Ján Plesník
A bound for the Steiner tree problem in graphs


Persistent URL: [http://dml.cz/dmlcz/128936](http://dml.cz/dmlcz/128936)

**Terms of use:**

© Mathematical Institute of the Slovak Academy of Sciences, 1981

Institute of Mathematics of the Academy of Sciences of the Czech Republic provides access to digitized documents strictly for personal use. Each copy of any part of this document must contain these *Terms of use.*

This paper has been digitized, optimized for electronic delivery and stamped with digital signature within the project *DML-CZ: The Czech Digital Mathematics Library* [http://project.dml.cz](http://project.dml.cz)
A BOUND FOR THE STEINER TREE PROBLEM
IN GRAPHS

JÁN PLESNÍK

1. Introduction

Given a graph $G$ with edge lengths, find a tree $S$ in $G$ which interconnects a prescribed subset $B$ of vertices and has the least possible total length. This is the Steiner problem in graphs and $S$ is called a Steiner minimal tree for $B$. As the Steiner problem is very difficult, one is usually satisfied with an approximate solution obtainable by an effective method. In this paper, two polynomial algorithms are considered for finding a tree $T$ for $B$ to approximate the length of $S$. One of these algorithms computes a minimal spanning tree $T$ for $B$. We show that in both cases the ratio of the lengths of $T$ and $S$ never exceeds 2 and in the worst case tends to 2.

At the beginning of the 19-th century, J. Steiner raised and solved the following problem: Given three points in the Euclidean plane, find a connecting network of the minimal total length. The generalization for $n \geq 3$ points has been studied by Jarník and Kőssler [17] and now is known as the Steiner problem in the Euclidean plane (or, generally, in a Euclidean space). As any solution looks like a tree, one calls it a Steiner minimal tree. Many properties of such trees are reviewed by Gilbert and Pollak [12].

A later version of this problem, known as the rectilinear Steiner problem, was first suggested by Hanan [15] in connection with routing wires on printed circuit boards for electronic components (only horizontal and vertical lines may be used).

The Steiner problem in graphs has been proposed and studied by Hakimi [14] and by Dreyfus and Wagner [9]. It can be stated as follows: Let $G$ be an undirected graph with vertex set $V(G)$ and edge set $E(G)$, where each edge $e \in E(G)$ has a positive length $L(e)$; given a set $B \subset V(G)$ of so-called basic (or regular) vertices, find a tree $S$ in $G$ containing $B$ and having the least possible total length $L(S)$ (the sum of the lengths of the edges of $S$). One can see [15] that this problem involves the rectilinear Steiner problem as a special case and has also other applications in problems concerning network design [14]. A connected
subgraph \( H \) of \( G \) containing all the basic vertices is usually called a Steiner graph; if \( H \) is a tree, it is called a Steiner tree; if \( H \) has the least possible total length, then \( H \) must be a tree and is called a Steiner minimal tree.

All the presented problems are very difficult. More precisely, certain discretized versions of these problems are NP-complete (see Karp [18] for the problem in graphs, Garey and Johnson [11] for the rectilinear problem, and Garey, Graham and Johnson [10] for the problem in the Euclidean plane). Such results give strong evidence for the impossibility of efficient algorithms for these problems. Up to the present time, only some special cases have been solved effectively [1, 14, 17].

Here we discuss two heuristic methods. The first one is based on finding a minimal spanning tree of a derived graph. The second method is recurrent. It consists of a sequence of contractions and is related to a minimal spanning tree too.

2. Steiner trees by spanning trees

Given a Steiner problem with the set \( B \) (of the basic points or vertices), we can easily compute the distance \( d(u, v) \) for every two points or vertices \( u, v \in B \) (for graphs, see, e.g. [4, Chap. 8]). We obtain a complete graph \( K(B) \) with the vertex set \( B \), where each edge \( uv \) has the length \( L(uv) = d(u, v) \). Then we can easily find a minimal spanning tree \( T \) of \( K(B) \) [4, Chap. 7]. Finally, in the case of the rectilinear or graph problem, the conversion of \( T \) to a tree in the original structure is necessary but it is straightforward. For every edge \( uv \) of \( T \) in \( K(B) \) we choose a \( u-v \) path \( P_{uv} \) in \( G \) with \( L(P_{uv}) = L(uv) \). The union of all such paths form a subgraph \( H \) of \( G \), which is connected and contains \( B \) (all the basic vertices). Thus \( H \) is a Steiner graph, from which one can choose a Steiner tree \( T' \). Obviously, for any Steiner minimal tree \( S \) we have: \( L(S) \leq L(T') \leq L(H) \leq L(T) \). Sometimes \( L(T') < L(T) \), but in the worst case (as we shall see) the equality can occur and therefore the value \( L(T) \) is considered. Excepting the trivial case when \( |B| = 1 \), there is \( L(S) > 0 \) and we can ask for the ratio of \( L(T) \) and \( L(S) \). More precisely, let \( \alpha \) denote the minimal number such that for all examples of a Steiner problem the ratio \( L(T)/L(S) \leq \alpha \). The symbol is specified by an index and we write \( \alpha_m \), or \( \alpha_{rect} \), or \( \alpha_{graph} \) for the Steiner problem in the \( m \)-dimensional Euclidean space, or the rectilinear problem, or the problem in graphs, respectively. Graham and Hwang [13] have shown that \( \alpha_m \leq \sqrt{3} - 1.73... \) and examples of Chung and Gilbert [6] show that \( \alpha_m \geq (4 - \sqrt{2})/\sqrt{3} = 1.49... \) if \( m \) tends to infinity. Further, Chung and Hwang [7] have proved that \( \alpha_2 \leq 3/(2\sqrt{3} + 2 - \sqrt{7} + 2\sqrt{3}) = 1.34... \). Obviously, \( \alpha_2 \geq 2/\sqrt{3} = 1.16... \), which is the conjectured value for \( \alpha_2 \) [12]. Finally, Hwang [16] has determined \( \alpha_{rect} = 3/2 \). As for graphs, we have
Theorem 1. \( \alpha_{\text{graph}} = 2. \)

Proof. To prove that \( \alpha_{\text{graph}} \leq 2, \) consider a Steiner minimal tree \( S \) for a basic set \( B \) in a graph \( G, \) where \( |B| = k > 1. \) Note that every endvertex of \( S \) must be basic. Choose a basic vertex and denote it by \( u_1. \) Consider a walk \( W \) in \( S \) beginning at \( u_1, \) containing all the vertices of \( S, \) including every edge of \( S \) at most twice, and ending in a basic vertex. Such a walk \( W \) can be easily found by using the classical Tarry or Trémaux algorithm [2, Chap. 4], or a modern version of the latter called the depth-first search [20]. Let \( u_1, u_2, ..., u_k \) be the sequence of basic vertices ordered in accordance with the first appearance in the walk \( W. \) (See Fig. 1, where the basic vertices are depicted by squares, the tree \( S \) is depicted by heavy lines, and the slight line with arrows shows a walk \( W \) in \( S. \) ) The \( u_1 - u_k \) walk \( W \) can be decomposed into \( k - 1 \) paths: \( u_1 - u_2 \) path \( P_1, u_2 - u_3 \) path \( P_2, ..., u_{k-1} - u_k \) path \( P_{k-1}. \) The distance \( d_G(u_i, u_{i+1}) \) does not exceed the length \( L(P_i) \) \( (i = 1, ..., k - 1). \)

Let \( T \) be a minimal spanning tree in \( K(B) \) and let \( T' \) be the spanning tree of \( K(B) \) consisting of the edges \( u_1u_2, u_2u_3, ..., u_{k-1}u_k. \) Then we can write
\[
L(T) \leq L(T') = \sum L(u_iu_{i+1}) = \sum d_G(u_i, u_{i+1}) \leq \sum L(P_i) = L(W) < 2L(S).
\]
Hence \( \alpha_{\text{graph}} \leq 2. \)

To prove that \( \alpha_{\text{graph}} \geq 2, \) it is sufficient to consider the example in Fig. 2, where we have a graph \( G \) with \( 2m \) vertices, \( m \) basic vertices \( v_1, v_2, ..., v_m \) (depicted as squares) and each edge has the length 1. The edges of a Steiner minimal tree \( S \) are depicted by continuous lines while those of a minimal spanning tree \( T \) by dashed lines. We see that \( L(S) = m \) and \( L(T) = 2m - 2. \) Hence the ratio \( L(T)/L(S) \) tends to 2 if \( m \) tends to infinity. This completes the proof.

Remark. One could suggest the following stronger version of the spanning tree method: For a fixed \( q \) form all \( K(B \cup Q) \) with \( Q \subseteq V(G) - B \) and \( |Q| \leq q, \) and solve the corresponding minimal spanning tree problems. However, a bit more complicated examples than that of Fig. 2 show that also now the ratio of lengths of a shortest obtained tree and a Steiner minimal tree tends to 2.

3. A method of contractions

The following observation is obvious.

Lemma 1. If some basic vertices \( x, y \) of \( G \) are joined by a shortest edge, then there is a Steiner minimal tree containing the edge \( xy. \)

If such an edge exists, then we can shrink it to a basic vertex and we obtain a smaller graph to consider. Here we shall give another idea of shrinking. In fact, the well known Kruskal algorithm for minimal spanning trees (see, e.g. [4, Chap. 7]) uses the same observation. However, in the case of Steiner trees it may happen that Lemma 1 is not applicable. (Note that the first known minimal
spanning tree algorithm is due to Borůvka [3]. A good historical survey can be found in [8].

Let \( G \) be a graph. If \( G \) is considered as a (e.g., road) network, then the distance \( d_G(x, y) \) of two points \( x, y \in G \) is defined in the obvious way. (Note that \( x \) and \( y \) may be not only vertices, but any points from edges thought as simple curves.)

Given a basic vertex \( v \in B \) and a number \( r > 0 \), we define the neighbourhood \( N(v) \) of \( v \) with radius \( r \) to be the set \( \{ x \in G \mid d_G(v, x) \leq r \} \). A point \( y \in G \) with \( d_G(v, y) < r \) is called an interior point of \( N(v) \). The set of all points of all neighbourhoods \( N(v), v \in B \), with the same radius \( r \), can be divided into classes as follows. Two points \( x \in N(u) \) and \( y \in N(v) \) belong to the same class \( C \) whenever there is a sequence of neighbourhoods \( N_1, N_2, \ldots, N_n \) such that \( N(u) = N_1, N_1 \cap N_2 \neq \emptyset, \ldots, N_{n-1} \cap N_n \neq \emptyset \), and \( N_n = N(v) \). A point \( x \in C \) is called an interior point of \( C \) whenever it is an interior point of a neighbourhood included in \( C \); in the opposite case \( x \) is called a boundary point of \( C \). In Fig. 3, we have illustrated an example, where the basic vertices are depicted as squares, the other vertices as circles, the boundary points as crosses, and each class for \( r = 1 \) is in a dotted covering. (The heavy, wavy, or crossed lines should not be distinguished this time.)

The contraction \( f(G) \) of a graph \( G \) in a set of vertices \( B \subseteq V(G) \) with the radius \( r \) is a graph or pseudograph \( \hat{G} \) formed as follows (cf. Fig. 3). Every class \( C \) is contracted to a new basic vertex \( f(C) \), i.e. \( f(u) = f(C) \) for all \( u \in C \). Such vertices together with those vertices \( u \) not belonging to any class of \( G \) form the vertex set of \( \hat{G} \) (here, \( f(u) = u \)). An edge \( v_1v_2 \) of \( G \) generates a new edge \( f(v_1v_2) \) or no edge, in accordance with the following rules:

(a) If neither \( v_1 \) belongs to a class nor \( v_2 \) belongs to a class, then the edge \( v_1v_2 \) remains unchanged for \( G \) with the original length, i.e. \( f(v_1v_2) = v_1v_2 \).

(b) One of the vertices, say \( v_1 \), belongs to a class \( C \) and the other, i.e. \( v_2 \), belongs to no class. Then the edge \( v_1v_2 \) contains a boundary point \( x \) of \( C \) and changes to the edge \( xv_2 \) of \( \hat{G} \) with the length \( L(xv_2) = L(v_1v_2) - d_G(v_1, x) \).

(c) Let \( v_1 \) belong to a class \( C \) and \( v_2 \) belong to a class \( D \). Let \( x \) and \( y \) be the
boundary points of $C$ and $D$, respectively, with $x, y \in v_1v_2$. Then the edge $v_1v_2$ changes to the edge $xy$ of $G$ with $L(xy) = L(v_1v_2) - d_G(v_1, x) - d_G(y, v_2)$ if $L(xy) > 0$.

After $f(G)$ has been formed, all loops can be deleted. Analogously, from each bundle of parallel edges only a shortest one is important and the other can be deleted.

![Diagram](image)

**Fig. 3**

Our second heuristic method for finding a Steiner tree of a graph $G_1$ for a basic set $B_1$ is recurrent and a reduction to smaller graphs can be described as follows:

1. **Find the minimal edge length** $r_1$ of $G_1$ and form classes $C_i$.
2. **Form Steiner trees** $S_{i_1}$ for $B_1 \cap C_i$ in $G_1 \cap C_i$ with $L(S_{i_1}) \leq 2r_1(|B_1 \cap C_i| - 1)$. (This can be done because a tree on $p$ vertices has $p - 1$ edges and if for some two neighbourhoods $N(u) \cap N(v) \neq \emptyset$, then $d(u, v) \leq 2r_1$.)
3. **If there is only one class** $C_i$, stop. **Otherwise make the contraction** $f(G_1)$ of $G_1$ in $B_1$ with radius $r_1$. Put $G_2 = f(G_1)$, $B_2 = f(B_1) = \{f(v) | v \in B_1\}$, and $w_i = f(C_i)$ ($i = 1, \ldots, |B_2|$).
4. **Form a Steiner tree** $S_2$ for $B_2$ in $G_2$ (by applying this algorithm). Then combine $S_2$ and the trees $S_{i_1}$ to a single tree $S_1$ by adding no more than $\sum \deg_S(w_i)$ lines of length $r_1$. (This can be done because each boundary point of $C_i$ has the distance $r_1$ from a vertex of $B_1 \cap C_i$.)
To understand this algorithm Fig. 3 can be useful. (Put $G_1 = G$ and $G_2 = f(G).$) The heavy lines in $G_2$ form a Steiner tree $S_2$ for $B_2$ (depicted as squares) in $G_2$ and the corresponding heavy lines are also in $G_1$. Steiner trees $S_{ii}$ consist of wavy lines. Each end of a wavy line is a boundary point which in step 4 we join (if necessary) to a vertex of $S_{ii}$ by a crossed line.

Note that this algorithm allows to use any effective method which gives shorter trees $S_{ii}$ or $S_2$. Also we admit to form Steiner trees $S_{ii}$ for $B_1 \cap C_i$ not only in $G_1 \cap C_i$ (step 2) but in all $G_1$; sometimes this can give a shorter tree $S_{ii}$ (see $S_{12}$ in Fig. 3). Finally, the reader can see that a proper choice from several parallel edges in $G_2$ can provide a shorter final tree $S_1$ (cf. edges between $w_3$ and $w_4$ of Fig. 3). We do not study these questions. Unfortunately, if $|B_2| = 1$, then our algorithm reduces basically to step 2. For such cases Fig. 2 shows that the algorithm is at least as bad as that from part 2. However, as we shall see, it is not worse. We need some lemmas.

**Lemma 2.** Let $S$ be a Steiner tree for $B = \{v_1, ..., v_k\}$ in a graph $G$. If $B$ contains all endvertices of $S$, then

$$
\sum_{i=1}^{k} \text{deg}_S(v_i) \leq 2(k - 1).
$$

The proof is immediate by induction on $k$.

**Lemma 3.** Let $S_2$ and $S_1$ be Steiner trees from our algorithm. Then we have

$$
L(S_1) \leq L(S_2) + 2r_1(|B_1| - 1).
$$

**Proof.** In accordance with step 4, we can write

$$
L(S_1) \leq L(S_2) + \sum_{i=1}^{[B_2]} [L(S_{i}) + r_1 \text{deg}_{S_1}(w_i)].
$$

Using step 2 and Lemma 2, we obtain that:

$$
L(S_1) \leq L(S_2) + \sum_{i=1}^{[B_2]} 2r_1(|B_1 \cap C_i| - 1) + 2r_1(|B_2| - 1) =
$$

$$
= L(S_2) + 2r_1|B_1| - 2r_1|B_2| + 2r_1(|B_2| - 1)
$$

and the proof follows.

**Lemma 4.** Let $S_1^*$ be a Steiner minimal tree of $G_1$ for $B_1$ and let $S_2^*$ be a Steiner minimal tree of $G_2$ for $B_2$. Then

$$
L(S_1^*) \geq L(S_2^*) + (|B_1| - 1)r_1.
$$

Moreover, if $|B_2| > 1$, then

$$
L(S_1^*) \geq L(S_2^*) + |B_1|r_1.
$$

160
Proof. Let $C_1, C_2, \ldots, C_k$ be all the classes of $G$ and let

$$f(C_j) = w_j (1 \leq j \leq k).$$

First let $|B_2| = 1$. Then $k = 1$ and $L(S^*) = 0$. The sole class $C_1$ contains $|B_1|$ old basic vertices. Consequently, $S^*$ has at least $|B_1| - 1$ edges and therefore $L(S^*) \geq (|B_1| - 1) \rho$, as desired.

Now let $|B_2| > 1$. Consider a basic vertex $w_t = f(C_i)$. Let $S^*$ be the tree which arises from $S^*$ by inserting a vertex into each boundary point of $G$ which belongs to $S^*$ but is not a vertex of $S^*$. Then the part $S^*(C_j)$ of $S^*$ which belongs to $C_j$ is a graph with the length of every edge at least $\rho$. In general, $S^*(C_j)$ is a forest containing, say, $b_j$ old basic vertices. Consider a connected component $S^*(C_j)$ of it. For every basic vertex $u$ of $S^*(C_j)$, and a fixed vertex $x \in C_i \neq C_j$ there is exactly one path in the tree $S^*$ from $u$ to $x$. Assign to $u$ the first edge of the $u - x$ path. As no basic vertex $u$ is a boundary point, the edge $e$ belongs to the component $S^*(C_j)$. Obviously, if $u' \neq u$ is another basic vertex of this component, then the assigned edge $e' \neq e$. And so, every component has at least as many edges as basic vertices. Thus, the forest $S^*(C_j)$ has at least $b_j$ edges, each of which has the length at least $\rho$.

Let us denote by $f(S^*)$ the subgraph of $f(G)$ with the vertex set $f(V(S^*)) = \{f(v) | v \in V(S^*)\}$ and the edge set $f(E(S^*)) = \{f(v_1v_2) | v_1v_2 \in E(S^*)\}$. Obviously, $f(S^*)$ is connected and contains all the new basic vertices, i.e. $B_2 \subseteq f(V(S^*))$. (Note that it may contain a cycle.) Therefore, $L(f(S^*)) \geq L(S^*)$. Hence, we can write:

$$L(S^*) \geq L(f(S^*)) + \sum_{j=1}^{k} r_j b_j \geq L(S^*_2) + |B_1| \rho,$$

which completes the proof.

**Theorem 2.** If $S$ is a Steiner tree obtained by the method of contractions and $S^*$ is a Steiner minimal tree, then

$$L(S) \leq 2L(S^*).$$

**Proof.** Using Lemmas 3 and 4 $t - 1$ times, we obtain:

$$L(S^*) + \sum_{j=1}^{k-1} r_j (|B_j| - 1) \leq L(S^*) \leq L(S) \leq L(S_1) + 2 \sum_{j=1}^{k-1} r_j (|B_j| - 1).$$

Supposing that $S$ is determined in accordance with step 2, we see that $L(S) \leq 2L(S^*)$ (note that $r_j (|B_j| - 1) \leq L(S^*) \leq 2r_j (|B_j| - 1)$). Consequently, we have

$$L(S^*) \leq L(S) \leq 2L(S^*).$$

Which completes the proof.
4. Open questions

It is easy to show that both the presented algorithms are polynomial ones. So the promised aim is attained. However, we have ensured only a weak approximation of the length of a Steiner minimal tree. Therefore, a polynomial algorithm giving better Steiner trees would be of a great interest. We believe that this task will be solved in the affirmative. (The Euclidean travelling salesman problem can serve as an excellent example. Namely, Christofides [5] improved the ratio bound 2 [19] to 3/2.)

Another open question is to decide about the NP-completeness of an approximate Steiner tree problem. More precisely, does there exist such a $\varrho > 1$ that the problem of determining a Steiner tree $T$ with $L(T)/L(S) \leq \varrho$ is NP-complete? Owing to our results only $\varrho < 2$ are recommended for consideration.

REFERENCES


Received May 6, 1979

Katedra numerickej matematiky
Matematicko-fyzikálna fakulta Univerzity Komenského
Mlynská dolina
816 31 Bratislava

ОДНА ГРАНИЦА ДЛЯ ЗАДАЧИ ДЕРЕВА ШТЕЙНЕРА НА ГРАФАХ

Ян Плесник

Резюме

Под задачей нахождения дерева Штейнера понимается: Для данного подмножества вершин реберно-взвешенного графа построить кратчайшую связывающую сеть (дерево Штейнера). Приводятся два эвристических алгоритма для этой задачи. Показано, что этими алгоритмами всегда получаются деревья, которых длина не превосходит дважды взятую длину минимального дерева.