数理解析研究所講究録 1457 巻 2005 年 107-114

A Tabu Search Algorithm for Fuzzy Random Minimum Spanning Tree Problems

広島大学大学院・工学研究科　片桐 英樹 (Hideki KATAGIRI)
坂和 正敏 (Masatoshi SAKAWA)
Graduate School of Engineering
Hiroshima University

1 Introduction

The Minimum Spanning Tree (MST) problem is to find a least cost spanning tree in an edge weighted graph. The efficient polynomial-time algorithms to solve MST problems have been developed by [11] and [14]. In the real world, MST problems are usually seen in network optimization. For instance, when designing a layout for telecommunication systems, if a decision maker wishes to minimize the cost for connection between cities, it is formulated as an MST problem. As other examples, the objective is to minimize the time for construction or to maximize the reliability.

Most research papers with respect to MST problems have been dealt with the case where each weight is constant. However, in order to investigate more realistic cases, it is necessary to consider the situation that one makes a decision on the basis of data involving randomness and fuzziness simultaneously. For instance, the cost for connection or construction often depends on economical environments which vary randomly, and experts often estimate the value of the cost not as a constant but as an ambiguous value. In order to take account of such situations, we shall deal with minimum spanning tree problems where each edge weight is a fuzzy random variable. We call it a Fuzzy Random Minimum Spanning Tree (FRMST) problem.

A fuzzy random variable was first defined by [12]. Recently, some researchers [9, 15] considered fuzzy random linear programming problems. We could take various approaches to an FRMST problem according to the interpretations of the problem.

In this paper, we take a possibilistic and stochastic programming approach, which is based on the idea provided in [9]. As it is shown later, the formulated problem is transformed into the deterministic equivalent nonlinear maximum ratio spanning tree problem, which is generally an NP-hard problem.

In order to obtain an approximate optimal solution, we employ a solution method through TS [7, 8].

2 MST problem with fuzzy random edge costs

Consider a connected undirected graph $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a finite set of vertices representing terminals or telecommunication stations etc., and $E =$

$\{e_1, e_2, \ldots, e_m\}$ is a finite set of edges representing connections between these terminals or stations. Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_m)$ be a vector defined by

$$x_i = \begin{cases} 1 & \text{if edge } e_i \text{ is selected} \\ 0 & \text{otherwise.} \end{cases}$$

In the present paper, we consider a minimum spanning tree problem involving fuzzy random weights as follows:

$$\left.\begin{array}{ll} \min & \tilde{\tilde{\boldsymbol{C}}}\boldsymbol{x} \\ \text{s. t.} & \boldsymbol{x} \in X, \end{array}\right\} \tag{1}$$

where $\boldsymbol{x} = (x_1, \ldots, x_m)^t$ is a decision variable column vector, $\tilde{\tilde{\boldsymbol{C}}} = (\tilde{\tilde{C}}_1, \ldots, \tilde{\tilde{C}}_m)$ is a coefficient vector and X stands for the collection of m-dimensional 0-1 vectors corresponding to all possible spanning trees of the given graph $\mathcal{G}$. Each $\tilde{\tilde{C}}_j$ is a fuzzy random variable taking a fuzzy number $\tilde{\tilde{C}}_j(\omega)$ as a realization for each ω where ω is an elementary event of the universal event Ω. The following is the membership function characterizing $\tilde{\tilde{C}}_j(\omega)$:

$$\mu_{\tilde{\tilde{C}}_j(\omega)}(t) = \begin{cases} L\left(\dfrac{\bar{d}_j(\omega) - t}{\alpha_j}\right) & (t \le \bar{d}_j(\omega),\ \forall\omega) \\ R\left(\dfrac{t - \bar{d}_j(\omega)}{\beta_j}\right) & (t > \bar{d}_j(\omega),\ \forall\omega), \end{cases}$$

where $L(t) \triangleq \max\{0,\ l(t)\}$ and $R(t) \triangleq \max\{0,\ r(t)\}$. The functions $l(t)$ and $r(t)$ are strictly decreasing functions on $[0, +\infty)$, satisfying (i) $l(0) = r(0) = 1$, (ii) $l(1) = r(1) = 0$, (iii) $l(t) = l(-t)$, $r(t) = r(-t)$ for any t.

The parameters $\bar{d}_j$, $j = 1, \ldots, m$ are normal random variables with mean m_j^d and the variance-covariance matrix V

By applying the calculation formula [5] with respect to L - R fuzzy numbers based on the extension principle [16] to the fuzzy number $\tilde{\tilde{Y}}(\omega)$ for each ω, it is easily shown that $\tilde{\tilde{Y}}$ is a fuzzy random variable with the following membership function:

$$\mu_{\tilde{\tilde{Y}}}(y) = \begin{cases} L\left(\dfrac{\bar{\boldsymbol{d}}\boldsymbol{x} - y}{\boldsymbol{\alpha x}}\right) & (y \le \bar{\boldsymbol{d}}\boldsymbol{x}) \\ R\left(\dfrac{y - \bar{\boldsymbol{d}}\boldsymbol{x}}{\boldsymbol{\beta x}}\right) & (y > \bar{\boldsymbol{d}}\boldsymbol{x}) \end{cases}$$

3 Possibilistic programming approach

In problem (1), the total edge weights represented by a fuzzy random variable cannot be minimized in the deterministic sense. Katagiri et al. [9] considered a linear programming problem where the right-hand side of a constraint is a fuzzy random variable. They first introduced a possibilistic and stochastic programming approach to fuzzy random programming problems by noting that the degree of possibility that the constraint is satisfied varies randomly. In this paper, we shall develop the idea to the case where the coefficients of an objective function are fuzzy random variables. Considering the

vagueness of the decision maker's judgment, the fuzzy goal $\tilde{G}$ such that the objective function value is substantially smaller than some value is introduced. If a decision maker wishes to maximize a degree of possibility that the fuzzy goal is satisfied, problem (1) is reformulated as the following problem:

$$\left.\begin{array}{ll} \max & \Pi_{\tilde{Y}}(\tilde{G}) \\ \text{s. t.} & \boldsymbol{x} \in X. \end{array}\right\} \tag{2}$$

where

$$\Pi_{\tilde{Y}}(\tilde{G}) = \sup_{y} \min \left\{\mu_{\tilde{Y}}(y), \mu_{\tilde{G}}(y)\right\}. \tag{3}$$

Since the value of $\Pi_{\tilde{Y}}(\tilde{G})$ varies randomly due to the randomness of $\mu_{\tilde{Y}}(y)$, we regard problem (2) as a stochastic programming problem. In stochastic programming [1, 4] introduced two-stage problems [3] introduced several stochastic programming model such as the expected optimization model, the variance minimization model and the probability maximization model. Another stochastic programming model, which is to optimize a satisficing level under the condition that the objective function value is better than the satisficing level, are considered [10] and [6].

In this paper, we consider the following problem which is based on the probability maximization model:

$$\left.\begin{array}{ll} \max & Pr\left(\omega \middle| \Pi_{\tilde{Y}(\omega)}(\tilde{G}) \geq h\right) \\ \text{s. t.} & \boldsymbol{x} \in X, \end{array}\right\} \tag{4}$$

where h is a satisficing level for the degree of possibility with respect to a fuzzy goal. This problem is to maximize the probability that the degree of possibility is greater than or equal to a satisficing level h.

For any elementary event, $\Pi_{\tilde{Y}(\omega)}(\tilde{G}) \geq h$ is transformed as follows:

$$\begin{array}{ll} & \sup_{y} \min \left\{\mu_{\tilde{Y}(\omega)}(y),\ \mu_{\tilde{G}}(y)\right\} \geq h \\ \Longleftrightarrow & \exists\, y \leq \bar{\boldsymbol{d}}(w)\boldsymbol{x} : \{\bar{\boldsymbol{d}}(\omega) - L^*(h)\boldsymbol{\alpha}\}\boldsymbol{x} \leq y \leq \mu^*_{\tilde{G}}(h) \\ \Longleftrightarrow & \{\bar{\boldsymbol{d}}(\omega) - L^*(h)\boldsymbol{\alpha}\}\boldsymbol{x} \leq \mu^*_{\tilde{G}}(h), \end{array} \tag{5}$$

provided $h > 0$ where $L^*(h)$ and $\mu^*_{\tilde{G}}(h)$ are pseudo inverse functions defined as follows:

$$\begin{array}{rcl} L^*(h) & = & \sup\{r | L(r) > h,\ r \geq 0\} \\ \mu^*_{\tilde{G}}(h) & = & \sup\{r | \mu_{\tilde{G}}(r) \geq h\}. \end{array}$$

Subtract the following expectation $E\left(\bar{\boldsymbol{d}}(\omega)\boldsymbol{x}\right)$ from both sides of the inequality (5) and divide all by this deviation $\sqrt{Var\left(\bar{\boldsymbol{d}}(\omega)\boldsymbol{x}\right)}$, where Var denotes the variance, then we get

$$\begin{array}{ll} & Pr\left(\Pi_{\tilde{Y}}(\tilde{G}) \geq h\right) \\ \Longleftrightarrow & Pr\left(\dfrac{\left\{\bar{\boldsymbol{d}} - L^*(h)\boldsymbol{\alpha}\right\}\boldsymbol{x} - \left\{\boldsymbol{m}^d - L^*(h)\boldsymbol{\alpha}\right\}\boldsymbol{x}}{\sqrt{\boldsymbol{x}^t V \boldsymbol{x}}} \leq \dfrac{-\left\{\boldsymbol{m}^d - L^*(h)\boldsymbol{\alpha}\right\}\boldsymbol{x} + \mu^*_{\tilde{G}}(h)}{\sqrt{\boldsymbol{x}^t V \boldsymbol{x}}}\right). \end{array}$$

By noting that the left-hand side of the above inequality becomes a standard normal random variable, problem (4) is transformed into the following deterministic equivalent problem:

$$\left.\begin{array}{ll} \max & \dfrac{-\{\boldsymbol{m}^d - L^*(h)\boldsymbol{\alpha}\}\boldsymbol{x} + \mu^*_{\tilde{G}}(h)}{\sqrt{\boldsymbol{x}^t V \boldsymbol{x}}} \\ \text{s. t.} & \boldsymbol{x} \in X \end{array}\right\} \tag{6}$$

4 Tabu search algorithm

In this section, we shall construct a solution algorithm based on TS incorporating strategic oscillation and path-relinking method. Starting from an initial spanning tree, the improvement strategy, which consists of exchanging a pair of edges, generates the neighborhood of the current solution. In order to prevent cycling between the same solutions, certain exchanges which are call "moves" can be forbidden, earning them the status of "tabu move". The set of tabu moves defines the tabu list. Tabu moves are not permanent; a short-term memory function enables them to leave the tabu list. The use of aspiration criterion permits certain moves on the tabu list to overcome any tabu status. Strategic oscillation was originally introduced to provide an effective interplay between intensification and diversification over the intermediate to long term. In the proposed algorithm, we used strategic oscillation to intensively explore the region around the current neighborhood. In addition to a short-term memory, we use a residence frequency memory as a long-term memory. A diversification procedure, using the residence frequency memory function, will lead to the exploration of region of the solution space not previously visited. On the other hand, an intensification produce undertakes to create solutions aggressively encouraging the incorporating of solutions from an elite solution set. The process goes on until the termination criterion is satisfied.

We recall that problems (6) can be written on the following form:

$$\left.\begin{array}{l} \max z(\boldsymbol{x}) = \dfrac{\boldsymbol{a}\boldsymbol{x} + g}{\sqrt{\boldsymbol{x}^t V \boldsymbol{x}}} \\ \text{s. t. } \boldsymbol{x} \in X, \end{array}\right\} \tag{7}$$

Let $\boldsymbol{x}^c$ be the current solution and T^c its corresponding spanning tree, and let $\boldsymbol{x}^b$ and T^b be the best solution and its corresponding spanning tree, respectively. We denote by $z(\boldsymbol{x})$ the objective function value of (6) at an element $\boldsymbol{x}$. Then, a TS algorithm for solving fuzzy random minimum spanning tree problem is given as follows. In the proposed algorithm, we use the following parameters:

NISmall: Number of iterations in the small depth procedure.
MAX_Small: Threshold of the counter *NISmall*.
NILarge: Number of iterations in the large depth procedure.
MAX_Large: Threshold of the counter *NISmall*.
UNRIter: Counts the iterations where the best solution is unrenewed.
MAX_Iter: Threshold of the counter *UNRIter*.
Max_k: Threshold of the counter, k, of the oscillation strategy.

Step 0 (Initial solution)
Set $NISmall = NILarge = UNRIter = k = 0$. Generate an initial solution $\boldsymbol{x}^0$ corresponding to an initial spanning tree T^0 Set $\boldsymbol{x}^c := \boldsymbol{x}^0$ and $\boldsymbol{x}^b := \boldsymbol{x}^0$.

Step 1 (Improvement)
Improve the obtained solution by the *improvement strategy.* Set $\boldsymbol{x}^b := \boldsymbol{x}^c$.

Step 2 (Strategic oscillation, small depth
Set $k := k + 1$. If $NISmall > MAX\_Small$, then go to step 4. Otherwise, add a_1 edges among $N(T^c)$ by using the *edge addition rule* and continue to remove one of the edges in a cycle by using the *edge remove rule* until a spanning tree is formed.

Step 3 If $z(\boldsymbol{x}^c) > z(\boldsymbol{x}^b)$, then set $NISmall = 0$ and return to step 2. Otherwise, set $NISmall = NISmall + 1$ and return to step 2.

Step 4 (Strategic oscillation, large depth)
If $NILarge > MAX\_Large$, then go to step 6. Otherwise, add a_2 $(a_1 < a_2)$ edges among $N(T^c)$ by the edge addition rule and continue to remove one of the edges in a cycle by the edge remove rule until a spanning tree is formed. Improve the current solution by the improvement strategy.

Step 5 If $z(\boldsymbol{x}^c) > z(\boldsymbol{x}^b)$, then set $NILarge = NISmall = 0$ and return to step 2. Otherwise, set $NILarge = NILarge + 1$ and return to step 4.

Step 6 If $k > Max\_k$, then go to step 8. Otherwise, go to step 7.

Step 7 (Diversification)
Remove a_3 edges in T^c that are resident for a long time in spanning trees. Slap a long tabu tenure to the removed edges. Add the edges whose resident time are short so as not to make a cycle until a spanning tree is formed. Return to step 1.

Step 8 (Intensification by elite solutions)
Set $k = 0$. Construct a set of connected components by adding edges that are in most of elite solutions. Add edges, except for the edges that are not in most of elite solutions, by the *edge addition rule* until a spanning tree is formed. Improve the obtained solution by the improvement strategy. If $z(\boldsymbol{x}^c) > z(\boldsymbol{x}^b)$, then set $\boldsymbol{x}^b := \boldsymbol{x}^c$, $k = UNRIter = 0$ and return to step 2. Otherwise, set $UNRIter = UNRIter + 1$ and go to step 9.

Step 9 (Path relinking method)
If $UNRIter > Max\_Iter$, then terminate. Otherwise, generate an initial solution by the path relinking method and return to step 1.

The essential features that have been considered in building a TS algorithm for solving a fuzzy random minimum spanning tree problem are: generating an initial solution, the neighborhood structure, the improvement strategies, short-term and long-term memories, oscillation strategy, intensification by an elite solution set, diversification procedure, termination criterion. The detail of those features is as follows:

Initial solution
Let $SCC(i)$ denote a Set of Connected Component that consists of i edges. To construct a spanning tree T, first, an edge $e \in E$ is chosen uniformly at random. With this edge, a subtree $SCC(1)$ which consists of only one edge is created. Then, a set of connected component $SCC(k + 1)$ is constructed by adding an edge $e \leftarrow$

argmin$\{z(SCC(k)+e')-z(SCC(k))|e' \in E_{NC}(SCC(k))\}$ to the current $SCC(k)$ under construction, where $E_{NC}(SCC(k))$ is defined as follows:

$$E_{NC}(SCC(k)) \leftarrow \{e \in E | SCC(k)+e \text{ has no cycle}\}.$$

Neighborhood structure

Let T be a set of edges which forms a spanning tree, and let $\mathcal{T}$ be a class of all possible spanning trees in a given graph. The neighborhood $N(T)$ consists of all spanning trees which can be generated by removing an edge $e \in T$ and by adding an edge from the set $E_{NH}(T-e) \setminus \{e\}$, where $E_{NH}(T-e)$ is defined as follows:

$$E_{NH}(T-e) \triangleq \{e' \in E | T-e+e' \in \mathcal{T}\}.$$

Improvement strategy

In order to improve the current solution $\boldsymbol{x}^c$, here are two major improvement strategies. One is a first improvement, which scans the neighborhood $N(T^c)$ of a current spanning tree T^c and chooses the first spanning tree T^f corresponding to the solution $\boldsymbol{x}^f$ such that $z(\boldsymbol{x}^f) > z(\boldsymbol{x}^c)$. The other is a best improvement, which exhaustively explores the neighborhood and returns one of the solutions with the lowest objective function value.

At the beginning, we use the first improvement strategy. If a better solution cannot be found, we switch to the best improvement strategy.

Edge addition rule

Again, we use $SCC(i)$, a set of connected component that consists of i edges, defined in the previous section. Then, by the edge addition rule, $SCC(k+1)$ is constructed by adding an edge $e \leftarrow$ argmin$\{z(SCC(k)+e')-z(SCC(k))|e' \in E_{NC}(SCC(k))\}$ to the current $SCC(k)$ under construction.

Edge remove rule

By the edge remove rule, $SCC(k-1)$ is constructed by removing an edge $e \leftarrow$ argmin$\{z(SCC(k)-e')-z(SCC(k))|e' \in CY(k)\}$ from a cycle in the current $SCC(k)$ under construction, where $CY(k)$ denotes the set of element $e' \in SCC(k)$ which are parts of cycles in the component $SCC(k)$.

Diversification procedure

The diversification procedure begins at the situation that some spanning tree is formed. Then it removes from the spanning tree a_3 edges which have been a part of spanning trees for a long time. Next, a spanning tree is again formed with the edges which have not been added so far by the *edge addition rule*. The diversification derives the search into a new region. Then, the strategic oscillation procedure begins at the new search region. If the strategic oscillation procedure is iterated in Max_k times, then the intensification procedure is started.

Intensification procedure using an elite solution set

The intensification procedure begins at the condition that no edge is selected. First, a connected component is constructed by continuing to selecting the edges that occur frequently in the elite solutions. The selected edges are never removed during

the procedure. After constructing a connected component, the process of adding edges, except ones that are not in most of elite solutions, are continued by the edge addition rule until a spanning tree is formed.

Path relinking method

Path relinking is initiated by selecting two solutions $\boldsymbol{x}'$ and $\boldsymbol{x}''$ from a collection of elite solutions produced during previous search phases. A path is then generated from $\boldsymbol{x}'$ to $\boldsymbol{x}''$, producing a solution sequence $\boldsymbol{x}' = \boldsymbol{x}'(1), \boldsymbol{x}'(2), \ldots, \boldsymbol{x}'(r) = \boldsymbol{x}''$, where $\boldsymbol{x}'(i+1)$ is created from $\boldsymbol{x}'(i)$ at each step by choosing a move that leaves the fewest number of moves remaining to reach $\boldsymbol{x}''$. Finally, once the path is completed, one or more of the solutions $\boldsymbol{x}''(i)$ is selected as a solution to initiate a new search phase.

Aspiration criterion

An aspiration criterion is activated to overcome the tabu status of a move whenever the solution then produced is better than the best historical solution achieved. This criterion will be effective only after a local optimum is reached.

Termination criterion

The counter $UNRIter$ counts the iterations where the best solution T^b is unrenewed. The proposed algorithm terminates if $UNRIter$ is greater than the threshold Max_Iter. The quality of the final solution and the computer running time are both influenced by the termination criterion

5 Conclusion

In this paper, we have considered fuzzy random spanning tree problems. Introducing a fuzzy goal, we formulated the problem to maximize the probability that the degree of possibility or necessity that an objective function satisfies the fuzzy goal. It has been shown that the problem was transformed into the deterministic equivalent nonlinear maximization ratio spanning programming problem. In order to solve the problem, we have constructed a TS algorithm based on oscillation strategy, intensification by a elite solution set and diversification by residence frequency and so on.

In the future, we will try to extend and apply this method to the problems to minimize the variance of the degree of possibility. Since the problems include the constraint with respect to the expected degree of possibility, we need to extend our method in order to deal with a constraint by changing a part of the oscillation strategy.

References

[1] Beale, M.L. 1955. On optimizing a convex function subject to linear inequalities, *Journal of Royal Statistics Society*, vol. 17, 173–184.

[2] Blum, C. and Blesa, M.J. New metaheuristic approaches for the edge-weighed k-cardinality tree problem, *Computer & Operations Research* (to appear).

[3] Charnes A. and Cooper, W.W. 1963. Deterministic equivalents for optimizing and satisficing under chance constraints, *Operations Research*, vol. 11, 18–39.

[4] Dantzig, G.B. 1995. Linear programming under uncertainty, *Management Science*, vol.1, 197–206.

[5] Dubois, D. and Prade, H. 1980. *Fuzzy Sets and Systems*, Academic Press, New York.

[6] Geoffrion, A.M. 1967. stochastic programming with aspiration or fractile criteria, *Management Science*, vol. 13, 672–679.

[7] Glover, F. 1989. Tabu Search-Part I., *ORSA Journal on Computing*, vol.1, 190–206.

[8] Glover, F. 1990. Tabu Search-Part II., *ORSA Journal on Computing*, vol.2, 4–32.

[9] Katagiri, H. and Ishii, H. 2000. Linear programming problem with fuzzy random constraint, *Mathematica Japonica*, vol. 52, 123–129.

[10] Kataoka, S. 1963. A stochastic programming model, *Econometrica*, vol. 31, 181-196.

[11] Kruskal, J.B. Jr. 1956. On the shortest spanning subtree of a graph and traveling salesman problem, *Proceedings of ACM* 7/1, 48–50.

[12] Kwakernaak, H. 1978. Fuzzy random variable-1, *Information Sciences*, vol. 15, 1–29.

[13] Luhandjula, M.K. and Gupta, M.M. 1996. On fuzzy stochastic optimization, *Fuzzy Sets and Systems*, vol. 81, 47–55.

[14] Prim, R.C. 1957. Shortest connection networks and some generations, *Bell System Technical Journal*, vol. 36, 1389–1401.

[15] Wang, G.Y. and Qiao, Z. 1993. Linear programming with fuzzy random variable coefficients, *Fuzzy Sets and Systems*, vol. 57, 295–311.

[16] Zadeh, L.A. 1978. Fuzzy sets as a basis for a theory of possibility, *Fuzzy Sets and Systems*, vol. 1, 3–28.