A primal-dual approximation algorithm for the Steiner forest problem

R. Ravi

Department of Computer Science
Brown University
Providence, Rhode Island 02912

CS-93-42
September 1993

A primal-dual approximation algorithm for the Steiner forest problem

R. Ravi *

Abstract

Given an undirected graph with nonnegative edge-costs, a subset of nodes of size k called the terminals, and an integer q between 1 and k, the minimum q-Steiner forest problem is to find a forest of minimum cost with at most q trees that spans all the terminals. When $q = 1$, we have the classical minimum-cost Steiner tree problem on networks. In this note, we adapt a primal-dual approximation algorithm for the latter problem due to Agrawal, Klein and Ravi to provide one for the former. The algorithm runs in time $O(n \log n + m)$ and outputs a solution of cost at most $2(1 - \frac{1}{k-q+1})$ times the minimum. Here n and m denote respectively the number of nodes and edges in the input graph.

Keywords: Approximation algorithm, primal-dual method, Steiner tree problem.

1 Introduction

We consider a generalization of the classical Steiner tree problem in networks called the *Steiner forest problem*. We are given an undirected graph with nonnegative costs on the edges, and a subset of k nodes called *terminals*. Given an integer q between 1 and k, the goal is to find a forest of minimum cost containing at most q trees spanning all the terminals. When $m = 1$, we have the classical Steiner tree problem where a single tree of minimum cost spanning all the terminals is required.

The Steiner forest problem was introduced by Duin and Volgenant [4] motivated by the following application. A tree in the forest spanning a subset of the terminals represents establishing a service for the terminals in the tree. The cost of the tree may be taken to be the cost of providing the service. The q-Steiner forest problem corresponds to minimizing the service cost for the terminals when there is provision for providing the service using q different agents. For instance, if the costs obey the triangle inequality, the Euler tour of each tree in the forest can be short-cut into a cycle, thus providing a tour covering all the terminals in the tree. When q agents are available to perform q tours that are required to cover all the terminals at minimum cost, then a solution to the the q-Steiner forest problem on these terminals is a good starting point for this simple heuristic for the q-Steiner tour problem. Applying our heuristic for the q-Steiner tour problem gives a solution of cost at most $4(1-\frac{1}{k-q+1})$ times the minimum.

The Steiner tree problem is one of the first few problems proved NP-complete by Karp in [10]. Since then, much work has gone towards finding exact and approximate solutions to the problem [7, 8, 9, 18, 19]. Several researchers [5, 14, 16, 17] have independently provided approximation algorithms for this problem with performance ratio $2(1 - \frac{1}{k})$ where k is the number of terminals specified to be connected in the problem. There have been several efficient algorithms devised to provide such an approximation [11, 13, 20] and the best-known running time for such an algorithm is that of Mehlhorn [13]. Mehlhorn's algorithm runs in time $O(n \log n + m)$ where n and m are the number of nodes and edges in the input graph respectively. Zelikovsky [21] devised the first approximation algorithm with a better performance ratio. His algorithm outputs a solution of cost at most $\frac{11}{6}$ times the minimum. Berman and Ramaiyer [2] have generalized Zelikovsky's result to provide approximation algorithms with even better performance ratios.

*Brown University, Providence, RI 02912, USA. Email: rr@cs.brown.edu. Research supported by an IBM Graduate Fellowship. Additional support provided by NSF PYI award CCR-9157620 and DARPA contract N00014-91-J-4052 ARPA Order No. 8225.

Agrawal, Klein and Ravi [1] provided an approximation algorithm for a different generalization of the Steiner problem. They consider the *generalized Steiner forest problem* wherein given an undirected graph with nonnegative edge-costs, and a set of *site-pairs* of nodes, a subgraph of minimum cost in which each site is connected to its mate is required. They provide an approximation algorithm with performance guarantee $2(1-\frac{1}{k})$ where k is the number of nodes specified as sites in the input. They prove the performance guarantee of their algorithm by relating the cost of the approximate solution they find to the value of a packing of cuts that separate some site-pair. They argue that the value of such a packing is a lower bound on the cost of any solution; Since they find a solution of cost at most roughly twice this value, the approximation guarantee follows. Note that the Steiner tree problem is a special case of a generalized problem where each pair of terminals is specified as a site-pair (or more efficiently, a "root" terminal is chosen and for every other terminal, a site-pair consisting of the root and this terminal is specified).

We adapt the algorithm and proof technique of Agrawal, Klein and Ravi to provide an approximation algorithm for the Steiner forest problem. In addition, we use Mehlhorn's technique in implementing our algorithm efficiently. The following is our main result.

Theorem 1.1 *For every positive integer q, there is an $O(n \log n + m)$-time approximation algorithm for the q-Steiner forest problem that outputs a solution of cost at most $2(1-\frac{1}{k-q+1})$ times the minimum, where k is the number of terminals specified in the problem, and n and m are the number of nodes and edges in the input graph respectively.*

In the special case when all the nodes in the graph are terminals, it is not hard to infer that our algorithm returns an optimal solution. We call this special case the q-spanning forest problem.

Theorem 1.2 *For every positive integer q, there is an $O(n \log n + m)$-time algorithm for the q-Spanning forest problem that outputs a solution of minimum cost, where n and m are respectively the number of nodes and edges in the input graph.*

2 Background

In this section, we define and use the notion of multicuts to derive a lower bound for our problem. The input to the problem is an undirected graph $G = (V, E)$ with nonnegative edge-costs, and a subset A of terminals that must be spanned by a q-Steiner forest. Let $|A| = k$. Let the cost on edge e be denoted by $c(e)$.

Definition: A *multicut* M is defined by a partition of the vertex set V into sets $V_1, V_2, \ldots, V_l$, i.e., $V_i \cap V_j = \emptyset$ whenever $i \neq j$ and $\bigcup_i V_i = V$. The set of edges going between nodes in different sets in the partition constitute the cut. Call $V_1, \ldots, V_l$ the *blocks* of the partition. A block is termed *active* if it contains at least one terminal and its complement contains at least one terminal. Thus a block V_i is active whenever $V_i \cap A \neq \emptyset$ and $(V - V_i) \cap A \neq \emptyset$. For a subset V_i of nodes, we use $\Gamma(V_i)$ to denote the set of edges with exactly one endpoint in V_i.

The following lemma is immediate.

Lemma 2.1 *Let F denote the set of edges of a q-Steiner forest and let $M = V_1, V_2, ..., V_l$ denote a multicut with q' active blocks where $q' > q$. Then*

$$\sum_{active\ blocks\ V_i} |\Gamma(V_i) \cap F| \geq q' - q + 1.$$

We can generalize the above lemma by considering an edge-disjoint collection of multicuts.

Lemma 2.2 *Let F denote the set of edges of a q-Steiner forest and let $M_1, M_2, ..., M_l$ be edge-disjoint multicuts where M_i has $q_i (> q)$ active blocks. Then*

$$\sum_{multicuts\ M_i} \sum_{active\ blocks\ V_j \in M_i} |\Gamma(V_i) \cap F| \geq \sum_i (q_i - q + 1).$$

We can use the above lemma to provide a lower bound on the number of edges in a q-Steiner forest using multicuts with more than q active blocks in them. To generalize this notion to the case when the edges have costs, we need more definitions.

We associate a rational weight λ_P with multicut M_P. Define the *load* on an edge of the graph due to a multicut M_P with weight λ_P as follows. If an edge has endpoints in two distinct blocks V_i and V_j of P, and if both V_i and V_j are active, then the load on the edge due to the multicut M_P is defined to be $2 \cdot \lambda_P$. If only one of V_i and V_j is active, then the load on the edge due to M_P is defined to be λ_P. Otherwise, the edge has zero load due to M_P. Lemma 2.1 generalizes as follows to the weighted case.

Lemma 2.3 *Let F denote the set of edges of a q-Steiner forest and let $M = V_1, V_2, ..., V_l$ denote a multicut of weight λ_M with q' active blocks where $q' > q$. Then the cost of the edges in F is at least $(q' - q + 1)\lambda_M$.*

A *c-packing* of multicuts in a graph is a collection of multicuts $M_1, M_2, ..., M_l$, each multicut M_i associated with a rational weight λ_i, such that for every edge e, the sum of the loads on e due to all the multicuts in the collection is at most $c(e)$, the cost of the edge. Further the number of active blocks q_i in multicut M_i is greater than q for all i. Applying Lemma 2.3 , we arrive at the following generalization of Lemma 2.2.

Lemma 2.4 *Let F denote the set of edges of a q-Steiner forest and let $M_1, M_2, ..., M_l$ denote a c-packing of multicuts, where multicut M_i has $q_i (> q)$ active blocks and weight λ_i. Then the cost of the edges in F is at least $\sum_i (q_i - q + 1)\lambda_i$.*

We use the above lower bound in our analysis of the performance guarantee.

3 The algorithm

The notion of a multicut is dual to that of a q-Steiner forest. Our algorithm is primal-dual in that it runs in iterations, and in each iteration, collects a greedy c-packing of multicuts and at the same time partially constructs the Steiner forest. During the iterations, we maintain that the cost of the partial forest constructed is roughly at most twice the lower bound we collect from the multicuts using Lemma 2.4.

The multicuts are collected by growing ordinary cuts greedily starting from all the terminals in a breadth-first fashion. This way of collecting a dual solution was introduced by Agrawal, Klein and Ravi in [1]. Each of these greedy cuts around terminals is considered an active block of the partition defining a multicut in the dual solution. All the nodes of the graph not in an active block together form an inactive block in the multicut partition. If we have q' active blocks that grow for distance δ, then using Lemma 2.3, the value of the lower bound collected by the multicut defined by this partition is $(q' - q + 1)\delta$. When two growing cuts around active blocks "collide" or load an edge completely, then we merge them into a single active block of the partition. Note that the multicuts we collect all have at most one inactive block.

As we grow such active blocks to form a series of multicuts, we also maintain a tree for each active block that connects all the terminals within the block. Whenever we merge two active blocks into one, we connect the corresponding trees using a shortest path that is testimony to the collision of these two blocks. The breadth-first growth of the active blocks lends a notion of time to the running of the algorithm. At any instant in the running of the algorithm, the total distance from the start of the algorithm that any of the currently active blocks have grown for is defined to be the current time. Using this definition of time, if two active blocks collide at time t, then there is a path of length at most $2t$ between a pair of terminals, one from each of these colliding blocks. We use this path to connect the trees of the colliding blocks.

We continue until there are at most q active blocks in the multicut at which point we have at most q trees spanning all the terminals. We stop and output this q-Steiner forest as the approximate solution.

4 Performance guarantee

The proof of the performance guarantee is based on the following Lemma that we prove by using induction on the steps of the algorithm.

Lemma 4.1 *At any time t in the running of the algorithm, let LB_t denote the total lower bound accumulated until time t using the multicuts collected so far in the running of the algorithm, and let $k_t(> q)$ denote the number of active blocks in the multicut. Let F_t denote the cost of the forest built up so far. Then*

$$c(F_t) \leq 2(LB_t - (k_t - q + 1) \cdot t)$$

where $c(F_t)$ is the sum of the costs of all the edges in F_t.

Proof: The proof uses induction on the running of the algorithm. Assume that the lemma holds for any time $t' < t$. We prove it for time t. There are two distinct kinds of steps that the algorithm performs: a grow step which happens over a period of time, and a build-network step that happens at certain instants of time. We show that during both types of steps of the algorithm, the claim in the lemma is maintained.

Suppose the algorithm grows a multicut for the period from t' to t without adding any edge to the solution F. Then $F_t = F_{t'}$ and $k_t = k_{t'}$. Moreover, the increase in the lower bound is exactly $(k_t - q + 1)(t - t')$. Using this and the induction hypothesis at time t', we see that the claim in the lemma continues to hold.

The build-network step of the algorithm happens at certain instants of time. Consider such an instant t. In this step, we merge two active blocks into one thus reducing k_t by one. But at the same time we connect the spanning trees of the two merging blocks using a path of length at most $2t$. Thus assuming that the statement of the lemma holds just before the merge, it continues to hold after the merge as well since the increase in the left-hand side in the cost of the solution is at most the increase in the right-hand side due to the decrease in k_t.

This completes the inductive verification of the Lemma. □

Suppose the algorithm stops at time t_s and outputs a q-Steiner forest F, then by applying the above lemma, we have that

$$c(F) \leq 2(LB_{t_s} - t_s)$$

since $k_{t_s} = q$. Since the maximum value of LB_{t_s} is attained if all the multicuts collected have the maximum number of active blocks, we have that

$$LB_{t_f} \leq (k - q + 1) \cdot t_s .$$

Substituting above and simplifying gives the performance guarantee in Theorem 1.1.

5 Implementation

To arrive at an efficient implementation of the algorithm, observe that the algorithm may be simply restated as follows: Compute an auxiliary distance-graph on the terminals where the edge between two terminals has weight equal to the shortest path between them in the graph. Run a Minimum Spanning Tree algorithm in this graph until there are at most q connected components. Add the shortest paths corresponding to the edges added in the auxiliary graph to obtain the final q-Steiner forest. This is because the uniform breadth-first growth of the active blocks would cause exactly the shortest-paths between distinct active blocks to be picked in the same order in which they would be picked by the MST algorithm.

We can now use the algorithm of Mehlhorn [13] to compute an auxiliary graph that is equivalent to the one we described above in terms of running the MST algorithm. However, Mehlhorn showed that this alternative graph can be computed using just one single-source shortest-path computation. This takes time $O(n \log n + m)$ using Fredman and Tarjan's implementation of Dijkstra's algorithm [6]. Running the MST algorithm in this alternative graph until at most q components result and converting this to a q-Steiner forest can be accomplished in this time as well. This proves the running time in Theorem 1.1.

6 Remarks

It is not hard to see that the algorithm reduces to a truncated version of Kruskal's algorithm for MSTs [12] in the case of q-spanning forests addressed in Theorem 1.2. At any step in Kruskal's algorithm when x edges have been added to the solution, these edges form a minimum-cost set of x edges that induce an acyclic subgraph. Thus it is also easy to see that this algorithm outputs a minimum-cost solution as claimed in Theorem 1.2.

Acknowledgements

Thanks to Ravi Sundaram for a careful reading of this paper.

References

[1] A. Agrawal, P. Klein and R. Ravi, "When trees collide: an approximation algorithm for the generalized Steiner tree problem on networks," *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing* (1991), pp. 134-144.

[2] P. Berman and V. Ramaiyer, "Improved approximations for the Steiner tree problem," *Proc., 3rd Annual ACM-SIAM Symposium on Discrete Algorithms* (1992), pp. 325-334.

[3] S. E. Dreyfus, and R. A. Wagner, "The Steiner problem in graphs," *Networks, vol. 1* (1971), pp. 195-207.

[4] C. W. Duin, and A. Volgenant, "Some generalizations of the Steiner problem in graphs," *Networks, 17*, pp. 353-364, (1987).

[5] C. El-Arbi, "Une heuristique pour le problem de l'arbre de Steiner", *R.A.I.R.O. Operations Research, vol. 12* (1978), pp. 207-212.

[6] M. L. Fredman, and R. E. Tarjan, "Fibonacci Heaps and their use in improved network optimization algorithms," *IEEE* (1984), pp. 338-346.

[7] S. L. Hakimi, "Steiner's problem in graphs and its implications," *Networks, vol. 1* (1971), pp. 113-133.

[8] F. K. Hwang and D. S. Richards, "Steiner tree problems," *Networks*, Vol. 22, No. 1, pp. 55-90 (1992).

[9] A. Jain, "Probabilistic analysis of an LP relaxation bound for the Steiner problem in networks", *Networks, vol. 19* (1989), pp. 793-801.

[10] R. M. Karp, "Reducibility among combinatorial problems", in R. E. Miller and J. W. Thatcher (eds.), *Complexity of Computer Computations*. Plenum Press, New York (1972) pp. 85-103.

[11] L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for Steiner trees", *Acta Informatica, vol. 15* (1981), pp. 141-145.

[12] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem", *Proc. American Mathematical Society*, 7(1), pp. 48-50, 1956.

[13] K. Mehlhorn, "A faster approximation algorithm for the Steiner problem in graphs," *Information Processing Letters, vol. 27(3)* (1988), pp. 125-128.

[14] J. Plesnik, "A bound for the Steiner tree problem in graphs", *Math. Slovaca, vol. 31* (1981) pp. 155-163.

[15] V. J. Rayward-Smith, "The computation of nearly minimal Steiner trees in graphs", *Int. J. Math. Educ. Sci. Tech., vol. 14* (1983), pp. 15-23.

[16] G. F. Sullivan, "Approximation algorithms for Steiner tree problems", Tech. Rep. 249, Dept. of Comp. Sci., Yale Univ. (1982).

[17] H. Takahashi, and A. Matsuyama, " An approximate solution for the Steiner problem in graphs", *Math. Japonica*, vol. 24 (1980) pp. 573-577.

[18] P. Winter, "Steiner Problem in Networks : A Survey", *BIT 25* (1985), pp. 485-496.

[19] R. T. Wong, "A dual ascent approach for Steiner tree problems on a directed graph", *Math. Program.*, vol. 28 (1984) pp. 271-287.

[20] Y. F. Wu, P. Widmayer and C. K. Wong, "A faster approximation algorithm for the Steiner problem in graphs," *Acta Informatica, vol. 23* (1986), pp. 321-331.

[21] A. Z. Zelikovsky, "The 11/6-approximation algorithm for the Steiner problem on networks," *Algorithmica*, 9, pp. 463-470.