MAX-PLANCK-INSTITUT
FÜR
INFORMATIK

A Faster 11/6–Aproximation Algorithm for the Steiner Tree Problem in Graphs

A. Zelikovsky

MPI–I–92–122 June 1992

INFORMATIK

Im Stadtwald
66123 Saarbrücken
Germany

A Faster 11/6–Aproximation Algorithm for the Steiner Tree Problem in Graphs

A. Zelikovsky

MPI–I–92–122 June 1992

"Das diesem Bericht zugrunde liegende Vorhaben wurde mit Mitteln des Bundesministers für Forschung und Technologie (Betreuungskennzeichen ITS 9103) gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor."

A Faster 11/6-Approximation Algorithm for the Steiner Tree Problem in Graphs

Alexander Z. Zelikovsky

Max-Planck-Institut für Informatik
Im Stadtwald, 6600 Saarbrücken, Germany
and
Institute of Mathematics of Moldavian Academy of Sciences
Akademicheskaya 5, Kishinev, 277028 Moldova

June 22, 1992

Abstract

The Steiner problem requires a shortest tree spanning a given vertex subset S within graph $G = (V, E)$. There are two 11/6-approximation algorithms with running time $O(VE + VS^2 + S^4)$ [9] and $O(VE + VS^2 + S^{3+\frac{1}{2}})$ [2], respectively. Now we decrease the implementation time to $O(ES + VS^2 + VlogV)$.

1 Introduction

Let $G = (V, E, d)$ be a graph with a vertex set V, an edge set E and distance function $d : E \to R^+$. A tree T is a *Steiner tree* of S, $S \subset V$, if S is contained in the vertex set of T. Given G and S, the *Seiner tree problem* requires the shortest Steiner tree (also called the *Steiner minimal tree*) of S.

It is known that the Steiner tree problem is NP-hard [5]. Therefore algorithms which in polynomial time construct an approximate Steiner minimal tree are investigated. The quality of an approximation is measured by its performance ratio: an upper bound on the ratio between achieved length and the optimal length.

A well-known heuristic (an *MST-heuristic*) for the Steiner tree problem approximates a Steiner minimal tree with a minimum length spanning tree of a complete graph G_S which has a vertex set S and edge lengths equal to shortest path lengths in the graph G. It was proved that the lowest performance ratio of this heuristic equals 2 [7]. The fastest known implementation of the MST-heuristic has a running time $O(E + VlogV)$ [6] (throughout this paper we use E, V, S to denote the $\#E, \#V, \#S$, respectively, in the order of a running time of an algorithm). For many years, the problem of finding a better heuristic remained open.

Two better heuristics were given recently [2,9]. Their better performance ratios appear while consideration of a $k-$restricted Steiner tree problem.

First we introduce some denotations: $SMT(S)$ and $smt(S)$ are a Steiner minimal tree of S and its length, respectively. For a complete graph with a vertex set S, G_S, $M(G_S)$ denotes the minimum length spanning tree of G_S, and $m(G_S)$ denotes its length.

$SMT(S)$ may in general contain vertices of $V \backslash S$. So $SMT(S)$ contains the set S of *given vertices* and some additional vertices. $SMT(S)$ is called a *full Steiner tree* if S coincides with the set of leaves of $SMT(S)$. If $SMT(S)$ is not full, then we can split it into the union of edge-disjoint full Steiner subtrees. $SMT(S)$ is called *$k-$restricted* if every full component has at most k given vertices. Let the shortest $k-$restricted Steiner tree for the set S, denoted by $SMT_k(S)$, has the length $smt_k(S)$. Note, that $SMT_2(S) = M(F)$.

Let $r_k = sup\{smt_k(S)/smt(S)\}$. The bound for the MST-heuristic implies $r_2 = 2$ [7]. It was proved that $r_3 = 5/3$ [8,9], $r_4 \leq \frac{3}{2}$ and $r_8 \leq \frac{4}{3}$ [1]. Moreover, $r_{2^k} \leq 1 + \frac{1}{k}$ [3].

The above bounds arise the *$k-$restricted Steiner tree problem* which requires a shortest $k-$restricted Steiner tree. A greedy algorithm finds a 2-restricted Steiner minimal tree [7]. Unfortunately, for $k \geq 4$ computing $SMT_k(S)$ is NP-hard [5]. The problem for $k = 3$ is open. Note that this problem can be generalized to the problem of a minimum spanning set of weighted $(k-1)-$polymatroids [10]. The main idea of the known heuristics with nontrivial performance ratios is to approximate $k-$restricted Steiner minimal trees instead of usual Steiner minimal trees.

A *greedy* heuristic achieves a performance ratio $r_2 - (r_2 - r_3)/2$ in time

$O(VE + VS^2 + S^4)$ [9]. A family of *evaluation* heuristics A_k was constructed in [2]. A_k achieves a performance ratio at most

$$r_2 - \sum_{i=3}^{k} \frac{r_{i-1} - r_i}{i-1}$$

in time $O(VE + V^{k-2}S^{k-1} + S^{k+\frac{1}{2}})$ [2].

We restrict our attention to the case $k = 3$. Then the approximation ratio for A_3 (as well as for greedy algorithm) is at most $\frac{11}{6}$. The problem of finding exact upper approximation bounds for the both algorithms is still open.

In this paper, we achieve the performance ratio of $\frac{11}{6}$ in time $O(ES + VS^2 + VlogV)$. A greedy approach to the 3−restricted Steiner problem, necessary definitions and facts are contained in the next section. In Section 3, we give a modified version of the greedy algorithm.

2 Greedy Approach

Some preliminary definitions: given a triple $z = \{a, b, c\} \in S$, a Steiner minimal tree z^* for z (called a *star*) may include one additional vertex $v = v(z)$ (called a *center* of a star). The length of $z^* = (v(z); a, b, c)$ is denoted by $d(z) = d(v, a) + d(v, b) + d(v, c)$. For a set Z of triples, $d(Z)$ is the sum of lengths of its elements. *Triples* denotes the set of all triples for S.

Let $T = M(G_S)$ and $d(T)$ denote the length of T. Given a pair of vertices a, b of T, we use $T[a, b]$ to denote an $MST(T \cup (a, b))$, where (a, b) is an edge of zero length. For any triple $z = \{a, b, c\}$ the graph $T[z]$ equals $T[(a, b)][(a, c)]$, i.e. it results from two reductions. For a set A consisting of pairs and triples we define $T[A]$ recursively: $T[\emptyset] = T$, and $T[A \cup e] = T[A][e]$.

Let a value

$$win_T(z) = m(F) - m(F \cup z^*) = d(T) - d(T[z]) - d(z)$$

is positive. Then $m(T \cup z^*)$ is better than the MST-heuristic approximate solution for the 3-restricted Steiner problem. For a set Z of triples, we define $win_T(Z) = d(T) - d(T[Z]) - d(Z)$. The equality $r_3 = \frac{5}{3}$ implies an existance of a set Z such that

$$d(T) - win_T(Z) \leq \frac{5}{3} smt(S) \tag{2.1}$$

The greedy heuristic choses the best possible reduction of a previously achieved approximate solution. Below we present a rough version of the greedy heuristic.

Algorithm 2.1 (greedy heuristic)

(0) $T \leftarrow M(G_S)$, $W \leftarrow \emptyset$;
(1) repeat forever
  (a) find $z = argmax\{win_T(z) | z \in Triples\}$;
  (b) if $win_T(z) \leq 0$ then exit repeat;
  (c) $T \leftarrow T[z]$; insert$(W, v(z))$;
(2) find a Steiner tree T_1 for $S \cup W$ in graph G using MST-heuristic.

A sequence of triples chosen by the greedy heuristic is called *greedy in* G_S.

Theorem 2.2 [9]. If H is the set of elements of a greedy sequence of triples, then for every set of triples Z

$$2win_T(H) \geq win_T(Z) \tag{2.2}$$

Inequalities (2.1) and (2.2) imply a performance ratio of $\frac{11}{6}$ for the greedy heuristic.

An implementation of the greedy heuristic given in [9] generates stars for all triples of given vertices in time $O(EV + VS^2)$. The same procedure is necessary for the evaluation heuristic. This generation needs the shortest path distances between given vertices and additional vertices. Therefore, we can decrease its running time to $O(ES + VS^2)$ using an $O(E)$-algorithm for every given vertex $s \in S$ to find all shortest paths from s to other vertices of V.

Now we describe computing of the function win_T.

For a pair $e = (a, b)$ of given vertices define $save_T(e) = d(T) - d(T[e])$. Let $List(T) = \{t_1, ..., t_n\}$ be an nondecreasing oreder of edges of T. Then $save_T(e)$ is the length of the last edge of $List$, say t_i, in the unique cycle of

$T \cup e$. The index i of t_i is denoted by $ind_T(e)$, i.e. $save_T(e) = d(t_{ind_T(e)})$. Note that $T[e] = e \cup T \backslash t_i$ and $List(T[e]) = \{e, t_1, ..., t_{i-1}, t_{i+1}, ..., t_n\}$.

Further, $\bar{z}$ denotes the set of three edges $\{(a,b),(b,c),(c,a)\}$, for a triple $z = \{a,b,c\}$.

Lemma 2.3. For any triple $z = \{a,b,c\}$, $\bar{z}$ contains a unique edge with the minimum index and two other edge indices equal to each other.

Proof. Let P_{ab}, P_{bc}, P_{ca} be simple paths in the tree T between a and b, b and c, c and a, respectively. One of these three paths is the symmetric subtraction of the two others. Let $P_{ca} = (P_{ab} \backslash P_{bc}) \cup (P_{bc} \backslash P_{ab})$, $ind_T(a,b) \geq ind_T(b,c)$ and $ind_T(c,a) = i$. If $t_i \in P_{ab} \backslash P_{bc}$, then $i = ind(a,b) = ind(c,a) > ind(b,c)$, otherwise $i = ind(a,b) = ind(b,c) > ind(c,a)$. []

Corollary 2.4[9].

$$win_T(z) = \max_{e \subset \bar{z}} save_T(e) + \min_{e \subset \bar{z}} save_T(e)$$

This corollary implies that it is sufficient to compute the function $save_T$. At first we find a binary tree T' which corresponds to T according to $List$. Inner vertices of T' correspond to edges of T, leaves correspond to the vertices of T. A root of T' is t_n and sons of t_i are the last edges in two components which appear after deletion of t_i from $T' \backslash A(t_i)$, where $A(t_i)$ is the set of ancestors of t_i. If such component does not contain edges, then the son is the corresponding vertex of T. Moreover, using $preprocess(T')$ (preprocessing of time $O(S)$) we can find in time $O(1)$ the nearest common ancestor of any pair e of given vertices [4] which corresponds to the edge of T with the length $save_T(e)$.

Thus, to fulfill the step (1)(a) of Algorithm 2.1 it is sufficient time $O(S^3)$, therefore, the step (1) demands time $O(S^4)$ and a running time of the whole algorithm is $O(ES + VS^2 + S^4)$.

3 A Faster Greedy heuristic

Now we present a faster modification of the greedy heuristic. At first, for every vertex $v \in V \backslash S$, the following algorithm finds a best possible star with the center v and then chooses the best one among all such stars. Therefore, it does not generate all stars with positive win.

Algorithm 3.1.

(0) find all shortest paths from S to V;
find $T = M(G_S)$;
$W \leftarrow \emptyset$;
(1) repeat forever
(a) find $List(T), T'$; $preprocess(T')$;
(b) for all $v \in V \backslash S$ do
$s_0 \leftarrow arg\min_{s \in S} d(v, s)$;
$s_1 \leftarrow arg\max_{s \in S}[save_T(s_0, s) - d(v, s)]$;
$s_2 \leftarrow arg\max_{s \in S} win_T(v; s_0, s_1, s)$;
(c) $z \leftarrow arg\max_{v \in V \backslash S} win_T(v; s_0, s_1, s_2)$;
(d) if $win_T(z) \leq 0$ then exit repeat;
(e) $T \leftarrow T \backslash \{e_1, e_2\} \cup \{(s_0, s_1), (s_0, s_2)\}$, where

$$e_1 = arg\max_{e \subset \bar{z}} save_T(e), e_2 = arg\min_{e \subset \bar{z}} save_T(e)$$

and $d(s_0, s_1) \leftarrow d(s_0, s_2) \leftarrow 0$;
$insert(W, v(z))$;
(2) find a Steiner tree T_2 for $S \cup W$ in graph G using MST-heuristic.

Lemma 3.2. The output Steiner tree T_2 of Algorithm 3.1 coincides with the output Steiner tree T_1 of Algorithm 2.1.

Proof. It is necessary to prove that a star resulted by steps (1)(b) and (1)(c) has a maximum win. Let

$$z' = (v; a, b, c) = arg \max_{a,b,c \in S} win_T(v; a, b, c)$$

Let i and j be the largest and the smallest indices of $\bar{z}'$. Then

$$win_T(v; a, b, c) = d(t_i) + d(t_j) - d(v, a) - d(v, b) - d(v, c) \qquad (3.1).$$

The forest $T \backslash \{t_i, t_j\}$ has three components with vertex sets A, B, C, such that $a \in A, b \in B, c \in C$. Let t_i connect A and B, t_j connect B and C. Then

$ind(a', b') \geq i$ for every two vertices $a' \in A$ and $b' \in B$. The same inequality holds for the components B and C. Note, that (3.1) implies that a, b, c are the nearest vertices to v in its components. Therefore, $s_0 \in z'$.

Let s_0 coincides with a. If s_1 belongs to the other component (B or C), then s_1 also coincides with b or c and $z = z'$, since the star z has the largest win_T among stars with the center v and given vertices s_0 and s_1.

Let $s_1 \in A$ and $ind_T(s_0, s_1) = k$. Then k cannot coincide with i and j and Lemma 2.3 implies that

$$win_T(v; s_0, s_1, b) = d(t_i) + d(t_k) - d(v, s_0) - d(v, s_1) - d(v, b)$$

This value is at least

$$d(t_j) - d(v, c) + d(t_i) - d(v, s_0) - d(v, b) = win_T(v; s_0, b, c),$$

since $d(t_k) - d(v, s_1) \geq d(t_i) - d(v, c) \geq d(t_j) - d(v, c)$

The cases of s_0 coincide with b or c are similar. []

Now we can present the main result of this paper.

Theorem 3.3. Algorithm 3.1 finds an $\frac{11}{6}$-approximation of a minimal Steiner tree in time $O(ES + VS^2 + VlogV)$.

Proof. Lemma 3.2 implies that it is sufficient to estimate the running time of Algorithm 3.1. The steps (0) and (2) have been considered already above: they can be implemented in time $O(E + VlogV)$ [6]. The updating time for the step (1) is $O(S)$ for *preprocess* and constant for substitution of edges in the current tree T. The most complicated step is (1)(a), which takes time $O(VS)$ per iteration. Thus, the whole running time of Algorithm 3.1 is $O(ES + VS^2 + VlogV)$.[]

References

[1] P. Berman and V. Ramaiyer. An approximation algorithm for the Steiner tree problem. *Tech. Rep.#CS-91-05, The Pennsylvania State University*, 1991.

[2] P. Berman and V. Ramaiyer. Improved approximations for the Steiner tree problem. *Proceedings, 3d SODA*, 325-334, 1992.

[3] Ding-Zhu Du, Yanjun Zhang and Qing Feng. On better heuristic for Euclidean Steiner minimum trees. *Proceedings, 32nd FOCS*, 431-439, 1991.

[4] D. Harel and R. E. Tarjan. Fast algorithms for finding nearest common ancestor. *SIAM J. Comp.* , 13: 338-355, 1984.

[5] R. M. Karp. Reducibility among combinatorial problems. In Miller and Thatcher (eds.), *Complexity of Computer Computations*, Plenum Press, New York 85-103, 1972.

[6] K. Mehlhorn. A faster approximation algorithm for the Steiner problem in graphs. *Inf. Process. Lett.*, 27: 125-128, 1988.

[7] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Math. Japonica*, 24: 573-577, 1980.

[8] A. Z. Zelikovsky. The Steiner problem in graphs without optimal vertices. *Matemat. Issled.* 121: 44-49, 1991 (in russian).

[9] A. Z. Zelikovsky. An 11/6-approximation algorithm for the Steiner problem on graphs. In Complexity and computation, *Proceedings of International Symposium on combinatorics, Prachatice (Czechoslovakia), June 10-16*, 1990 (to appear)

[10] A. Z. Zelikovsky. Minimum base of weighted k-polymatroids and the Steiner tree problem. *Tech. Rep. #MPI-92-121, Max-Planck-Institut für Informatik*, 1992.