A Fast Algorithm for Steiner Trees

Article *in* Acta Informatica · June 1981

DOI: 10.1007/BF00288961 · Source: DBLP

3 authors, including:

George Markowsky
Kennesaw State University
142 PUBLICATIONS **3,830** CITATIONS

Acta Informatica 15, 141–145 (1981)

Acta Informatica

A Fast Algorithm for Steiner Trees

L. Kou, G. Markowsky, and L. Berman

IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598, USA

Summary. Given an undirected distance graph $G=(V, E, d)$ and a set S, where V is the set of vertices in G, E is the set of edges in G, d is a distance function which maps E into the set of nonnegative numbers and $S \subseteq V$ is a subset of the vertices of V, the Steiner tree problem is to find a tree of G that spans S with minimal total distance on its edges. In this paper, we analyze a heuristic algorithm for the Steiner tree problem. The heuristic algorithm has a worst case time complexity of $O(|S||V|^2)$ on a random access computer and it guarantees to output a tree that spans S with total distance on its edges no more than $2\left(1-\frac{1}{l}\right)$ times that of the optimal tree, where l is the number of leaves in the optimal tree.

Introduction

Let $G=(V,E,d)$ be an *undirected distance graph*, where $V=\{v_1, v_2, \ldots, v_n\}$ is the set of *vertices* in G, $E \subseteq \{\{v_i, v_j\} | v_i \in V, v_j \in V \text{ and } v_i \neq v_j\}$ [1] is the set of *edges* in G and d: $E \to R$ is a *distance function* which maps E into the set R of nonnegative numbers. G is *complete* if, for all pairs of distinct vertices v_i and v_j, $\{v_i, v_j\} \in E$. Let $S \subseteq V$ be a subset of distinguished vertices of V which we shall call *Steiner points*.

We shall denote a *path* in G by a sequence of vertices, $u_1, u_2, \ldots, u_p$, such that for all k, $1 \leqq k < p$, $\{u_k, u_{k+1}\} \in E$ and $u_k \in V$. We shall say that the path is from u_1 to u_k and its distance is $\sum_{k=1}^{p-1} d(\{u_k, u_{k+1}\})$. The path is *simple* if all the vertices on the path are distinct. A *shortest path* from u_1 to u_p is a path from u_1 to u_p whose distance is minimal among all the possible paths from u_1 to u_p. A *loop* is a path, $u_1, u_2, \ldots, u_p$, such that $u_1 = u_p$.

A *tree* of G is a connected subgraph of G such that the removal of any edge in the subgraph will make it disconnected. Let Q be any subset of vertices in a

connected subgraph G' of G. We shall say that G' *spans* Q. A *spanning tree* of G is a tree that spans V. The *minimal spanning tree* of G is a spanning tree of G such that the total distance on its edges is minimal among all spanning trees. Given an undirected distance graph G and a set of Steiner points S, a *Steiner tree* for G and S is a tree in G that spans S. The *minimal Steiner tree* for G and S, is a Steiner tree for G and S such that the total distance on its edges is minimal among all Steiner trees for G and S.

The problem of finding a minimal Steiner tree for any given G and S has been shown to be NP-complete [8]. The problem has also been shown to be NP-complete even with a restricted class of function mappings for the distance function [2]. Due to the wide application of the Steiner tree problem, we will analyze a heuristic algorithm for the problem. The Steiner tree produced by the suggested heuristic algorithm is not necessarily minimal. However, it will be shown that D_H, the total distance on the edges of the Steiner tree produced by the heuristic algorithm, is not very far from D_{MIN}, the total distance on the edges of the minimal Steiner tree. In fact, we are going to show that, $D_H/D_{\text{MIN}} \leqq 2\left(1-\frac{1}{l}\right)$, where l is the number of leaves in the minimal Steiner tree. The upper bound for D_H/D_{MIN} will also be shown to be sharp. As far as the computational time complexity is concerned, for a given $G=(V,E,d)$ and $S\subseteq V$, the worst case time complexity of the heuristic algorithm is $O(|S||V|^2)$. The algorithm we consider has been considered in a more restricted setting by Hwang [9].

Heuristic Algorithm

Given a connected undirected distance graph $G=(V,E,d)$ and a set of Steiner points $S\subseteq V$, consider the complete undirected distance graph $G_1=(V_1,E_1,d_1)$ constructed from G and S in such a way that $V_1=S$ and, for every $\{v_i,v_j\}\in E_1$, $d(\{v_i,v_j\})$ is set equal to the distance of the shortest path from v_i to v_j in G. Notice that, for each edge in G_1, there corresponds a shortest path in G. Given any spanning tree in G_1, we can construct a subgraph of G by replacing each edge in the tree by its corresponding shortest path in G. Our heuristic algorithm for the Steiner tree problem is simply the following.

Algorithm H

INPUT: an undirected distance graph $G=(V,E,d)$ and a set of Steiner points $S\subseteq V$

OUTPUT: a Steiner tree, T_H, for G and S

Step 1. Construct the complete undirected distance graph $G_1=(V_1,E_1,d_1)$ from G and S.

Step 2. Find the minimal spanning tree, T_1, of G_1. (If there are several minimal spanning trees, pick an arbitrary one.)

Step 3. Construct the subgraph, G_S, of G by replacing each edge in T_1 by its corresponding shortest path in G. (If there are several shortest paths, pick an arbitrary one.)

Step 4. Find the minimal spanning tree, T_S, of G_S. (If there are several minimal spanning trees, pick an arbitrary one.)

Step 5. Construct a Steiner tree, T_H, from T_S by deleting edges in T_S, if necessary, so that all the leaves in T_H are Steiner points.

The readers are referred to the literatures with regard to the algorithms for constructing the shortest path as required in Step 1 [3–5] and for finding a minimal spanning tree as required in Step 2 and Step 3 [3, 6, 7]. We would only mention that, as far as computational complexity is concerned, using algorithms as mentioned above, in the worst case, Step 1 could be done in $O(|S||V|^2)$ time, Step 2 could be done in $O(|S|^2)$ time, Step 3 could be done in $O(|V|)$ time, Step 4 could be done in $O(|V|^2)$ time and Step 5 could be done in $O(|V|)$ time. Overall speaking, Step 1 dominates the computational time. Hence, our heuristic algorithm has a worst case time complexity of $O(|S||V|^2)$.

Example

$G=(V,E,d)$ is given in Fig. 1a. Each edge is labeled with a distance. Let $S=\{v_1,v_2,v_3,v_4\}$. Figure 1b shows the graph G_1. Figure 1c shows the minimal

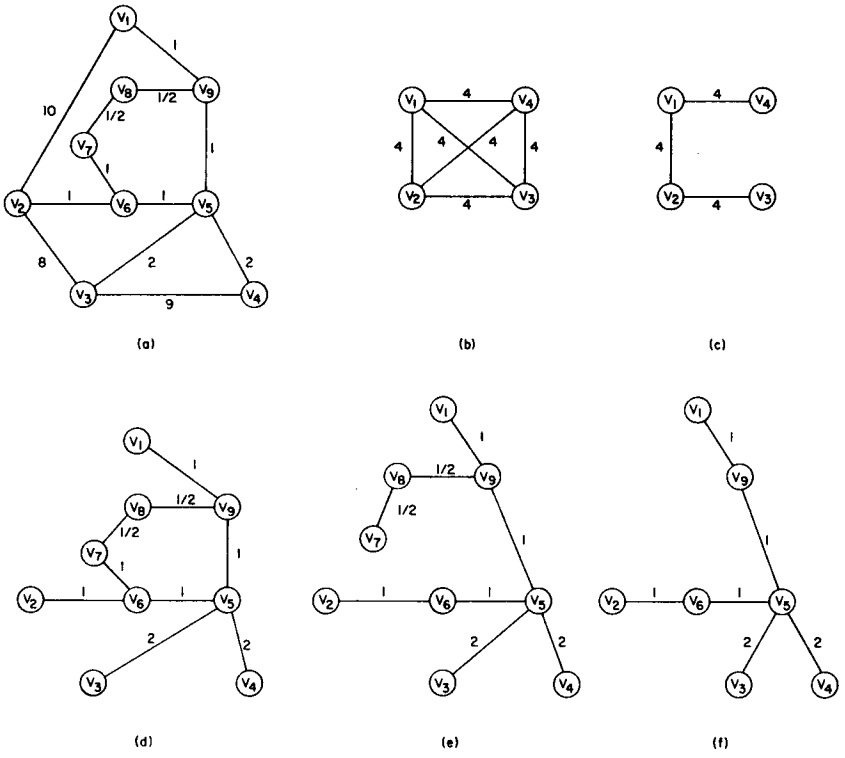

Fig. 1a–f

spanning tree T_1 of G_1. Figure 1d shows the corresponding subgraph, G_S, of G. The minimal spanning tree T_S of G_S is shown in Fig. 1e. Figure 1f shows the final output for this example, T_H. The output turns out to be the minimal Steiner tree. Notice that the example is chosen intentionally to demonstrate each step of the heuristic algorithm and to reveal the fact that T_1, G_S, T_S might not be unique.

Worst Case Performance of Algorithm H

Let an undirected distance graph $G=(V,E,d)$ and a set of Steiner points S be the input to algorithm H. Let D_H be the total distance on the edges of the Steiner tree, T_H, produced by algorithm H and let D_{MIN} be the total distance on the edges of the minimal Steiner tree. In this section, we shall give a worst case upper bound for the ratio, D_H/D_{MIN}. First, we need the following lemma.

Lemma. *Let T be a tree with $m \geqq 1$ edges. Then, there exists a loop in T, $u_0, u_1, u_2, \ldots, u_{2m}$, where every u_i, $1 \leqq i \leqq 2m$, is a vertex in T, such that (i) every edge in T appears exactly twice in the loop, (ii) every leaf in T appears exactly once in the sequence, $u_0, u_1, \ldots, u_{2m}$[2] and if u_i, u_j are two leaves in the loop, with no other leaf between them then, $u_i, u_{i+1}, \ldots, u_j$ is a simple path.*

Proof. We shall prove this lemma by induction on m. For $m=1$, let $\{u_1, u_2\}$ be the vertices in T. Consider the loop u_1, u_2, u_1 which satisfies (i) and (ii) in the lemma. Assume for $m=k \geqq 1$ the lemma is true. Then, for $m=k+1$, let v_p be a leaf in T and $\{v_p, v_q\}$ be the edge connected to v_p. Consider the tree T' constructed by deleting the edge, $\{v_p, v_q\}$, and the vertex v_p from T. By induction, there exists a loop, $u'_0, u'_1, \ldots, u'_{2k}$, in T' satisfying (i) and (ii). Replacing the first appearance of v_q in the loop by v_q, v_p, v_q, the lemma then follows. □

We now give the worst case upper bound for the ratio D_H/D_{MIN} in the next theorem.

Theorem 1. *Let the undirected distance graph $G=(V,E,d)$ and the set of Steiner points S be the input to algorithm H. Let T_H be the Steiner tree produced by algorithm H and D_H be the total distance on its edges. Let T_{MIN} be the minimal Steiner tree, l be the total number of leaves in T_{MIN} and D_{MIN} be the total distance on its edges. Then, $D_H/D_{\text{MIN}} \leqq 2\left(1-\frac{1}{l}\right) \leqq 2\left(1-\frac{1}{|S|}\right)$.*

Proof. By the preceding lemma, there is a loop L in T_{MIN} such that (i) every edge in T_{MIN} appears exactly twice in T_{MIN}, (ii) every leaf in T_{MIN} appears exactly once in T_{MIN} and if u_i, u_j are two "consecutive" leaves in the loop, then the subpath connecting u_i, u_j in the loop is a simple path. We may regard the loop L as composed of l simple subpaths each connecting a leaf to another leaf. By deleting from the loop L the longest simple subpath, we could construct a path P such that (i) total distance of P is no more than $\left(1-\frac{1}{l}\right)$ times the total distance of L (ii) every edge in T_{MIN} appears at least once in P. Let $w_1, w_2, \ldots, w_k$ be the $k=|S|$ distinct Steiner points appearing, in P, in that order. We then have the total distance of $P \leqq \left(1-\frac{1}{l}\right) \times$ total distance of $L = \left(1-\frac{1}{l}\right) \times 2 \times$ total distance on the edges of $T_{\text{MIN}} = 2\left(1-\frac{1}{l}\right) D_{\text{MIN}}$. On the other hand, the total distance of $P \geqq$ the total distance on the edges of a spanning tree for G_1 consisting of edges, $\{w_1, w_2\}, \{w_2, w_3\}, \ldots, \{w_{k-1}, w_k\} \geqq$ the total distance on the edges of the minimal spanning tree for $G_1 \geqq D_H$. Hence $D_H \leqq 2\left(1-\frac{1}{l}\right) D_{\text{MIN}}$. The theorem follows. □

[2] $u_0 = u_{2m}$ is counted as one appearance in the loop

We want now to show that the upper bound for D_H/D_{MIN} given in Theorem 1 is the best possible.

Theorem 2. *For every positive integer $l \geqq 2$, there exist an undirected distance graph $G=(V,E,d)$ and a set of Steiner points $S \subseteq V$ such that the minimal Steiner tree, T_{MIN}, for G and S has l leaves and has total distance D_{MIN} on its edges whereas the algorithm H, in the worst case, could produce a Steiner tree, T_H, for G and S with total distance D_H on its edges and $D_H/D_{\text{MIN}} = 2\left(1-\frac{1}{l}\right)$.*

Proof. Consider $G=(V,E,d)$ defined as an complete undirected distance graph such that $V=\{v_1, v_2, \ldots, v_{l+1}\}$ and, for all $\{v_i, v_j\} \in E$,

$$d(\{v_i, v_j\}) = \begin{cases} 2 & \text{if } v_i \neq v_{l+1} \text{ and } v_j \neq v_{l+1} \\ 1 & \text{otherwise.} \end{cases}$$

The set of Steiner points, S, is $\{v_1, v_2, \ldots, v_l\}$. In the worst case, T_H could be a Steiner tree consisting of edges, $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{l-1}, v_l\}$. The minimal Steiner tree, T_{MIN}, is a Steiner tree consisting of edges, $\{v_1, v_{l+1}\}, \{v_2, v_{l+1}\}, \ldots, \{v_l, v_{l+1}\}$. Therefore, $D_H/D_{\text{MIN}} = 2(l-1)/l = 2\left(1-\frac{1}{l}\right)$. □

References

1. Gilbert, E.N., Pollak, H.O.: Steiner minimal tree. SIAM J. Appl. Math. **16**, 1-29 (1968)
2. Garey, M.R., Graham, R.L., Johnson, D.S.: Some NP-complete geometric problems. Proc. of the 8th Annual ACM Symposium on Theory of Computing, pp. 10-22 1976
3. Dijkstra, E.N.: A note on two problems in connection with graphs. Numer. Math. **1**, 269-271 (1959)
4. Floyd, R.W.: Algorithm 97: shortest path, CACM **5**, 345 (1962)
5. Tabourier, Y.: All shortest distances in a graph: an improvement to Dantzig's inductive algorithm, Discrete Math. **4**, 83-87 (1973)
6. Yao, A.C.C.: An $O(|E| \log\log |V|)$ algorithm for finding minimal spanning tree. Information Processing Lett. **4**, 21-23 (1975)
7. Cheriton, D., Tarjan, R.E.: Finding minimal spanning tree, SIAM J. Comput. **5**, 724-742 (1976)
8. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of computer computations (R.E. Miller, J.W. Thatcher eds.), pp. 85-104. New York: Plenum Press 1972
9. Hwang, F.K.: On Steiner minimal trees with rectilinear distance, SIAM J. Appl. Math. **30**, 104-114 (1976)

Received December 14, 1979