New Approximation Algorithms for Routing with Multiport Terminals

Christopher S. Helvig, Gabriel Robins, and Alexander Zelikovsky

***Abstract*—Previous literature on very large scale integration routing and wiring estimation typically assumes a one-to-one correspondence between terminals and ports. In practice, however, each "terminal" consists of a large collection of electrically equivalent ports, a fact that is not accounted for in layout steps such as wiring estimation. In this paper, we address the general problem of minimum-cost routing tree construction in the presence of multiport terminals, which gives rise to the *group Steiner minimal tree problem.* Our main result is the first known approximation algorithm for the group Steiner problem with a *sublinear* performance bound. In particular, for a net with k multiport terminals, previous heuristics have a performance bound of $(k-1) \cdot OPT$, while our construction offers an improved performance bound of $2 \cdot (2 + \ln(k/2)) \cdot \sqrt{k} \cdot OPT$. Our Java implementation is available on the Web.**

***Index Terms*—Approximation algorithms, combinatorial optimization, group Steiner problem, multiport terminals, routing, Steiner trees.**

I. Introduction

PREVIOUS works on routing often assume a one-to-one correspondence between terminals and ports, either implicitly or explicitly requiring each terminal to consist of a *single* port. However, in actual layouts (e.g., in a gridded routing regime), a "terminal" to which a wire is to be routed can consist of a large collection of separate ports. Even though a wire may connect to any one of these ports, this degree of freedom is often not fully exploited, particularly in routing or in wiring estimation.

In this paper, we address the general problem of minimum-cost Steiner tree construction in the presence of multiport terminals. Clearly, the problem of interconnecting a net with multiport terminals is a direct generalization of the NP-hard Steiner problem and is, therefore, itself NP-hard (i.e., in the classical Steiner problem each terminal contains exactly one port). The classical Steiner minimal tree problem is defined as follows.

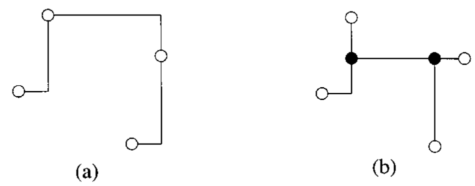


Fig. 1. (a) The minimum spanning tree (MST) and (b) the Steiner minimal tree (SMT) in the rectilinear plane. Solid dots represent Steiner nodes.

The Steiner Minimal Tree (SMT) problem: 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. A rectilinear instance of the Steiner problem is shown in Fig. 1.

We address the generalization of the Steiner Minimal Tree problem, where rather than spanning a set of nodes, the objective is to connect *groups* of nodes. This problem, which models the routing of nets with multiport terminals, is formalized as follows.

The Group Steiner Problem [9], [19]: given an undirected weighted graph $G = (V, E)$ and a family $N = \{N_1, \ldots, N_k\}$ of k disjoint groups of nodes $N_i \subseteq V$, find a minimum-cost tree which contains at least one node (i.e., port) from each group N_i.

As in the classical Steiner problem, we are allowed to include optional (i.e., Steiner) nodes, in order to reduce the cost of the tree interconnecting the groups of N. One version of the group Steiner problem, known as the *strong connectivity* version, allows different connections to the same group to attach to *different* nodes in that group [i.e., all the nodes of a group are implicitly connected to each other, which allows the solution to the group Steiner problem to be a forest—see Fig. 2(b)]. The version of the group Steiner problem that we study involves *weak connectivity*: the solution must be strictly a tree, and intra-group edges must be explicitly part of the solution [see Fig. 2(a)].

The group Steiner problem captures several practical scenarios in very large scale integration layout design.

- Even in single-port-per-terminal scenarios, the possibility of rotating and flipping a module induces multiple locations for the given port. For a general module, there are up to eight possible orientations [19] [see Fig. 3(a)], and a given terminal will induce a group of up to eight nodes

Manuscript received July 1, 1997; revised March 15, 1998 and November 22, 1998. This work was supported by a Packard Foundation Fellowship and by National Science Foundation (NSF) Young Investigator Award MIP-9457412. A preliminary version of this work appeared in Proc. International Symposium on Physical Design, Napa Valley, CA, April 1997 and in Proc. Conf. Design Automation and Test in Europe, Paris, France, 1988. This paper was recommended by Associate Editor M. Sarrafzadeh.

C. S. Helvig and G. Robins are with the Department of Computer Science, University of Virginia, Charlottesville, VA 22903-2442 USA.

A. Zelikovsky is with the Department of Computer Science, Georgia State University, Atalanta, GA 30303 USA.

Publisher Item Identifier S 0278-0070(00)09149-1.

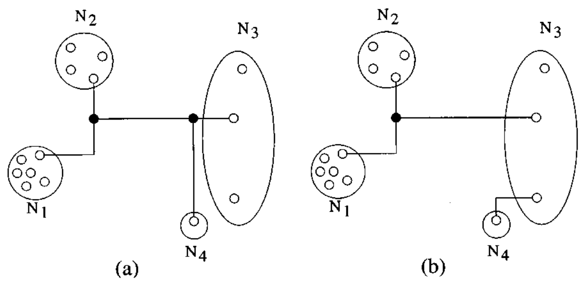


Fig. 2. (a) A feasible solution for the weak-connectivity version of the group Steiner problem. (b) A solution to the same group Steiner problem instance under the strong-connectivity assumption. Ovals represent terminals (i.e., groups), hollow dots represent ports within a terminal, and solid dots represent Steiner nodes.

in the group Steiner problem formulation[1] [see Fig. 3(b)]. The use of "virtual" ports is mutually exclusive, and a version of the *weak connectivity* model applies.

- The group Steiner problem also models the pin assignment problem [15], which seeks to optimally determine pin locations on module boundaries. Usually, exactly one pin is assigned to each module [18], and the *weak connectivity* model applies.
- In grid-based maze routing regimes, "dot-models" or similar techniques are commonly used to create multinode routing abstracts for each terminal within the multilayer grid. A complicated terminal geometry can easily have 30 or more ports located on multiple fabrication layers. These ports form a group in the group Steiner problem formulation. The ports are electrically equivalent, and a routing tree may connect to more than one port of a given terminal. Hence, the *strong connectivity* model applies.
- On a larger length scale, multiple ports on a block boundary may be electrically equivalent by virtue of being connected inside the block. Again, the strong-connectivity group Steiner problem version applies with these sets of ports forming groups.

Despite such applications, the freedom to connect to any of multiple port locations has neither been explored in geometric or graph-based routing tree constructions, nor has it been accounted for in wiring estimation.[2] In deep submicrometer technologies, the error incurred by approximating multiple ports with, e.g., their center of gravity can be substantial when propagated through multiple logic stages. Furthermore, instances of this problem become common when hierarchical design methodologies are applied (e.g., when global nets are partially prerouted).

The only existing approximation algorithms for the weak group Steiner problem produce solutions $k-1$ times worse than optimal[3] where k is the number of groups [10]. In this paper, we propose a new heuristic with an improved performance ratio[4] of $2 \cdot (2 + \ln(k/2)) \cdot \sqrt{k}$, where k is the number of groups. On the negative side, we show that this problem is not likely to be approximable to within polynomial time with a performance bound of less than $\ln k \cdot$ OPT [6], [11]. This implies that one can not expect to find efficient heuristics with sublogarithmic performance bounds. Note that in practice our heuristic produces solutions that are much better than the theoretical bound indicates. This is apparent from our experimental results and from examining actual trees produced by our implementation. Note that although our algorithms are general and apply to arbitrary weighted graphs, our implementation (and the experimental results section) uses the rectilinear metric to determine the distances between ports.

The rest of the paper is organized as follows. In Section II, we introduce a special type of *depth-bounded* Steiner tree. We prove that an optimal depth-2 bounded Steiner tree approximates the optimal Steiner tree to within a factor of $2 \cdot \sqrt{k}$. In Section III, we present our main heuristic that approximates the optimal depth-2 Steiner tree to a $2+\ln(k/2)$ factor (our overall method, therefore, has a performance bound that is the product of these two factors). Analysis given in the Appendix shows that our heuristic cannot be improved substantially (i.e., a logarithmic factor is a lower bound). We finish Section III with the proof of the claimed performance bound for our main heuristic. Section IV analyzes its time complexity, and discusses some practical enhancements. Section V shows how to improve the bound even further by specially treating groups of size one. Section VI extends our basic group Steiner approach into a bounded-radius formulation that minimizes tree cost as well as source-to-sink pathlengths in a provably good manner. Finally, we discuss our implementation and experimental results in Section VII. A preliminary version of this work appeared in [3], [8].

II. Depth-Bounded Steiner Trees

In this section, we introduce the concept of Steiner depth-bounded[5] trees. Our motivation for using depth-bounded trees is twofold: 1) optimal depth-2-bounded trees can be used to approximate optimal group Steiner trees to within a factor of $2\cdot\sqrt{k}$, and 2) optimal depth-2-bounded trees in turn can be approximated efficiently, as discussed in the next section. Our overall method is thus a composition of these two approximations and, therefore, enjoys a performance bound that is the product of the two corresponding bounds.

In general, the given graph G may 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-v path in G. 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 by its *metric closure*.[6] In order to further simplify our

[1]For standard cells there are really only two (or possibly four) orientations possible.

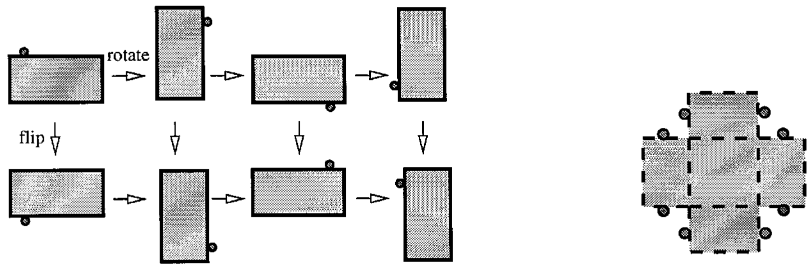
[2]Detailed maze routers implicitly exploit this degree of freedom, but their solutions may have unbounded error.

[3]In contrast, the strong connectivity version, though also NP-hard, is somewhat more tractable than the weak connectivity version: by converting an instance of the strong connectivity version into an instance of the graph Steiner problem, then setting to zero the weight of every intra-group edge, we can efficiently solve the strong group Steiner problem to within a factor of two or less of optimal using known graph-based Steiner tree algorithms [16], [21], [22].

[4]The *performance ratio* is an upper bound on the ratio of heuristic solution cost divided by optimal solution cost, over all problem instances [i.e., the worst-case of $cost(APPROX)/cost(OPT)$].

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

[6]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-v path in G.



(a) (b)

Fig. 3. (a) A module is rotated and flipped to induce a group of eight terminal positions, shown in (b).

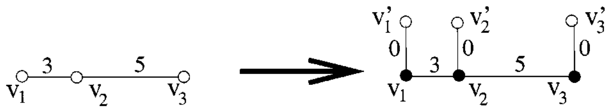
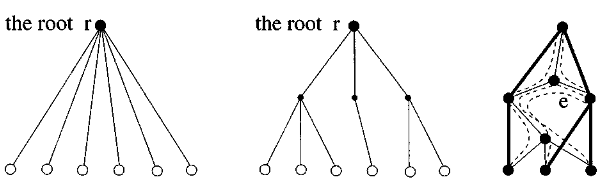


Fig. 4. Transformation of G: each terminal node v_i in G is connected to a newly created node v_i' with a zero-cost edge (i.e., terminal nodes thus become nonterminals, and newly created corresponding nodes replace the roles of the original terminal nodes). Note that this transformation alters G in a way that enables us to transform a solution in the modified graph back into a solution of the same cost in the unmodified graph.



(a) (b) (c)

Fig. 5. (a) A 1-star, and (b) a 2-star with root r. (c) Graph edges are thin, while 2-star edges are thick; the edge e is (re)used here three times in dashed paths between nodes adjacent in the 2-star.

analysis, we also modify G as follows. For any port v_i, we create a new node v_i' and a new zero-cost edge (v_i, v_i'); we let v_i' take on the role of v_i, i.e., v_i' becomes a port and v_i becomes a nonport, as shown in Fig. 4. An optimal tree in the modified graph has the same cost as an optimal tree in the original graph. Hence, this transformation allows us to seek the optimal Steiner tree in which every port is a leaf.

We define *d-stars* to be rooted trees of depth of at most d [see Fig. 5(a), (b)]. 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 2-star spanning the leaves of T. This will imply that an optimal group Steiner tree can be approximated by a low-cost *group Steiner 2-star* (defined as a 2-star which spans all of the groups). Our overall strategy is to specify a low-cost 2-star and derive upper bounds on its total cost.

In deriving upper bounds on the cost of 2-stars, we will sum the costs of tree paths between nodes which are adjacent in 2-stars. Such paths are not necessarily disjoint, and the same tree edge may be counted multiple times in this sum. We refer to this situation as *edge reuse* [see Fig. 5(c)]. For the tree T, edge reuse provides a loose upper bound on the ratio cost(2-star)/cost(T): if no edge is used more than j times when replacing edges of a 2-star by the corresponding paths in T, then the 2-star has cost no more than j times the cost of the tree T. Our strategy for deriving upper bounds on the cost of a 2-star is to bound its edge reuse.

More formally, given a tree T, a 1-star S_1, and a 2-star S_2 (collectively denoted S_d), let $reuse_T(S_d)$ denote the maximum number of times that any tree edge from T is used in tree paths connecting nodes adjacent in S_d. In order to establish an upper bound on the cost of a 2-star, we first select an appropriate common root r for S_1 and S_2. The following is a known result from graph theory (which we prove here for the sake of completeness).

Lemma 1: Any tree T has at least one node r, called the *center*, such that each connected component of $T-\{r\}$ contains at most half of the leaves of T (See Fig. 6).

Proof: We direct each edge $e = (u, v)$ in T from u to v if the number of leaves in the connected component of $T - \{e\}$ containing v is strictly more than the number of leaves in the other component [See Fig. 6(a)]. Since T is a tree, we can start from an arbitrary node and walk along directed edges until we reach a node r without any incident outgoing edges. The node r is a center since no connected component of $T - \{r\}$ contains more leaves than the sum of the leaves in all of the other (disjoint) components. □

Placing the root r of the 1-star S_1 at a center of the tree T minimizes the edge reuse of S_1, as follows.

Lemma 2: Let r be a center of the tree T with the set of leaves L. The 1-star S_1 with the root r and leaves L costs no more than $(|L|/2) \cdot cost(T)$.

Proof: Since r is a center of T, $reuse_T(S_1)$ is not larger than $|L|/2$ [See Fig. 6(b)]. Indeed, the reuse of an edge e of T is the number of root-to-leaf paths in T that contain e. Since r is a center, any edge of T lies on at most half of all such paths. □

We now construct a 2-star S_2 from the tree T with cost no more than $2 \cdot \sqrt{|L|} \cdot cost(T)$, where L is the set of leaves of T. Note that a center r of the tree T will serve as a root of both S_1 and S_2. Note also that S_2 and T (and S_1) have the same set of leaves, namely L. Let C_i be connected components of $T - \{r\}$. In order to connect nodes in different C_i, we must pass through the root r. Define T_i as a component C_i plus the root r and the (unique) edge between the root and C_i [Fig. 6(a) shows three such components C_i]. Finally, we denote by $L_i = C_i \cap L$ the set of leaves of T that are contained in T_i. Because the root r is a center of T, the size of L_i is at most $|L|/2$ in each rooted subtree T_i. We construct a 2-star S_2 for the entire tree T by taking the union of the 2-stars for each of the subtrees T_i. Note that the edge reuse of S_2 in T is the maximum of the edge reuses over all T_i. We define a node u as an *ancestor* of v if the path from the root to v passes through u.

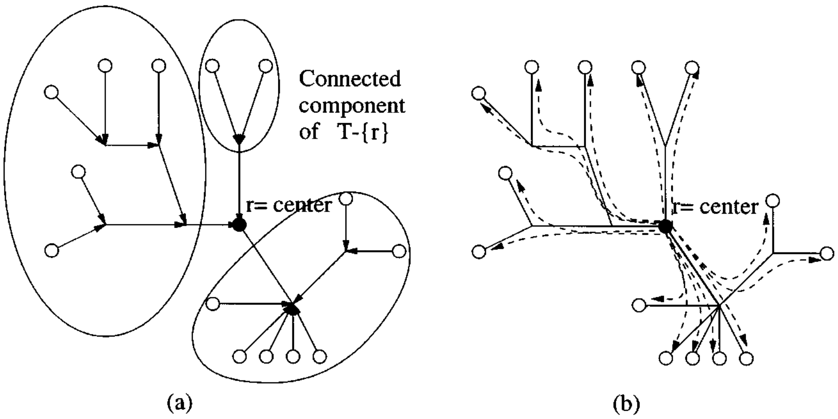


Fig. 6. (a) When a tree center is removed, no subtree has more than $|L|/2$ leaves. (b) When we place the root at a center of the tree, no edge is reused more than $|L|/2$ times (in dashed paths). In this example, the edge reuse of the 1-star is 7.

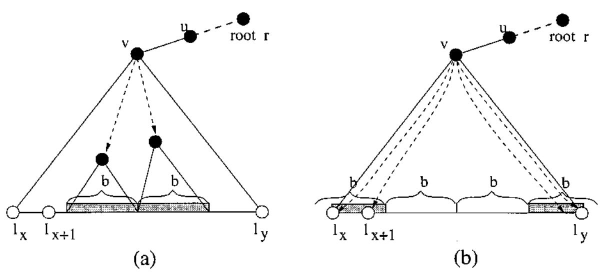


Fig. 7. Triangles represent depth-first -ordered subtrees. (a) Type (i) paths which reuse the edge (u, v) terminate at intermediate nodes below v (there are no more than $|L_i|/b$ of these). (b) Type (ii) paths which reuse the edge (u, v) terminate at leaves that lie in the leftmost and the rightmost blocks that together contain at most $2 \cdot (b - 1)$ leaves.

Now we determine an appropriate set of intermediate nodes for a 2-star S_2 in each T_i. First, we label the leaves of T_i in an arbitrary depth-first manner, $l_1, l_2, \ldots, l_{|L_i|}$. Next, we partition this sequence into fixed-size blocks of contiguously numbered leaves. Finally, we choose the intermediate nodes of our 2-star for T_i as the set of least common ancestors[7] of the leaves in each of the blocks. In our 2-star, the root is connected to these intermediate nodes, and each intermediate node is connected to the leaves of the corresponding block.

Note that the edge reuse is determined by two kinds of paths: (i) paths from the root to intermediate nodes [Fig. 7(a)], and (ii) paths from intermediate nodes to leaves [Fig. 7(b)]. The number of paths of type (i) is clearly bounded by the number of blocks, i.e., $|L_i|/b$, where b is the block size.

We now estimate the contribution to edge reuse of paths of type (ii). Since we have ordered the nodes in a depth-first manner, any node v will be a common ancestor for a sequence of contiguously numbered leaves, say $l_x, l_{x+1}, \ldots, l_y$. Clearly, v is an ancestor of the least common ancestors of all blocks that are contained completely in this sequence. Because the edge between v and its parent u does not lie on the paths from the intermediate nodes to leaves of such blocks, we are concerned only with paths to the leaves outside of such blocks. Such leaves may occupy only the first or the last $b - 1$ positions in the sequence $l_x, \ldots, l_y$. This induces a bound of $2 \cdot (b - 1)$ on the contribution of type (ii) paths to the reuse of edge (u, v).

Thus, the total edge reuse is at most $(|L_i|/b) + 2 \cdot b$. Choosing $b = \sqrt{|L_i|/2}$ yields an upper bound on edge reuse of $2 \cdot \sqrt{2 \cdot |L_i|}$. Since the root is the center of T, we have $|L_i| \leq \frac{1}{2}|L|$, and the edge reuse is at most $2 \cdot \sqrt{|L|}$. This proves that for any tree T with the set of leaves L and a center r, there is a 2-star rooted at r with the same set of leaves L, and with cost at most $2 \cdot \sqrt{|L|} \cdot cost(T)$. If we define T to be an optimal group Steiner tree with $k = |L|$ leaves,[8] then the above argument implies that there exists a group Steiner 2-star of cost at most $2 \cdot \sqrt{k} \cdot cost(T)$, where k is the number of groups. We, therefore, obtain the following result.

Theorem 1: Let *Opt* be an optimal group Steiner tree, let k be the number of groups, and let r be a center of *Opt*. Then

1) the cost of an optimal Steiner 1-star rooted at r is at most $(k/2) \cdot cost(Opt)$; and
2) the cost of an optimal Steiner 2-star rooted at r is at most $2 \cdot \sqrt{k} \cdot cost(Opt)$.

III. A Provably-Good Heuristic

We have established that an optimal Steiner 2-star is a reasonable approximation of an optimal group Steiner tree, denoted *Opt*. It can be shown that even the problem of approximating an optimal Steiner 2-star is as difficult as approximating a minimum set cover (see the Appendix). Therefore, for any $\epsilon > 0$, it is unlikely that there exists a polynomial-time heuristic with performance ratio $(1-\epsilon) \cdot \ln k$, where k is the number of groups [6]. We present an algorithm for approximating a Steiner 2-star with performance ratio $2 + \ln(k/2) \approx 1.307 + \ln k$. Therefore, the overall performance bound for our Group Steiner Minimal Tree heuristic will be the product of these two factors (namely the approximation bound of 2-stars with respect to optimal, times the bound with which we can approximate 2-stars themselves).

Theorem 1 states that there exists a low-cost Steiner 2-star with the root placed in a center of an optimal group Steiner tree *Opt*. Although we do not know which node of the graph G is

[7]A common ancestor of a set of nodes is the *least* common ancestor if it is not an ancestor of any other common ancestor of these nodes.

[8]Note that by the transformation described in Fig. 4, at most one leaf node in each group will be spanned by an optimal group Steiner tree and, therefore, we can set $k = |L|$ here.

Group Steiner heuristic for arbitrary weighted graphs

Input: A graph $G = (V, E)$, a family N of k disjoint groups $N_1, \ldots, N_k \subseteq V$
Output: A low-cost tree *Approx* spanning at least one vertex from each group N_i

For each node $r \in V$ **do**
Find a low-cost 2-star $Approx_2(r)$ rooted at r intersecting each group $N_i, i = 1, ..., k$
Output the least-cost 2-star *Approx*, i.e. $cost(Approx) = \min_{r \in V} cost(Approx_2(r))$

Fig. 8. Our main approximation algorithm for the group Steiner problem on arbitrary graphs produces a low-cost Steiner 2-star.

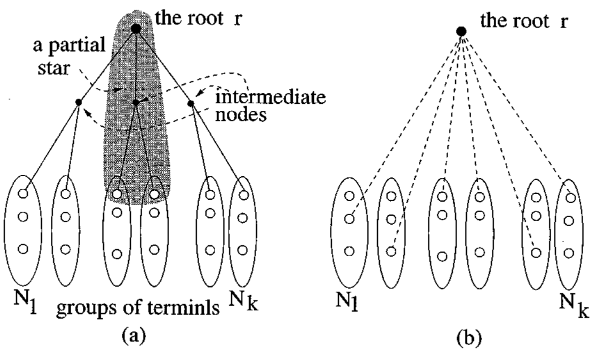


Fig. 9. (a) A Steiner 2-star, and (b) the corresponding optimal 1-star. The shaded area in (a) is representative of our so-called "partial stars," the determination of which is essential to our algorithm.

a center of *Opt*, this is not an obstacle: for each node r of G, we construct a low-cost Steiner 2-star $Approx_2(r)$ with root r, and then we select the least-cost $Approx_2(r)$ over all possible choices of r (see Fig. 8). Thus, we only need to specify how we will construct a Steiner 2-star $Approx_2(r)$ with a given root r.

Let $Opt_1(r)$ be the optimal Steiner 1-star rooted at r, and let $Opt_2(r)$ be the optimal Steiner 2-star rooted at r. The main idea in constructing $Approx_2(r)$ 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 $Opt_2(r)$: first, unlike $Opt_2(r)$, the 1-star $Opt_1(r)$ can be computed efficiently; secondly, the cost of $Opt_1(r)$ is bounded by $(k/2) \cdot cost(Opt)$ if r is a center of *Opt* (see Theorem 1). Therefore, to measure the approximation quality of a 2-star, we will compare its cost to the cost of an optimal 1-star with the same root and with leaves taken from the same groups spanned by the 2-star.

Formally, let S be a 2-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 $groups(S)$ (see Fig. 9). We define the *norm* of S as $norm(S) = cost(S)/cost(S')$.

In order to specify our low-cost 2-star $Approx_2(r)$, we need to select the intermediate nodes and also to determine the set of groups that should be connected to each intermediate node. It is, therefore, natural to represent $Approx_2(r)$ as a union of subtrees, each consisting of a single intermediate node that is connected to the root as well as to certain leaf ports. Such rooted subtrees will be called *partial stars* [see Fig. 9(a)].

We select the partial stars for $Approx_2(r)$ in the following greedy manner. First, we find a partial star P with the minimum norm (i.e., the minimum ratio of the cost of P over the cost of the corresponding 1-star). Next, we remove the groups that it spans [i.e., $groups(P)$] from the set of all groups. Finally, we determine the next partial star with minimum norm, and iterate until all groups are spanned. Fig. 10 gives a formal definition of this procedure.

In order to complete the description of our heuristic, we will next present an efficient procedure that, given a root r and set of groups M, finds a minimum-norm partial star $P(r, M)$ rooted at r and spanning some of the groups in M. This procedure is formally described in Fig. 12. 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. Fig. 11 illustrates our main heuristic from Fig. 8.

For the remainder of this section, we analyze our heuristic, which is formally described in Figs. 8, 10, and 12. Lemma 3 establishes the correctness of the Minimum-Norm Partial Star Algorithm (Fig. 12), proving that it actually finds the minimum-norm partial star. Lemma 4 yields a performance ratio for the Rooted Steiner 2-star Heuristic (see Fig. 10). Together with Theorem 1, this will imply our main result, namely Theorem 2.

Lemma 3: Given a family $M \subseteq N$ of groups, the Minimum-Norm Partial Star Algorithm (Fig. 12) outputs a minimum-norm partial star.

Proof: Let v be the intermediate node of the minimum-norm partial star P with root r. The Minimum-Norm Partial Star Algorithm (Fig. 12) sorts the family of groups $M \in \{N_1, \ldots, N_{|M|}\}$ such that for any i from $1, \ldots, |M|$

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

We exploit the *ratio property*[9] which states that $a/b \leq c/d$ if and only if $a/b \leq (a + c)/(b + d) \leq c/d$ for any positive a, b, c, d. 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. Then

$$\begin{aligned} norm(P) = \frac{cost(P)}{cost(P')} &\leq \frac{cost(P) - cost(v, N_j)}{cost(P') - cost(r, N_j)} \\ &= \frac{cost(v, N_j) - cost(P)}{cost(r, N_j) - cost(P')}. \end{aligned}$$

By the ratio property, this implies

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

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

and, therefore

$$\frac{cost(v, N_j)}{cost(r, N_j)} \leq \frac{cost(P)}{cost(P')}. \tag{1}$$

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

[9] The proof of the property is as follows: $(a/b) \leq (c/d)$ if and only if $ad \leq bc$ which is true if and only if both $ad + cd \leq bc + cd$ and $ad + ab \leq bc + ab$ which holds if and only if $(a/b) \leq (a + c/b + d)$. We can similarly obtain the analogous form $(a/b) \leq (c/d)$ if and only if $(a + c)/(b + d) \leq (c/d)$.

Rooted Steiner 2-star Heuristic

Input: A graph $G = (V, E)$, a family N of k disjoint groups $N_1, \ldots, N_k \subseteq V$ and a root $r \in V$
Output: A low-cost 2-star $Approx_2(r)$ with the root r intersecting each group N_i

$Approx_2(r) \leftarrow \{r\}$
$N' \leftarrow N$ /* We begin with all groups remaining */
While $N' \neq \emptyset$ **do**
  Find a partial star $P = P(r, N')$ with the minimum norm (see Figure 12)
  $N' \leftarrow N' - groups(P)$
  $Approx_2(r) \leftarrow Approx_2(r) \cup P$
Output $Approx_2(r)$

Fig. 10. The greedy heuristic for a given fixed root. At each iteration of the loop, we add the minimum-norm partial 2-star to the solution, and remove its groups from future consideration. The algorithm terminates when no groups remain to be spanned. At this point, all groups are in the solution, which is the output of this heuristic.

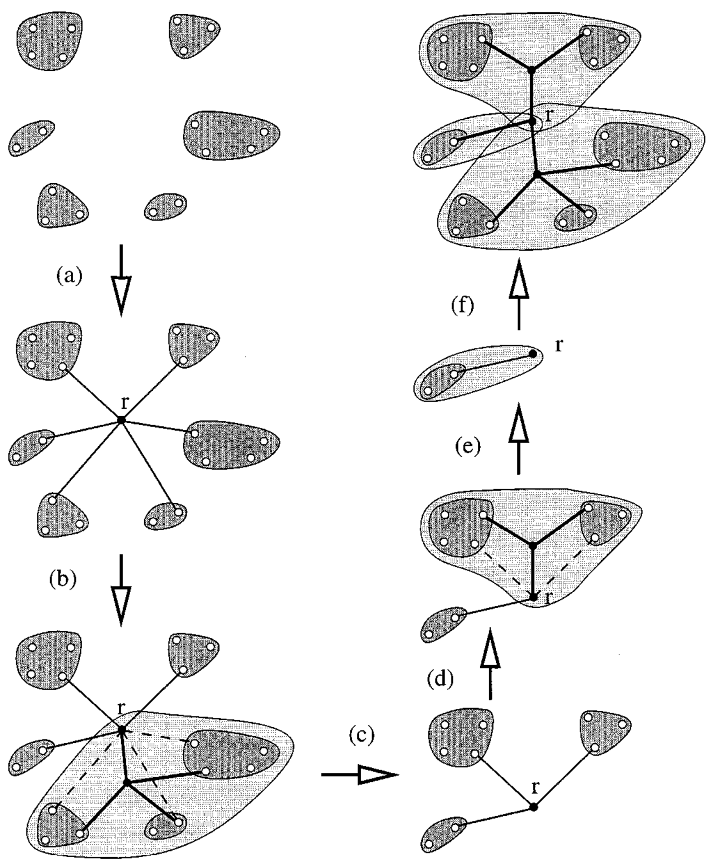


Fig. 11. Given an instance of the group Steiner problem, for each possible root r our heuristic: (a) finds the optimal 1-star, (b) finds the minimum-norm partial star (shaded region), (c) stores this star in the solution and removes its groups from future consideration, (d) finds the next minimum-norm partial star (shaded region), (e) repeat step (c) for the new partial star, and finally (f) finds the last minimum-norm partial star and outputs the union of all stored partial stars.

Minimum-Norm Partial Star Algorithm

Input: A graph $G = (V, E)$, a family M of k disjoint groups $N_1, \ldots, N_k \subseteq V$ and a root $r \in V$
Output: The minimum-norm partial star $P(r, M)$ with the root r and leaves from some groups of M

For each $v \in V$ **do**
  Sort $M = \{N_1, ..., N_k\}$ such that $\frac{cost(v,N_i)}{cost(r,N_i)} \leq \frac{cost(v,N_{i+1})}{cost(r,N_{i+1})}$
  Find $j \in \{1, ..., k\}$ that minimizes
  $norm(v) = \frac{cost(r,v)+\sum_{i=1}^{j} cost(v,N_i)}{\sum_{i=1}^{j} cost(r,N_i)}$
  $M(v) \leftarrow \{N_1, ..., N_j\}$
Find v with the minimum $norm(v)$
Output the partial star $P(r, M)$ with the intermediate node v adjacent to the root r and groups $M(v)$

Fig. 12. Our algorithm for finding a minimum-norm partial star. For each candidate v for the intermediate node, we sort groups N_i according to the potential improvement of inserting v between the root r and each group. Then, we add consecutive groups from the list while the addition decreases the norm of the partial star.

only decrease. Let v and a group N_i with $i < j$, be nonadjacent in P. Then we have

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

The ratio property together with inequality (1) 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. □

Lemma 3 allows us to substitute the Minimum-Norm Partial Star Algorithm of Fig. 12 into the Rooted Steiner 2-star Heuristic of Fig. 10. With the algorithm completely described, we are now ready to prove bounds on its performance.

Lemma 4: Let $Opt_d(r)$, $d = 1, 2$, be an optimal Steiner d-star rooted at r. The cost of the output of the Rooted Steiner 2-star Heuristic (Fig. 10) is at most

$$cost(Approx_2(r)) \le \left(\ln \frac{cost(Opt_1(r))}{cost(Opt_2(r))} + 2\right) \cdot cost(Opt_2(r)).$$

Proof: To prove the lemma, we must compare the optimal 2-star rooted at r to the approximate solution produced by our heuristic. Since the root r is fixed, we will omit it from the notation in the proof. The optimal 2-star Opt_2 can be partitioned into partial stars denoted $R_1, \ldots, R_s$. The approximate Steiner 2-star $Approx_2$ is a union of t minimum-norm partial stars $P_1, P_2, \ldots, P_t$ selected by the Rooted Steiner 2-star Heuristic.

Our algorithm works by greedily choosing partial stars, removing the groups spanned by the newly chosen partial star, and iterating until no groups remain. At each iteration, the algorithm chooses a minimum-norm partial star P_i with the greatest improvement over the corresponding optimal one-star (hereafter, denoted P_i'). Let C_i denote the cost of the optimal Steiner 1-star at the ith iteration of the loop when groups $groups(P_1) \cup \cdots \cup groups(P_i)$ have been removed. Recall that $groups(P_i)$ denotes the set of groups that are spanned by the partial star P_i. In the first iteration, we have $C_0 = cost(Opt_1)$. Inductively, we obtain

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

We may consider the partial stars of the optimal Steiner 2-star, $R_1, \ldots, R_s$, to be sorted by nondecreasing order of their norms. Applying the ratio property, we obtain

$$norm(R_1) \le \frac{\sum_{j=1}^{s} cost(R_j)}{\sum_{j=1}^{s} cost(R_j')} = \frac{cost(Opt_2)}{C_0}.$$

Therefore, by our greedy choice of P_1, we have

$$\frac{cost(P_1)}{cost(P_1')} = norm(P_1) \le norm(R_1) \le \frac{cost(Opt_2)}{C_0}.$$

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 we find the optimal Steiner 2-star Opt_2' over the remaining groups. We compare P_2 with R_1' to get

$$\begin{aligned}\frac{cost(P_2)}{cost(P_2')} &= norm(P_2) \le norm(R_1') \le \frac{cost(Opt_2')}{C_1} \\ &\le \frac{cost(Opt_2)}{C_1}.\end{aligned}$$

Applying this observation inductively, we get

$$\begin{aligned}\frac{cost(P_i)}{cost(P_i')} &\le \frac{cost(Opt_2)}{C_{i-1}}, \qquad i = 1, \ldots, t \\ cost(P_i') &\ge \frac{C_{i-1} \cdot cost(P_i)}{cost(Opt_2)}.\end{aligned} \tag{3}$$

Substituting relationship (3) into relationship (2) gives us

$$C_i \le C_{i-1}\left(1 - \frac{cost(P_i)}{cost(Opt_2)}\right).$$

Unraveling the inequalities above, we obtain

$$\begin{aligned}C_n &\le C_0 \prod_{i=1}^{n}\left(1 - \frac{cost(P_i)}{cost(Opt_2)}\right) \\ \frac{C_0}{C_n} &\ge \left[\prod_{i=1}^{n}\left(1 - \frac{cost(P_i)}{cost(Opt_2)}\right)\right]^{-1}.\end{aligned}$$

Taking the natural logarithm of both sides, we get

$$\begin{aligned}\ln \frac{C_0}{C_n} &\ge -\ln\left[\prod_{i=1}^{n}\left(1 - \frac{cost(P_i)}{cost(Opt_2)}\right)\right] \\ &= -\sum_{i=1}^{n} \ln\left(1 - \frac{cost(P_i)}{cost(Opt_2)}\right).\end{aligned}$$

Using the fact that $\ln(1+x) \le x$ and thus $\ln(1-x) \le -x$, we conclude

$$\ln \frac{C_0}{C_n} \ge \frac{\sum_{i=1}^{n} cost(P_i)}{cost(Opt_2)}. \tag{4}$$

Since $C_t = 0$, there exists an n such that $C_n > cost(Opt_2) \ge C_{n+1}$ (see Fig. 13). Note that $cost(P_{n+1}') \le C_n$. Therefore, inequality (3) yields

$$\frac{cost(P_{n+1})}{cost(Opt_2)} \le \frac{cost(P_{n+1}')}{C_n} \le 1. \tag{5}$$

The performance ratio of the Rooted Steiner 2-star Heuristic can be bounded as follows:

$$\begin{aligned}\frac{cost(Approx_2)}{cost(Opt_2)} &= \frac{\sum_{i=1}^{t} cost(P_i)}{cost(Opt_2)} \\ &\le \frac{\sum_{i=1}^{n} cost(P_i) + cost(P_{n+1}) + C_{n+1}}{cost(Opt_2)} \\ &\le \frac{\sum_{i=1}^{n} cost(P_i)}{cost(Opt_2)} + \frac{cost(P_{n+1})}{cost(Opt_2)} + \frac{C_{n+1}}{cost(Opt_2)}.\end{aligned}$$

Finally, because of inequalities (4) and (5) and because $cost(Opt_2) \ge C_{n+1}$ (see Fig. 13), we get

$$\begin{aligned}\frac{cost(Approx_2)}{cost(Opt_2)} &\le \ln \frac{C_0}{C_n} + 1 + 1 \\ &\le \ln \frac{cost(Opt_1)}{cost(Opt_2)} + 2.\end{aligned}$$

□

Together with Theorem 1, Lemma 4 implies our main result.

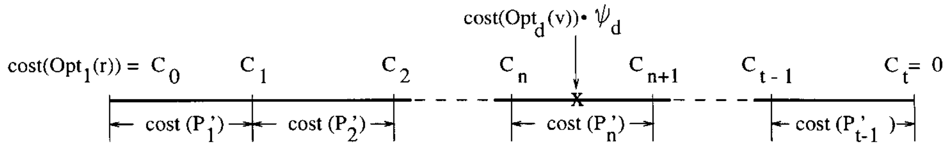


Fig. 13. The rooted Steiner 2-star heuristic (Fig. 10) iteratively modifies the problem instance by repeatedly removing groups associated with minimum-norm partial stars P_i. The cost of the optimal 1-star C_0 originally exceeds (or is equal to) the cost of the optimal 2-star $cost(Opt_2)$, but finally becomes equal to zero at step t. Therefore, there exists an n between one and t such that $cost(Opt_1)$ drops to strictly less than $cost(Opt_2)$ between steps n and $n+1$.

Theorem 2: The group Steiner heuristic (Figs. 8, 10, and 12) solves the group Steiner minimal tree problem with performance ratio $2 \cdot (2 + \ln(k/2)) \cdot \sqrt{k}$, where k is the number of groups.

Proof: Our overall group Steiner heuristic (Fig. 8) runs the rooted Steiner 2-Star heuristic (Fig. 10) for all possible roots $r \in V$. If the root is a center r_c of the optimal group Steiner tree *Opt*, then from Theorem 1, we know that $cost(Opt_1(r_c)) \leq (k/2) \cdot cost(Opt) \leq (k/2) \cdot cost(Opt_2(r_c))$, where k is the number of groups. Therefore, Lemma 4 implies that the cost of the tree produced by our main algorithm is at most $2 + \ln(k/2)$ times the cost of the optimal Steiner 2-star. Finally, using Theorem 1, we obtain a group Steiner tree that costs no more than $2 \cdot (2 + \ln(k/2)) \cdot \sqrt{k}$ times the optimum. □

IV. Runtime and Practical Optimizations

In this section, we first estimate the runtime of our heuristic, and then we propose several ways to improve its runtime and performance in practice. Let n denote the total number of ports in all of the groups, i.e. $n = |\cup_{i=1}^{k} N_i|$. Let α denote the time complexity of computing all-pairs shortest paths in the graph $G = (V, E)$. As part of our preprocessing, we shall also compute in time $O(n \cdot |V|)$ all vertex-to-group distances (i.e., distances between each vertex and the closest port in each group). The time complexity of the Minimum-Norm Partial Star Heuristic (Fig. 12) is $O(|V| \cdot k \cdot \log k)$. Therefore, the Rooted 2-star Heuristic (Fig. 10) has runtime $O(|V| \cdot k^2 \cdot \log k)$, where k is the number of groups. Thus, we obtain the following result:

Theorem 3: The total runtime of the group Steiner heuristic (Fig. 8) is $O(\alpha + |V|^2 \cdot k^2 \cdot \log k)$, where k is the number of groups, and α is the time complexity of computing all-pairs shortest paths.

The performance ratios derived in previous sections pertain to *worst-case* analysis. However, in practice we are also interested in the average-case behavior of our heuristics, in terms of both the solution quality and the runtime. One such practical improvement entails omitting the removal of the set of groups spanned by the minimum-norm 2-star (see the inner loop of the algorithm in Fig. 10). Instead, every time we accept an intermediate node, we 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. We then use this distance to sort the groups in the minimum-partial star algorithm (Fig. 12).

Another practical enhancement entails computing a *group minimum spanning tree* instead of a *group Steiner minimal tree*, that is, a minimum spanning tree for a set of nodes containing *exactly* one port from each group. It can be shown that the optimal group minimum spanning tree is at most twice as long as the optimal group Steiner minimal tree. Thus, in approximating the group Steiner minimal tree by a group minimum spanning tree, we lose only a factor of two, which does not asymptotically increase the overall bound of $2 \cdot (2 + \ln(k/2)) \cdot \sqrt{k}$, yet yields substantial savings in runtime.

We may further modify our algorithm with a post-processing step which finds the minimum spanning (or approximate Steiner) tree for the set of intermediate nodes and ports chosen by the group Steiner heuristic (Fig. 8). We may also make local (one at a time) node substitutions in groups to re-arrange the tree topology and reduce the overall cost.

Although provably good heuristics are frequently outperformed by local optimization methods, the output of the former can serve as a good starting point for local-improvement post-processing schemes. For example, it was shown that Christofides' heuristic (i.e., the best-known heuristic for traveling salesperson in graphs) also provides excellent initial traveling salesperson tours for further local rearrangements [17], [20].

V. Instances with Degenerate Groups

We now show how to more effectively handle instances of the group Steiner problem with some *degenerate* groups, i.e. groups of size one. We will see that treating degenerate groups differently will yield improvements in solution quality as well as in runtime.

The degenerate groups by themselves induce an instance of the classic Steiner problem, and such an instance can be approximated efficiently by known methods (with a constant performance ratio). Thus, to solve the SMT problem for degenerate groups, we may choose a provably good heuristic from among the numerous existing ones [4], [7], [12], [13], [16], [21]. For example, in time $O(|V|^3)$ we may find a Steiner tree which is at most 11/6 times longer than the optimal [22]. The remaining issue now is how to combine the Steiner minimal tree over the degenerate groups with a tree spanning the other, nondegenerate groups.

More formally, let $N = M_1 \cup M_2$ be a partition of all the groups in N into those containing one terminal (M_1), and those containing at least two terminals (M_2). We define the *combined group Steiner heuristic* as follows. First, we find the usual Steiner tree $Approx_1$ for the terminals M_1 using the algorithm from [22]. Next, using our group Steiner heuristic (Fig. 8), we find the group Steiner tree $Approx_2$ for the family of groups that includes M_2 and a single arbitrary group from M_1. Finally, we output a minimum spanning tree over the union $Approx_1 \cup Approx_2$ (see Fig. 14).

If the number of degenerate groups is large, then the combined group Steiner heuristic will enjoy considerable runtime

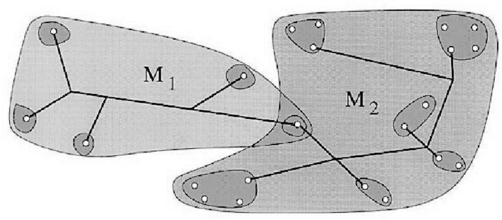


Fig. 14. We span the set of degenerate groups (M_1) with an approximate Steiner tree (on the right). Then we span all nondegenerate groups (M_2), together with an arbitrary degenerate group, with a group Steiner tree.

savings as compared to the group Steiner heuristic (Fig. 8). Moreover, the following theorem shows that this heuristic also enjoys an improved performance bound.

Theorem 4: The combined group Steiner heuristic solves the group Steiner problem with a performance ratio of at most:

$$\frac{11}{6} + 2 \cdot \left(2 + \ln \frac{k'}{2}\right) \cdot \sqrt{k'}$$

where M_2 is the set of groups of size at least two, and $k' = |M_2| + 1$.

Proof: The optimal Steiner tree for M_1, denoted by Opt_1, cannot cost more than the optimal group Steiner tree (denoted by *Opt*), i.e.

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

Moreover, if we remove from N all but one of the degenerate groups, then the cost of the optimal group Steiner tree over the remaining groups can not be more than *Opt*.

$$\begin{aligned} cost(Approx_2) &\le 2 \cdot \left(2 + \ln \frac{k'}{2}\right) \cdot \sqrt{k'} \cdot cost(Opt_2) \\ &\le 2 \cdot \left(2 + \ln \frac{k'}{2}\right) \cdot \sqrt{k'} \cdot cost(Opt) \end{aligned}$$

Therefore, the cost of the combined tree (i.e., the union of the two trees constructed above) is at most:

$$\begin{aligned} &cost(Approx_1 \cup Approx_2) \\ &\quad \le \left[\frac{11}{6} + 2 \cdot \left(2 + \ln \frac{k'}{2}\right) \cdot \sqrt{k'}\right] \cdot cost(Opt) \end{aligned}$$

□

VI. Bounded-Radius Group Steiner Trees

The objective of delay-minimization can induce wiring geometries that are substantially different from those dictated by an optimal-area objective, particularly in deep submicrometer regimes. This has motivated a number of bounded-radius[10] routing constructions [1], [2], [5], [13], [14]. Our basic group Steiner tree approach can be easily extended to a bounded-radius construction, thereby yielding routing trees with source-to-sink pathlengths bounded by a user-specified parameter.

[10]The radius of a graph is defined as the maximum pathlength of any shortest source-sink path. Note that 2-stars implicitly have a radius bound of $2 \cdot OPT$, although an MST post-processing step does not preserve this bound.

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 ≥ 1 nodes 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 node of T **do**
        Find the group $N(l) = N_i$ 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

Fig. 15. The RW heuristic for the group Steiner problem [19].

For example, we can utilize the tree produced by our main algorithm (Fig. 8) as the starting point in the bounded-radius/bounded-cost construction of [5]. For an arbitrary instance of the group Steiner problem (with k groups), this hybrid approach yields a routing tree with radius $(1 + \epsilon) \cdot \mathrm{OPT}_{\mathrm{radius}}$ and total cost $(1 + (2/\epsilon)) \cdot 2 \cdot (2 + \ln(k/2)) \cdot \sqrt{k} \cdot \mathrm{OPT}_{\mathrm{cost}}$ for *any* user-specified parameter $\epsilon > 0$.

VII. Experimental Results

We have implemented our heuristic for the group Steiner problem using the Java programming language. Our implementation can be executed on the Web at:

http://www.cs.virginia.edu/robins/groupSteiner

We compared our heuristics with the heuristic proposed by Reich and Widmayer [19], henceforth, referred to as "RW." Their 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 heuristic then repeatedly deletes such a leaf node which is incident to the longest edge among all such nodes (see Fig. 15 for a formal description of this algorithm).

Table I compares two versions of our heuristic (Fig. 8) to the RW algorithm [19] (Fig. 15). We created each group by first defining a randomly placed square area of predetermined size, and then uniformly and independently distributing nodes inside this square-shaped area. We varied the predetermined group areas among 10%, 50%, and 100% of the overall square routing region. All table numbers represent the average relative improvement over 100 trials, given as a percent improvement over the tree cost of the RW algorithm [19] (negative numbers represent disimprovements).

In the context of VLSI layout, we are primarily concerned with the *rectilinear* (or *Manhattan*) plane, where the cost of routing between two nodes (a_x, a_y) and (b_x, b_y) is defined to be $|a_x - b_x| + |a_y - b_y|$. While our implementation uses the rectilinear metric to determine the distances between ports, our algorithms are general and apply to arbitrary weighted graphs, including the rectilinear case.

The first version of the group Steiner heuristic (Fig. 8) that we implemented has the following three modifications.

1) Intermediate nodes are selected strictly from among the ports (i.e., all of the constructions benchmarked are *spanning* trees—they do not use Steiner nodes other than ports).
2) The root of the 2-star is selected from a single randomly chosen group.

TABLE I
EXPERIMENTAL RESULTS: ALL NUMBERS REPRESENT THE AVERAGE PERCENT IMPROVEMENT OVER 100 TRIALS, WITH RESPECT TO THE COST OF THE OUTPUT TREE OF THE RW ALGORITHM [19] (NEGATIVE NUMBERS REPRESENT DISIMPROVEMENTS)

Area of each group is 10% of total routing region						
number of groups	group size = 3		group size = 5		group size = 8	
	2-star	2-star +MST	2-star	2-star +MST	2-star	2-star +MST
3	4.0	4.0	5.0	5.0	7.2	7.2
5	-6.2	4.6	-2.0	5.6	-1.2	8.5
10	-23.0	6.5	-14.6	10.1	-10.5	11.0
20	-40.0	8.0	-32.2	12.3	-26.0	16.1
30	-52.2	9.8	-34.8	10.9	-36.1	18.5
Average	-23.5	6.6	-15.7	8.8	-13.3	12.3
Area of each group is 50% of total routing region						
number of groups	group size = 3		group size = 5		group size = 8	
	2-star	2-star +MST	2-star	2-star +MST	2-star	2-star +MST
3	10.2	10.2	15.2	15.2	24.7	24.7
5	4.5	10.4	16.4	21.2	23.6	27.6
10	-3.4	15.2	9.0	23.0	20.3	32.4
20	-21.9	14.5	-7.2	23.1	11.2	34.3
30	-28.8	16.7	-8.5	19.8	3.0	30.9
Average	-7.9	13.4	5.0	20.5	16.6	30.0
Area of each group is 100% of total routing region						
number of groups	group size = 3		group size = 5		group size = 8	
	2-star	2-star +MST	2-star	2-star +MST	2-star	2-star +MST
3	13.9	13.9	28.0	28.0	31.4	31.4
5	11.8	17.7	25.9	30.2	28.5	31.3
10	-1.9	14.1	16.8	29.6	26.8	36.2
20	-13.9	18.0	6.6	31.4	15.4	37.2
30	-22.2	28.8	-1.6	22.2	12.0	35.2
Average	-2.5	18.5	15.1	28.3	22.8	34.3

3) After accepting an intermediate node in the inner loop of Fig. 12, the node is removed from further consideration in subsequent iterations.

The second version that we implemented is a hybrid approach, which is our main algorithm discussed above, except the solutions are then post-processed with a minimum spanning tree algorithm (i.e., we output the minimum spanning tree over the nodes selected by the modified heuristic described above). The table column labeled "2-star" shows the average percent improvements of our modified heuristic over the RW algorithm, while data for the hybrid approach is given in the column labeled "2-star + MST." As can be seen from the table, the hybrid approach significantly outperforms the RW algorithm as the group sizes and the group areas increase.

VIII. CONCLUSION

We have addressed the problem of minimum-cost routing of multiport terminals, a direct generalization of the Steiner problem. Our main result is the first known heuristic with a *sublinear* performance bound. In particular, for a net with k multiport terminals, our construction has a performance bound of $2 \cdot (2 + \ln(k/2)) \cdot \sqrt{k} \cdot OPT$. Our implementation and benchmark results indicate that our approach is effective in practice. Future work includes further reducing the bounds, and also improving the time complexity of the algorithms.

APPENDIX
GROUP STEINER TREE APPROXIMATION COMPLEXITY

It is known that the group Steiner tree problem is not easier to approximate than the set cover problem [11]. Combining this result with the recent result of Feige [6], we obtain the following.

Theorem 5: Unless $NP \subseteq DTIME[n^{\log\log n}]$, the group Steiner tree cannot be approximated to a factor of better than $(1 - o(1)) \cdot \ln k$, where k is the number of groups.

Now we will show that even approximating depth-2 group Steiner tree with the given root r is not easier than approximating the set cover problem. On the other hand, in Section III we described a Steiner 2-star heuristic (Fig. 10) for finding a depth-2 group Steiner tree with the approximation ratio $2 + \ln(k/2)$. We, therefore, conclude that set cover and finding depth-2 group Steiner tree have the same approximation complexity.

Theorem 6: Unless $NP \subseteq DTIME[n^{\log\log n}]$, a depth-2 group Steiner tree with the given root r cannot be approximated to a factor of better than $(1 - o(1)) \ln k$ where k is the number of groups.

Proof: An instance I of the set cover problem consists of a finite set $X = \{x_1, \ldots, x_k\}$ and a family of subsets of X namely $P = \{p_1, \ldots, p_m\} \subseteq 2^X$. The set cover problem seeks a minimum-size subfamily $M \subseteq P$ that covers X, i.e. such that $X \subseteq \cup\{p_i | p_i \in M\}$. In order to prove the theorem, we will give polynomial-time L-reduction from the set cover problem to the group Steiner problem. Formally, we will construct an instance I' of the group Steiner problem (i.e. a graph G with a node r and groups of terminals $N_i, i = 1, \ldots, k$), such that

1) any subfamily $M \subseteq P$ that covers X will correspond to a Steiner 2-star M' rooted at r for I' such that $cost(M') = |M|$;
2) any Steiner 2-star M' rooted at r for I' will correspond to a subfamily $M \subseteq P$ that covers X such that $cost(M') = |M|$; and
3) given I, the corresponding I' can be found in polynomial time and the correspondence $M \leftrightarrow M'$ is also polynomial.

First, we construct a graph G for the instance I'. The node set of G consists of the pairs (i, j) such that $x_i \in p_j$ and an auxiliary pair $r = (0, 0)$. The cost of an edge between (i, j) and (i', j') equals zero if $j = j'$ and one otherwise. Given $i = 1, \ldots, k$ the group of terminals N_i consists of all pairs (i, j). Now we are ready to prove (1)–(3).

(1) Given $M \subseteq P$, in each $p_j \in M$ we fix one element $x_i \in p_j$. In the 2-star M', we connect each such node (i, j) with the root $r = (0, 0)$ and all nodes that are within distance zero from (i, j). Clearly, if M covers the entire set X, then the 2-star M' intersects all groups. Moreover, M' contains exactly $|M|$ edges of cost one (they are incident to the root r).

(2) Given a Steiner 2-star M', we build a subfamily M from the sets p_j's for which there are intermediate nodes (i, j) in the 2-star M'. Clearly if M' intersects all groups, then M covers the entire set X, and the number of sets in M is equal to the number of intermediate nodes of M' which is the cost of M'.

(3) Clearly, all reductions above can be done in polynomial time. □

ACKNOWLEDGMENT

The authors are grateful to D. Bateman for help with the Java implementation, and to A. Kahng and S. Friend for their valuable feedback and discussions.

REFERENCES

[1] C. J. Alpert, T. C. Hu, J. H. Huang, A. B. Kahng, and D. Karger, "Prim-Dijkstra tradeoffs for improved performance-driven routing tree design," *IEEE Trans. Computer-Aided Design*, vol. 14, no. 7, pp. 890–896, 1995.

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

[3] C. D. Bateman, C. S. Helvig, G. Robins, and A. Zelikovsky, "Provably-good routing tree construction with multi-port terminals," in *Proc. Int. Symp. Physical Design*, Napa Valley, CA, Apr. 1997, pp. 96–102.

[4] P. Berman and V. Ramaiyer, "Improved approximations for the Steiner tree problem," in *Proc. ACM/SIAM Symp. Discrete Algorithms*, San Francisco, CA, Jan. 1992, pp. 325–334.

[5] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh, and C. K. Wong, "Provably good performance-driven global routing," *IEEE Trans. Computer-Aided Design*, vol. 11, pp. 739–752, June 1992.

[6] U. Feige, "A threshold of ln n for approximating set cover," in *Proc. ACM Symp. Theory of Computing*, May 1996, pp. 314–318.

[7] 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*, vol. 13, pp. 1351–1365, Nov. 1994.

[8] C. S. Helvig, G. Robins, and A. Zelikovsky, "Improved approximation bounds for the group Steiner problem," in *Proc. Conf. Design Automation and Test in Europe*, Paris, France, Feb. 1998, pp. 406–413.

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

[10] E. Ihler, "Bounds on the quality of approximate solutions to the group Steiner problem," in *Lecture Notes in Computer Science*, 1991, vol. 484, pp. 109–118.

[11] ——, "The complexity of approximating the class Steiner tree problem," Institut für Informatik, Universität Freiburg, Tech. Rep., 1991.

[12] A. B. Kahng and G. Robins, "A new class of iterative Steiner tree heuristics with good performance," *IEEE Trans. Computer-Aided Design*, vol. 11, pp. 893–902, July 1992.

[13] ——, *On Optimal Interconnections for VLSI*. Boston, MA: Kluwer Academic, 1995.

[14] S. Khuller, B. Raghavachari, and N. Young, "Balancing minimum spanning and shortest path trees," in *Proc. ACM/SIAM Symp. Discrete Algorithms*, Jan. 1993, pp. 243–250.

[15] N. L. Koren, "Pin assignment in automated printed circuit board design," in *Proc. Design Automation Workshop*, June 1972, pp. 72–79.

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

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

[18] L. E. Liu and C. Sechen, "Multi-layer pin assignment for macro cell circuits," in *Proc. ACM/SIGDA Physical Design Workshop*, Reston, VA, Apr. 1996, pp. 249–255.

[19] G. Reich and P. Widmayer, "Beyond Steiner's problem: A VLSI oriented generalization," in *Lecture Notes in Computer Science*, 1989, vol. 411, pp. 196–211.

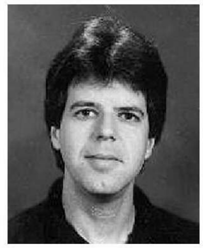
[20] G. Reinelt, *The Traveling Salesman: Computational Solutions for TSP Applications*. Berlin, Germany: Springer-Verlag, 1994.

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

[22] ——, "A faster approximation algorithm for the Steiner tree problem in graphs," *Inform. Processing Lett.*, vol. 46, no. 2, pp. 79–83, 1993.

Christopher S. Helvig received the B.S. degree with honors from the University of North Carolina, Chapel Hill, in 1996, and the M.S. degree from the Department of Computer Science at the University of Virginia, Charlottesville, in 1998.

He is currently a Software Engineer with Volition, Inc., Urbana, IL. He co-authored several papers in the areas of VLSI CAD, computer vision, and computational geometry. He is also interested in computer game development.



Gabriel Robins received the Ph.D. degree in computer science from the University of California, Los Angeles, in 1992.

He is an Associate Professor in the Department of Computer Science at the University of Virginia, Charlottesville. His primary area of research is VLSI CAD, with emphasis on physical design. He co-authored a book on high-performance routing as well as over 70 refereed papers, including a Distinguished Paper at the 1990 IEEE International Conference on Computer-Aided Design.

Dr. Robins received a Packard Foundation Fellowship, a National Science Foundation Young Investigator Award, a University Teaching Fellowship, an All-University Outstanding Teaching Award, a Faculty Mentor Award, and the Walter N. Munster endowed chair. He also received an IBM Fellowship and a Distinguished Teaching Award at the University of California at Los Angeles. He is a member of the U.S. Army Science Board, and an alumni of the Defense Science Study Group, an advisory panel to the U.S. Department of Defense. He also served on panels of the National Academy of Sciences and the National Science Foundation. He was General Chair of the 1996 ACM/SIGDA Physical Design Workshop, and a co-founder of the International Symposium on Physical Design. He also serves on the technical program committees of several other leading conferences and on the Editorial Board of the IEEE Book Series. He is a member of ACM, SIAM, MAA, SIGDA and SIGACT.

Alexander Zelikovsky received the A.B. and M.S. degree in mathematics from Kishinev University, Moldova. In April 1989, he received the Ph.D. degree in computer science from the Institute of Mathematics of the Belorussian Academy of Sciences, Minsk, Belarus. From 1982 to October 1985, he was with the Institute of Mathematics of Moldova Academy of Sciences in Kishinev, where he worked in system programming. He was a Senior Research Scholar with the Institute of Mathematics in Kishinev until September 1995. Dr. Zelikovsky was awarded the Young Investigator award of the Moldova Academy of Sciences and a Humboldt Fellowship (Germany). In 1992–1995, he participated in Volkswagen Stiftung project and was on leave of absence from Institute of Mathematics as Visiting Researcher at the Computer Science Department of Bonn University in Germany, and at Institut fur Informatik in Saarbrueken, Germany. During 1995–1997, Dr. Zelikovsky was a Research Scientist at the University of Virginia, Charlottesville, and in 1997–1998 he was a Postdoctoral Scholar at University of California, Los Angeles. Since January 1999, he has been an Assistant Professor at the Computer Science Department of Georgia State University, Atlanta. He has authored more than 40 refereed publications. His research interests include VLSI physical layout design and performance analysis, approximation algorithms, combinatorial optimization, and computational geometry.