Approximation Algorithms for the Covering Steiner Problem*

Goran Konjevod[†] R. Ravi[‡] Aravind Srinivasan[§]

Abstract

The covering Steiner problem is a generalization of both the k-MST and the group Steiner problems: given an edge-weighted graph, with subsets of vertices called the groups, and a nonnegative integer value (called the requirement) for each group, the problem is to find a minimum-weight tree spanning at least the required number of vertices of every group. When all requirements are equal to 1, this becomes the group Steiner problem, while if there is only one group which contains all vertices of the graph the problem reduces to k-MST with k equal to the requirement of this unique group.

We discuss two different (but equivalent) linear relaxations of the problem for the case when the given graph is a tree and construct polylogarithmic approximation algorithms based on randomized LP rounding of these relaxations. By using a probabilistic approximation of general metrics by tree metrics due to Bartal, our algorithms also solve the covering Steiner problem on general graphs with a further polylogarithmic worsening in the approximation ratio.

1 Introduction

1.1 Problem statement

Let $G = (V, E)$ be an undirected graph with a cost function $c : E \to \mathbf{Q}^+$ defined on the edges. Let a family of subsets of V be given,

$$\mathcal{G} = \{g_1, \ldots, g_m\} \subseteq 2^V.$$

We call the sets $g_1, \ldots, g_m$ groups. In addition, for each group g_i a nonnegative integer $k_i \leq |g_i|$ is given, called the *requirement* of the group. The *covering Steiner problem*

*A preliminary version of part of this work appears in the paper "An approximation algorithm for the covering Steiner problem" by G. Konjevod and R. Ravi, in the *Proc. ACM-SIAM Symposium on Discrete Algorithms*, 2000.

[†]Dept. of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287-5406. Work done partly in the Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA 15213-3890 and Los Alamos National Laboratory, supported in part by the NSF CAREER grant CCR-9625297 and the DOE Contract W-7405-ENG-36. goran@asu.edu

[‡]GSIA, Carnegie Mellon University, Pittsburgh, PA 15213-3890. Work partly done while visiting IBM SRC, New Delhi. Supported in part by an NSF CAREER grant CCR-9625297. ravi@cmu.edu

[§]Department of Computer Science and University of Maryland Institute for Advanced Computer Studies, University of Maryland at College Park, College Park, MD 20742. Part of this work was done while at Bell Laboratories, Lucent Technologies, 600-700 Mountain Avenue, Murray Hill, NJ 07974-0636. srin@cs.umd.edu

(*covering Steiner Tree*) on G is the problem of finding a minimum-cost connected subgraph of G that contains at least k_i vertices of group g_i for all $i \in \{1, \ldots m\}$. We denote the size of the largest group by N, and the largest requirement of a group by K. We call the group vertices *terminals*.

1.2 (Non)standard notation and terminology

If $S \subseteq V$ is a set of vertices of G, we write ∂S for the set of edges with exactly one endpoint in S. If $y : E \to \mathbf{Q}$ is a function defined on a set E, we write $y(E')$ for $\sum_{e \in E'} y(e)$ for any set $E' \subseteq E$. In the sequel, we use two words—*cost* and *weight*—to mean one and the same thing.

1.3 Assumptions

In order to simplify the exposition we make a number of other assumptions. For clarity we first list all of them and then explain why each may be made with almost no loss of generality.

(1) The graph G is a weighted tree.
(2) The groups are disjoint.
(3) We know one vertex spanned by the optimal solution (we give this vertex a special role and call it *the root*).
(4) Every vertex belonging to a group has degree 1.
(5) Every vertex of degree 1 belongs to some group.

All our algorithms assume that the given graph G is a weighted tree. Unfortunately we do not know how to generalize our algorithms (or, for that matter, the linear relaxations on which the algorithms are based) to general graphs. However, the results of Bartal [3, 4] and Charikar et al. [6, 7], give a way to reduce any one from a very broad class of combinatorial optimization problems defined on a general weighted graph to the same problem on a tree. We describe this in slightly greater detail in Section 5. Briefly, this assumption costs us an extra factor $O(\log n \log \log n)$ in the approximation guarantee, where n is the number of nodes in the input graph, i.e. $n = |V(G)|$.

In some arguments, it will be convenient to assume that the groups are pairwise disjoint. There is no loss of generality: if a vertex v occurs in p groups, $p \geq 1$, attach p new vertices to v using newly created zero-cost edges. Assign each leaf of this star to one of the groups and remove v from all groups. In this new graph, the groups are disjoint sets of vertices, and there is a cost-preserving bijection between covering Steiner trees in the two graphs.

Another useful assumption is that one vertex, called the *root*, of the optimal solution is known in advance. This assumption is necessary in order to run our rounding algorithms because they are "centralized": they grow the solution subtree from a single vertex. There is no loss of generality (only loss of efficiency), because we can run the algorithm for the *rooted covering Steiner problem* $\leq N$ times, once for every possible choice of the root vertex from say the smallest group.

Finally, we assume that every vertex belonging to a group has degree 1 in G. This may be achieved by adding a new vertex for each group-vertex of degree greater than 1, connecting it to its original vertex by an edge of weight 0 and then removing the original vertex from the group. Also, it is easy to check that the problem remains unchanged if any vertex of degree 1 that does not belong to any group is removed.

1.4 Special cases

The covering Steiner problem generalizes two different NP-hard network design problems that have been studied recently. The first is the k-MST problem (see Ravi et al. [18], Fischetti et al. [9], Blum, Ravi and Vempala [5], and Garg [10]). The second is the group Steiner problem (see Reich and Widmayer [19], Garg, Konjevod and Ravi [12] and Charikar et al. [6]).

The k-MST problem is that of finding a minimum-cost connected subgraph that contains at least k nodes in an undirected graph with nonnegative edge-costs. The covering Steiner problem reduces to the k-MST problem when there is only one group and when all the vertices in V belong to this group. The best-known approximation ratio is 2, achieved by an algorithm of Garg [11] as a culmination of the series of improvements on his original 3-approximation [10]; see the papers by Arya and Ramesh [2] (giving a 2.5-approximation) and Arora and Karakostas [1] (a $(2+\epsilon)$-approximation for any fixed $\epsilon > 0$). This problem can be solved in polynomial time on trees using dynamic programming.

The group Steiner problem is the restriction of the covering Steiner problem to unit-valued group requirements. This problem is at least as hard to approximate as the set cover problem, because even the special case where the underlying graph is a star generalizes the classical set cover problem (Klein and Ravi [15]). Garg, Konjevod and Ravi [12] and Charikar et al. [6] give randomized and deterministic approximation algorithms for the group Steiner problem with (asymptotically equal) polylogarithmic approximation ratios.

Subsequent to our work, Even et al. [8] use an edge-cost-flow formulation for the covering Steiner problem and derive approximation algorithms with the same guarantees as those presented in this paper.

1.5 Results

We first describe two linear relaxations of the covering Steiner problem on a tree. We show that they are equivalent and exhibit a graph for which they have a large integrality gap (arbitrarily close to K). However, as shown in the preliminary version of this work [16], even this does not preclude using either of these relaxations as a basis for a rounding algorithm with a polylogarithmic approximation guarantee. In particular, the main result in [16] is an $O(\log N \log m \log K)$ randomized approximation algorithm for the covering Steiner problem on trees. Recall that N denotes the maximum size of a group (which is at most n, the number of nodes in G), K denotes the maximum requirement of a group (which in turn is at most N) and m denotes the number of groups. In this paper, building on this previous paper, we present an improved approximation guarantee by using the stronger relaxation and a more involved analysis. Our algorithm gives a randomized approximation for the covering Steiner problem on a tree with guarantee $O(\log N \log(mK))$. As defined earlier, N denotes the maximum size of a group (which is at most n, the number of nodes in G), K denotes the maximum requirement of a group (which in turn is at most N) and m denotes the number of groups. For the group Steiner problem with $K = 1$, this approximation ratio matches the best-known ratio. The basic idea is that we grow a covering Steiner tree in several phases; we argue that the expected *deficit* (which is related to the yet-remaining coverage required) goes down fast enough, to establish the algorithm's performance.

We then present a second algorithm which is a refinement of the first, and which gives a better approximation guarantee when the maximum requirement of a group is large compared to the number of groups. This is achieved via a more careful rounding procedure at every phase. Randomization is a key ingredient in the rounding process of all our algorithms, and a tail bound of Janson [14] helps much in our analyses.

We do not make a complete attempt at optimizing the constants in our results.

2 Linear relaxations

2.1 The first relaxation

We present an integer programming formulation of the covering Steiner problem on a tree. Let the indicator variable x_e denote whether the edge e is contained in the solution, $x_e = 1$ if $e \in T^*$ (the covering Steiner tree), and $x_e = 0$ otherwise. The cost of the solution x then equals $\sum_{e \in E} c_e x_e$, and we use this quantity as the objective function to minimize.

In addition to x we use variables y^i for $i \in \{1, \ldots, K\}$ on the edges E. We think of y^1 as supporting a unit flow from the root to one vertex of every group (whose requirement is at least 1). Similarly, y^2 supports a unit flow from the root to one vertex of every group whose requirement is at least 2. For a general $i \leq K$, we require y^i to support a unit of flow to every group whose requirement is at least i. That is,

$$\begin{aligned} &y^i(\partial S) \geq 1 \\ &\qquad \text{for all } S \subseteq V \text{ such that } r \in S \text{ and} \\ &\qquad \text{for all } i \text{ such that for some group } g, \\ &\qquad S \cap g = \emptyset \text{ and } k_g \geq i. \end{aligned} \tag{2.1}$$

Of course, x must support y^i for all i:

$$x_e \geq y^i_e \quad \text{for all } i. \tag{2.2}$$

We refer to the y^i as "commodities" to underscore the similarity of our problem's formulation to multicommodity flow problems. We are asked to build a tree that connects k_i vertices of group g_i to the root vertex r. This can be thought of as installing enough capacity on the edges of G to send a unit of flow to k_i different vertices of group g_i. The similarity ends here, however, because the capacity may be shared between the commodities, unlike the case in standard multicommodity flow.

Consider a group g with requirement k. If there is no repetition among the k vertices of group g reached by $y^1, \ldots, y^k$, then the edges given value 1 by at least one of these commodities span at least k vertices of g. However, in general, there may be vertices counted by more than one commodity $y^1, \ldots, y^k$. In order to prevent this, we strengthen the "support" constraints (2.2) and enforce

$$x_e = \sum_i y_e^i, \text{ for every edge } e \text{ incident on a terminal.} \tag{2.3}$$

The above constraints along with the integrality upper bound of one unit on the x-variables ensure that no vertex is counted in more than one commodity (to satisfy more than one unit of requirement) for its group.

The constraints (2.1), (2.2) and (2.3) together with integrality constraints

$$x_e, y_e^i \in \{0, 1\} \quad \text{for all } e \text{ and all } i$$

give an integer programming formulation of the covering Steiner tree.

We relax this integer program by allowing the variables y (and consequently x) to take on fractional values between 0 and 1. However, unnatural fractional solutions are still possible and to prevent them, we add "monotonicity" constraints:

$$x_e \geq x_f \text{ for all } (e, f) \text{ where } e \text{ is the parent edge of } f. \tag{2.4}$$

A drawback to using these constraints is that our linear relaxation is now only valid if G is a tree. This is not a problem as we have already made this assumption. The complete linear programming relaxation is summarized below.

$$\begin{aligned}
&\min \sum_{e \in E} c_e x_e \\
&y^i(\partial S) \geq 1 \\
&\qquad \text{for all } S \subset V \text{ such that } r \in S \text{ and} \\
&\qquad \text{for all } i \text{ such that for some group } g, \\
&\qquad S \cap g = \emptyset \text{ and } k_g \geq i \\
&x_e \geq y_e^i \\
&\qquad \text{for all non-pendant edges } e \\
&x_e = \sum_i y_e^i \\
&\qquad \text{for all pendant edges } e \\
&x_e \geq x_f \\
&\qquad \text{for all } (e, f) \text{ where } e \text{ is the parent edge of } f \\
&0 \leq \ x_e, y_e^i \leq 1 \ \forall (e, i).
\end{aligned} \tag{2.5}$$

Despite its exponential size, the above linear program can be solved in one of two ways: One can reformulate the minimum cut constraints using the max-flow min-cut theorem more compactly using flow variables. Alternately, one can employ the ellipsoid method [13] by supplying a polynomial-time separation oracle for the constraints, the crux of which is easily worked out to be a set of minimum cut problems. At any rate, since we will introduce and employ an alternate compact formulation later in Section 2.3, the solution of the above linear program is not critical to achieving our results.

2.2 Integrality gap

Even for the version of the problem with a single group (the k-MST problem), this relaxation is not tight. For instance, let G be the tree in Fig. 1, a star with $k-1$ leaves whose center is connected to another star with k leaves by a single edge. Let the root vertex be the center of the first star. Denote the first star by A, the second by B, and the edge joining them by e_0. The single group consists of all the leaves of G and its requirement is k.

Consider the solution to the linear program where each of the k commodities sends $1/k$ of flow from the root to each of the $k-1$ leaves of A and where each commodity sends $1/k$ of flow to a distinct leaf of B. The packing constraints force $x_e = 1$ for all edges e in A, but since only one commodity is served by every edge of B, $x_f = 1/k$ for all edges f of B and for the edge e_0.

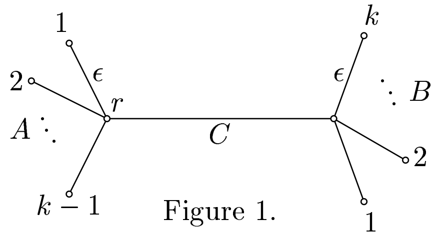

Figure 1.

Let the cost of the edges belonging to the stars A and B be ϵ, and the cost of edge e_0 be C. Clearly, the cost of the optimal tree that contains the root and covers k terminals is at least $C + k\epsilon$, since at least one of the leaves of B must be included. However, the relaxed solution described above costs only $C/k + k\epsilon$. Thus, the ratio between the optimal integral and fractional solutions of the relaxation (2.5) can be arbitrarily close to k.

2.3 Another relaxation

We now propose a relaxation which is simpler but equivalent to (2.5). In the integer programming formulation, there is an indicator variable x_e for each edge e of G, to say whether e is chosen or not and thus the objective is again to minimize $\sum_e c_e x_e$.

For any non-root node u, let $\mathrm{pe}(u)$ denote the edge connecting u to its parent. Similarly, for any edge e not incident on the root, let $\mathrm{pe}(e)$ denote the parent edge of e. Also, given an edge $e = uv$ where u is the parent of v, both $T(v)$ and $T(e)$ denote the subtree of G rooted at v. Let k_g denote the requirement of group g.

To reach sufficiently many vertices we require that

$$\sum_{j \in g} x_{\mathrm{pe}(j)} = k_g \quad \text{for every group g.} \tag{2.6}$$

Consider an edge e and the subtree $T(e)$ below this edge. If e is included in the covering Steiner tree then up to k_g vertices of g may be reached in $T(e)$; if e is not included in the solution then no vertex of g will be reached in $T(e)$. Thus the constraint

$$\sum_{j \in (T(e) \cap g_i)} x_{\mathrm{pe}(j)} \leq k_g x_e \tag{2.7}$$

is valid for any edge e and any group g. This constraint will be crucial later in bounding the parameter in Janson's inequality.

Finally, we still impose the monotonicity constraints (2.4):

$$x_{\mathrm{pe}(e)} \geq x_e \quad \text{for all edges } e \text{ not incident on the root.} \tag{2.8}$$

By allowing each x_e to lie in $[0,1]$, we get our second LP relaxation of the covering Steiner problem.

$$\begin{aligned} \min \sum_{e \in E} c_e x_e & \\ \sum_{j \in g} x_{\mathrm{pe}(j)} &= k_g \\ & \text{for every group g,} \\ \sum_{j \in (T(e) \cap g_i)} x_{\mathrm{pe}(j)} &\leq k_g x_e, \\ & \text{for every edge } e \text{ and every group } g \\ x_{\mathrm{pe}(e)} &\geq x_e \\ & \text{for every edge } e \text{ not incident on } r, \\ 0 \leq \ x_e &\leq 1 \\ & \text{for every edge } e. \end{aligned} \tag{2.9}$$

We now show that relaxations (2.9) and (2.5) are equivalent.

Theorem 2.1. *Let x be a feasible solution to LP (2.9). Then there exist $y^1, \ldots y^K$ such that (x, y) is a feasible solution to LP (2.5). Conversely, given a solution (x, y) to LP (2.5), it is also feasible for LP (2.9).*

Proof. First we show that LP (2.9) is at least as strong as LP (2.5). Let g be a group with requirement k_g. Let $v_1, \ldots v_j \in g$ be all vertices of g spanned by the support tree of x. Define

$$w_e^{(g)} = x_e / k_g \tag{2.10}$$

for every edge of the form $e = \mathrm{pe}(v_i)$, and

$$w_e^{(g)} = 0 \tag{2.11}$$

for every other edge e incident on a terminal.

Define $w^{(g)}$ for the other edges working bottom-up from the leaves along the tree G: if $w^{(g)}$ has been defined for all children of e, let $w_e^{(g)} = \sum_f w_f^{(g)}$, where the sum is taken over all the children of e.

Our first claim is that

$$w_e^{(g)} \leq x_e$$

for all $e \in E$. In fact, we prove

$$w_e^{(g)} = \frac{1}{k_g} \sum_{j \in (T(e) \cap g)} x_{\mathrm{pe}(j)},$$

and the claim then follows from the subtree constraint (2.7) for edge e.

The proof is by induction on the number of edges in the longest path "descending" from e to a leaf of G.

By (2.10) and (2.11), the claim holds for the edges incident on terminals.

Consider an edge $e \in E$. By the induction hypothesis, the claim is true for all children edges of e: if $e = \mathrm{pe}(f)$ then $w_f^{(g)} = \frac{1}{k_g} \sum_{j \in (T(f) \cap g)} x_{\mathrm{pe}(j)}$. Now

$$w_e^{(g)} = \sum_f w_f^{(g)} = \frac{1}{k_g} \sum_{j \in (T(e) \cap g)} x_{\mathrm{pe}(j)},$$

proving the claim.

After defining $w^{(g)}$ for every group g, we set $y_e^i = \max_{g:\, k_g \geq i} w_e^{(g)}$ for all i and e. Since $x_e \geq w_e^{(g)}$ for all g and e, the pair (x, y) satisfies the support constraints (2.2). By definition of w for terminal edges, all strengthened support constraints (2.3) are satisfied. Constraints (2.4) and (2.8) are identical. Finally, note that by the definition of w, $w^{(g)}$ supports a unit flow from r to the terminals of the group g. Therefore, the capacity (as defined by $w^{(g)}$) of a minimum cut separating the root from the vertices of g is at least 1. Thus the pair (x, y) satisfies cut-covering constraints (2.1), and is thus a feasible fractional solution for LP (2.5).

Now, consider a feasible solution (x, y) to LP (2.5). Let g be a group with requirement k and let e be an edge in the tree. Then $y_e^1, \ldots y_e^k$ denote the "flow" values on the edge e of commodities supporting group g. Now we have

$$\sum_{j \in T(e) \cap g} x_{\mathrm{pe}(j)} = \sum_{j \in T(e) \cap g} \sum_{i=1}^{k} y_{\mathrm{pe}(j)}^i \leq \sum_{i=1}^{k} y_e^i \leq k x_e;$$

that is, constraints (2.7) are satisfied by x. The justification is as follows. The equality follows from (2.3). The first inequality follows because: (1) all vertices of g are leaves of the tree, and so no two edges of the form $\mathrm{pe}(j)$, where $j \in T \cap g$, are contained in any single path from the root to a terminal of g, and (2) the values y^i form a flow originating from the root. The second inequality follows from constraints (2.2). All other constraints of LP (2.9) are obviously satisfied by (x, y) and so x is a feasible solution to LP (2.9).

$\square$

3 The main algorithm

We will proceed in phases, with each phase satisfying a part of the coverage requirements. Suppose a subset S_i of g_i has already been chosen (initially, $S_i = \emptyset$). We will work with $g_i' = g_i - S_i$, and the remaining requirement of the ith group is $r_i = k_i - |S_i|$.

The constraints of the linear program for the residual problem in this more general notation become

$$\sum_{j \in g'_i} x_{\mathrm{pe}(j)} = r_i \quad \text{for every group } g'_i, \tag{3.12}$$

$$x_{\mathrm{pe}(e)} \geq x_e \quad \text{for every edge } e \text{ not incident on the root} \tag{3.13}$$

$$\sum_{j \in (T(e) \cap g'_i)} x_{\mathrm{pe}(j)} \leq r_i x_e \quad \text{for every edge } e \text{ and every group } g'_i. \tag{3.14}$$

Suppose we are given an optimal solution $(x_e : e \in E)$ for this LP. We use the rounding procedure described in [12]. For every edge e incident on the root, include e in the set of edges to be added to the solution with probability x_e. For every other edge e with its parent denoted f, "round up" e with probability x_e/x_f. This experiment is performed for each edge of G independently. Let H denote the subgraph of G induced by the edges that were "rounded up". Discard all connected components of H except the one containing the root, and denote the resulting tree by T. We choose all edges of T in our solution. This constitutes one phase of "relax-and-round". We repeat such phases until all groups have their coverage satisfied. Also, after each phase, we reset the costs of all chosen edges to zero, so as to not count their costs in future phases.

Consider the generic phase described above. It is not difficult to see that the expected cost of T is equal to the cost of the initial linear programming solution (which in turn is at most the optimal objective function value of the original integer programming problem). This can be seen as follows: An edge e is included in T if and only if all the edges in the path from r to e, say $e_1, \ldots, e_p, e$ are picked in their respective independent random experiments, the probability of which is

$$\frac{x_{e_1}}{1} \cdot \frac{x_{e_2}}{x_{e_1}} \cdots \frac{x_e}{x_{e_p}} = x_e. \tag{3.15}$$

To analyze our random process and to show that we do not need too many phases, we state and use a probabilistic inequality. Let Ω be a universal set, and $R \subseteq \Omega$ determined by the experiment in which each element $r \in \Omega$ is independently included in R with probability p_r. Let $\{A_i \mid i \in I\}$ be a family of subsets of Ω, and denote by B_i the event that $A_i \subseteq R$. Write $i \sim j$ if $i \neq j$ and $A_i \cap A_j \neq \emptyset$. Define $\Delta = \sum_{i \sim j} \Pr[B_i \cap B_j]$ (the sum is over ordered pairs). Let $X = \sum_i X_i$, where X_i is an indicator variable for the event B_i, let $\mu_i = \mathbf{E}[X_i] = \Pr[B_i]$ and $\mu = \mathbf{E}[X] = \sum_i \mu_i$. Finally, let e denote the base of the natural logarithm.

Theorem 3.1. *(Janson's inequality [14].) With the notation as above and with* $0 \leq \delta \leq 1$,

$$\Pr\big[X \leq (1-\delta)\mu\big] \leq \mathrm{e}^{-\delta^2 \mu / (2 + \frac{\Delta}{\mu})}.$$

We will use Theorem 3.1 to show that we get a guaranteed amount of coverage from any fixed group, with non-neglibile probability. Let us fix a group g_i. Then, in the setting of Theorem 3.1, we have $\Omega = E(G)$ and $p_e = x_e/x_{\mathrm{pe}(e)}$, where $\mathrm{pe}(e)$ is the parent edge of e. Each subset A_j is the edge-set of a path from r to a leaf belonging to g_i. Thus, X is the random variable denoting the amount of g_i's residual requirement

r_i that a generic phase covers. The argument underlying (3.15) easily helps show that $\mu_i = r_i$. Thus, the key issue is to upper-bound the parameter $\Delta_i = \Delta$ of Theorem 3.1. Suppose $j, j' \in g'_i$ (recall that $g'_i = g_i - S_i$, where S_i is the set of elements of g_i that have been covered in previous phases). We will say that $j \sim j'$ if and only if **(i)** $j \neq j'$ and **(ii)** the least common ancestor of j and j' in G is not the root r. If $j \sim j'$, let $lca(j, j')$ denote the least common ancestral *edge* of j and j' in T'. A little reflection shows that

$$\Delta_i = \sum_{j,j' \in g'_i:\ j \sim j', x_{lca(j,j')} > 0} \frac{x_{pe(j)} x_{pe(j')}}{x_{lca(j,j')}}.$$

We aim to show

Theorem 3.2. $\Delta_i \leq r_i(r_i - 1 + r_i \ln(|g'_i|/r_i))$. *In particular,* $\Delta_i \leq k_i(k_i - 1 + k_i \ln(|g_i|))$.

The proof of this useful theorem is shown next in Section 3.1.

3.1 Proof of Theorem 3.2

We use a technical result.

Lemma 3.3. *If* $x_{pe(j)} > 0$, *then*

$$x_{pe(j)} \cdot \sum_{j' \in g'_i:\ j \sim j'} \frac{x_{\mathrm{pe}(j')}}{x_{lca(j,j')}} \leq x_{\mathrm{pe}(j)}(r_i - 1 + r_i \ln(1/x_{\mathrm{pe}(j)})).$$

Lemma 3.3 suffices to prove Theorem 3.2. To see this, first note that the function $z \mapsto z(r_i - 1 + r_i \ln(1/z))$ is concave for $z > 0$, since its second derivative is non-positive. Thus, since $\sum_{j \in g'_i} x_{\mathrm{pe}(j)} = r_i$, Lemma 3.3 shows that

$$\Delta_i \leq |g'_i| \cdot (r_i/|g'_i|) \cdot (r_i - 1 + r_i \ln(|g'_i|/r_i)),$$

as required.

We now prove Lemma 3.3. Suppose $x_{\mathrm{pe}(j)} = z \in \langle 0, 1]$. We need some extra notation. Let $e_0, e_1, \ldots, e_t$ be the sequence of edges that we encounter as we walk up the tree starting from j; let $y_\ell = x_{e_\ell}$. Thus we have $z = y_0 \leq y_1 \leq y_2 \leq \cdots \leq y_t \leq 1$. Next, for $\ell = 0, 1, \ldots, \ell$, let $S_\ell = \sum_{j' \in (T(e_\ell) \cap g'_i)} x_{\mathrm{pe}(j')}$. Then, it is not hard to see that the left-hand side in the statement of the lemma equals

$$z \cdot \sum_{\ell=1}^{t} \frac{S_\ell - S_{\ell-1}}{y_\ell}. \tag{3.16}$$

The sum in (3.16) is clearly bounded by the maximum of the following optimization problem, whose variables are the y_ℓ and S_ℓ. (The optimization problem has a maximum since the domain is a polytope and since the objective function is continuous in the domain.)

$OPT(z, t)$: maximize $\sum_{\ell=1}^{t} \frac{S_\ell - S_{\ell-1}}{y_\ell}$ subject to

$$\begin{aligned}
S_0 &= z; \\
y_0 &= z; \\
y_t &\leq 1; \\
S_\ell &\leq S_{\ell+1}, \quad \ell = 0, 1, \ldots, t-1; \\
y_\ell &\leq y_{\ell+1}, \quad \ell = 0, 1, \ldots, t-1; \\
S_\ell &\leq r_i y_\ell, \quad \ell = 0, 1, \ldots, t. \qquad (3.17)
\end{aligned}$$

Constraint (3.17) holds since (3.14) is a valid constraint in our problem formulation and in the LP relaxation.

Let $\{y_\ell, S_\ell : \ell \geq 0\}$ be any feasible solution to the above optimization problem. We now bound the objective function value v of this solution. We have

$$\begin{aligned}
v &= S_t/y_t - S_0/y_1 + \sum_{\ell=1}^{t-1} S_\ell \cdot (1/y_\ell - 1/y_{\ell+1}) \\
&\leq r_i - z/y_1 + \sum_{\ell=1}^{t-1} S_\ell \cdot (1/y_\ell - 1/y_{\ell+1}) \\
&\leq r_i - z/y_1 + \sum_{\ell=1}^{t-1} r_i y_\ell \cdot (1/y_\ell - 1/y_{\ell+1}) \\
&= r_i - z/y_1 + r_i \sum_{\ell=1}^{t-1} (1 - y_\ell/y_{\ell+1}). \qquad (3.18)
\end{aligned}$$

Note the use of constraints (3.14) in the second inequality above.

Take any ℓ, $2 \leq \ell \leq t-1$. If we keep all variables but y_ℓ fixed, we see that (3.18) is maximized when $y_\ell = \sqrt{y_{\ell-1} y_{\ell+1}}$, i.e., when $y_{\ell-1}/y_\ell = y_\ell/y_{\ell+1}$. Thus, for any fixed choice of y_1 and y_t, (3.18) is maximized when $y_1, y_2, \ldots, y_t$ are in geometric progression. Therefore, if we fix y_1 and y_t and let $\psi = (y_1/y_t)^{1/(t-1)}$, we have

$$v \leq r_i - z/y_1 + r_i(t-1)(1-\psi). \qquad (3.19)$$

Now, $\psi \geq y_1^{1/(t-1)} = \mathrm{e}^{(\ln y_1)/(t-1)} \geq 1 + (\ln y_1)/(t-1)$. Thus,

$$v \leq r_i - z/y_1 + r_i \ln(1/y_1). \qquad (3.20)$$

Subject to $y_1 \in [z, 1]$ and $r_i \geq 1$, the r.h.s. of (3.20) is maximized when $y_1 = z$. Thus, $v \leq r_i - 1 + r_i \ln(1/z)$, concluding the proof.

Remark. In the preliminary version of this work [16], a bound on Δ_i that is weaker by a constant factor is shown using a different approach. That approach has the property of letting us assume that the tree has "small" depth, which is useful in some other contexts [20].

3.2 Analysis

Suppose some ℓ iterations of "relax-and-round" have been run, with remaining groups g'_i and residual requirements r_i. Run the $(\ell+1)$st iteration. For each i, let X_i be the number of elements of g'_i covered in the above randomized rounding of the $(\ell+1)$st iteration; the *deficit* D_i is $\max\{r_i - X_i, 0\}$. Since $\mu_i = r_i$ and $\Delta_i/\mu_i \le r_i(1+\ln(g_i))$, Theorem 3.1 gives for any $\delta \in [0,1]$ that

$$\Pr[D_i \ge r_i\delta] = \Pr[X_i \le (1-\delta)\mu_i] \le \mathrm{e}^{-\delta^2 r_i/(2+r_i(1+\ln(g_i)))}.$$

Now, if $y \in [0,1]$, $\mathrm{e}^{-y} \le 1-(1-1/\mathrm{e})y$. Thus,

$$\Pr[D_i \ge r_i\delta] \le 1-\gamma\delta^2, \tag{3.21}$$

where

$$\gamma = \Theta(1/\log N). \tag{3.22}$$

So, since D_i is an integer taking values in $[0, r_i]$,

$$\mathbf{E}[D_i] = \sum_{j=1}^{r_i} \Pr[D_i \ge j] \le \sum_{j=1}^{r_i} (1-\gamma j^2/r_i^2) \le r_i(1-\gamma/3);$$

Linearity of expectation yields

$$\mathbf{E}[\sum_i D_i] \le (1-\gamma/3)\sum_i r_i. \tag{3.23}$$

Now let Y_ℓ be the total residual requirement after ℓ iterations; Y_0 is deterministic, and has value $\sum_i k_i$. We see from (3.23) that for any $y > 0$,

$$\mathbf{E}[Y_{\ell+1} | (Y_\ell = y)] \le (1-\gamma/3)y.$$

Hence, $\mathbf{E}[Y_{\ell+1}] \le (1-\gamma/3)\mathbf{E}[Y_\ell]$. Induction gives $\mathbf{E}[Y_\ell] \le (1-\gamma/3)^\ell \sum_i k_i$. Choosing $\ell = \ell_0 \doteq \lceil (3/\gamma)\cdot \ln(2\sum_i k_i)\rceil$, we get $\mathbf{E}[Y_{\ell_0}] \le 1/2$. Thus, by Markov's inequality,

$$\Pr[Y_{\ell_0} \ge 1] \le 1/2. \tag{3.24}$$

Also, as argued just before (3.15), the expected total cost C_ℓ of the edges rounded in each iteration ℓ is at most OPT. Thus, $\mathbf{E}[\sum_{\ell=1}^{\ell_0} C_\ell] \le \ell_0 \cdot OPT$. Markov's inequality implies, e.g., that

$$\Pr[\sum_{\ell=1}^{\ell_0} C_\ell \ge 2.1\ell_0 \cdot OPT] \le 1/(2.1). \tag{3.25}$$

So, by (3.24) and (3.25), there is a probability of at least $1/2 - 1/(2.1)$ that after ℓ_0 iterations, all requirements have been satisfied, and that the total cost of the tree produced is at most $2.1\ell_0 \cdot OPT$. Note from (3.22) that $\ell_0 = O((\log N)\cdot(\log(Km)))$. Thus we get

Theorem 3.4. *There is a randomized polynomial-time approximation algorithm for the covering Steiner problem on trees, which with constant probability produces a solution of value at most $O((\log N)\cdot(\log(Km)))$ times optimal.*

4 Large requirements—the second algorithm

In this section, we present an approximation algorithm with approximation guarantee

$$O\left(\frac{(\log N)\cdot\log^2 m}{\log(2(\log N)\cdot(\log m)/\log K)}\right). \tag{4.26}$$

This bound is better than that of Theorem 3.4 if, e.g., $K \geq 2^{a(\log m)^2}$ where $a > 0$ is a certain absolute constant. The main idea behind the refinement in this algorithm is to partition the terminals from a group more carefully based on the support values on their parent edges: the part with the largest support values are rounded (as are the other edges on the path to the root) without much increase in cost since these edges have sufficient support value to begin with. When the rounding does not achieve rapid progress with this part, the remaining support edges are boosted in their fractional value and rounded; when the number of groups is small, this boosted rounding also ensures rapid progress leading to an overall smaller number of rounding steps.

As in Section 3, suppose we have run ℓ iterations of "relax-and-round", and that the residual version of g_i is g_i', with remaining requirement r_i. Call i *active* iff $r_i \neq 0$. As described in Section 3, we solve the LP relaxation for the residual instance to get a vector $(x_e : e \in E)$. Let $\lambda > 1$ be a parameter that is $\Theta((\log N)\cdot(\log m))$; its actual value is defined by (4.31). For certain positive constants a_0, a_1 such that $a_0 + a_1 < 1$ and $a_1 \geq \lambda^{-1}$, we define the following. For each active i, partition g_i' into three sets:

$$\begin{aligned} S_{i,1} &= \{j \in g_i' :\ x_{\mathrm{pe}(j)} \geq a_1\}, \\ S_{i,2} &= \{j \in g_i' :\ x_{\mathrm{pe}(j)} \in (\lambda^{-1}, a_1)\} \quad \text{and} \\ S_{i,3} &= \{j \in g_i' :\ x_{\mathrm{pe}(j)} \leq \lambda^{-1}\}. \end{aligned}$$

Also, for any vector $w = (w_e : e \in E)$ and for $t = 1,2,3$, define $F_{i,t}(w) = \sum_{j \in S_{i,t}} w_{\mathrm{pe}(j)}$.

Let i be any active index. We will say that i is *Type A* iff $F_{i,1}(x) \geq a_0 r_i$; otherwise we say that i is *Type B*. Also let $0 < z < 1$ be a parameter to be defined later. We are now ready to describe our rounding in the current, i.e., $(\ell+1)$st, iteration. There are two cases:

Case I: *At least a z fraction of the currently active groups are of Type A*. In this case, our rounding is the following simple *deterministic* scheme: choose an edge e iff $x_e \geq a_1$.

Case II: This is the complement of Case I. In this case, we set $x_e' = x_e \cdot \min\{\lambda, 1/x_e\}$, and run the randomized rounding scheme of Section 3 using these new values x'. (More precisely, we repeat this randomized rounding algorithm independently a sufficient—$O(\log n)$—number of times so that a certain property holds whp; see Section 4.1.) Note that

$$\forall i\ \forall j \in (S_{i,1} \cup S_{i,2}),\ x'_{\mathrm{pe}(j)} = 1. \tag{4.27}$$

As described above, the main idea behind this improved algorithm is to argue that in Case II, the boosted probabilities used in rounding allow us to use fewer rounding iterations overall than in the previous case. In this analysis, the second of the three sets of support edges for a group is used to dispose of an easy case - intuitively, when $F_{i,2}(x)$ is a constant fraction of the requirement r_i, the boosting immediately ensures (as in Case I) that we are making rapid progress in coverage. In the remaining case, a direct application of Janson's inequality (Theorem 3.1) with the boosted probability shows that few repetitions of rounding are sufficient to finish covering all active indices in such iterations.

4.1 Analysis of the rounding

Let us first upper-bound the total cost incurred by the iterations in which Case I held. It is easily seen that in each such iteration, the total cost of the edges chosen is at most OPT/a_1.

We next bound the number of iterations in which Case I could have been true. The idea is roughly as follows: In every iteration in which Case I held, a z-fraction of active groups all have an a_0 fraction of their terminal support values in the first partition (with support value at least a_1). Total deficit thus reduces roughly by a fraction of za_0 in every Case I iteration. Since we start with total deficit at most mK, the number of iterations is roughly $O(\frac{\ln mK}{\ln(1/za_0)})$ and the cost incurred per iteration by rounding up edges with original support values at least a_1 is a $\frac{1}{a_1}$ factor. We do this more formally below.

Let s be an integer with $0 \leq s \leq \lceil \ln m \rceil$. Let I_s be the sequence of iterations (arranged in increasing order) in which the number of active indices was in the range $(\mathrm{e}^{s-1}, \mathrm{e}^s]$, **and** in which Case I was true; we will now bound $|I_s|$. Consider any i that was active at the beginning of I_s. Equation (4.27) shows that for every iteration in I_s in which i was Type A, at least an a_0 fraction of g_i's requirement is satisfied; so i could have been Type A in at most $\lceil (\ln K)/(\ln(1/a_0)) \rceil$ iterations in I_s. Since each iteration in I_s had at least e^{s-1} active indices, at least $z\mathrm{e}^{s-1}$ active indices are of Type A in each iteration in I_s. So, since there were at most e^s active indices at the beginning of I_s, we can check that

$$|I_s| \leq \frac{\mathrm{e}}{z} \cdot \left\lceil \frac{\ln K}{\ln(1/a_0)} \right\rceil.$$

Summing over the $(1 + \lceil \ln m \rceil)$ possible values of s and recalling that each "Case I" iteration incurs a cost of at most OPT/a_1, we get

$$\text{The total cost from "Case I" iterations is at most } \frac{\mathrm{e}}{a_1 z} \cdot \left\lceil \frac{\ln K}{\ln(1/a_0)} \right\rceil \cdot (1 + \lceil \ln m \rceil) \cdot OPT. \tag{4.28}$$

We now analyze Case II. Fix an iteration in which Case II held; As before, let r_i be the residual demand of group g_i. The terms "Active", "Type B" etc. below, refer to these predicates at the beginning of this iteration.

We aim to show that if λ is chosen large enough, then the residual demands of *all* active Type B indices will be satisfied by this iteration whp. We first dispose of an easy case. Recall that $a_0 + a_1 < 1$. Consider any active i of Type B, for which $F_{i,3}(x) \leq (1 - a_0 - a_1) r_i$. Since i was Type B, we have $F_{i,1}(x) < a_0 r_i$. However, $F_{i,1}(x) + F_{i,2}(x) + F_{i,3}(x) = r_i$. So, we must have $F_{i,2}(x) \geq a_1 r_i$. Now, we can check that for each $j \in S_{i,2}$, $x'_{\mathrm{pe}(j)} = 1 \geq x_{\mathrm{pe}(j)}/a_1$. Therefore, since $F_{i,2}(x) \geq a_1 r_i$, all of group g_i's residual demand will get satisfied with probability 1, by this iteration.

So, we can just focus on those Type B indices that are not covered by the above easy case. Define i to be relevant if it is active, Type B, and has $F_{i,3}(x) > (1 - a_0 - a_1)r_i$. We will now show that if λ is chosen large enough, then the residual demands of all relevant indices will be satisfied by this iteration whp. To do this, we will actually prove the following. For each edge e, let Y_e be the indicator random variable for whether e is chosen by this iteration or not. We will show that whp, $F_{i,3}(Y) \geq r_i$ for all relevant indices i.

Fix a relevant i. To prove that $\Pr[F_{i,3}(Y) \geq r_i]$ is sufficiently high, we will use Janson's inequality (Theorem (3.1)). To do so, we now modify the definition of the relation $\sim$, and also redefine μ_i and Δ_i. Suppose j and j' belong to $S_{i,3}$. We will say that $j \sim j'$ iff: (i) $j \neq j'$ and (ii) $x'_{lca(j,j')} < 1$. [Note that this requirement (ii) is different from before; this is crucial for our bound (4.30) on Δ'_i.] Define

$$\mu'_i = \mathbf{E}[F_{i,3}(Y)] = F_{i,3}(x') = \lambda F_{i,3}(x) > \lambda(1 - a_0 - a_1)r_i, \text{ and} \tag{4.29}$$
$$\Delta'_i = \sum_{j,j' \in S_{i,3}:\ j \sim j'} \frac{x'_{\mathrm{pe}(j)} x'_{\mathrm{pe}(j')}}{x'_{lca(j,j')}}.$$

For each $j, j' \in S_{i,3}$, we have $x'_{\mathrm{pe}(j)} = \lambda x_{\mathrm{pe}(j)}$ and $x'_{\mathrm{pe}(j')} = \lambda x_{\mathrm{pe}(j')}$ by definition of $S_{i,3}$. The definition of $\sim$ also implies that if $j \sim j'$, then $x'_{lca(j,j')} = \lambda x_{lca(j,j')}$. So, Theorem 3.2 yields

$$\Delta'_i \leq c r_i \mu'_i \log N \tag{4.30}$$

for some constant c. Let $\exp(y)$ denote e^y. Theorem 3.1 gives, for any $\delta \in [0,1]$, that

$$\Pr[F_{i,3}(Y) \leq \mu'_i(1-\delta)] \leq \exp(-\delta^2 \mu'_i/(2 + \Delta'_i/\mu'_i)).$$

Then, using (4.29) and (4.30), we derive the bound

$$\begin{aligned}\Pr[F_{i,3}(Y) < r_i] &\leq \Pr\left[F_{i,3}(Y) < \frac{\mu'_i}{\lambda(1-a_0-a_1)}\right] \\ &\leq \exp\left(-\frac{\lambda(1-a_0-a_1)}{2(1+c\log N)} \cdot \left(1 - \frac{1}{\lambda(1-a_0-a_1)}\right)^2\right).\end{aligned}$$

Choosing

$$\lambda = \frac{3}{1-a_0-a_1} \cdot (1 + c\log N) \cdot \ln(2m), \tag{4.31}$$

we get $\Pr[F_{i,3}(Y) < r_i] \leq 1/(2m)$. Thus,

$$\Pr[\text{there is some Type B index that is not completely covered}] \leq 1/2. \tag{4.32}$$

It is also easy to see that the expected total cost of the edges chosen is at most $\lambda \cdot OPT$; Markov's inequality shows that the probability of this cost being more than, e.g., $2.1 \cdot \lambda \cdot OPT$ is at most $1/(2.1)$. So, we have from (4.32) that with at least the constant probability of $1/2 - 1/(2.1)$, the chosen edges cover all the Type B indices, and have a total cost of at most $2.1 \cdot \lambda \cdot OPT$. This probability can be amplified to, say, $1 - 1/n^2$ by repeating this process $O(\log n)$ times. Thus, since each "Case II" iteration covers at least an $(1 - z)$ fraction of the currently active indices, at most $\left\lceil \frac{\ln m}{\ln(1/z)} \right\rceil$ such iterations need to be run, whp. Therefore,

$$\text{Whp, the total cost from "Case II" iterations is at most } 2.1 \cdot \left\lceil \frac{\ln m}{\ln(1/z)} \right\rceil \cdot \lambda \cdot OPT. \tag{4.33}$$

Thus, the total cost is whp at most the sum of the quantities from (4.28) and (4.33). We choose $a_0 = a_1 = 1/3$ and

$$z = \min\left\{ \frac{(\log K) \cdot \log(2(\log N) \cdot (\log m)/\log K)}{(\log N) \cdot \log m}, \frac{1}{2} \right\},$$

though again a more careful choice of the constants is possible. This completes the analysis and gives the following theorem.

Theorem 4.1. *There is a randomized polynomial-time approximation algorithm for the covering Steiner problem on trees, which with constant probability produces a solution of value at most* $O(\frac{(\log N) \cdot \log^2 m}{\log(2(\log N) \cdot (\log m)/\log K)})$ *times optimal.*

5 Extensions

5.1 General metrics

Definition 5.1. *A set of metric spaces $\mathcal{S}$ over V is said to α-probabilistically approximate a metric space M over V, if* **(1)** *for all $x, y \in V$ and $S \in \mathcal{S}$, $d_S(x, y) \geq d_M(x, y)$, and* **(2)** *there exists a probability distribution D over metric spaces in $\mathcal{S}$ such that for all $x, y \in V$, $\mathbf{E}[d_D(x, y)] \leq \alpha d_M(x, y)$.*

Bartal [3, 4] proved the following theorem.

Theorem 5.2. *Every weighted connected graph G on n vertices can be α-probabilistically approximated by a set of weighted trees, where $\alpha = O(\log n \log\log n)$. Moreover, we can sample from the probability distribution in polynomial time.*

The trees that we get from Bartal's algorithm are not subtrees of the original graph. Only their leaves are the original vertices of G. To solve the covering Steiner tree problem on a general graph G, first find a set of trees and the distribution on them that $O(\log n \log\log n)$-approximates G. Then pick a tree from the distribution and solve the covering Steiner tree problem approximately on it. Now this solution subtree must be transformed into a subgraph of G, and this can be done by simply taking the tour that visits all the leaves of the solution tree, as in the classical 2-approximation for the metric TSP. The distances in the tree are greater than those in the original graph, so this tour will at most double the cost of the solution tree. The expected cost of this tour is $O(\rho \log n \log\log n)$ times the optimum, where ρ is the approximation ratio of the covering Steiner approximation algorithm on trees. By using Markov's inequality, we finally get the following theorem.

Theorem 5.3. *The algorithm described above with high probability finds a covering Steiner tree of cost $O(\rho \log n \log \log n)$ times the cost of the optimal tree, where ρ is the approximation ratio of the algorithm for trees.*

A slight improvement of Bartal's result for graphs that exclude small minors is presented by Konjevod et al. [17], and can be applied to improve the performance ratio on instances of the covering Steiner problem for these classes of graphs.

Acknowledgements. Thanks to Naveen Garg, Madhav Marathe and Neal Young for helpful conversations. We also thank the referees for their many helpful comments.

References

[1] S. Arora and G. Karakostas. A $2 + \epsilon$ approximation algorithm for the k-MST problem. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 754–759, 2000.

[2] S. Arya and H. Ramesh. A 2.5-factor approximation algorithm for the k-MST problem. *Information Processing Letters*, 65:117–118, 1998.

[3] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science*, pages 184–193, October 1996.

[4] Y. Bartal. On approximating arbitrary metrics by tree metrics. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 161–168, 1998.

[5] A. Blum, R. Ravi, and S. Vempala. A constant-factor approximation for the k-MST problem. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pages 442–448, 1996.

[6] M. Charikar, C. Chekuri, A. Goel, and S. Guha. Rounding via trees: deterministic approximation algorithms for group Steiner trees and k-median. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 114–123, 1998.

[7] M. Charikar, C. Chekuri, A. Goel, S. Guha, and S. Plotkin. Approximating a finite metric by a small number of tree metrics. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*, pages 379–388, 1998.

[8] G. Even, G. Kortsarz, and W. Slany. On network design problems: fixed cost flow and the covering Steiner problem. manuscript, May 2001.

[9] M. Fischetti, H. W. Hamacher, K. Jørnsten, and F. Maffioli. Weighted k-cardinality trees: complexity and polyhedral structure. *Networks*, 24:11–21, 1994.

[10] N. Garg. A 3-approximation for the minimum tree spanning k vertices. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science*, pages 302–309, Oct. 1996.

[11] N. Garg. Personal communication, September 1999.

[12] N. Garg, G. Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group Steiner tree problem. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 253–259, 1998.

[13] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, 1988.

[14] S. Janson. Poisson approximations for large deviations. *Random Structures & Algorithms*, 1:221–230, 1990.

[15] P. N. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted Steiner tree. *J. Algorithms*, 19:104–115, 1995.

[16] G. Konjevod and R. Ravi. An approximation algorithm for the covering Steiner problem. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 338–344, 2000.

[17] G. Konjevod, R. Ravi, and F. S. Salman. On approximating planar metrics by tree metrics. *Information Processing Letters*, 80:213–219, 2001.

[18] R. Ravi, R. Sundaram, M. V. Marathe, D. Rosenkrantz, and S. S. Ravi. Spanning trees—short or small. *SIAM J. Discrete Math.*, 9:178–200, 1996.

[19] G. Reich and P. Widmayer. Beyond Steiner's problem: A VLSI oriented generalization. In *Graph-Theoretic Concepts in Computer Science WG89*, volume 411 of *Lecture Notes in Computer Science*, pages 196–210. Springer, 1990.

[20] A. Srinivasan. New approaches to covering and packing problems. In *Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 567–576, 2001.