat{D} = \begin{cases} \frac{\alpha_1}{\lambda_1}(d_3 - d_4) + d_4 & \text{if } \alpha_1 \leqslant \lambda_1 \\ \frac{1-\alpha_1}{1-\lambda_1}(d_2 - d_1) + d_1 & \text{if } \alpha_1 > \lambda_1 \end{cases} \tag{4}$$

*and $f$ is given by Eq. 3.*

**Proof.** By Lemma 3, the constraint $m_{\lambda_1}\left\{diam(x) \leqslant \widetilde{D}\right\} \geqslant \alpha_1$ is equivalent to $diam(x) \leqslant \widehat{D}$, where

$$\widehat{D} = \begin{cases} \frac{\alpha_1}{\lambda_1}(d_3 - d_4) + d_4 & \text{if } \alpha_1 \leqslant \lambda_1 \\ \frac{1-\alpha_1}{1-\lambda_1}(d_2 - d_1) + d_1 & \text{if } \alpha_1 > \lambda_1. \end{cases}$$

Now since $diam(x)$ cannot be fraction, the result follows with the help of Theorem 2. □

**Lemma 5.** *Let $\xi = (r_1, r_2, r_3, r_4)$ be a trapezoidal fuzzy variable. Then*

$$m_\lambda\{\xi \leqslant \overline{C}\} = \begin{cases} 1 & \text{if } r_4 \leqslant \overline{C} \\ \frac{\lambda r_4 - r_3 + (1-\lambda)\overline{C}}{r_4 - r_3} & \text{if } r_3 \leqslant \overline{C} \leqslant r_4 \\ \lambda & \text{if } r_2 \leqslant \overline{C} \leqslant r_3 \\ \lambda \frac{\overline{C} - r_1}{r_2 - r_1} & \text{if } r_1 \leqslant \overline{C} \leqslant r_2 \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

**Proof.** We have, $m_\lambda\{\xi \leqslant \overline{C}\} = \lambda \mathrm{Pos}\{\xi \leqslant \overline{C}\} + (1-\lambda)\mathrm{Nec}\{\xi \leqslant \overline{C}\}$

Now, $\mathrm{Pos}\{\xi \leqslant \overline{C}\} = \sup_{x \leqslant \overline{C}} \mu_\xi(x)$

$$= \begin{cases} 1 & \text{if } r_4 \leqslant \overline{C} \\ 1 & \text{if } r_3 \leqslant \overline{C} \leqslant r_4 \\ 1 & \text{if } r_2 \leqslant \overline{C} \leqslant r_3 \\ \frac{\overline{C} - r_1}{r_2 - r_1} & \text{if } r_1 \leqslant \overline{C} \leqslant r_2 \\ 0 & \text{otherwise,} \end{cases}$$

and $\mathrm{Nec}\{\xi \leqslant \overline{C}\} = 1 - \mathrm{Pos}\{\xi > \overline{C}\} = \sup_{x > \overline{C}} \mu_\xi(x)$.

$$= \begin{cases} 1 & \text{if } r_4 \leqslant \overline{C} \\ 1 - \frac{\overline{C} - r_4}{r_3 - r_4} & \text{if } r_3 \leqslant \overline{C} \leqslant r_4 \\ 0 & \text{if } r_2 \leqslant \overline{C} \leqslant r_3 \\ 0 & \text{if } r_1 \leqslant \overline{C} \leqslant r_2 \\ 0 & \text{otherwise.} \end{cases}$$

Hence the result follows. □

**Theorem 6.** *Let $\xi_i = (r_{i1}, r_{i2}, r_{i3}, r_{i4}), i = 1, 2, \ldots, m$ are independent trapezoidal fuzzy numbers. Then the crisp equivalent of Model – III is given by*

$$\begin{aligned} &\max f \\ &\text{s.t. } diam(x) \leqslant D \\ &x_i \in \{0,1\}, i = 1, 2, \ldots, m \\ &x \in \Gamma \end{aligned}$$

*where, $f$ is given by left hand side expression of Eq. 5 with $r_k = \sum_{i=1}^{m} r_{ik} x_i, k = 1,2,3,4$.*

**Proof.** Since $x_i \geqslant 0$, for all $i = 1,2,\ldots,m$, and $\xi_i = (r_{i1}, r_{i2}, r_{i3}, r_{i4}), i = 1,2,\ldots,m$ are trapezoidal fuzzy numbers, $\sum_{i=1}^{m} \xi_i x_i$ is also a trapezoidal fuzzy number determined by the quadruple $(r_1, r_2, r_3, r_4)$, where $r_k = \sum_{i=1}^{m} r_{ik} x_i, k = 1,2,3,4$.

So, by Lemma 5, the value of the measure $m_\lambda \left\{\sum_{i=1}^{m} \xi_i x_i \leqslant \overline{C}\right\}$ is given by left hand side expression of Eq. 5.

Hence the theorem. □

**Theorem 7.** *Let $\xi_i = (r_{i1}, r_{i2}, r_{i3}, r_{i4}), i = 1,2,\ldots,m$ and $\widetilde{D} = (d_1, d_2, d_3, d_4)$ are independent trapezoidal fuzzy numbers. Then the crisp equivalent of Model – IV is given by*

$$\begin{aligned} &\max f \\ &\text{s.t. } diam(x) \leqslant \lceil \widehat{D} \rceil \\ &x_i \in \{0,1\}, i = 1,2,\ldots,m \\ &x \in \Gamma \end{aligned}$$

*where, $f$ is given by left hand side expression of Eq. 5 with $r_k = \sum_{i=1}^{m} r_{ik}, k = 1,2,3,4$ and $\widehat{D}$ is given by Eq. 4.*

**Proof.** Equivalent expression of the objective function is obtained in Theorem 6 and that of the constraint is obtained in Theorem 4. Combining them yields the result. □

**Note:** A trapezoidal fuzzy number $\xi = (r_1, r_2, r_3, r_4)$ coincides with a triangular fuzzy number when $r_2 = r_3$. Thus, all the above models also work when the fuzzy costs or the fuzzy bound on the diameter are characterized by triangular fuzzy numbers.

## 6. Fuzzy simulation based genetic algorithm approach

Zhou and Gen[26] devised genetic algorithm to solve quadratic MST problem, where as Gao and Lu extended it to solve the FQMST problem. In the following, we further extended it solve the DCFMST problem.

### 6.1. Fuzzy simulation techniques

If the weights of the edges of a graph are characterized by trapezoidal fuzzy numbers, then the different models of the DCFMST problem can be converted to their deterministic equivalents, as shown in the earlier section. But those may not be done so easily, if the weights are other types of fuzzy numbers, e.g., normal fuzzy numbers. Those are also not applicable when the weights are fuzzy numbers of different kinds. In that case, we may apply the fuzzy simulation technique to compute the $m_\lambda$ measure and pessimistic value defined by

$$m_\lambda \left\{ \sum_{i=1}^{m} \xi_i x_i \leqslant \overline{C} \right\} \tag{6}$$

and

$$\inf \left\{ m_\lambda \left\{ \sum_{i=1}^{m} \xi_i x_i \leqslant r \right\} \geqslant \alpha \right\}. \tag{7}$$

The techniques have been adopted from Yang and Iwamura.[21]

**Fuzzy simulation for $m_\lambda$ measure**

1. Randomly generate $v_{ik}$ from the $\varepsilon$-level set of $\xi_i, i = 1,2,...,m$, respectively, where $k = 1,2,...,N$ and $\varepsilon$ is a sufficiently small positive number.
2. Let $v_k = \mu_{\xi_i}(v_{ik})$, where $\mu_{\xi_i}(.)$ is the membership function of $\xi_i$ (as like Eq. 2).
3. Return the value of $L(\overline{C})$ using the following formula:

$$\begin{aligned} L(b) = \lambda &\left( \max_{1 \leqslant i \leqslant N} \left\{ v_k : \sum_{i=1}^{m} v_{ik} \leqslant b \right\} \right) \\ +(1-\lambda) &\left( \min_{1 \leqslant i \leqslant N} \left\{ v_k : \sum_{i=1}^{m} v_{ik} > b \right\} \right). \end{aligned} \tag{8}$$

**Fuzzy simulation for $(\lambda;\alpha)$-pessimistic value**

1. Randomly generate $v_{ik}$ from the $\varepsilon$-level set of $\xi_i, i = 1,2,...,m$, respectively, where $k = 1,2,...,N$ and $\varepsilon$ is a sufficiently small positive number.

2. Let $v_k = \mu_{\xi_i}(v_{ik})$, where $\mu_{\xi_i}(.)$ is the membership function of $\xi_i$ (as like Eq. 2).

3. Find the value of $b$ such that $L(b) \geqslant \alpha$, where $L(b)$ is defined by Eq. 8.

4. Return $b$.

### 6.2. Overview of GA

Starting from an initial pool of possible solutions, generally called as *chromosomes*, a GA *evolves* the fittest candidate solution through *crossover* and *mutation*. Each chromosome is evaluated and is assigned a *fitness measure* by means of the objective function value. In every generation *better chromosomes* (solutions with higher fitness values) have more possibilities to survive and to produce offspring in the next generations. In order to solve the models I – IV, we may design genetic algorithm based on the fuzzy simulation to obtain the approximate optimal solution, in which simulation algorithms may be employed to check the feasibility of solutions and to compute the objective values if it can not be converted to its crisp equivalent. The procedure of the hybrid genetic algorithm is listed below.

**Procedure Fuzzy_simulation_based_hybrid_GA**

1. Initialize *pop_size* chromosomes in which the simulation algorithms may be employed to check the feasibility of solution;

2. Calculate the objective values for all chromosomes by simulation algorithm or analytic method if possible;

3. Compute the fitness of each chromosome according to the objective value;

4. Select the chromosomes by spinning the roulette wheel;

5. Update the chromosomes by crossover and mutation operations in which the feasibility of offspring may be checked by simulation algorithm or analytic method;

6. Repeat Step 2 to Step 5 for a given number of cycles;

7. Output the best chromosome as the approximate optimal solution.

#### 6.2.1. Chromosome representation

One of the main tasks to devise a genetic algorithm is to represent each of the possible solution as chromosomes, which are usually strings of numbers. For the DCFMST problem, each of the possible solution is a spanning tree. Prüfer[27] showed that every spanning tree of a complete graph with $n$ vertices, which are labeled as $\{1, 2, \dots, n\}$ can be represented by a unique sequence of $n-2$ integers from the set $\{1, 2, \dots, n\}$. But, one drawback of the Prüfer code is its lack of structure for directly determining the diameter or the center(s) from the code without constructing the tree. Neville proposed three methods for encoding trees.[28] Neville's second method of encoding a tree leads to a simple algorithm for computing the diameter of a tree directly from the code. However, there are no known linear time algorithm for encoding or decoding a tree using Neville's second method. Deo and Micikevicious[22] proposed a method to obtain a Prüfer like code of a tree from which the diameter of the tree can be computed directly, as well as, the encoding and decoding can be done in liner time. In the proposed genetic algorithm, we use this encoding procedure to represent a spanning tree. The said procedures for encoding, for decoding and for computation of diameter of a spanning tree are given below.

**Procedure Encoding**
**Input:** A tree of size $n$.
**Output:** $P$, a code of length $n-2$.

1. Sort the leaf nodes in ascending order of their labels.

2. In a list $P$, record the parent nodes of the sorted list of leaf nodes.

3. Discard the leaves.

4. In $P$, append the parent nodes of the new leaves in the order in which they become leaves.

5. Repeat steps 3 and 4 until the number of elements in $P$ is $n-2$. At this stage, a single edge will be left out.

**Procedure Decoding**
**Input:** $P$, a code of $n-2$ integers from $\{1,2,\ldots,n\}, n>2$.
**Output:** A tree of size $n$.

1. Scan the given list from right to left. Push the elements one by one in a stack $S$, skipping those which have already appeared once.
2. Find $\overline{P}$, the list of integers from $\{1,2,\ldots,n\}$ which are not in $P$, sorted in descending order. Actually, $\overline{P}$ is the list of leaf nodes at first stage.
3. Push the elements of $\overline{P}$ one by one in the stack $S$.
4. Pop the element, say $i$, from the top of the stack $S$. Let $j$ be the left most integer in the list $P$. Add the edge from $i$ to $j$ and remove $j$ from $P$.
5. Repeat step 4 until $P$ is exhausted. Still there will be two elements, say $r$ and $s$, in the stack $S$. Add an edge between $r$ and $s$ to obtain a tree with $n-1$ edges.

**Procedure Diameter**
**Input:** $P$, a code of $n-2$ integers from $\{1,2,\ldots,n\}, n>2$.
**Output:** Diameter of the tree represented by $P$.

1. Find $\overline{P}$, the list of integers from $\{1,2,\ldots,n\}$ which are not in $P$. Then $|\overline{P}|$ is the number of leaves in the first stage.
2. Construct an array of $n-2$ elements, called *last*, whose $i$th element is 1, if $P[i]$ does not appear in $P$ furthermore, and 0, otherwise. This can be done in linear time, by scanning the list $P$ from right to left.
3. Let $\ell_k$ be the number of leaf nodes in the $k$th stage, and $L_k$ be the total number of leaf nodes up to $k$th stage, i.e., $L_k = \sum_{i=0}^{k} \ell_i$, provided $L_0 = \ell_0 = 0$ and $L_1 = \ell_1 = |\overline{P}|$. Since the number of leaves of any non-trivial tree is not less than two, $\ell_k \geqslant 2$ for $k \geqslant 1$ if the tree remaining after the deletion of leaves in the previous stage is not a star, i.e., $L_{k-1} \neq n-1$. It is also obvious that, once the number of leaves at any stage becomes 2, the remaining tree is only a path. Now, we find the $\ell_k$'s and $L_k$'s recursively by the relations $\ell_k = \sum_{i=L_{k-2}+1}^{L_{k-1}} last[i]$ and $L_k = \sum_{i=0}^{k} \ell_i$ until either $\ell_k = 2$, or $L_k = n-1$. Find $m = k-1$, where $k$ is the first index for which either $\ell_k = 2$, or $L_m = n-1$. So, after the $m$th stage, a path of length $d = n - L_m - 1$ is left out.
4. The diameter of the tree is given by $2m+d$.

### 6.2.2. *Crossover and mutation operation*

Crossover and mutation operation are two distinguishing features of a GA. The diversity of the population is preserved through these operations. Hence the population have a great chance to be evolved to the optimal solution. In crossover, new chromosomes are created by exchanging information among a pair of strings called parents, chosen at random with a given probability $p_c$ from the mating pool. The resulting strings are known as offspring. Since any string of $n-2$ integers always represent a labeled tree, we apply the simple one point crossover, i.e., we just exchange their digits at randomly selected positions.

Mutation operator randomly changes a digit in the string with the mutation probability $p_m$. Actually, mutation insures the population against a permanent fixation at any particular digit. For a chromosome to mutate, we randomly select a position and randomly generate an integer between 1 and $n$, both inclusive, to replace the original one.

### 6.2.3. Evaluation and selection process

Evaluation and selection process play important roles in genetic algorithm. The GA most often requires a fitness function to evaluate and to assign a score called the fitness to each chromosome in the current population. The fitness of a chromosome depends on how well that chromosome solves the problem under consideration. Selection or reproduction operator is used to select good strings in a population and to form the mating pool. There are several selection operators available in the literature, but the central concept in all of them is to pick strings with above average fitness from the current population and they get copied in the mating pool in a stochastic manner. In our genetic algorithm approach for DCFMST problem, the evaluation perform the following functions: (i) decoding all the chromosomes and calculating their $(\lambda,\alpha)$-pessimistic cost and chance function in terms of $m_\lambda$ measure; (ii) assigning each chromosome a fitness by a rank-based method according to its objective value. Then in the selection process, by spinning the roulette wheel *pop_size* times, we get a new population to go further.

### 6.2.4. Illustration

Now we illustrate the coding scheme with an example. Let $G$ be the graph with 10 vertices $1,2,\ldots,10$, shown in Figure 1(a). The edges are enumerated as in the order of adjacency list, i.e., $e_1=(1,2), e_2=(1,3), e_3=(2,3), e_4=(2,4), e_5=(2,5), e_6=(3,6), e_7=(3,7), e_8=(4,5), e_9=(4,8), e_{10}=(5,6), e_{11}=(5,8), e_{12}=(6,7), e_{13}=(6,9), e_{14}=(7,9), e_{15}=(8,9), e_{16}=(8,10), e_{17}=(9,10)$ and the corresponding weights are assigned as $\xi_1=(4.5,5,5.2,5.5)$, $\xi_2=(1.8,2.1,2.2,2.5)$, $\xi_3=(0.8,1,1.2,1.5)$, $\xi_4=(3,3,3.5,3.8)$, $\xi_5=(0.8,0.9,1,1.2)$, $\xi_6=(4.3,4.4,4.5,4.8)$, $\xi_7=(1.9,2,2,2.2)$, $\xi_8=(8.2,8.3,8.3,8.5)$, $\xi_9=(4,4.5,4.8,5)$, $\xi_{10}=(2.2,2.3,2.5,2.5)$, $\xi_{11}=(2.7,2.8,3,3.2)$, $\xi_{12}=(2.8,3,3.2,3.5)$, $\xi_{13}=(1.9,2,2.2,2.4)$, $\xi_{14}=(4.8,5,5,\ 5.3)$, $\xi_{15}=(1,1.1,1.2,1.5)$, $\xi_{16}=(2,2.2,2.3,2.5)$, $\xi_{17}=(1.1,1.2,1.2,1.4)$.

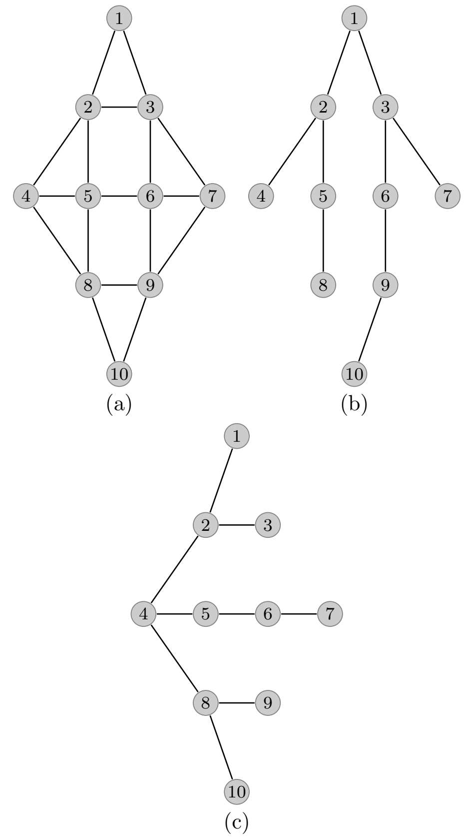


Fig. 1. (a) Graph $G$; (b) Spanning tree $x$; (c) Spanning tree $x'$.

Let us consider the spanning trees $x$, shown in Fig. 1(b), i.e., consisting of the edges $e_1,e_2,e_4,e_5,$ $e_6,e_7,e_{11},e_{13}$ and $e_{17}$. So, the binary decision variable corresponding to the spanning tree $x$, as defined in Section 3, is given by $x=(1,1,0,1,1,1,1,0,0,$ $0,1,0,1,0,0,0,1)$ and its cost is given by $C(x,\xi)=(22,23.4,24.8,26)$. Now we find the code $P$ of the spanning tree $x$ using the **Procedure Encoding** as below.

1. The leaf nodes are sorted in ascending order of their labels as $4,7,8,10$.
2. In $P$, record the parent nodes $2,3,5,9$ of the sorted list of leaf nodes $\{4,7,8,10\}$, i.e., $P=\{2,3,5,9\}$.
3. Discard the leaves $4,7,8,10$. Now, deletion of 4 and 7 create no new leaves, and 5, 9 become leaves when 8, 10 are deleted, respectively.

4. In $P$, append the nodes 2, 6, the parents of 5, 9, i.e., $P = \{2,3,5,9,2,6\}$.

5. Discard the leaves $5,9$. The nodes 2, 6 become leaves when 5, 9 are deleted, respectively.

6. In $P$, append the nodes 1, 3, the parents of 2, 6, i.e., $P = \{2,3,5,9,2,6,1,3\}$.

7. Discard the leaves $2,6$. Only the edge $e_2$ is left out now. So,the required code of the spanning tree $x$ is $P = \{2,3,5,9,2,6,1,3\}$.

Now we illustrate how the diameter of the tree $x$ can be found from its code $P = \{2,3,5,9,2,6,1,3\}$ directly, using the **Procedure Diameter**.

1. Here, $\overline{P} = \{4,7,8,10\}$ is the list of integers from $\{1,2,\ldots,10\}$ which are not in $P$. Then $L_1 = \ell_1 = |\overline{P}| = 4$.

2. The array *last* can be found as $last = \{0,0,1,1,1,1,1,1\}$.

3. Now $L_0 = \ell_0 = 0$ and $L_1 = \ell_1 = 4$. So, $\ell_2 = \sum_{i=L_0+1}^{L_1} last[i] = \sum_{i=1}^{4} last[i] = 2$, i.e., only after the removal of leaf nodes at the first stage, we are left with a path of length $10 - L_1 - 1 = 5$.

4. Therefore, the diameter of the tree $x$ is $2+5 = 7$.

Again let $P = \{2,2,6,8,8,4,5,4\}$ be a given code of a spanning tree of the graph $G$, and we wish to decode it, using **Procedure Decoding**.

1. The stack $S$ is formed by the distinct elements of $P$, scanning from right to left, as below.

| |
|---|
| 2 |
| 6 |
| 8 |
| 5 |
| 4 |
| $S$ |

2. So, $\overline{P} = \{1,3,7,9,10\}$ is the list of integers from $\{1,2,\ldots,10\}$ which are not in $P$.

3. Push the elements of $\overline{P}$ one by one in the stack $S$.

   Then $S$ becomes

| |
|---|
| 1 |
| 3 |
| 7 |
| 9 |
| 10 |
| 2 |
| 6 |
| 8 |
| 5 |
| 4 |
| $S$ |

4. Pop the element 1 from the top of the stack $S$ and 2 is the left most element of $P$. Add the edge from 1 to 2, i.e., $e_1$ and remove 2 from $P$.

5. Now 3 is on the top of the stack $S$ and 2 is the left most element of $P$. Add the edge from 3 to 2, i.e., $e_3$ and remove 2 from $P$. In this way, we add the edges from 7 to 6, i.e., $e_{12}$, 9 to 8, i.e., $e_{15}$, 10 to 8, i.e., $e_{16}$, 2 to 4, i.e., $e_4$, 6 to 5, i.e., $e_{10}$, 8 to 4, i.e., $e_9$.

6. Still there are two elements, namely, 4 and 5 are in the stack $S$. Add an edge between 4 and 5, i.e., $e_8$. The resulted tree is the spanning tree $x'$ of the graph $G$, as shown in Figure 1(c).

Clearly, the binary decision variable corresponding to $x'$ is $\{1,0,1,1,0,0,0,1,1,\ 1,0,1,0,0,1,1,0\}$ and its cost is $C(x',\xi) = (28.5,30.4,32.2,34.3)$. Also, applying the **procedure Diameter**, the diameter of $x'$ can be found as 5.

## 7. Conclusion

Introduction of the diameter constraint makes the fuzzy minimum spanning tree problem harder from its unconstrained counterpart. Chance-constrained programming and dependent-chance programming techniques are used to formulate the problem. Also we have used the $m_\lambda$ measure, a generalization of credibility measure. Finally, a fuzzy simulation based hybrid genetic algorithm is designed to solve the proposed models using Prüfer like code representing labeled trees. Fuzzy quadratic minimum spanning tree problem can also be extended to its constrained version, in the same manner.


## Acknowledgement

The authors are grateful to an anonymous reviewer for pointing out some mistakes and for providing fruitful suggestions.


## References


1. J.B. Kruskal, Jr., "On the shortest spanning subtree of a graph and travelling salesman problem," *Proceedings of AMS*, **7**, 45–50 (1996).
2. R.C. Prim, "Shortest connection networks and some generalizations," *Bell System Technical Journal*, **36**, 1389–1401 (1957).
3. E.W. Dijkstra, "A note on two problems in connection with graphs," *Numerische Mathematik*, **1**, 269–271 (1959).
4. N. Chritofides, *Graph Theory: An Algorithmic Approach*, Academic Press, New York, 1975.
5. J.A. Bondy, U.S.R. Murty, *Graph Theory with Applications*, MacMillan and Elsevier, London and New York, 1976.
6. H.N. Gabow, Z. Galil, T. Spencer, R.E. Tarjan, "Efficint algorithms for finding minimum spanning trees in undirected and directed grphs," *Combinatorica*, **6**, 109–122 (1986).
7. M.R. Garey, D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, W.H. Freeman, New York, 1979.
8. L. Gouveia, T.L. Magnanti, "Modelling and solving the diamemter-constrained minimum spanning tree problem," *Technical Report*, DEIO-CIO, Faculdade de Ciencias, 2000.
9. A. Bookstein, S.T. Klein, "Compression of correlated bivectors," *Information Systems*, **16**, 110–118 (2001).
10. K. Raymond, "A tree-based algorithm for distributed mutual exclusion, ACM transactions on Computers", **7**, 61–77 (1989).
11. N. Deo, A. Abdalla, "Computing a diameter-constrained minimum spanning tree in parallel," In *G. Bongiovanni, G. Gambosi, R. Petreschi ed. Algorithms and Complexity*, **1767**, 17–31 (2000).
12. N.R. Achuthan, L. Caccetta, P.A. Caccetta, and J.F. Geelen, 'Computational methods for the diameter restricted minimum weight spanning tree problem," *Australasian Journal of Combinatorics*, **10**, 51–71 (1994).
13. G.Y. Handler, "Minimax location of a facility in an undirected graph," *Tranportation Science*, **7**, 287–293 (1978).
14. A.C. dos Santos, A. Lucena, C.C. Ribeiro, "Solving diameter contrained minimum spanning tree problems in dense graphs," *Technical Report*, PUC-RioInf.MCC13, 2004.
15. T. Itoh, H. Ishii, "An approach based on necessity measure to the spanning tree problem," *Journal of the Operations Research Society of Japan*, **39**, 247–257 (1996).
16. P.-T. Chang, E.S. Lee, "Ranking fuzzy sets on the concept of existence," *Computer Mathematics with Applications*, **27**, 1–21 (1994).
17. P.-T. Chang, E.S. Lee, "Fuzzy decision networks and deconvolution," *Computer Mathematics with Applications*, **37**, 53–63 (1999).
18. B. Liu, *Theory and Practice of Uncertain Programming*, Physica-Verlag, Heidelberg, 2002.
19. B. Liu, *Uncertainty Theory: An introduction to its axiomatic foundations*, Springer-Verlag, Berlin, 2004.
20. J. Gao, M. Lu, "Fuzzy quadratic minimum spanning tree problem," *Applied Mathematics and Computation*, **164**, 773–788 (2005).
21. L. Yang, K. Iwamura, "Fuzzy chance-constrained programming with linear combination of possibility measure and necessity measure," *Applied Mathematical Sciences*, **2**, 2271–2288 (2008).
22. N. Deo, P. Micikevicius, "A new encoding for labeled trees employing a stack and a queue," *Bulletin of the ICA*, **34**, 77–85 (2002).
23. L.A. Zadeh, "Fuzzy sets as a basis for a theory of possibility", *Fuzzy Sets and Systems*, **1**, 3–28 (1978).
24. D. Dubois, H. Prade, *Possibility Theory*, Plenum, New York, 1988.
25. A. Charnes, W.W. Cooper, "Chance-constrained programming," *Management Science*, **6**, 73–79 (1959).
26. G. Zhou, M. Gen, "A note on genetic algorithms for degree-constrained spanning tree problems," *Networks*, **30**, 91–95 (1997).
27. H. Prüfer, "Nuuer Beweis Elines Satzes über Permutation," *Archives of Mathematical Physics*, **27**, 742–744 (1918).
28. E. H. Neville, "The codifying of tree-structure," *Proceedings of Cambridge Philosophical Society*, **49**, 381–385 (1953).