An Improved Approximation Scheme for the Group Steiner Problem[*]

C. S. Helvig Gabriel Robins Alexander Zelikovsky[†]

Department of Computer Science, University of Virginia, Charlottesville, VA 22903-2442

[†]Department of Computer Science, Georgia State University, Atlanta, GA 30303

Abstract

We address a practical problem which arises in several areas, including network design and VLSI circuit layout. Given an undirected weighted graph $G = (V, E)$ and a family $N = \{N_1, \ldots, N_k\}$ of k disjoint groups of nodes $N_i \subseteq V$, the Group Steiner Problem asks for a minimum-cost tree which contains at least one node from each group N_i. In this paper, we give polynomial-time $O(k^\epsilon)$-approximation algorithms for any fixed $\epsilon > 0$. This result improves the previously known $O(k)$-approximation. We also apply our approximation algorithms to the Steiner problem in directed graphs, while guaranteeing the same performance ratio.

Keywords: Combinatorial optimization, approximation algorithms, Steiner trees, Steiner problem, Group Steiner Problem, Graph algorithms, arborescences.

1 Overview

The classical Steiner problem can be formulated as follows: given an undirected weighted graph $G = (V, E)$ and $M \subseteq V$, find a minimum-cost tree which spans all of M. Nodes in $V - M$ (referred to as *Steiner* nodes) may be optionally included in order to reduce the total tree cost [14]. In this paper we address a generalization of this problem, namely the Group Steiner Problem, which is motivated by practical applications such as computer-aided design (CAD) of VLSI circuits, and network design. This problem, which was first formulated in [19] and reviewed in [10], is formalized as follows:

The Group Steiner Problem [10, 19]: given an undirected weighted graph $G = (V, E)$ and a family $N = \{N_1, ..., N_k\}$ of k disjoint groups of nodes $N_i \subseteq V$, find a minimum-cost tree which contains at least one node from each group N_i.

[*]This work was supported by a Packard Foundation Fellowship, by NSF Young Investigator Award MIP-9457412, and by NSF grant CCR-9988331. The corresponding author is Professor Gabriel Robins, Department of Computer Science, University of Virginia, Charlottesville, VA 22903-2442, robins@cs.virginia.edu, (804) 982-2207, http://www.cs.virginia.edu/robins

Optional *Steiner nodes* may be included in order to reduce the cost of the spanning tree interconnecting the groups of N (see Figure 1). The Group Steiner Problem captures practical scenarios in VLSI circuit design [4, 9, 19], where circuit modules may be rotated and flipped when positioned on a VLSI chip. This induces multiple potential connection points for a given circuit module, namely, one for each of the eight possible orientations [4, 19] (see Figure 1(b)). These locations correspond to a group of up to eight nodes in the Group Steiner Problem formulation when there is only one connection point for the given module (see Figure 1(c)). This formulation also captures the *pin assignment* step in VLSI physical design [18] where the locations of interconnection points (i.e., "pins") on module boundaries must be determined. Finally, the Group Steiner Problem also arises in various network design domains (e.g., when a set of buildings or towns need to be interconnected with a communication network).

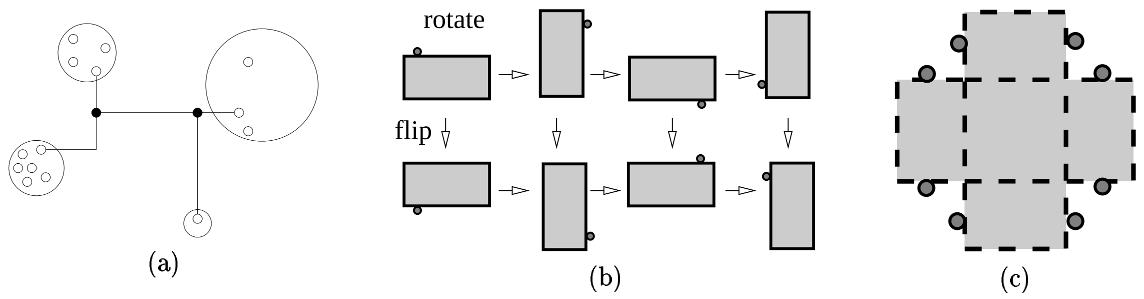

Figure 1: (a) An example of an optimal group Steiner tree (solid dots represent Steiner nodes). (b) A module is rotated and flipped to induce a group of up to eight virtual positions (c).

Existing approximation algorithms for the Group Steiner Problem produce solutions which are $O(k)$ times worse than optimal [10, 11]. The main result of this paper is a new series of heuristics with an improved performance ratio[1] of $O(k^{\epsilon})$ for any fixed $\epsilon > 0$, where k is the number of groups. On the negative side, it is known that this problem cannot be efficiently approximated with a performance ratio of less than $\ln k$ times the optimal [7, 12]. Our work has appeared in preliminary form in [4, 9].

The organization of the remainder of this paper is as follows. Section 2 introduces *depth-d -bounded* Steiner trees and proves that they approximate the optimal group Steiner tree to within a factor of $2d\cdot\sqrt[d]{k}$. Section 3 presents our main heuristic for approximating optimal depth-d -bounded Steiner trees to within a factor of $(2+\ln(2k))^{d-1}$, where d is the tree depth. In Section 4, we complete the description

[1]The *performance ratio* is an upper bound on the ratio of a heuristic solution cost divided by the optimal solution cost.

(and the performance ratio proof) of our main heuristic by approximating *minimum-norm partial d-stars.* The overall performance ratio of our heuristic is the product of these two bounds, which yields our main result. In Section 5, we generalize our results to *directed Steiner trees.* Section 6 analyzes the time complexity and suggests practical enhancements. Section 7 describes how to further improve the performance ratio when groups of size one are present, and Section 8 extends our construction to minimize the radius as well as the cost of group Steiner trees. We discuss our implementation and compare our heuristic with a previous approach [19] in Section 9.

2 Depth-Bounded Steiner Trees

We now introduce the concept of Steiner depth-bounded[2] trees with the following two-fold motivation: (1) depth-d -bounded trees can be used to approximate optimal group Steiner trees to within a factor of $2d \cdot \sqrt[d]{k}$, and (2) optimal depth-bounded Steiner trees can be approximated efficiently, as discussed in the next two sections. Our overall method composes these two approximations, and its performance bound is therefore the product of the two corresponding bounds.

A given graph G may in general violate the triangle inequality, i.e., there may be edges (u, v) in G whose cost is greater than the cost of the minimum u-to-v path in G. Clearly, an optimal group Steiner tree will contain no such edges, since replacing such edges with the corresponding shortest paths will decrease the total tree cost. Therefore, without loss of generality, we replace G with its *metric closure*[3]. In order to further simplify our analysis, we also modify G as follows. For any group node $v \in N_i$, we create a new node v' and a new zero-cost edge (v, v'); thus, we let v' take on the role of v, as shown in Figure 2. This transformation preserves the cost of Steiner trees, while allowing us to consider only Steiner trees in which every group node is a leaf.

We define *d-stars* to be rooted trees of depth at most d. The goal of the rest of this section is to show that for any arbitrary (but henceforth fixed) tree T with *root* r, there exists a low-cost d-star spanning the leaves of T. This will imply that an optimal group Steiner tree can be approximated by a low-cost *group Steiner d-star.* Formally, a Steiner d-star with a root r is a tree rooted at r spanning at least one node from each group such that any root-to-leaf path has at most d edges. It was shown in [11] that even a 1-star (see Figure 3(b)) provides a k -approximation to optimal group Steiner trees.

[2]We define the *depth* of a rooted tree T as the maximum number of edges in any root-to-leaf path.

[3]The *metric closure* is defined as the complete graph where the cost of each edge (u, v) is equal to the cost of the minimum u-to-v path in G.

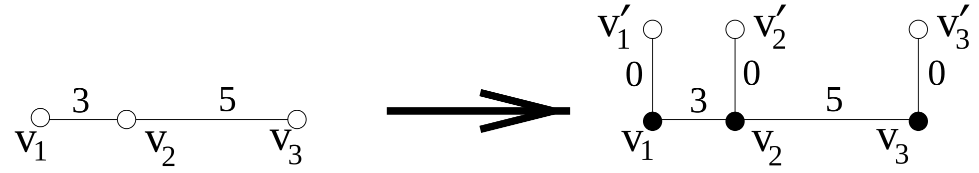

Figure 2: Transformation of G: each group node v in the graph G connects to a newly-created node v' with a zero-cost edge (i.e., newly-created nodes replace the corresponding original group nodes). This transformation alters G in a way that enables us to transform a solution in the modified graph back into a solution of the same cost in the unmodified graph.

Our overall strategy is to specify a low-cost d-star and derive upper bounds on its total cost.

We now construct a low-cost d-star S_d from the tree T with the same root and same set of leaves L. Thus, to completely specify the d-star S_d, we need to select an appropriate set of intermediate nodes that lie on paths between the root r and the leaves. Note that because our approximate tree is a d-star, it must have at most $d-1$ levels of intermediate nodes. The leaves form the set of level-d nodes in S_d. Similarly, we refer to the lowest level of intermediate nodes as level-$(d-1)$ nodes and so on. Finally, the root is connected to the level-1 intermediate nodes.

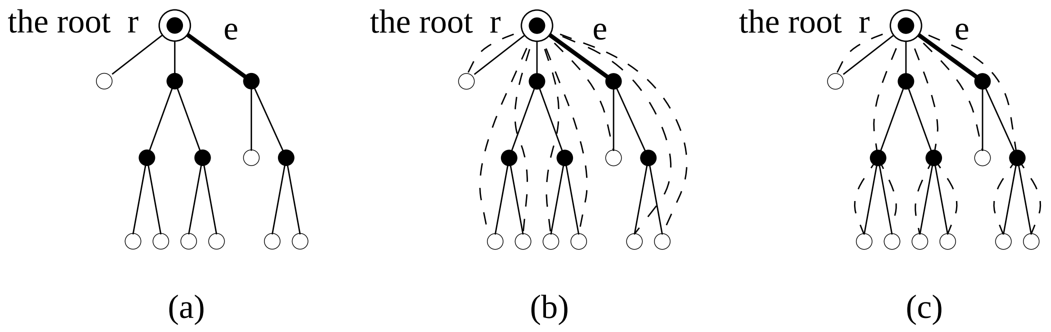

Figure 3: (a) A tree T rooted at r may have an arbitrary depth. (b) A 1-star and (c) a 2-star are represented by dashed lines which connect the root r to all leaves. The edge e is (re)used three times by edges of the 1-star in (b) and twice by edges of the 2-star in (c).

We determine an appropriate set of intermediate nodes for a d-star S_d in T, as follows. First, we sort the nodes of T in a depth-first manner (so that we can extract ordered subsequences of nodes from this sorted list). Then, we partition the sorted subsequence of leaves into contiguous blocks of fixed size b (later we will determine an appropriate value for b). We select the level-$(d-1)$ nodes of

T to be the set of least common ancestors[4] of the leaves in each of the blocks. Next, we partition these level-$(d-1)$ intermediate nodes into contiguous blocks of fixed size b, and we thus obtain the level-$(d-2)$ intermediate nodes by selecting the least common ancestors of each of these blocks (see the bottom of Figure 4(a)). We repeatedly use this procedure to define each level of intermediate nodes until we reach the root. Thus, in our final tree S_d, each level-i node is connected to b level-$(i+1)$ nodes below and to one level-$(i-1)$ node above. In particular, all level-1 nodes are connected to the root.

To prove upper bounds on the cost of d-stars, we will sum the costs of tree paths between nodes which are adjacent in d-stars. However, since such paths are not necessarily disjoint, the same tree edge may be counted multiple times in this sum, a situation we refer to as *edge reuse* (see Figure 3). For a tree T, edge reuse provides a loose upper bound on the ratio $cost(\text{d-star})/cost(T)$, since if no edge is used more than j times when replacing edges of a d-star by the corresponding paths in T, then the d-star has cost no more than j times the cost of the tree T. Our strategy for deriving upper bounds on the cost of S_d is to bound its edge reuse.

Let $reuse_T(S_d)$ denote the maximum number of times that any tree edge is used in tree paths connecting nodes adjacent in S_d. We distinguish two types of paths that contribute to the edge reuse of the resulting d-star: (a) paths from the root to level-1 intermediate nodes (top part of Figure 4(a)), and (b) paths from level-i intermediate nodes to level-$(i+1)$ nodes, where $1 \leq i \leq d-1$ (Figure 4(b)). The number of paths of type (a) is bounded by the number of level-1 intermediate nodes. Note that each level contains a factor of b fewer intermediate nodes than the level below it. Thus, there are no more than $|L|/b^{d-1}$ level-1 nodes, where b is the block size (see Figure 4(a)).

We now estimate the contribution of type-(b) paths to the reuse of an arbitrary (but henceforth fixed) edge (u,v) of the tree T with u being the parent of v. Let S be the depth-first -ordered sequence of level-$(i+1)$ intermediate nodes which descend from v. Note that S forms a contiguous subsequence in the sequence of all level-$(i+1)$ intermediate nodes. If a size-b block is completely contained in the sequence S, then its least common ancestor (which is a level-i intermediate node) necessarily descends from v (Figure 4). Therefore, the edge (u,v) does not lie on paths to level-$(i+1)$ nodes that belong to such completely-contained blocks, and no contribution to the reuse of the edge (u,v) occurs here.

We are thus only concerned with blocks which may be not completely contained inside S, because type-(b) paths ending only in such blocks can contribute to the reuse of the edge (u,v). Since the

[4] We define a node u as an *ancestor* of v (in other words, v *descends* from u) if the path from the root to v passes through u.

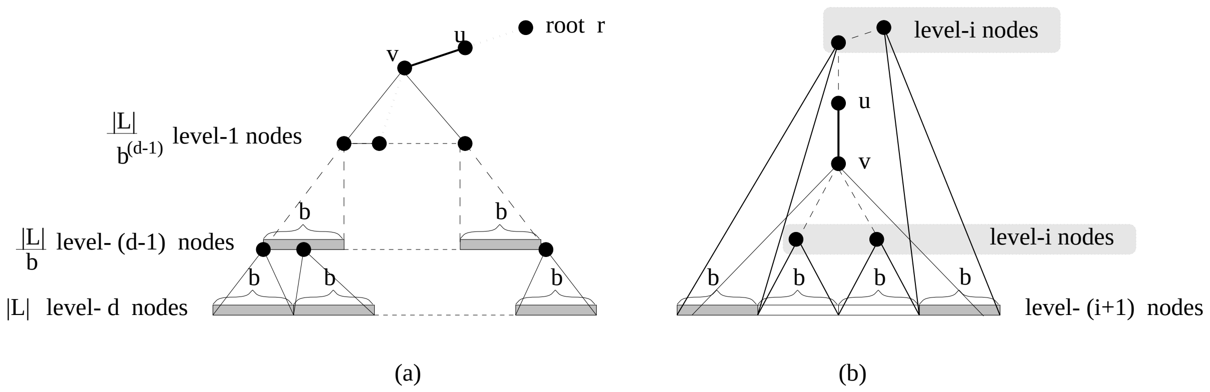

Figure 4: Triangles represent depth-first -ordered subtrees. (a) Type-(a) paths which reuse the edge (u, v) terminate at level-1 intermediate nodes below v (there are no more than $|L|/b^{d-1}$ of these). (b) Type-(b) paths which reuse the edge (u, v) terminate at intermediate nodes (or leaves) contained in only the leftmost and the rightmost blocks. These two blocks together contain at most $2b$ level-$(i + 1)$ nodes.

nodes of S are sorted in depth-first order, the only blocks relevant to this analysis are the leftmost and rightmost blocks. The total contribution to the reuse of edge (u, v) due to the paths ending in the leftmost and rightmost blocks cannot exceed the total size of these two blocks, namely $2b$ (Figure 4).

The edge (u, v) may be used simultaneously in paths starting from different levels. Thus, we bound the total contribution of type-(b) paths for all levels by $2b \cdot (d - 1)$, and the total edge reuse is at most $|L|/b^{d-1} + 2b \cdot (d - 1)$. Choosing $b = \sqrt[d]{|L|/2}$ yields an upper bound on edge reuse of $2d \cdot \sqrt[d]{|L|/2}$. This proves that for any tree T, with the set of leaves L and root r, there is a d-star rooted at r with the same set of leaves L, having cost at most $2d \cdot \sqrt[d]{|L|/2} \cdot cost(T)$. Therefore, by approximating the optimal group Steiner tree (spanning k groups and thus having k leaves) with an *optimal* Steiner d-star (i.e. the minimum-cost Steiner d-star), we obtain the following result.

Theorem 1 *Let Opt be an optimal group Steiner tree over k groups, and let r be an arbitrary node of Opt. Then the cost of an optimal Steiner d-star rooted at r is at most $2d \cdot \sqrt[d]{k/2} \cdot cost(Opt)$.*

3 The Main Heuristic

As shown above, an optimal Steiner d-star is a reasonable approximation of an optimal group Steiner tree (denoted *Opt*). On the other hand, it can be proven that even the problem of approximating an optimal Steiner d-star is as difficult as approximating a minimum set cover. Therefore, for any $\epsilon > 0$, it is unlikely that there exists a polynomial-time approximation algorithm with performance ratio $(1 - \epsilon) \cdot \ln k$, where k is the number of groups [7]. In this section, we will indirectly approximate *Opt* by approximating an optimal Steiner d-star.

In order to apply Theorem 1, we must ensure that the root of the Steiner d-star produced by our algorithm belongs to the optimal group Steiner tree *Opt*. Although we do not know which nodes are in *Opt*, this is not an obstacle: For each node r of an arbitrary fixed group (say N_1), we construct a low-cost Steiner d-star $Approx_d$ with root r. We then select the least-cost $Approx_d$ over all possible choices of r. Because *Opt* contains at least one node from each group, this guarantees (within polynomial time) that at least one of the $|N_1|$ trees thus constructed had the proper choice for the root. Therefore, without loss of generality we may fix the root r, i.e., we consider the rooted version of the Group Steiner Problem.

Define $Opt_d(r)$ as the optimal Steiner d-star rooted at r, for any positive integer d. The main idea in constructing the approximate Steiner d-star $Approx_d$ is to successively refine an initial approximation coinciding with $Opt_1(r)$. There are two advantages to using $Opt_1(r)$ as an initial approximation for $Opt_d(r)$: first, unlike $Opt_d(r)$ for $d \geq 2$, the 1-star $Opt_1(r)$ can be computed efficiently; secondly, the cost of $Opt_1(r)$ is bounded by $k \cdot cost(Opt)$ (from Theorem 1, for $d = 1$). To measure the approximation quality of a d-star, we will therefore compare its cost to the cost of an optimal 1-star with the root r and with leaves taken from the same groups spanned by the d-star.

Let S be a d-star with a root $v \in V$ and let $groups(S)$ be the set of groups spanned by S. We denote by S' an optimal 1-star with the root r connected to $groups(S)$. We define the *norm* of S as $norm(S) = cost(S)/cost(S')$. Note that S and S' may have different roots (see Figure 5(a)).

We represent our low-cost d-star $Approx_d$ as a union of subtrees, each consisting of the root, exactly one level-1 intermediate node, and all of the descendants of this intermediate node. Such rooted subtrees of depth d will be called *partial d-stars* (see Figure 5(b)). We select partial d-stars for $Approx_d$ in the following greedy manner. First, we find a partial d-star P with an approximately minimum norm. Next, we remove the groups that are spanned by P (i.e., $groups(P)$) from further consideration. Finally, we determine the next partial d-star with an approximately minimum norm and iterate this process until all groups are spanned. Figure 6 formally describes the algorithm, and an execution example (for $d = 2$) is given in Figure 7.

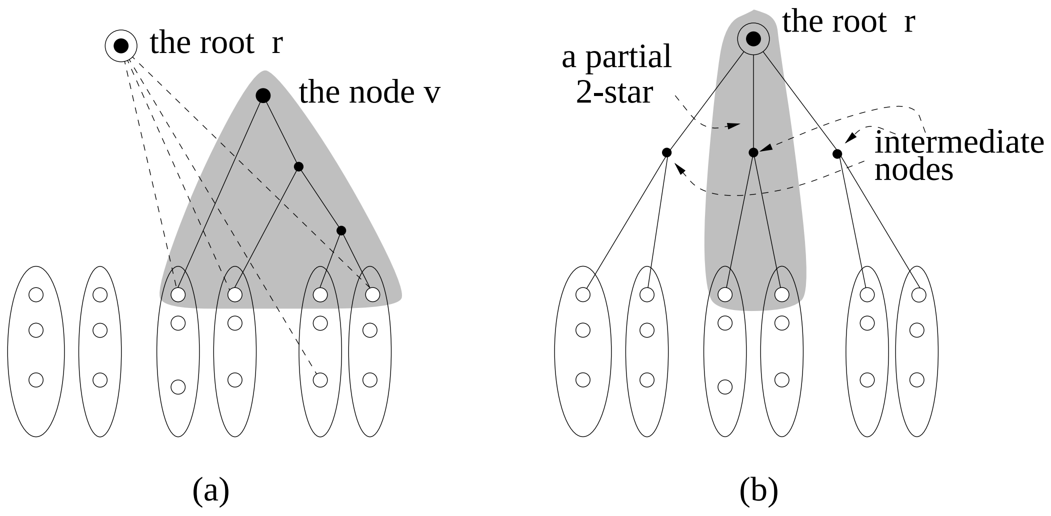

Figure 5: (a) A d-star S rooted at v (shaded area) and the corresponding optimal 1-star S' rooted at r (dashed lines). (b) A Steiner 2-star, with a (shaded) partial 2-star.

Rooted Steiner d-Star Heuristic

Input: A graph $G = (V, E)$, a family N of k disjoint groups $N_1, \ldots, N_k \subseteq V$, a root $r \in V$, and a node $v \in V$

Output: A low-cost d-star $Approx_d$ rooted at node v and intersecting each group N_i

```
$Approx_d \leftarrow \{v\}$
$N' \leftarrow N$
While $N' \neq \emptyset$ do
    Find a low-norm (with respect to $r$) partial $d$-star $P = Partial_d(v, N')$
        (see Section 4)
    $N' \leftarrow N' - groups(P)$
    $Approx_d \leftarrow Approx_d \cup P$
Output $Approx_d$
```

Figure 6: Our greedy d-star heuristic for a given (fixed) root. At each loop iteration, the heuristic approximates the lowest-norm partial d-star, adds it to the solution, and removes its groups from future consideration. Termination occurs when no groups remain to be spanned.

To complete the description of our heuristic, we need to describe an efficient procedure which, given a root r and set of groups M, finds a low-norm partial d-star $Partial_d(r, M)$ rooted at r spanning some of the groups of M. We call this procedure the Partial d-Star Heuristic and describe it formally in the next section. Before stating the performance ratio of the Partial d-Star Heuristic, we first define ψ_d as follows:

$$\psi_d \equiv (2 + \ln(2k))^{d-2} \tag{1}$$

Lemma 2 *Let $Partial_d$ be the partial d-star produced by the Partial d-Star Heuristic, and let $Partial\_Opt_d$ be the corresponding partial d-star with minimum norm. The performance ratio of the Partial Star Heuristic for $d \geq 2$ is at most:*

$$\frac{norm(Partial_d)}{norm(Partial\_Opt_d)} \leq \psi_d$$

Proof: (See Section 4.)

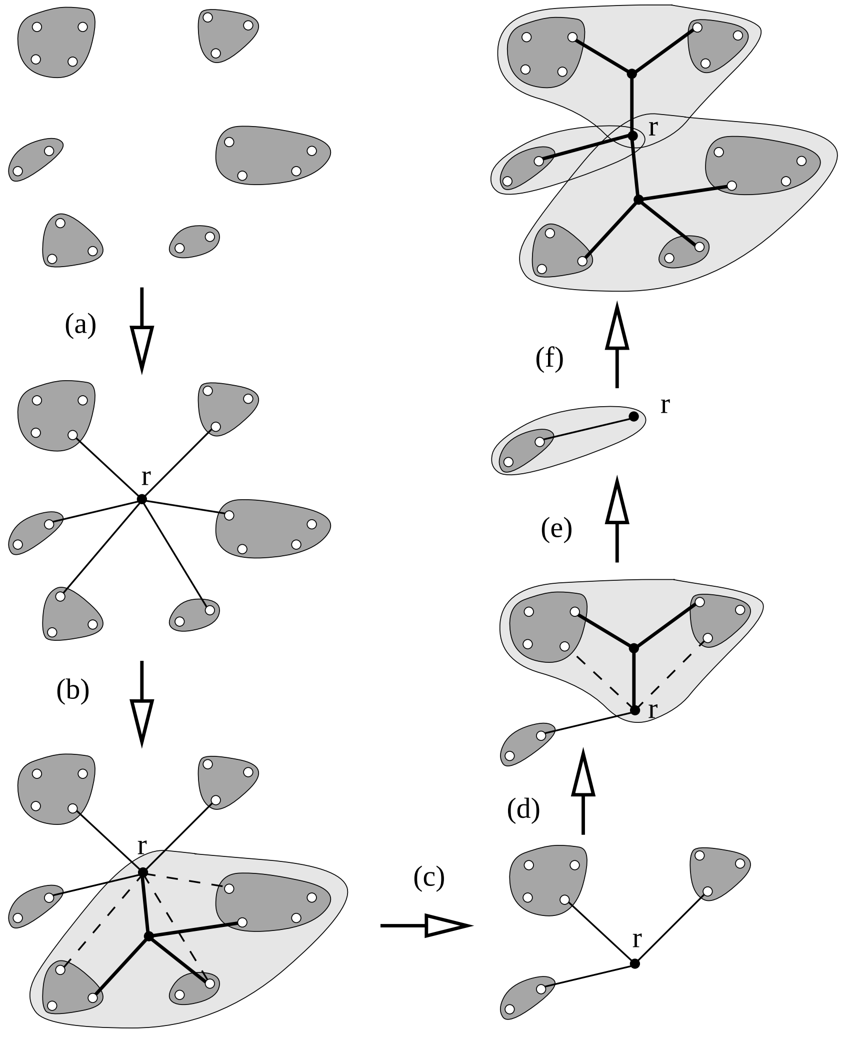

Figure 7: For an input instance of the Group Steiner Problem, our heuristic does the following for each possible root r: (a) find the optimal 1-star, (b) find a low-norm partial d-star (shaded region), (c) store this star in the solution and remove its groups from future consideration, (d) find the next low-norm partial d-star (shaded region), (e) repeat step (c) for the next partial d-star, and finally (f) find the last low-norm partial d-star and output the union of all stored partial d-stars.

In the remainder of this section, we analyze the Rooted Steiner d-Star Heuristic described in Figure 6. In particular, Lemma 3 below gives the performance ratio for this heuristic. Together with Theorem 1, this will imply our main result, namely Theorem 4.

Lemma 3 *Let $Opt_d(v)$ be an optimal Steiner d-star rooted at v, and let $Opt_1(r)$ be the optimal Steiner 1-star rooted at r. The cost of the tree produced by the Rooted Steiner d-Star Heuristic is at most:*

$$cost(Approx_d) \leq \left(2 + \ln \frac{cost(Opt_1(r))}{cost(Opt_d(v))}\right) \cdot cost(Opt_d(v)) \cdot \psi_d$$

Proof: We compare the optimal d-star rooted at v to the approximate solution produced by our heuristic. The optimal d-star $Opt_d(v)$ can be partitioned into partial d-stars denoted $R_1, \ldots, R_s$. Let $P_1, P_2, \ldots, P_t$ be the t partial d-stars selected by the Rooted Steiner d-Star Heuristic, i.e., the approximate Steiner d-star $Approx_d$ is a union of these t partial d-stars.

The algorithm greedily chooses a partial d-star, removes the groups spanned by this partial d-star, and then repeats this process until no more groups remain. At each iteration, the algorithm chooses a low-norm partial d-star P_i which has lower cost than the corresponding optimal 1-star (hereafter denoted P_i'). Let C_i denote the cost of the optimal Steiner 1-star at the i^{th} iteration of the loop, when the groups spanned by $P_1, \ldots, P_i$ have already been removed (see Figure 8). In the first iteration, we have $C_0 = cost(Opt_1(r))$. Inductively, we obtain:

$$C_i = C_{i-1} - cost(P_i'), \quad i = 1, \ldots, t \tag{2}$$

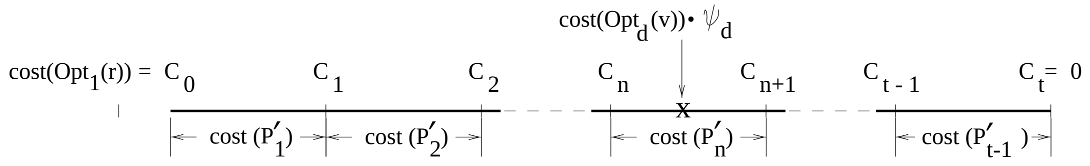

Figure 8: The Rooted Steiner d-Star Heuristic (Figure 6) iteratively modifies the problem instance by repeatedly removing groups associated with low-cost partial d-stars P_i. The cost of the optimal 1-star C_0 originally exceeds $cost(Opt_d(v)) \cdot \psi_d$, but finally the cost of the optimal 1-star for the remaining groups becomes equal to zero at step t. Therefore, there exists an n such that $1 \leq n \leq t$ and $cost(Approx_d)$ drops to strictly less than $cost(Opt_d(v)) \cdot \psi_d$ between steps n and $n+1$.

Assume that the partial d-stars $R_1, \ldots, R_s$ of the optimal Steiner d-star are sorted by non-decreasing order of their norms. Applying the ratio property[5], we obtain:

$$norm(R_1) \leq \frac{\sum_{j=1}^{s} cost(R_j)}{\sum_{j=1}^{s} cost(R'_j)} = \frac{cost(Opt_d(v))}{C_0}$$

Therefore, by Lemma 2, we have:

$$\frac{cost(P_1)}{cost(P'_1)} = norm(P_1) \leq norm(R_1) \cdot \psi_d \leq \frac{cost(Opt_d(v))}{C_0} \cdot \psi_d$$

After the removal of $groups(P_1)$, the cost of the Steiner 1-star rooted at r reduces to $C_1 = C_0 - cost(P'_1)$, and the cost of the optimal Steiner d-star cannot increase, that is:

$$\frac{cost(P_2)}{cost(P'_2)} \leq (\frac{cost(Opt_d(v))}{C_1}) \cdot \psi_d$$

Applying this observation inductively, we get:

$$\frac{cost(P_i)}{cost(P'_i)} \leq \frac{cost(Opt_d(v))}{C_{i-1}} \cdot \psi_d \quad \text{for} \quad i = 1, ..., t \tag{3}$$

The relationships (2) and (3) together imply:

$$C_i \leq C_{i-1} \cdot \left(1 - \frac{cost(P_i)}{cost(Opt_d(v)) \cdot \psi_d}\right)$$

Unraveling the inequality above, we obtain:

$$C_n \leq C_0 \cdot \prod_{i=1}^{n} \left(1 - \frac{cost(P_i)}{cost(Opt_d(v)) \cdot \psi_d}\right)$$

Taking the natural logarithm of both sides and using the fact that $\ln(1+x) \leq x$, we conclude:

$$\ln \frac{C_0}{C_n} \geq \frac{\sum_{i=1}^{n} cost(P_i)}{cost(Opt_d(v)) \cdot \psi_d} \tag{4}$$

[5] The ratio property states that $a/b \leq c/d$ if and only if $(a+c)/(b+d) \leq c/d$, for any positive a, b, c, d. Its proof is as follows: $\frac{a}{b} \leq \frac{c}{d}$ if and only if $ad \leq bc$ which is true if and only if $ad + cd \leq bc + cd$ which holds if and only if $\frac{a+c}{b+d} \leq \frac{c}{d}$.

Since $C_t = 0$, there exists an n such that $C_n > cost(Opt_d(v)) \cdot \psi_d \geq C_{n+1}$ (see Figure 8). Otherwise, even the cost of the simple 1-star is less than $cost(Opt_d(v)) \cdot \psi_d$, and Lemma 3 is clearly true. Note that $cost(P'_{n+1}) \leq C_n$. Therefore, inequality (3) yields:

$$\frac{cost(P_{n+1})}{cost(Opt_d(v)) \cdot \psi_d} \leq \frac{cost(P'_{n+1})}{C_n} \leq 1 \tag{5}$$

Finally, using (4) and (5), we can bound the performance ratio of the Rooted Steiner d-Star Heuristic as follows (see Figure 8):

$$\begin{aligned} \frac{cost(Approx_d)}{cost(Opt_d(v))} &= \frac{\sum_{i=1}^{t} cost(P_i)}{cost(Opt_d(v))} \\ &\leq \frac{\sum_{i=1}^{n} cost(P_i) + cost(P_{n+1}) + C_{n+1}}{cost(Opt_d(v))} \\ &\leq \left(\ln \frac{C_0}{C_n} + 2\right) \cdot \psi_d \\ &\leq \left(\ln \frac{cost(Opt_1(r))}{cost(Opt_d(v))} + 2\right) \cdot \psi_d \end{aligned}$$

□

Together with Theorem 1, Lemmas 2 and 3 imply our main result:

Theorem 4 *The Rooted Steiner d-star Heuristic with v coinciding with r (see Figure 6) solves the Group Steiner Problem with performance ratio* $2d \cdot (2 + \ln(2k))^{d-1} \cdot \sqrt[d]{k}$*, where k is the number of groups.*

Proof: By Theorem 1, we know that $cost(Opt_1(r)) \leq k \cdot cost(Opt) \leq k \cdot cost(Opt_d(r))$, where k is the number of groups. Therefore, Lemma 3 implies that the cost of the tree produced by our main algorithm (when v coincides with r) is at most ψ_{d+1} times the cost of the optimal Steiner d-star. Using Theorem 1 and identity (1), we obtain a group Steiner tree which costs no more than $2d \cdot (2 + \ln(2k))^{d-1} \cdot \sqrt[d]{k}$ times the optimal. □

Corollary 5 *The optimal group Steiner tree can be approximated in polynomial time with a performance ratio of* $O(k^{\epsilon})$ *for any fixed* $\epsilon > 0$.

4 Approximating Minimum-Norm Partial d-Stars

Although minimum-norm partial 2-stars can be found efficiently, finding minimum-norm partial d-stars becomes more difficult for $d \geq 3$. Indeed, the minimum-norm partial d-star S is an optimal Steiner $(d-1)$-star for the groups which are spanned by S, and the set-cover problem (which is NP-hard) can be reduced to finding an optimal Steiner 2-star [12]. Thus, in this section, we describe our heuristic for finding low-norm partial d-stars. First, we will describe how to find minimum-norm partial 2-stars, and then we will show how to approximate minimum-norm partial d-stars for $d \geq 3$. Finally, we will bound the error of our approximation (i.e., we will prove Lemma 2).

Figure 9 formally describes a procedure for finding minimum-norm partial 2-stars. Note that in this procedure we use $cost(u, N_i)$ to denote the cost of the shortest edge between u and any node in group N_i.

```
Partial d-Star Heuristic (for $d = 2$)
Input: A graph $G = (V, E)$, a family $M \subseteq N$ of disjoint groups and root $r \in V$
Output: The minimum-norm partial 2-star $P(r, M)$ rooted at $r$ with leaves from some groups of $M$
For each $v \in V$ do
    Sort $M = \{N_1, ..., N_{|M|}\}$ such that $\frac{cost(v,N_i)}{cost(r,N_i)} \leq \frac{cost(v,N_{i+1})}{cost(r,N_{i+1})}$
    Find $j \in \{1, ..., |M|\}$ that minimizes $norm(v) \leftarrow \frac{cost(r,v)+\sum_{i=1}^{j} cost(v,N_i)}{\sum_{i=1}^{j} cost(r,N_i)}$
    $M(v) \leftarrow \{N_1, ..., N_j\}$
Find $v$ having minimum $norm(v)$
Output the partial 2-star $P(r, M)$ with the intermediate node $v$ with root $r$ and groups $M(v)$
```

Figure 9: Our algorithm for finding a minimum-norm partial 2-star (i.e., d-star for $d = 2$). For each candidate v for the intermediate node, we sort groups N_i according to the potential improvement of inserting node v between the root r and each group. Then, we include consecutive groups from the list while their inclusion decreases the norm of the partial 2-star.

The following lemma establishes the correctness of the Partial 2-Star Heuristic.

Lemma 6 *Given a family $M \subseteq N$ of groups, the Partial 2-Star Heuristic (Figure 9) outputs a minimum-norm partial 2-star.*

Proof: Note that any partial 2-star (see Figure 5(b)) contains exactly one intermediate node (possibly coinciding with the root). Let v be the intermediate node of a minimum-norm partial 2-star P with root r. The Partial 2-Star Heuristic sorts the family of groups $M = \{N_1, ..., N_{|M|}\}$ such that for any

$i = 1, \ldots, |M|$:

$$\frac{cost(v, N_i)}{cost(r, N_i)} \leq \frac{cost(v, N_{i+1})}{cost(r, N_{i+1})}$$

Let N_j be the last group (in the sorted family of groups) which is connected to v in P. Since P is optimal, disconnecting v from N_j can only increase the norm of P. Let P' denote the optimal 1-star corresponding to P (i.e., the 1-star rooted at r spanning $groups(P)$). Then:

$$norm(P) = \frac{cost(P)}{cost(P')} \leq \frac{cost(P) - cost(v, N_j)}{cost(P') - cost(r, N_j)}$$

By the ratio property, this implies that $\frac{cost(v,N_j)}{cost(r,N_j)} \leq \frac{cost(P)-cost(v,N_j)}{cost(P')-cost(r,N_j)}$ and therefore:

$$\frac{cost(v, N_j)}{cost(r, N_j)} \leq \frac{cost(P)}{cost(P')} \tag{6}$$

It is sufficient to show that if we connect v to the contiguous sequence of groups $N_1, N_2, \ldots, N_j$, then the norm of P may only decrease. Indeed, let N_i (where $i < j$) be a group that is not adjacent to v in P. Then we have:

$$\frac{cost(v, N_i)}{cost(r, N_i)} \leq \frac{cost(v, N_j)}{cost(r, N_j)} \leq \frac{cost(P)}{cost(P')}$$

and the ratio property together with inequality (6) yields:

$$\frac{cost(P) + cost(v, N_i)}{cost(P') + cost(r, N_i)} \leq \frac{cost(P)}{cost(P')}$$

Thus, connecting v to N_i cannot increase the norm of P. $\square$

Our Partial d-Star Heuristic finds a low-norm partial d-star $Partial_d = Partial_d(r, M)$ with a given root r spanning elements from some groups of a given subfamily $M \subseteq N$, where N is the family of all groups (Figure 10 gives a formal description). The Partial d-Star Heuristic is recursive: in order to find a partial d-star, we need to first find certain partial $(d-1)$-stars.

Now we have completed the description of our main heuristic, and we are ready to prove Lemma 2, which states that the performance ratio of the Partial d-Star Heuristic is at most:

```
Partial d-Star Heuristic (for $d \geq 3$)
Input: A graph $G = (V, E)$, a family $M \subseteq N$ of disjoint groups and root $r \in V$
Output: A low-norm partial $d$-star $Partial_d(r, M)$ with root $r$ and leaves from some groups of $M$
For each $v \in V$ do
    $Partial_d \leftarrow (r, v)$
    $norm(Partial_d) \leftarrow \infty$
    $M' \leftarrow M$
    While $M' \neq \emptyset$ do
        Use the Partial $d$-Star Heuristic for depth $d-1$: $Partial_{d-1} \leftarrow Partial_{d-1}(v, M')$
        If $norm(Partial_d) \leq norm(Partial_{d-1})$ then exit while
        $Partial_d \leftarrow Partial_d \cup Partial_{d-1}$
        $M' \leftarrow M' - groups(Partial_{d-1})$
    $norm(v) \leftarrow norm(Partial_d)$
Find $v$ having minimum $norm(v)$
Output the partial $d$-star $Partial_d(r, M)$ with the intermediate node $v$
```

Figure 10: Our heuristic for finding a low-norm partial d-star for $d \geq 3$.

$$\frac{norm(Partial_d)}{norm(Partial\_Opt_d)} \leq \psi_d \tag{7}$$

where $Partial_d = Partial_d(r, M)$ is the output partial d-star of the Partial d-Star Heuristic, $Partial\_Opt_d$ is a partial d-star with minimum norm, and $\psi_d = (2 + \ln(2k))^{d-2}$.

Proof of Lemma 2: We prove inequality (7) by induction on d, assuming that it is true for all depths less than d (the base case $d = 2$ follows from Lemma 6). Because our algorithm examines each node as a possible candidate for being the intermediate node in $Partial_d$, we assume that the intermediate node v in $Partial_d$ is the same as the intermediate node v in $Partial\_Opt_d$. We distinguish three cases (see Figure 11) depending on the relationship between $groups(Partial_d)$ and $groups(Partial\_Opt_d)$.

Case I. $groups(Partial_d) \subseteq groups(Partial\_Opt_d)$.

If $groups(Partial_d) \subset groups(Partial\_Opt_d)$, then for the purposes of analysis, we continue to include partial $(d-1)$-stars covering only groups of $Partial\_Opt_d$ until we run out of such groups. This may only increase the norm of $Partial_d$ and therefore, without loss of generality, we may assume that $groups(Partial_d) = groups(Partial\_Opt_d)$.

The partial d-star $Partial\_Opt_d$ consists of the edge (r, v) and the optimal Steiner $(d-1)$-star (the shaded cone in Figure 11) for the set $groups(Partial\_Opt_d)$ rooted at v. By induction, we assume that inequality (7) is true for all depths $\leq d-1$. Therefore, we may apply Lemma 3 for depth $d-1$ to the

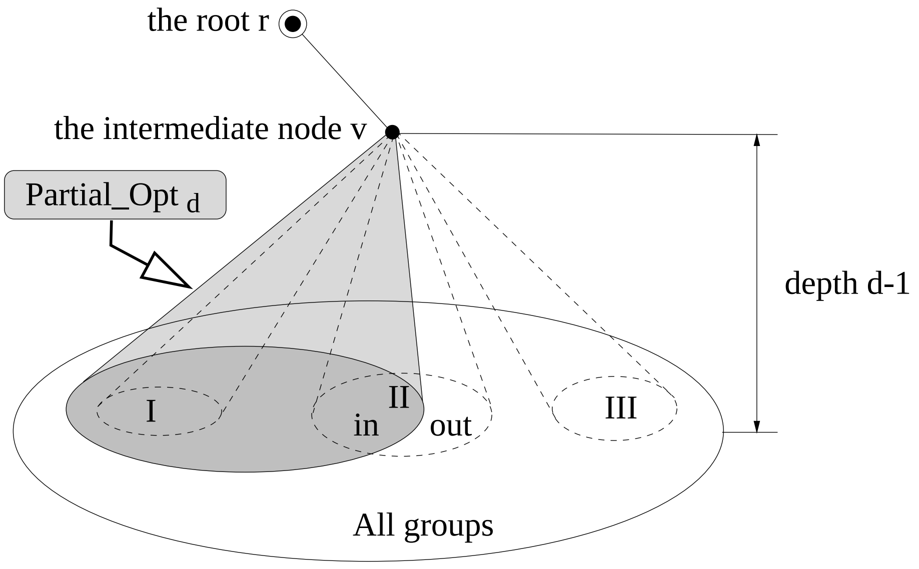

Figure 11: The three possible relationships (based on set inclusion) between $groups(Partial\_Opt_d)$ and $groups(Partial_d)$ correspond to Cases I, II, and III in the proof.

instance of the Group Steiner Problem with the root v and $groups(Partial\_Opt_d)$ as follows:

$$cost(Partial_d) - cost(r,v) \leq$$
$$\left(2 + \ln\left(\frac{cost(Opt_1(r))}{cost(Partial\_Opt_d) - cost(r,v)}\right)\right) \cdot \psi_{d-1} \cdot (cost(Partial\_Opt_d) - cost(r,v))$$

The right-hand-side of the inequality above cannot be more than:

$$\left(2 + \ln\left(\frac{cost(Opt_1(r))}{cost(Partial\_Opt_d) - \alpha \cdot cost(r,v)}\right)\right) \cdot \psi_{d-1} \cdot (cost(Partial\_Opt_d) - \alpha \cdot cost(r,v)) \qquad (8)$$

for any $0 < \alpha \leq 1$. This can be shown by taking the derivative with respect to α. For the purposes of our analysis, we choose $\alpha = 1/2$. The edge (r,v) belongs to $Partial\_Opt_d$, and therefore $cost(r,v) \leq cost(Partial\_Opt_d)$. Thus:

$$\frac{cost(Opt_1(r))}{cost(Partial\_Opt_d) - \frac{1}{2} \cdot cost(r,v)} \leq \frac{cost(Opt_1(r))}{cost(Partial\_Opt_d) - \frac{1}{2} \cdot cost(Partial\_Opt_d)}$$

By Theorem 1 for $d = 1$, we know that $\frac{cost(Opt_1(r))}{cost(Opt_d)} \leq k$. Therefore,

$$\frac{cost(Opt_1(r))}{cost(Partial\_Opt_d)} = \frac{cost(Opt_1(r))}{cost(Opt_d)} \cdot \frac{cost(Opt_d)}{cost(Partial\_Opt_d)} \leq k$$

Thus, we have:

$$\frac{cost(Opt_1(r))}{cost(Partial\_Opt_d) - \frac{1}{2} \cdot cost(Partial\_Opt_d)} = 2 \cdot \frac{cost(Opt_1(r))}{cost(Partial\_Opt_d)} \leq 2 \cdot k$$

Inserting this inequality into expression (8), we get:

$$cost(Partial_d) - cost(r, v) \leq (2 + \ln(2k)) \cdot \psi_{d-1} \cdot (cost(Partial\_Opt_d) - \frac{1}{2} \cdot cost(r, v))$$

By identity (1), we know that $(2 + \ln(2k)) \cdot \psi_{d-1} = \psi_d$. Thus, we have:

$$cost(Partial_d) - cost(r, v) \leq \psi_d \cdot cost(Partial\_Opt_d) - \psi_d \cdot \frac{1}{2} \cdot cost(r, v)$$

Since $\psi_d \geq 2$, we infer:

$$cost(Partial_d) \leq \psi_d \cdot cost(Partial\_Opt_d)$$

Thus, we finally have proven inequality (7) for Case I, since $Partial_d$ and $Partial\_Opt_d$ cover the same groups and therefore have the same corresponding 1-star.

Case II. $groups(Partial_d) \cap groups(Partial\_Opt_d) \neq \emptyset$, but $groups(Partial_d) \not\subseteq groups(Partial\_Opt_d)$.

Let $P_1, \ldots, P_t$ be partial $(d-1)$-stars of $Partial_d$ found consecutively inside the while loop of the Partial d-Star Heuristic (Figure 10). Also, let P'_i denote the optimal 1-star corresponding to P_i (i.e., the 1-star rooted at r spanning $groups(P_i)$).

We split the cost of each 1-star P'_i into two parts, $in\_cost(P'_i)$ and $out\_cost(P'_i)$, where $cost(P'_i) = in\_cost(P'_i) + out\_cost(P'_i)$, as shown in Figure 11. The first part, $in\_cost(P'_i)$, is the cost of edges from r to the groups that are contained in $groups(Partial\_Opt_d)$, while the second part, $out\_cost(P'_i)$, is the cost of edges from r to the groups that are not contained in $Partial\_Opt_d$ (see Figure 11). We also split the cost of P_i in the same proportion as the cost of P'_i, i.e. we pick $in\_cost(P_i)$ and $out\_cost(P_i)$ such that $cost(P_i) = in\_cost(P_i) + out\_cost(P_i)$ and:

$$\frac{in\_cost(P_i)}{in\_cost(P'_i)} = \frac{out\_cost(P_i)}{out\_cost(P'_i)}$$

which is always possible if $in\_cost(P_i') \neq 0$ and $out\_cost(P_i') \neq 0$. Otherwise, if $in\_cost(P_i') = 0$, we set $in\_cost(P_i) = 0$, and if $out\_cost(P_i') = 0$, we set $out\_cost(P_i) = 0$. The ratio property implies:

$$\begin{aligned} norm(Partial_d) &= \frac{cost(r,v) + \sum_{i=1}^{t} in\_cost(P_i) + \sum_{i=1}^{t} out\_cost(P_i)}{\sum_{i=1}^{t} in\_cost(P_i') + \sum_{i=1}^{t} out\_cost(P_i')} \\ &\leq \frac{cost(r,v) + \sum_{i=1}^{t} in\_cost(P_i)}{\sum_{i=1}^{t} in\_cost(P_i')} \end{aligned}$$

Note that the previous argument for Case I is true for the "*in*-costs" (i.e., $in\_cost(P_i)$ and $in\_cost(P_i')$) instead of the costs of P_i and P_i' (if we omit the partial stars P_i for which $in\_cost(P_i') = 0$). Indeed, the minimum norm of partial $(d-1)$-stars over all groups cannot be more than the minimum norm of such stars over a subset of groups (namely, from $groups(Partial\_Opt_d)$). Therefore,

$$\begin{aligned} norm(Partial_d) &\leq \frac{cost(r,v) + \sum_{i=1}^{t} in\_cost(P_i)}{\sum_{i=1}^{t} in\_cost(P_i')} \\ &\leq norm(Partial\_Opt_d) \cdot \psi_d \end{aligned}$$

Case III. $groups(Partial_d) \cap groups(Partial\_Opt_d) = \emptyset$ (see Figure 11).

For the purposes of analysis, we include $Partial_{d-1}(v, groups(Partial\_Opt_d))$ in our d-star $Partial_d$. By the exit condition of the while loop (see Figure 10), this inclusion may only increase the norm of $Partial_d$. Since the intersection of $groups(Partial_d)$ and $groups(Partial\_Opt_d)$ can be forced to be non-empty, any instance of Case III can be transformed to an instance of Case II. $\square$

5 The Steiner Problem in Directed Graphs

The techniques above can be adapted to address a generalization of the Group Steiner Problem, namely the Steiner problem in directed graphs, also known as the *Steiner Arborescence Problem* [10, 1]:

The Directed Steiner Problem [10]: given a directed weighted graph $G = (V, E)$, a set of k *terminals* $N \subseteq V$, and a *root* $r \in N$, find a minimum-cost subgraph T which contains directed paths from r to each terminal.

Without loss of generality, we may assume that the subgraph T is a *directed tree*[6], otherwise T would contain arcs which can be removed without increasing the cost of T. As in the beginning of

[6]Each node of T is reachable from the root via a unique directed path. This means that replacing all arcs (i.e., directed edges) by undirected edges would produce an undirected tree from T.

Section 2, we replace the directed graph G with its *transitive closure*[7] (an analog of metric closure), and we may assume that each terminal is a leaf of T.

We define *directed d-stars* to be directed trees of depth at most d. Again, as in Section 2, it can be shown that the minimum-cost directed d-star costs at most $2d \cdot \sqrt[d]{k/2}$ times the optimal directed Steiner tree. As in Section 3, we represent our low-cost directed d-star $Approx_d$ as a union of *directed partial d-stars.* Each such partial d-star consists of the root r, exactly one arc from the root to a level-1 intermediate node, and the directed subtree rooted at this intermediate node.

The norm is still defined as it was in Section 3. Given a directed d-star S, let S' be the corresponding directed 1-star (with the same terminals and root). The norm of S is defined to be $cost(S)/cost(S')$. Now we can use (an analog of) the Partial d-Star Heuristic for directed graphs in order to find a low-norm directed partial d-star. In this way, the norm of the directed partial d-star obtained is bounded as in Lemma 2.

In order to approximate an optimal directed Steiner d-star, we use (an analog of) the Rooted Steiner d-Star Heuristic for directed graphs (see Section 3). Again, starting from an optimal 1-star, we update the current solution with low-norm partial d-stars found by the Partial d-Star Heuristic. The algorithm stops when all terminals are spanned with directed partial d-stars. The output of this algorithm is a d-star with the same cost bound as in Lemma 3. Finally, combining the analogs of Lemmas 2 and 3 with Theorem 1, we obtain the following result[8]:

Theorem 7 *Optimal directed Steiner trees can be approximated in polynomial time with a performance ratio of $O(k^\epsilon)$ for any fixed $\epsilon > 0$.*

6 Time Complexity and Practical Enhancements

We now analyze the time complexity of our heuristic and discuss several ways of improving its runtime and performance. The time complexity for preprocessing is dominated by the time to compute all-pairs shortest paths in a graph G (we denote this time complexity by τ). All node-to-group distances (i.e., distances between each node and the closest node to it in each group) can be computed in time less than τ.

[7]If a node v is reachable from a node u in G, then in the transitive closure of G there is an arc (u,v) whose cost is equal to the cost of the minimum directed u-to-v path.

[8]Although the case of acyclic directed graphs was considered in [24], the technique suggested there does not generalize to arbitrary directed graphs.

The time complexity of the Partial d-Star Heuristic (Figure 10) is $O(|V|^{d-1} \cdot k^d)$, where k is the number of groups, and d is the depth. Therefore, the Rooted d-Star Heuristic (Figure 6) has runtime $O(|V|^{d-1} \cdot k^{d+1})$. In order to find a low-cost Steiner d-star, we run our Rooted d-Star Heuristic for all possible roots from the smallest group (i.e., size at most $|V|/k$). Therefore, an overall runtime of $O(\tau + (|V| \cdot k)^d)$ is sufficient to find a group Steiner tree with cost at most $2d \cdot (2 + \ln(2k))^{d-1} \cdot \sqrt[d]{k}$ times optimal.

In practice, it would be of interest to improve the average-case runtime and solution quality of our heuristics beyond the theoretical worst-case bounds derived above. For example, we may omit the removal of the set of groups spanned by the minimum-norm d-star (see the inner loop of the algorithm in Figure 6). Instead, every time we accept an intermediate node, we can update the best possible current star by calculating the distance to a particular group, not from the root, but rather from the closest already-accepted intermediate node. This will tend to improve the solution cost in practice.

The fact that an optimal group minimum spanning tree is at most twice longer than the optimal group Steiner tree [16, 21] suggests another practical enhancement. This involves computing a *group minimum spanning tree* instead of a *group Steiner tree*, i.e., a minimum spanning tree for a set of nodes containing *exactly* one node from each group. Since approximating the group Steiner tree by a group minimum spanning tree incurs an approximation penalty of at most a factor of 2, this does not asymptotically increase the overall bound yet can yield substantial runtime savings.

Finally, a post-processing step can be added which finds the minimum spanning (or approximate Steiner) tree for the set of intermediate nodes chosen by the group Steiner heuristic. We may also make local (one at a time) node substitutions in groups to re-arrange the tree topology in order to reduce the overall cost. Although provably-good heuristics are frequently outperformed by local optimization methods, the output of provably-good heuristics can serve as a good starting point for local-improvement post-processing schemes. For example, it was shown in [20] that Christofides' traveling salesperson heuristic [17] provides excellent initial traveling salesperson tours for further local rearrangements.

7 An Improved Bound for Instances with Degenerate Groups

Additional solution quality and runtime improvements may be realized when an instance of the Group Steiner Problem contains *degenerate* groups (groups of size 1). The degenerate groups, when considered by themselves, induce an instance of the classical Steiner problem, and such an instance can be approximated efficiently by known methods (with a constant performance ratio). Thus, to solve the Steiner Problem for degenerate groups, we may choose a provably-good heuristic from among the numerous existing ones [5, 8, 13, 16, 22]. For example, in time $O(|V|^3)$ we may find a Steiner tree which is at most $\frac{11}{6}$ times longer than the optimal [23].

We now need to effectively combine the Steiner tree over the degenerate groups with a tree which spans the non-degenerate groups. Let $N = M_1 \cup M_2$ be the partition of all the groups in N into those containing one node (M_1), and those containing at least two nodes (M_2). We define the *Combined Group Steiner Heuristic* as follows. First, we find the usual Steiner tree approximation $Approx_1$ for the degenerate groups M_1 using the algorithm from [23]. Next, using our group Steiner heuristic, we find the group Steiner tree $Approx_2$ for the family of groups that includes the non-degenerate groups M_2 and a single arbitrary degenerate group from M_1. Finally, we output a minimum spanning tree over the union $Approx_1 \cup Approx_2$ (Figure 12).

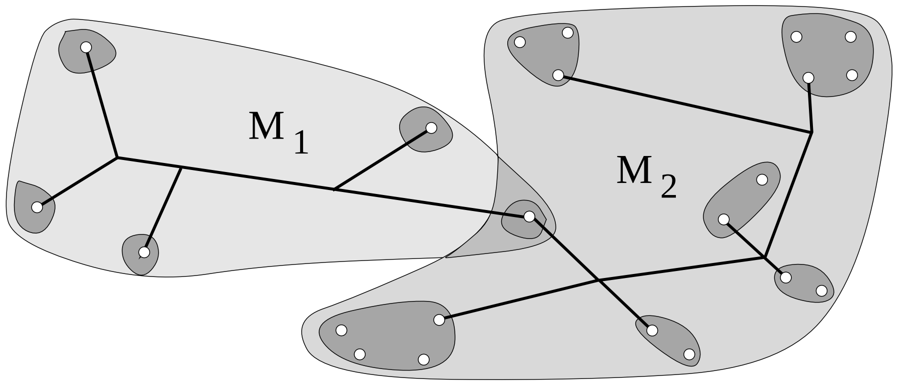

Figure 12: The Combined Group Steiner Heuristic spans the set M_1 of degenerate groups with an approximate Steiner tree $Approx_1$ (right). It then spans all non-degenerate groups M_2 together with an arbitrary degenerate group using a group Steiner tree $Approx_2$ (left).

In cases when the number of degenerate groups is large, the Combined Group Steiner Heuristic will run considerably faster than the basic group Steiner heuristic described above. Moreover, the bound on the solution quality will also improve, as shown in the following theorem.

Theorem 8 *The Combined Group Steiner Heuristic solves the Group Steiner Problem for k' non-degenerate groups with a performance ratio of at most:*

$$\frac{11}{6} + 2d \cdot [2 + \ln(2 \cdot (k' + 1))]^{d-1} \cdot \sqrt[d]{k' + 1}$$

Proof: The optimal Steiner tree Opt_1 for the set of degenerate groups M_1, cannot cost more than the optimal group Steiner tree Opt. Therefore, for the approximate Steiner tree $Approx_1$ spanning the degenerate groups of M_1 (see [23]), we have:

$$cost(Approx_1) \leq \frac{11}{6} \cdot cost(Opt_1) \leq \frac{11}{6} \cdot cost(Opt)$$

If we remove from the family of groups N all but 1 of the degenerate groups, then the cost of the optimal Steiner tree over the remaining groups can not be more than Opt. Theorem 4 implies that the cost of the approximate Steiner tree $Approx_2$ spanning the remaining $k' + 1$ groups is at most:

$$cost(Approx_2) \quad \leq \quad 2d \cdot [2 + \ln(2 \cdot (k' + 1))]^{d-1} \cdot \sqrt[d]{k' + 1} \cdot cost(Opt)$$

Combining these two bounds on the cost of the trees $Approx_1$ and $Approx_2$ yields the desired result.

□

8 Group Steiner Trees With Bounded Radius

In deep-submicron VLSI design, the objective of delay minimization often induces wiring geometries that are substantially different from those dictated by an optimal-area objective. This motivated several bounded-radius[9] routing algorithms [2, 3, 6, 14, 15]. Our basic group Steiner tree approach can be easily extended into a bounded-radius construction, thus yielding routing trees with source-to-sink pathlengths bounded by a user-specified parameter.

Towards this goal we can utilize the tree produced by our main algorithm as the starting point in the bounded-radius construction of [6]. For an instance of the Group Steiner Problem with k groups, this approach yields a routing tree with the following simultaneous cost/radius bounds: its radius is

[9]The *radius* of a graph is defined as the maximum length of any shortest source-sink path. Note that the radius of d-stars is implicitly bounded by d times the optimal radius.

$(1+\epsilon)$ times the optimal radius, and its total cost is $(1+\frac{2}{\epsilon}) \cdot 2d \cdot (2+\ln(2k))^{d-1} \cdot \sqrt[d]{k}$ times the optimal cost, for *any* fixed user-specified parameter $\epsilon > 0$. This scheme provides a provably-good smooth tradeoff between tree cost and tree radius for the group Steiner problem.

9 An Empirical Study

We used Java to implement our main heuristic for the group Steiner problem (our implementation is executable on the Web at http://www.cs.virginia.edu/robins/groupSteiner). We also implemented a variant of our basic heuristic where the solutions are post-processed with a minimum spanning tree algorithm (i.e., we output the minimum spanning tree over the nodes selected by our main heuristic). We compared our heuristics with the heuristic proposed by Reich & Widmayer [19], denoted by RW. The RW heuristic begins by first finding the minimum spanning tree T for the entire set of nodes of all the groups. If a leaf node is not the last member of its group in the tree T, then it may be removed. The RW heuristic then repeatedly deletes such a leaf node incident to the longest edge among all such nodes, as detailed in Figure 13.

```
The RW heuristic for the Group Steiner Problem [19]
Input: A graph $G = (V, E)$, $k$ disjoint groups $N_1, \ldots, N_k \subseteq V$
Output: A low-cost tree $T$ spanning at least one node from each $N_i$
$M \leftarrow \cup_{i=1}^{k} N_i$
Find minimum spanning tree $T$ over $M$, i.e. $T \leftarrow MST(M)$
Repeat forever
    For each leaf $l$ of $T$ do
        Find the group $N(l)$ containing $l$
        If $N(l) \cap T = l$ then mark $l$
    If all leaves of $T$ are marked then exit repeat
    Find an unmarked leaf $l$ incident to the longest edge
    Remove $l$ from $T$, i.e. $T \leftarrow T - l$
Output the tree $T$
```

Figure 13: The heuristic of Reich & Widmayer [19] for the Group Steiner Problem

Figure 14 gives a comparison between our heuristic and the RW algorithm [19] (Figure 13). We generated instances of the Group Steiner Problem by uniformly and independently distributing the nodes of each group inside a randomly-placed square area of predetermined size. We varied the group areas among 10%, 50%, and 100% of the overall region. All chart data represent the average relative improvement over 100 trials, given as a percent improvement over the tree cost of the RW algorithm.

We implemented two versions of our group Steiner heuristic. The first variant embodies the following three modifications over the basic version described in previous sections: (1) Intermediate nodes are selected strictly from among the groups (i.e., all of the constructions benchmarked are *spanning* trees since they do not use Steiner nodes other than group nodes); (2) the root of the d-star is selected from a single randomly-chosen group; and (3) after accepting an intermediate node in the inner loop of Figure 9, it is removed from further consideration in subsequent iterations.

Our second heuristic variant adds a minimum spanning tree post-processing step to the first version above (i.e., we output the minimum spanning tree over the nodes selected by our heuristic modified as described above). The data in Figure 14 labeled "2-star" shows the average percent improvements of our modified heuristic for $d = 2$ over the RW algorithm, while data for the MST-postprocessed variant is given by the data labeled "2-star + MST". As can be seen from the data, the second variant significantly outperforms the RW algorithm with increasing group sizes and areas.

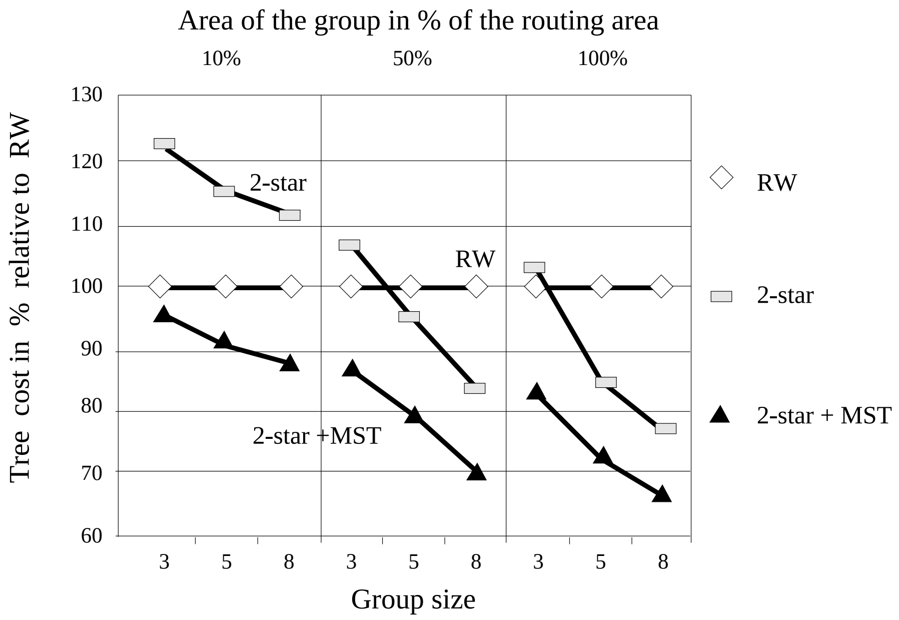

Figure 14: Empirical data: average tree cost produced by our main heuristic (rectangles) and our MST post-processing heuristic (triangles), normalized with respect to the RW heuristic [19] (diamonds), for various group sizes.

10 Conclusion

In this paper we addressed a generalization of the classical Steiner problem, with the goal of finding a minimum-cost tree spanning k groups of nodes. Our main result is a series of polynomial-time approximation algorithms with performance ratio $O(k^\epsilon)$ for any fixed $\epsilon > 0$. This result improves the previously known $O(k)$-approximation and also applies to the Steiner problem in directed graphs. Empirical results produced by our implementation indicate that our approach is also effective in practice.

11 Acknowledgments

We are grateful to Douglass Bateman for help with the implementation, and to Andrew Kahng, Sarah Friend, and the referees for their valuable feedback. This work was supported by a Packard Foundation Fellowship, by NSF Young Investigator Award MIP-9457412, and by NSF grant CCR-9988331.

References

[1] M. J. Alexander and G. Robins. New performance-driven fpga routing algorithms. *IEEE Trans. Computer-Aided Design*, 15(12):1505–1517, December 1996.

[2] C. J. Alpert, T. C. Hu, J. H. Huang, and A. B. Kahng. A direct combination of the Prim and Dijkstra constructions for improved performance-driven global routing. In *Proc. IEEE Intl. Symp. Circuits and Systems*, pages 1869–1872, Chicago, IL, May 1993.

[3] B. Awerbuch, A. Baratz, and D. Peleg. Cost-sensitive analysis of communication protocols. In *Proc. ACM Symp. Principles of Distributed Computing*, pages 177–187, 1990.

[4] C. D. Bateman, C. S. Helvig, G. Robins, and A. Zelikovsky. Provably-good routing tree construction with multi-port terminals. In *Proc. International Symposium on Physical Design*, pages 96–102, Napa Valley, CA, April 1997.

[5] P. Berman and V. Ramaiyer. Improved approximations for the Steiner tree problem. In *Proc. ACM/SIAM Symp. Discrete Algorithms*, pages 325–334, San Francisco, CA, January 1992.

[6] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh, and C. K. Wong. Provably good performance-driven global routing. *IEEE Trans. Computer-Aided Design*, 11(6):739–752, 1992.

[7] U. Feige. A threshold of ln n for approximating set cover. In *Proc. ACM Symp. on the Theory of Computing*, pages 314–318, May 1996.

[8] J. Griffith, G. Robins, J. S. Salowe, and T. Zhang. Closing the gap: Near-optimal Steiner trees in polynomial time. *IEEE Trans. Computer-Aided Design*, 13(11):1351–1365, November 1994.

[9] C. S. Helvig, Gabriel Robins, and Alexander Zelikovsky. Improved approximation bounds for the group Steiner problem. In *Proc. Conference on Design Automation and Test in Europe*, pages 406–413, Paris, France, February 1998.

[10] F. K. Hwang, D. S. Richards, and P. Winter. *The Steiner Tree Problem.* North-Holland, 1992.

[11] E. Ihler. Bounds on the quality of approximate solutions to the group Steiner problem. In *Lecture Notes in Computer Science*, volume 484, pages 109–118, 1991.

[12] E. Ihler. The complexity of approximating the class Steiner tree problem. Technical report, Institut für Informatik, Universität Freiburg, 1991.

[13] A. B. Kahng and G. Robins. A new class of iterative Steiner tree heuristics with good performance. *IEEE Trans. Computer-Aided Design*, 11(7):893–902, July 1992.

[14] A. B. Kahng and G. Robins. *On Optimal Interconnections for VLSI.* Kluwer Academic Publishers, Boston, MA, 1995.

[15] S. Khuller, B. Raghavachari, and N. Young. Balancing minimum spanning and shortest path trees. In *Proc. ACM/SIAM Symp. Discrete Algorithms*, pages 243–250, January 1993.

[16] L. Kou, G. Markowsky, and L. Berman. A fast algorithm for Steiner trees. *Acta Informatica*, 15:141–145, 1981.

[17] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. *The Traveling Salesman Problem: a Guided Tour of Combinatorial Optimization.* John Wiley and Sons, Chichester, New York, 1985.

[18] B. T. Preas and M. J. Lorenzetti. *Physical Design Automation of VLSI Systems.* Benjamin/Cummings, Menlo Park, CA, 1988.

[19] G. Reich and P. Widmayer. Beyond Steiner's problem: A vlsi oriented generalization. In *Lecture Notes in Computer Science*, volume 411, pages 196–211, 1989.

[20] G. Reinelt. *The Traveling Salesman: Computational Solutions for TSP Applications.* Springer Verlag, Berlin, Germany, 1994.

[21] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Mathematica Japonica*, 24(6):573–577, 1980.

[22] A. Z. Zelikovsky. An 11/6 approximation algorithm for the network Steiner problem. *Algorithmica*, 9:463–470, 1993.

[23] A. Z. Zelikovsky. A faster approximation algorithm for the Steiner tree problem in graphs. *Information Processing Letters*, 46(2):79–83, May 1993.

[24] A. Z. Zelikovsky. A series of approximation algorithms for the acyclic directed Steiner tree problem. *Algorithmica*, 18:99–110, 1997.