King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

ORIGINAL ARTICLE

An efficient method to solve least-cost minimum spanning tree (LC-MST) problem

M.R. Hassan

South Valley University, Faculty of Science, Department of Computer Science, Aswan, Egypt

Received 29 June 2011; revised 2 October 2011; accepted 7 December 2011
Available online 19 December 2011

KEYWORDS

Computer networks;
Minimum spanning tree;
LC-MST problem

Abstract In this paper, least-cost minimum spanning tree (LC-MST) problem is defined as a method to construct a minimum cost spanning tree that has the least-cost edges in the network by using the distance (cost) matrix. The paper presents a new algorithm based on the distance matrix to solve the LC-MST problem. The studied cases show that the presented algorithm is efficient to solve the LC-MST problem in less time. Also, the presented algorithm can be modified to solve the DC-MST (Delay Constrained-Minimum Spanning Tree) problem presented by Lee and Atiquzzaman (2007) and the MRCT (Minimum Routing Cost Tree) problem presented by Cambos and Ricardo (2008), given as the applications of the presented algorithm.

1. Introduction

Minimum-cost spanning trees have many applications such as building cable networks that join n locations with minimum cost and building a road network that joins n cities with minimum cost. A tree is a connected graph without cycles. A spanning tree is a subgraph that is a tree and that spans (reaches out) to all vertices of the original graph. Among all the spanning trees of a weighted and connected graph, the one (possibly more) with the least total weight is called a minimum spanning tree (MST).

The problem of identifying a minimum spanning tree (MST) of a connected, undirected graph belongs to the group of classical combinatorial optimization problems, which can be solved by greedy heuristics, i.e. Kruskal or Prim in polynomial time (Graham and Hell, 1985). Recently, there exist several practically relevant variants of MST problem that were proven to be NP-hard. One of these problems is the degree-constrained MST problem (Pagacz et al., 2006). The other is the Bounded Diameter MST (BDMST) problem, which has been formulated by Nghia and Binh (2008) as follows: among spanning trees of G whose diameters do not exceed a given upper bound $k \geqslant 2$, find the spanning tree with the minimal cost (sum of the weights on edges of the trees). The last one is the capacitated minimum spanning tree problem (see Jothi and Raghavachari, 2005; Zhou et al., 2006).

The paper presents a new method to solve the LC-MST problem, i.e. finding the least-cost edges (links) of the network to construct a minimum spanning tree with the least cost. The presented method depends on the distance matrix (cost matrix) to construct a preferred link matrix (least-cost link matrix) and then by executing at most two steps, the preferred link matrix can be used to obtain LC-MST. In the following sections, we will explain the presented method to obtain LC-MST. Also, we will show how to use LC-MST algorithm to solve both the DC-MST problem presented by Lee and Atiquzzaman

E-mail address: m_r_hassan73@yahoo.com

1319-1578 © 2011 King Saud University. Production and hosting by Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.jksuci.2011.12.001

(2007) and MRCT problem presented by Cambos and Ricardo (2008).

The rest of the paper is organized as follows: The formulation and description of the LC-MST problem is given in Section 2. The LC-MST algorithm is given in Section 3. Section 4 presents network examples to explain the use of the algorithm. Time analysis is shown in Section 5. The applications of the LC-MST algorithm are given in Section 6. Section 7 gives the conclusion and future work.

2. Problem formulation and description

In graph theory, a tree T is defined as a connected graph without cycles. The number of edges in T is $n - 1$ (where n is the number of nodes) and any two vertices in T are connected by exactly one path. A spanning tree for a graph $G = (V, E, w)$, where V is the number of vertices (nodes), E is the number of edges (links) and w is the weights assigned to edges, is defined as a subgraph of G that is a tree, contains all vertices of G, and has $|V| - 1$ edges. The total cost for T is given by

$$C_{total}(T) = \sum_{e=1}^{n-1} W_T(e), \quad (1)$$

where $W_T(e)$ represents the weight assigned to edge $e \in E_T$ and E_T defines the set of edges of T.

A minimum spanning tree (MST) represents a spanning tree T^* such that

$$C_{total}(T^*) = \min\left(\sum_{e=1}^{n-1} W_T(e)\right), \quad (2)$$

For all spanning trees, T can be computed from G.

In this paper, we define the least-cost minimum spanning tree (LC-MST) as: for each node i, there are a set of nodes $ni \subset n$ that connect to i by a set of edges ek (links); among these links ek there is at least one link ei^* (preferred link) that has the least cost. Applying this idea to construct the set of links e^* to build the LC-MST, its total cost is given by:

$$\text{LC-MST} = \sum_{e=1}^{n-1} W_T(e^*) \quad (3)$$

Where $W_T(e^*)$ defines the weight of the set of edges e^* (set of preferred links) that connects each node-pair in LC-MST.

Consider the following network example, shown in Fig. 1.

The LC-MST of the network given in Fig. 1 is shown in Fig. 2.

In the following section, we will introduce the idea of our algorithm to find the LC-MST by using the weight (distance) matrix of a given network.

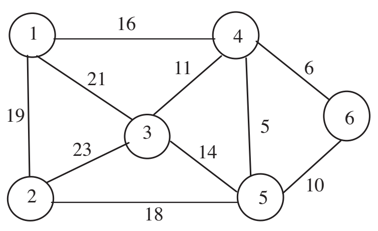

Figure 1 An example network topology.

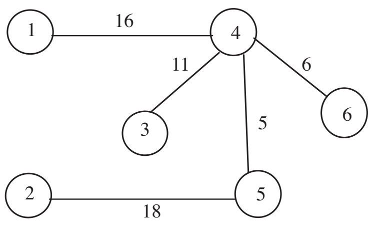

Figure 2 Least-cost minimum spanning tree.

3. The LC-MST algorithm

The aim of the LC-MST algorithm is to find a least-cost tree of a given network. The idea of the algorithm is to read the distance matrix (weight matrix) of a given network and construct a preferred link matrix that contains the set of least-cost links to construct the least-cost minimum spanning tree.

Algorithm: Least-Cost Minimum Spanning Tree

1. Input the distance matrix D = [dij]nxn for the weighted graph $G(V, E)$, where V is the set of vertices and E is the set of edges.
2. For all i, j, find the least-cost element (preferred link) in each column j and set the other elements to zero.
3. Construct the preferred link matrix (PLM) by using step 2.
4. Construct the nodes-set matrix (NSM) by using PLM matrix constructed in step 3 (each element in this matrix contains the node-pairs that correspond to the preferred link in PLM).
5. Combining the node-pairs in step 4 to construct the candidate spanning tree.
6. If there are any duplicating node-pairs, keep one of them, and if there is a set of node-pairs, construct a cycle, remove the one that has the largest cost.
7. Output the least-cost minimum spanning tree.

Note: The algorithm has been implemented by using Borland C+ + 5.0.

4. Numerical examples

4.1. Five nodes example

The following network example is taken from Lee and Atiquzzaman (2007). The computer network and its distance matrix are given in Fig. 3 and Fig. 4, respectively.

The Preferred link matrix PLM (least-cost link matrix) can be constructed from the distance matrix as shown in Fig. 5.

The corresponding nodes-set matrix NSM is constructed as shown in Fig. 6.

The set of node-pairs that construct the candidate spanning tree is:

$$\{(1, 0),\ (0, 1),\ (0, 2),\ (2, 0),\ (2, 3),\ (2, 4)\}$$

Removing the repeated one (both (0,1) and (1,0) are the same), the LC-MSPT is:

$$\{(0, 1),\ (0, 2),\ (2, 3),\ (2, 4)\}$$

The corresponding graph of the obtained spanning tree is shown in Fig. 7.

The cost of the obtained LC-MST is equal to 14.

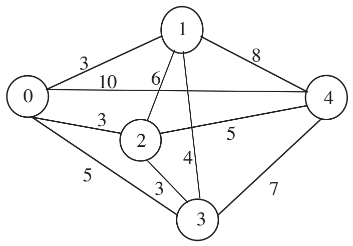

Figure 3 Computer network of Example 4.1.

$$D = \begin{bmatrix} 0 & 3 & 3 & 5 & 10 \\ 3 & 0 & 6 & 4 & 8 \\ 3 & 6 & 0 & 3 & 5 \\ 5 & 4 & 3 & 0 & 7 \\ 10 & 8 & 5 & 7 & 0 \end{bmatrix}$$

Figure 4 Distance matrix.

$$\boldsymbol{PLM} = \begin{bmatrix} 0 & 3 & 3 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 3 & 5 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 5 Preferred link matrix.

$$\boldsymbol{NSM} = \begin{bmatrix} 0 & (0{,}1) & (0{,}2) & 0 & 0 \\ (1{,}0) & 0 & 0 & 0 & 0 \\ (2{,}0) & 0 & 0 & (2{,}3) & (2{,}4) \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 6 Nodes set matrix.

4.2. Another five nodes example

The following network example is taken from Sriram et al. (1998). The computer network and its distance matrix are given in Figs. 8 and 9, respectively.

The Preferred link matrix (least-cost link matrix) can be constructed from the distance matrix as shown in Fig. 10.

The corresponding nodes-set matrix NSM is constructed as shown in Fig. 11.

By combining these node pairs, we get the following set:

$\{(2, 1), (5, 1), (3, 2), (5, 2), (2, 3), (4, 3), (3, 4), (2, 5)\}$

Removing the repeated node-pairs, we get:

$\{(2, 1), (5, 1), (2, 3), (3, 4), (2, 5)\}$

The set of nodes-pair: $\{(2,1), (2,5), (5,1)\}$ perform a cycle.

To solve this problem, we remove the largest one $\{(5,1)\}$ (that has the big cost), then the LC-MST is:

$\{(2, 1), (2, 3), (3, 4), (2, 5)\}$

The corresponding graph of the obtained spanning tree is shown in Fig. 12.

The cost of it is equal to 6.

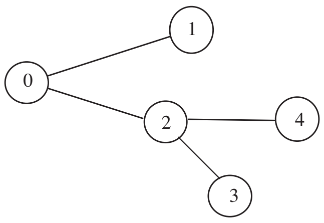

Figure 7 The least-cost minimum spanning tree.

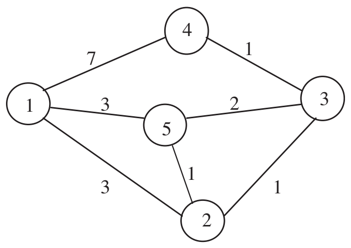

Figure 8 Computer network of Example 4.2.

$$D = \begin{bmatrix} 0 & 3 & 0 & 7 & 3 \\ 3 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 2 \\ 7 & 0 & 1 & 0 & 0 \\ 3 & 1 & 2 & 0 & 0 \end{bmatrix}$$

Figure 9 Distance matrix.

$$\boldsymbol{PLM} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 3 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Figure 10 Preferred link matrix.

$$\boldsymbol{NSM} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ (2{,}1) & 0 & (2{,}3) & 0 & (2{,}5) \\ 0 & (3{,}2) & 0 & (3{,}4) & 0 \\ 0 & 0 & (4{,}3) & 0 & 0 \\ (5{,}1) & (5{,}2) & 0 & 0 & 0 \end{bmatrix}$$

Figure 11 Nodes-set matrix.

4.3. Eight nodes example

The following network example is taken from Cambos and Ricardo (2008). The computer network and its distance matrix are given in Figs. 13 and 14, respectively.

The preferred link matrix (least-cost link matrix) can be constructed from the distance matrix as shown in Fig. 15.

The corresponding nodes-set matrix NSM is constructed as shown in Fig. 16.

After combining these node pairs, we get the set:

$\{(2,1), (4,1), (1,2), (4,3), (4,5), (7,3), (7,8), (2,6), (8,7)\}$

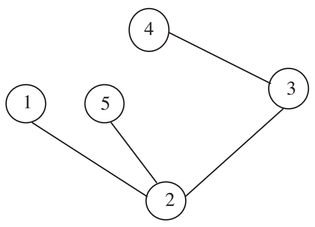

Figure 12 The least-cost minimum spanning tree.

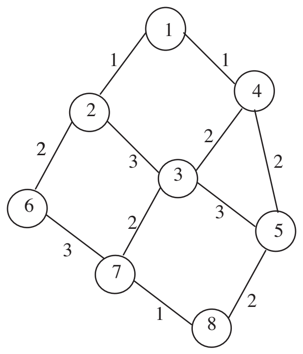

Figure 13 Computer network of Example 4.3.

$$D=\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 3 & 0 & 0 & 2 & 0 & 0 \\ 0 & 3 & 0 & 2 & 3 & 0 & 2 & 0 \\ 1 & 0 & 2 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 2 & 0 & 0 & 0 & 2 \\ 0 & 2 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 2 & 0 & 0 & 3 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure 14 Distance matrix.

$$PLM=\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 2 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure 15 Preferred link matrix.

$$NSM=\begin{bmatrix} 0 & (1,2) & 0 & (1,4) & 0 & 0 & 0 & 0 \\ (2,1) & 0 & 0 & 0 & 0 & (2,6) & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ (4,1) & 0 & (4,3) & 0 & (4,5) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & (7,3) & 0 & 0 & 0 & 0 & (7,8) \\ 0 & 0 & 0 & 0 & 0 & 0 & (8,7) & 0 \end{bmatrix}$$

Figure 16 Nodes-set matrix.

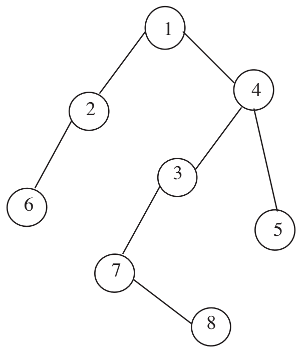

Figure 17 The least-cost minimum spanning tree.

Removing the repeated one, the LC-MSPT is:

{(2, 1), (4, 1), (4, 3), (4, 5), (7, 3), (2, 6), (7, 8)}

The corresponding graph of the obtained spanning tree is shown in Fig. 17.

The cost of it is 11.

5. Applications of the LC-MST algorithm

5.1. Solving the DC-MST problem

The delay constrained minimum spanning tree problem (DC-MST) is formulated by Lee and Atiquzzaman (2007) as: the objective function of the DC-CMST problem is to find a collection of trees with minimal link cost subject to the following three constraints:

(i) Average traffic flow on the link should be smaller than the capacity of the link.
(ii) Mean delay time of network has to be dropped within allowable value (δ).
(iii) Maximum traffic flow on one tree must be below ε (maximum traffic).

The problem solved in Lee and Atiquzzaman (2007) uses two phases: Subtree generation phase and Matching phase. And matching phase contains a final step called the Link capacity allocation phase.

The total execution time of DC-CMST algorithm = execution time of subtree generation phase + execution time of Matching phase + execution time of link capacity allocation phase = $O(kV^2)$, where k is the number of links and V is the number of nodes in the network.

By using the presented algorithm, we obtained the same spanning tree as shown in Section 3.1 that satisfies the above three constraints.

5.2. Solving the MRCT problem

The MRCT (Minimum Routing Cost Tree) problem is presented in Cambos and Ricardo (2008) and is defined by the optimal spanning tree from the standpoint of the routing cost and always represents a spanning tree closer to the optimal solution defined by the union of the Shortest Path Trees.

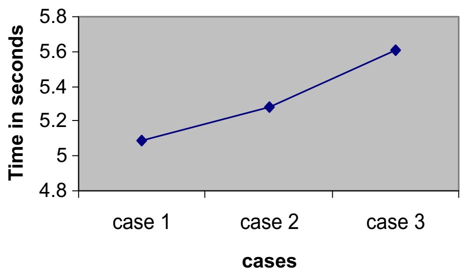

Figure 18 Time for the algorithm to reach the solution for each case.

Table 1 Comparison of algorithms.

Algorithm	Application problem	Time complexity
EW algorithm (Esau and Williams, 1996)	CMST	$O(n^2 \log(n))$
Gavish (Gavish and Altinkemer, 1986)	CMST	$O(n^3)$
Sharma (Sharma, 1983)	CMST	$O(n \log(n))$
Lee algorithm (Lee, 2006)	CMST	$O(2^L V^2)$
DC-MST algorithm (Lee and Atiquzzaman, 2007)	DC-MST	$O(kV^2)$
MRCT algorithm (Cambos and Ricardo, 2008)	MRCT	$O(m + n \log(n))$
LC-MST algorithm	LC-MST	$O(n^2)$

The MRCT is given by

$$C_r(T^*) = \min\left(\sum\sum c_T(i,j),\ i{\neq}j\right) \tag{4}$$

For all spanning trees T that can be computed from G.

The time of the algorithm is $O(m + n\log(n))$, where m is the number of links and n is the number of nodes.

By using the presented algorithm, we obtained the same MRCT as shown in Section 3.2.

6. Time analysis

The algorithm searches the least cost links that construct the least cost minimum spanning tree. So, the algorithm produces the exact solution to the LC-MST problem

Consider a network with n nodes. This network can be represented by an array of *nxn* dimensions. Time complexity to search the least cost links to construct the candidate LC-MST is $O(n^2)$. Time complexity to construct the final LC-MST is $O(n^2)$. Then, the time complexity of LC-MST algorithm is $O(n^2)$, n is the number of nodes.

Fig. 18 shows the time (in s) required by the proposed algorithm to obtain the LC-MST for each studied case.

Table 1 presents the comparison of some algorithms including the proposed LC-MST algorithm. Lee and Atiquzzaman (2007) and Lee (2006) proved that the time complexity of the DC-MST and CMST respectively is less than the time complexity of the algorithms presented in Esau and Williams (1996), Gavish and Altinkemer, (1986) and Sharma (1983). The time complexity of the proposed algorithm LC-MST is equal to $O(n^2)$ and is less than the time complexity of both the DC-MST and CMST algorithms.

7. Conclusion and future work

The paper presented a simple and efficient method to solve the LC-MST problem in less time. Also, the presented method has been applied on the DC-MST and MRCT problems. In future work, we hope to use the presented method to solve other constrained spanning tree problems and also apply it when there is a multicriteria for choosing the link (such as cost and delay) to construct the minimum spanning tree.

References

Cambos, Rui, Ricardo, Manuel, 2008. A fast algorithm for computing minimum routing cost spanning trees. Computer Networks 52, 3229–3247.

Esau, L.R., Williams, K.C., 1996. On teleprocessing system design, part II. IBM System Journal 5 (3), 142–147.

Gavish, B., Altinkemer, K., 1986. Parallel savings heuristic for the topological design of local access tree networks. INFOCOM, 130–139.

Graham, R.L., Hell, Pavol, 1985. On the history of the minimum spanning tree problem. Annals of the History of Computing 7 (1), 44–57.

Jothi, Raja, Raghavachari, Balaji, 2005. Approximation algorithms for the capacitated minimum spanning tree problem and its variants in network design. ACM Transactions on Algorithms 1 (2), 265–282.

Lee, Y., 2006. Multimedia traffic distribution using capacitated multicast tree. Lecture Notes in Computer Science 4317, 212–220.

Lee, Y.J., Atiquzzaman, M., 2007. Exact algorithm for delay-constrained capacitated minimum spanning tree network. IET Communication 1 (6), 1238–1247.

Nghia, Nguyen Duc, Binh, Huynh Thi Thanh, 2008. Heuristic algorithms for solving bounded diameter minimum spanning tree problem and its application to genetic algorithm development, advances in greedy algorithms. Witold Bednorz.

Pagacz, Anna, Raidl, Günther, Zawislak, Stanisaw, 2006. Evolutionary approach to constrained minimum spanning tree problem – commercial software based application. Evolutionary Computation and Global Optimization, 331–341.

Sharma, R., 1983. Design of an economic multi-drop network topology with capacity constraints. IEEE Transactions on Communication 31 (4), 590–591.

Sriram, R., Manimaran, G., Siva Ram Murthy, C., 1998. Preferred link based delay-constrained least-cost routing in wide area networks. Computer Communications 21, 1655–1669.

Zhou, Gengui, Cao, Zhenyu, Cao, Jian, Meng, Zhiqing, 2006. A genetic algorithm approach on capacitated minimum spanning tree problem. International Conference on Computational Intelligence and Security, 215–218.