Matemática Contemporânea, Vol. 46, 265–274
http://doi.org/10.21711/231766362020/rmc4628
©2019, Sociedade Brasileira de Matemática

An algorithmic strategy to select vertices as router candidates in a Steiner tree

João Guilherme Martinez **Rosiane de Freitas**
Altigran da Silva **Fábio Protti**

Abstract

The Steiner tree problem in graphs is NP-Hard, so the development of algorithms that provide good polynomial-time solutions are desired. In this paper, we present a heuristic algorithm based on the concepts of an efficient exact enumerative algorithm proposed by Dourado *et al* (2014). Despite the existence of several approximation algorithms for this problem, we present a simple heuristic method and show that it achieves competitive results on benchmark instances from the literature, with an empirical average approximation ratio of no more than 1.01% and a maximum of 1.12%.

1 Introduction

Let $G = V, E$ be a connected undirected weighted graph with $|V(G)| = n$ vertices and $|E(G)| = m$ edges. A Steiner tree is a connected subgraph T of G such that $W \subseteq V(T)$ and $|E(T)|$ is minimum. The vertices belonging to W are called *terminal vertices* and the remaining vertices of T, i.e. the

2000 AMS Subject Classification: 05C12, 68R10 and 90C35.
Key Words and Phrases: algorithm, Steiner tree, heuristic, minimum spanning tree, shortest path.

vertices belonging to $V(T) \setminus W$, are called *Steiner vertices.* It is known that the Steiner tree problem in graphs is NP-Hard [6], so proposals for new heuristics helps the development of better approximation algorithms. The Steiner tree problem in graphs has many practical applications in the design of transport, electrical and computer networks [2, 5]. In Table 1, we present the state-of-the-art of exact and approximative algorithms for the Steiner tree problem, with their time complexity and the approximation ratio.

Table 1: Main algorithms for the Steiner tree problem in graphs. l is the number of leaves of an optimal Steiner tree and α is the number of optimal Steiner trees enumerated.

Title	**Authors**	**Year**	**Time Complexity**	**Approximation Ratio**
The Steiner problem in graphs	Dreyfus e Wagner	1971	$O(n^3 + n^2 2^{k-1} + n3^{k-1})$	1
An approximate solution for the steiner problem in graphs	Takahashi e Matsuyama	1980	$O(kn^2)$	2
A fast algorithm for Steiner trees	Kou, Markowsky e Berman	1981	$O(kn^2)$	$2(1 - 1/l)$
A faster approximation algorithm for the Steiner problem in graphs	Wu, Widmayer e Wong	1986	$O(m \log n)$	$2(1 - 1/l)$
An 11/6-approximation algorithm for the network steiner problem	Zelikovsky	1993	$O(nm + k^4)$	1.833
Improved Steiner Tree Approximation in Graphs	Robins e Zelikovsky	2000	$O(kn^2)$	1.55
Faster algorithm for optimum Steiner trees	Vygen	2011	$O(nk2^{k+\log_2 k \log_2 n})$	1
Steiner Tree Approximation via Iterative Randomized Rounding	Byrka, Grandoni, Rothvoss e SanitÃ	2013	Linear Programming	1.38
Algorithmic aspects of Steiner convexity and enumeration of Steiner trees	Dourado, Oliveira e Protti	2014	$O(n^2(n + m) + n^{k-2} + n\alpha)$	1
A Robust and Scalable Algorithm for the Steiner Problem in Graphs	Pajor, Uchoa e Werneck	2018	$O(m(min\{nk, m\}))$	experimental

Based on the exact enumerative algorithm proposed by Dourado *et al.* [3], cited in Table 1, we use some properties proved by then but replacing the more complex and costly part of their algorithm with a greedy strategy to select vertices as router candidates in the Steiner tree to be generated. The remaining of this work is organized as follows: Section 2 presents the new heuristic algorithm, some theoretical properties and its computational complexity proof; and, Section 3 is devoted to the concluding remarks and future works.

2 The proposed heuristic algorithm

Dourado *et al.* [3] propose an enumerative exact algorithm to obtain all minimum Steiner trees of a given graph in exponential time. The authors define a partition $R_T \cup L_T$ of the Steiner vertices of T as follows: $R_T = \{v \in V(T)\backslash W | deg_T(v) > 2\}$ and $L_T = \{v \in V(T)\backslash W | deg_T(v) = 2\}$. Vertices in R_T and L_T are called *routers* and *linkers* of T, respectively. A *chain* of T is a path $P = v_1v_2...v_p$ in T such that $v_1, v_p \in W \cup R_T$ and $v_i \in L_T$, for $i = 2, ..., p-1$. That is, P starts at a terminal or a router, ends at a terminal or a router, and has only linkers as internal vertices. The collection of all chains of T is denoted by $C(T)$. The Proposition 1 was taken from the original paper.

Proposition 1. *Let G be a connected graph, $W \subseteq V(G)$ and T a Steiner W-tree of G. Then: (i) Every chain in $C(T)$ is a minimum path in G; (ii) $|C(T)| = |W| + |R_T| - 1$; (iii) $|R_T| \leq |W| - 2$.*

The authors also define the weighted complete graph G_{RW} as follows: $V(G_{RW}) = W \cup R$, and the weight of an edge $uv \in E(G_{RW})$ is equal to $d_G(u,v)$ and the weight of a spanning tree T of G_{RW} as $weight(T) = \sum_{uv \in E(T)} d_G(u,v)$. So we rewrite Definition 1 from the original paper to Lemma 1.

Lemma 1. *A tree $\mathcal{T}$ is a template for W if there exists a set $R \subseteq V(G)$ such that: (a) $|R| \leq |W| - 2$; (b) $\mathcal{T}$ is a minimum spanning tree of G_{RW}; (c) $deg_T(v) > 2$ for all $v \in R$.*

In their proposed algorithm, all possible subsets of routers that can be part of the optimal solution are explored. They consider all possible subsets of $V(G) \setminus W$ with cardinality at most $k-2$, where $k = |W|$ which can be enumerated in time $O(n^{k-2})$.

The heuristic proposed in this paper analyzes all non-terminal vertices and test them as routers, one at a time, and chooses the one that generates the minimum cost template $\mathcal{T}$. However, this procedure is executed at most $k-2$ times, as this is the maximum possible amount of routers in an optimal Steiner tree [9]. Algorithm 1 shows a pseudo-code of the proposed heuristic.

Algorithm 1: Proposed heuristic algorithm for the Steiner tree in graphs.

```
   Data: Connected graph $G$, set $W$ of terminals
   Result: Steiner tree $T$
1  begin
2      $d_G \leftarrow$ all shortest paths distances between all vertices;
3      $R \leftarrow \emptyset$;
4      $G_{RW} \leftarrow$ complete graph with $W \cup R$ vertices and $d_G$ distances;
5      $\mathcal{T} \leftarrow MST(G_{RW})$;
6      $best \leftarrow cost(\mathcal{T})$;
7      $changed \leftarrow true$;
8      while $changed = true$ and $count(R) < k - 2$ do
9          $changed \leftarrow$ false;
10         for each vertex $v$ of $V(G) \setminus (W \cup R)$ do
11             $G_{RW} \leftarrow$ complete graph with $W \cup R \cup v$ vertices and
                $d_G$ distances;
12             $\mathcal{T}' \leftarrow MST(G_{RW})$;
13             $c \leftarrow cost(\mathcal{T}')$;
14             if $c < best$ and $deg_{\mathcal{T}}(v) > 2$ then
15                 $best \leftarrow c$;
16                 $\mathcal{T} \leftarrow \mathcal{T}'$;
17                 $changed \leftarrow true$;
18                 $v' \leftarrow v$;
19             end
20         end
21         if $changed = true$ then
22             $R \leftarrow R \cup v'$;
23         end
24     end
25     $T \leftarrow \mathcal{T}$ with expanded shortest paths;
26     return $T$;
27 end
```

As in the original algorithm [3], the main idea is to use a matrix d_G with the distances of all shortest paths between all vertices from G and build templates $\mathcal{T}$. By Lemma 1, we show that our algorithm can find the optimal solution.

In our heuristic algorithm, the MST procedure retrieves only one arbitrary minimum spanning tree from graph G_{RW}, given that more distinct trees would not improve the quality of the solution. Also, all non-terminal vertices are always visited at each *for loop* iteration in an arbitrary order, as the order in which they are visited does not affect the result. In its last step, the algorithm expands all the contracted shortest paths from the best $MST(G_{RW})$ to the Steiner tree solution. The computational complexity of Algorithm 1 is given as follows, by Theorem 1.

Theorem 1. *Let G be a connected graph and $W \subset V(G)$ such that $|W| = k$, for a fixed positive integer k. Algorithm 1 can generate an approximate Steiner tree of G in $O(n^3 + nk^3)$, or just $O(n^3)$ time complexity.*

Proof 1 First, we compute the distance matrix d_G in $O(n^3)$, applying the Floyd-Warshall algorithm. Next, in the nested loop, we have the process for determining the routers set R and, thus, construct the Steiner tree from it. Thus, in the WHILE loop, Line 8, the set R starts empty and with each interaction a router is included and so at the end, by Lemma 1, there will be at most $k - 2$ routers, hence $O(k)$ time complexity. Next, in the FOR loop, Line 10, we have to visit all the vertices of the graph to test without distinction (except for the terminal vertices and the vertices already included in R), wherein the end it is known who is the best vertex that should be added to the router set R. And within this innermost loop, the most costly step is in Line 12, where for each G_{RW} graph, an arbitrary minimum spanning tree is extracted. Since the G_{RW} graph is complete, its number of edges is always k^2, and therefore, regardless of the algorithm used to extract such an MST, the complexity in this step is always $O(k^2)$. Thus, we have $O(nk^3)$ for the entire nested loop, or just $O(n)$ as k is a constant. Finally, the Algorithm 1 has $O(n^3 + nk^3)$, that is, $O(n^3)$ total time complexity.

Note that if the distance matrix d_G is given (in a preprocessing), our proposed algorithm will have linear time complexity.

2.1 Empirical analysis of results

Table 2 presents the results of computational experiments in test datasets from a known benchmark [7]. The detailed description of the test bases can be found on the benchmark website[1]. The algorithm was implemented in C++ and the experiments were performed on an Ubuntu 18.04 server. We compare the solution value (the sum of the Steiner tree edge weights generated) found by the algorithm to their respective expected solution.

The proposed algorithm was able to get all the optimal solutions for the sparse graphs with Euclidian weights. In the case of complete graphs, both with random and Euclidian weights, the algorithm was able to get the optimal solution in more than 85% of the time. Exceptionally in the case of sparse graphs with random weights or incident weights, the performance of the algorithm was not so good: in the first case, it was able to get the optimal solution in a bit more than half the time; and in the second case, only in a bit more than 30% of the time. The average empirical approximation ratios found was very close to 1, which seems to be a good indicator to determine the approximation ratio, which is in progress.

Table 2: Quality of the solutions generated by our heuristic algorithm.

	Sparse with random weight	**Complete with random weight**	**Sparse with euclidian weight**	**Complete with euclidian weight**	**Sparse with incident weight**	**TOTAL**
Number of instances	60	27	14	14	395	514
Number of optimal solutions found	31	23	14	12	130	212
Percentage of optimal solutions found	51,67%	85,19%	100%	85,71%	32,91%	41,25%
Average of the approximation ratios found	1,01	1,00	1,00	1,00	1,01	1,01
Worst approximation ratio found	1,09	1,04	1,00	1,00	1,12	1,12

[1]http://steinlib.zib.de

3 Concluding remarks

The proposed heuristic algorithm presents good competitive results in terms of solution quality obtained in polynomial time, precisely in $O(n^3)$ time complexity, or even $O(n)$ if the distance matrix is given with input, while it is simpler and easier to understand than other algorithms with approximation ratio certificate, and achieves competitive results on benchmark instances, with an empirical average approximation ratio of no more than 1.01% and a maximum of 1.12%. As future work, we intend to determine the approximation ratio of our algorithm. Additionally, we shall improve the criteria adopted to select vertices as routers.

References

[1] J. Byrka, F. Grandoni, T. Rothvoss, and L. Sanità. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM*, 60(1):6:1–6:33, Feb. 2013.

[2] J.-D. Cho. *Steiner Tree Problems in VLSI Layout Designs*, pages 101–173. Springer US, Boston, MA, 2001.

[3] M. Dourado, R. Oliveira, and F. Protti. Algorithmic aspects of steiner convexity and enumeration of steiner trees. *Annals of Operations Research*, 223(1):155–171, 2014.

[4] S. E. Dreyfus and R. A. Wagner. The steiner problem in graphs. *Networks*, 1(3):195–207, 1971.

[5] D.-Z. Du and X. Hu. *Steiner Tree Problems in Computer Communication Networks.* 01 2008.

[6] R. M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972.

[7] T. Koch, A. Martin, and S. Voß. SteinLib: An updated library on steiner tree problems in graphs. Technical Report ZIB-Report 00-

37, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Takustr. 7, Berlin, 2000.

[8] L. Kou, G. Markowsky, and L. Berman. A fast algorithm for steiner trees. *Acta Informatica*, 15(2):141–145, Jun 1981.

[9] J. G. Martinez, R. de Freitas, A. da Silva, and F. Protti. Uma estratÃ©gia para selecionar vÃ©rtices como candidatos a roteadores em uma Ã¡rvore de steiner. In *Anais do III Encontro de Teoria da ComputaÃ§Ã£o*, Porto Alegre, RS, Brasil, 2018. SBC.

[10] T. Pajor, E. Uchoa, and R. F. Werneck. A robust and scalable algorithm for the steiner problem in graphs. *Mathematical Programming Computation*, 10(1):69–118, Mar 2018.

[11] G. Robins and A. Zelikovsky. Improved steiner tree approximation in graphs. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '00, pages 770–779, Philadelphia, PA, USA, 2000. Society for Industrial and Applied Mathematics.

[12] H. Takahashi and A. Matsuyama. An approximate solution for the steiner problem in graphs. In *Mathematica Japonica*, pages 24:573–577, 1980.

[13] J. Vygen. Faster algorithm for optimum steiner trees. *Information Processing Letters*, 111(21):1075 – 1079, 2011.

[14] Y. F. Wu, P. Widmayer, and C. K. Wong. A faster approximation algorithm for the steiner problem in graphs. *Acta Informatica*, 23(2):223–229, Apr. 1986.

[15] A. Z. Zelikovsky. An 11/6-approximation algorithm for the network steiner problem. *Algorithmica*, 9(5):463–470, May 1993.

João Guilherme Martinez
Instituto de Computação
Universidade Federal do Amazonas
Manaus - Brazil
joaogam@icomp.ufam.edu.br

Rosiane de Freitas
Instituto de Computação
Universidade Federal do Amazonas
Manaus - Brazil
rosiane@icomp.ufam.edu.br

Altigran da Silva
Instituto de Computação
Universidade Federal do Amazonas
Manaus - Brazil
alti@icomp.ufam.edu.br

Fábio Protti
Instituto de Computação
Universidade Federal Fluminense
Niterói - Brazil
fabio@ic.uff.br