Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

# ON A NEW CLASS OF ITERATIVE STEINER TREE HEURISTICS WITH GOOD PERFORMANCE

Andrew Kahng
Gabriel Robins

May 1990
CSD-900014

# On A New Class of Iterative Steiner Tree Heuristics with Good Performance

*Andrew Kahng and Gabriel Robins*

Department of Computer Science
UCLA
Los Angeles, CA 90024

*ABSTRACT*

Virtually all previous methods for the rectilinear Steiner tree problem begin with a minimum spanning tree *topology* and rearrange edges to *induce* Steiner points. This paper gives a more direct approach which makes a significant departure from such spanning tree based strategies: we iteratively find *optimal* Steiner points to be added to the layout. Our method not only gives improved average-case performance, but also escapes the worst-case examples of existing approaches. Sophisticated computational geometry techniques allow efficient and practical implementation, and the method is naturally suited to real-world VLSI regimes where, e.g., via costs can be high. Extensive performance results show almost 3% wirelength reduction over the best existing methods. We describe a number of variants and extensions, and also suggest directions for further research.

## On A New Class of Iterative Steiner Tree Heuristics with Good Performance

*Andrew Kahng and Gabriel Robins*

Department of Computer Science
UCLA
Los Angeles, CA 90024

### 1. Introduction.

The minimum rectilinear Steiner tree (MRST) problem is as follows: given N points in the plane, find a minimum-length tree of rectilinear edges which connects the points. This problem has been extensively studied, and important applications arise in such circuit design phases as wirability analysis and global routing. The problem is NP-complete [9], and a number of heuristics have been proposed [20] which resemble classic minimum spanning tree (MST) construction methods. Hwang [15] showed that the rectilinear MST itself is an approximation to the MRST (see Figure 1) with worst-case ratio

$$\frac{\text{length(MST)}}{\text{length(MRST)}} \leq \frac{3}{2},$$

and a fundamental open question over the years has been whether there exists an MRST heuristic with worst-case performance ratio less than 3/2.

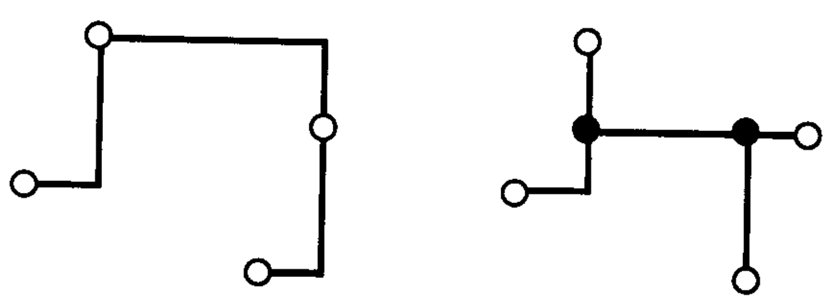

**Figure 1:** MST and MRST for the same 4-point set.

The MRST heuristics proposed thus far have very similar performance on random instances (i.e., average heuristic RST length being 7-9% smaller than MST length), and have tight worst-case bounds of 3/2, the same as for the simple rectilinear MST. This effective similarity has been noted in [20] and [27], and is confirmed by our own empirical studies. The three most recent MRST methods are due to Ho, Vijayan and Wong [14], Bern [3], and Hasan, Vijayan, and Wong [13]; the first gives an elegant construction of the *optimal* rectilinear Steiner tree (RST) that is derivable from a minimum spanning tree, the second (also discussed in [20]) is an analog of Kruskal's spanning tree algorithm, while the third at each iteration adds as many "locally-independent" (with respect to the MST) Steiner points as possible. All of these methods have recently joined the list of methods which have the worst-possible worst-case performance bound 3/2 (i.e., as large as that of the simple MST [17] (see figures 2 and 3), and it seems unlikely that any MST-inspired heuristic variant will have a performance ratio less than 3/2. Steiner routing for VLSI is surveyed in [16], [20] and [27]; the latter paper also discusses the more general problem of Steiner routing in networks (as opposed to the VLSI problem, which is embeddable in a metric space).

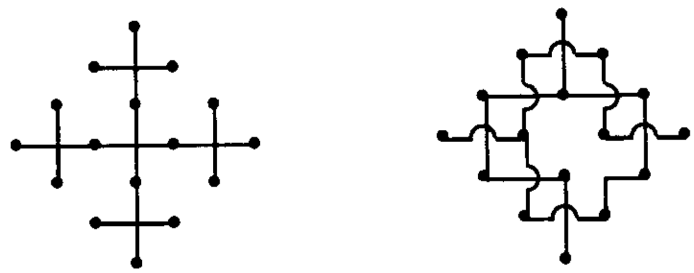

**Figure 2:** An example where the cost ratio MST-derived-RST/MRST is equal to 3/2. On the left is the optimal MRST (cost 20); any Steiner tree derived from the MST on the right has cost 30.

**Figure 3:** An example where the cost ratio of a *separable* (in the terminology of [14]) MST-derived-RST to the MRST is arbitrarily close to 3/2. The MRST on the left has cost (4/3)(N-1), while any RST derivable from the MST on the right has cost 2(N-2).

In retrospect, it is natural that the MST has been used as a starting point for solving the MRST problem. A body of theoretical work on subadditive functionals in the $L_p$ plane (such functionals include the MST, the MRST, and the optimal traveling salesman problem (TSP) tour) indicates that optimal solutions to random N-point instances of these problems have expected length $\beta\sqrt{N}$, where the constant $\beta$ depends on both the problem, e.g., TSP versus MST, and the $L_p$ norm. (The $L_p$ distance function in the plane is given by $\Delta = \sqrt[p]{(\Delta x)^p + (\Delta y)^p}$, i.e., $p = 1$, $p = 2$ and $p = \infty$ define the Manhattan, Euclidean and Chebyshev norms, respectively.) A classic result, that the minimum spanning tree can be used to find a provably good heuristic TSP solution, lies in this vein.

Recent surveys of results on subadditive functionals are contained in [2][25], and there are many implications for practical optimization. For example, we note that many VLSI global routers (e.g., TimberWolfSC) use bounding-box MRST estimates for computational simplicity. The growth function above immediately implies that such estimates can be refined by using an $O(\sqrt{N})$ scaling factor, with negligible CPU cost.

Although the optimal MST and MRST may have "similar" growth rates, an MST-derived solution method may not be appropriate for VLSI routing applications. It has been shown [4][11] that the optimal Steiner tree, as well as heuristic MST-based RSTs, will have a linear number of Steiner points. With preferred-direction wiring planes, each Steiner point therefore corresponds to an additional via. In certain (board) wiring technologies, or for chip reliability considerations, having so many Steiner points may not be desirable. Ideally, we would like to *prescribe* the relative incidence of Steiner points as a routing parameter which might depend on technology or estimated layout congestion, but this is not a natural concept when we use an MST-based method.

When we consider the extreme case where extra vias are very expensive, it is natural to ask the following: if we are allowed to introduce *exactly one Steiner point* into a net, where should it be placed? This is the motivation for our *Iterated 1-Steiner* heuristic, which repeatedly finds the best possible Steiner point and adds it to the pointset until no further improvement is possible. The advantages of this method are several:

- we can limit the algorithm so that it introduces only k Steiner points (e.g., in a layout regime where vias are expensive), and we will end up with a very good approximation to the optimal k-Steiner point solution;
- the method can be efficiently implemented by applying elegant computational geometry results, including those of Georgakopoulos and Papadimitriou [10];
- the performance of the method is significantly better than all previous MST-based methods, yielding an average improvement of 10 to 11 percent over MST lengths; and
- the method is amenable to a number of extensions: randomization, partial amortization of computations, parallel implementation, and applications to higher-dimensional or alternate-metric geometries.

In the following section, we review several important attributes of Steiner trees before developing the new method in detail. Section 2 also gives the formal statement of our algorithm and several variants. Section 3 presents a theoretical analysis of the method and a large body of empirical results. The paper concludes by listing directions for further research.

## 2. A New Approach.

### 2.1. Steiner Tree Attributes.

*Definition*: Given a set of P of points $\{p_1, \ldots, p_N\}$ in the plane, the *1-Steiner point* is the point x such that the length of MST(P ∪ {x}) is minimized, and MST(P ∪ {x}) < MST(P). (Where unambiguous, we use MST(P) to denote the length of the MST on point set P.) The *1-Steiner tree* is the minimum spanning tree on P ∪ {x}.

It is useful to view a Steiner tree as an MST on the union of P, the original point set, and S, a set of Steiner points. Our approach is to iteratively calculate optimum 1-Steiner points and add them to S. With each added point, the length of the MST on P ∪ S will decrease. If there is no x such that MST(P ∪ S ∪ {x}) < MST(P ∪ S), we terminate the construction.

We may begin bounding the complexity of the 1-Steiner tree computation by using the following facts.

*Lemma 1*: Every node in the minimum spanning tree on a point set in the $L_1$ plane has degree less than or equal to 8. □

This follows from noting that a point cannot have two neighbors in the MST which both lie in a single octant of the plane. (Simple perturbative arguments can slightly reduce the constant 8 if desired.) It is also easy to show the following:

*Lemma 2*: The optimal 1-Steiner point for $k \leq 8$ points in the plane can be computed in constant time. □

We can therefore determine the 1-Steiner point for N points by examining all C(N,8) combinations of eight points, determining the 1-Steiner point in O(1) time for each of these sets and then taking the best MST length. Thus, the naive algorithm for finding the optimal 1-Steiner point requires $O(N^8)$ minimum spanning tree computations.

For a net with N pin locations in the grid, define the *Steiner candidate set* to consist of all points whose x and y coordinates are both shared with two of the N pins. A result of Hanan [12] states that an optimal MRST exists whose Steiner points are taken from this set. This implies that we may find the 1-Steiner tree by examining $N^2$ Steiner candidates, constructing a new MST for each set of N + 1 points, then picking the best candidate. This solves the 1-Steiner problem with $O(N^2)$ MST computations, each of which (by sorting edges) takes at most $O(N^2 \log N)$ time, yielding an $O(N^4 \log N)$ time bound. Note that this is the time required to find just one 1-

Steiner point, and that we will surely have to add more than one point to find a good Steiner tree. The following theorem, which generalizes a classic result of Gilbert and Pollak [11] and is of independent interest, addresses this question with an upper bound on the number of Steiner points in the MRST. We show that the MRST will always have at most a *linear* number of points, and that this is an artifact of the tree topology, not the geometry of the metric. Proofs and further discussion of this result are contained in Appendix I.

*Theorem 1*: In any finite two-colored tree such that the degree of every red node must be d or more, the ratio of blue to red nodes is at least $d - 2$. □

Because any Steiner point must have degree at least three, the following corollary implies that there are at most N Steiner points in the MRST on N points.

*Corollary*: A set of points P has cardinality greater than the number of Steiner points in the MRST on P. □

In Appendix I, we show that this upper bound can be refined to $N - 2$ Steiner points; we also give an additional theorem on MRST decomposition.

Theorem 1 and its corollary justify an MRST heuristic which iteratively selects the optimum 1-Steiner point on $P \cup S$ and adds it to S. When there are a linear number of Steiner points in the final solution, the method has an obvious $O(N^5 \log N)$ time bound. To make the approach practical, we only require more efficient methods for finding the 1-Steiner point. By using the Voronoi diagram [19], the MST computation can be done in $O(N \log N)$ time, which reduces the time per added Steiner point from $O(N^4 \log N)$ to $O(N^3 \log N)$. Our main result, described below, gives a method for adding the optimal new Steiner point in $O(N^2)$ time. A linear number of Steiner points can thus be found with $O(N^3)$ effort, and finding heuristic solutions with $\leq k$ Steiner points (k a constant), in a regime with high via costs, can be accomplished in $O(N^2)$ time. By recent results of Eppstein et al. [8], it seems unlikely that this time bound can be improved.

Extensions to this "Iterated 1-Steiner" approach abound. We have examined randomized variants, several complex (i.e., more global) criteria for picking 1-Steiner points, etc. A very useful extension involves amortizing the computation needed to determine a single 1-Steiner point: we determine an entire set of "independent" Steiner points, all of which can be added to the layout in a single round. In practice, only a very small number of rounds (less than three) is required by this "batched" approach. Before we discuss these and other versions of the iterative MRST approach, we review the $O(N^2)$ method for finding the 1-Steiner point.

### 2.2. A Result of Georgakopoulos and Papadimitriou.

Georgakopoulos and Papadimitriou [10] give an $O(N^2)$ method for computing the 1-Steiner *tree* N points in the Euclidean plane. We use an adaptation of their method for the Manhattan norm. The idea is summarized as follows (to make the discussion self-contained, we give a more complete synthesis of [10] in Appendix II below):

- By Lemma 1, a point p cannot have two neighbors in the MST which lie in the same octant of the plane with respect to p. We can fix eight "orientations" at 45-degree intervals, each of which induces a Voronoi-like partition (the *oriented Dirichlet cell*) of the plane.
- In $O(N^2)$ time, these eight partitions can be overlaid into a "coarsest common partition" which has $O(N^2)$ regions (Theorem 4 in Appendix II). The regions of this partition are *iso-dendral*: introducing any point from within a given region will result in a constant MST *topology*.
- The minimum spanning tree on the N points is constructed, and we perform preprocessing in $O(N^2)$ time such that whenever a new point is added to the point set, updating the MST to include the new point can be done in *constant* time (Theorem 5 in Appendix II).
- We then go through the $O(N^2)$ regions of the overlaid partitions and determine, in constant time per region, the optimal Steiner point in each region. Each such point will induce an MST on N + 1 points that can be computed in constant time using the information obtained from the preprocessing. Comparing the costs of these trees and selecting the smallest one will give the minimum-length MST on N + 1 points. The total time for all phases is $O(N^2)$.

### 2.3. The Algorithm.

We now state the **Iterated 1-Steiner** heuristic:

> **while** the number of Steiner points is less than N and a 1-Steiner point x exists,
> add x to the point set.

There are at most N iterations, each requiring $O(N^2)$ computation, and therefore the time complexity of this method is $O(N^3)$. Empirical results in Section 3 show that Iterated 1-Steiner significantly outperforms all existing heuristics (see Table 1). Empirically it is observed that the actual number of iterations the algorithm performs for random pointsets is nowhere near N, and in fact on the average is less than N/2.

An important variant of this heuristic is motivated by observing that it may not be necessary to search for the best candidate Steiner points to be added at each iteration. In particular, the quality of the final tree might be acceptable even if each step simply chooses a *random* improving point. Both this method and the original heuristic may be improved by removing Steiner points which become degree-1 or degree-2 points in subsequent MSTs; by the triangle inequality the latter can be removed without increasing the MST length, and the former can trivially be removed from the layout. The advantage of this refinement is that the variant performs no worse than the original algorithm and by Theorem 1 above will produce a layout with at most $N-2$ Steiner points. The **Iterated Random 1-Steiner** heuristic is:

> **while** 1-Steiner points exist
>
> add a randomly selected 1-Steiner point to the point set.
>
> if a Steiner point x is of degree 1 or 2 in the MST, delete x.

Iterated Random 1-Steiner lends itself well to a simple, compact implementation. Performance is worse than Iterated 1-Steiner, but remains slightly better than MST-derived solutions for typical instances. Iterated Random 1-Steiner will clearly terminate because the MST length decreases monotonically, but we can show that the final cost can take on any one of an exponential number of distinct values for certain instances). A simple random model suggests that there may be an $O(N \log N)$ expected upper bound on the number of iterations. However, we do not have a polynomial worst-case complexity bound. A variant which requires that a point cannot return to the layout after it has been deleted will have a trivial $O(N^2)$ bound on the number of iterations, and we can construct a family of instances for which the randomized method actually produces this quadratic number of Steiner points.

In surveying the vast Steiner tree literature, we found that the Iterated 1-Steiner heuristic and its variants have as their closest conceptual relative a method of Smith and Liebman [22][23] which involves a highly ad hoc examination of a linear-size subset of the candidate Steiner set. Our method is preferable on several grounds: (i) *performance*: the method in [22] gives less than 8% average improvement over MST length for random point sets [16] and thus seems to fall in with the other methods in the literature, while our method gives almost 11% average improvement; (ii) *efficiency*: [22] gives an $O(N^4)$ method, while the Iterated 1-Steiner algorithm is $O(N^3)$; (iii) *simplicity*: the algorithm in [22] requires seven pages to describe while our method is simply described and coded.

**2.4. Reducing the Number of Iterations.**

It is possible to consider combinations of k Steiner candidates at each step, with corresponding improvements in performance. Although the number of iterations is reduced, the overall time complexity grows as $O(N^{2k+1})$ and so the method is not practical for large N. However, the approach gives insight into a "k-opt" local-search formulation for the Steiner problem. We note that Sarrafzadeh and Wong [21] have independently considered exactly such a k-opt criterion, but within the usual MST-derived solution framework.

In order to reduce the running time of our heuristic, one is tempted to consider salvaging intermediate computational results of the algorithm [10] from one iteration to the next. Unfortunately, it turns out that even the addition of a single Steiner point to a pointset can modify each of the $O(N^2)$ oriented Dirichlet cells induced by that pointset (see Figure 4). Moreover, an arbitrarily large sequence of points may be added, each of which causes a quadratic number of isodendral regions to change. This suggests that "on-line" maintenance of an oriented Dirichlet cell partition for a pointset would require up to $O(N^2)$ time per new point added (in both the worst-case and amortized analyses), and is thus no cheaper than computing the entire new partition from scratch.

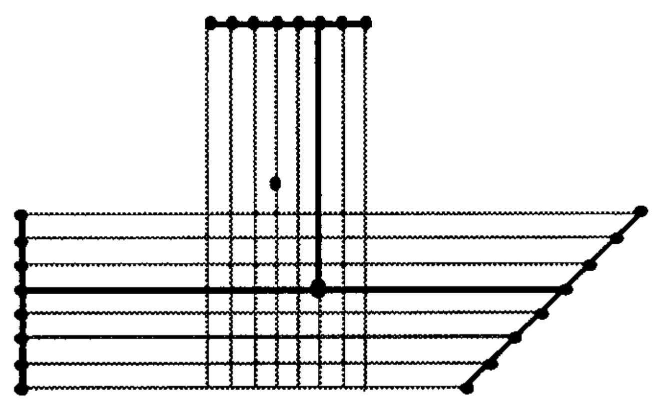

**Figure 4:** An example where a pointset gives rise to a quadratic number of isodendral regions, each of which induces a distinct MRST (one of these is highlighted). The addition of the new point shown will change each and every one of the quadratic number of associated MRST topologies; this can be repeated arbitrarily many times by introducing a long sequence of points, each one slightly underneath the preceding one.

As noted earlier, a promising variant amortizes some of the computational expense, as follows: we use the approach of [10] to compute the optimal 1-Steiner point and its associated MST cost savings *within each isodendral region*; instead of selecting only the Steiner candidate with highest cost savings, we select a maximal "independent" set of SPs, similar to the approach of [13]. The criterion for "independence" here is that no candidate SP is allowed to "interfere" with, i.e., reduce, the MST cost savings of any other candidate SP in the added set. In particular, for a set of points P, candidate Steiner points x and y may be added in the same round only if

$$\Delta MST(P,\{x\}) + \Delta MST(P,\{y\}) \leq \Delta MST(P,\{x,y\}),$$

where $\Delta MST(P,S) = \max(0, MST(P) - MST(P \cup S))$ For a candidate SP x we assume that $\Delta MST(P,\{x\}) > 0$. A round of this method is formally described as follows:

- Compute the MST over P in $O(N \cdot \log N)$ time using a Voronoi diagram-based method [19]. Also construct the weighted undirected graph G=(P,E) where E = {(x,y) | (x,y) is an edge in the MST over P} and the weight of each edge in G is the rectilinear distance between its two endpoints.
- Compute the $O(N^2)$ isodendral regions over P and the associated O(1) potential MST neighboring points for each, as outlined in Appendix II and [10]. This requires $O(N^2)$ time.
- Preprocess the $O(N^2)$ isodendral regions, now treated as a planar subdivision, so that future planar subdivision searches (i.e., determining which planar region a given point lies lies in) may be performed in $O(\log N)$ time [19]. This preprocessing requires $O(N^2 \log N)$ time, using $O(N^2 \log N)$ space.
- For each candidate SP x, compute the cost savings $\Delta MST(P,\{x\})$ associated with x. We determine the isodendral region R to which x belongs in $O(\log N)$ time by virtue of the planar subdivision search preprocessing done earlier, and then for each point y among the O(1) potential MST neighbors of x, we add the weighted edge e = (x,y) to the graph G (recall that the weight of e is the rectilinear distance between x and y). The MST of a planar weighted graph can be maintained using $O(\log N)$ time per addition/insertion of a point/edge [8]. Thus, we can determine in $O(\log N)$ time the MST cost savings for each candidate SP; since by Hanan's theorem there are at most $N^2$ candidate SPs, the time for this entire phase is $O(N^2 \log N)$.
- Next, sort the $O(N^2)$ Hanan SP candidates in order of decreasing MST cost saving; this requires $O(N^2 \log N)$ time using any reasonable sorting algorithm (e.g., Mergesort).
- Determine a maximal set of "independent" candidate SPs to be added during this round, by successively adding one candidate at a time (in order of decreasing MST cost saving), as long as the latest SP is "independent" of all SPs previously added during the round. In other words, for an original pointset P, a set of already added SPs S, and a new candidate SP x, add x to S if and only if $\Delta MST(P,\{x\}) \leq \Delta MST(P \cup S,\{x\})$ Again, MST cost saving differences due to the addition/deletion of a single point can be determined in time $O(\log N)$ [8], bringing the total total time for this entire step to $O(N^2 \log N)$.

- Iterate with $P = P \cup S$, until we reach a round which fails to add at least one Steiner point to P.

The total time required for the entire round is $O(N^2 \log N)$. Given a pointset P, the **Batched 1-Steiner** algorithm is summarized as follows:

```
while there exists a set S={ x | ΔMST(P,{x}) > 0 } ≠ ∅ do
  P′=P
   for x in {S sorted by descending ΔMST} do
       if ΔMST(P,{x}) ≤ ΔMST(P′,{x}) then P′ = P′ ∪ {x}
  P=P′
```

Empirical data indicates that the number of rounds required grows much more slowly than the number of Steiner points produced. For example, for pointsets of size 40, where the average number of SPs produced is about 17 (with a max of 22), the average number of rounds for Batched 1-Steiner is only 2.05 (with a max of 4). We conjecture that the number of rounds grows only sub-linearly as a function of $|P|$.

## 3. Analysis of the Approach and Computational Results.

### 3.1. Performance Ratio.

*Theorem 3*: The 1-Steiner heuristic is optimal for four or fewer points.

*Proof*: For three points, there can be at most one Steiner point, and since our heuristic looks at all candidates, it is optimal. For a set of four points, the MRST can have zero, one or two Steiner points, and our method is trivially optimal when this number is less than two. When the MRST has two Steiner points, it must have one of the two topologies shown in Figure 5 [15]. A simple case analysis can show that our heuristic selects both Steiner points; the selection order is irrelevant because after either is chosen, the other must follow. □

We have found a 9-point example where the 1-Steiner heuristic performs as badly as 13/11 times optimal (Figure 6a). Even after considerable effort (and some exhaustive search) we have not found any instance for which Iterated 1-Steiner has a worse performance ratio than 13/11.

It is encouraging to note that for every other MRST heuristic in the literature, there are five- or six-point examples which *force* a performance ratio of 3/2; in contrast, the worst-case performance ratio of Iterated 1-Steiner for a five-point example seems to be 7/6 (Figure 6b). We conjecture that the Iterated 1-Steiner method has a performance ratio strictly less than 3/2, and that this ratio could in fact be 13/11.

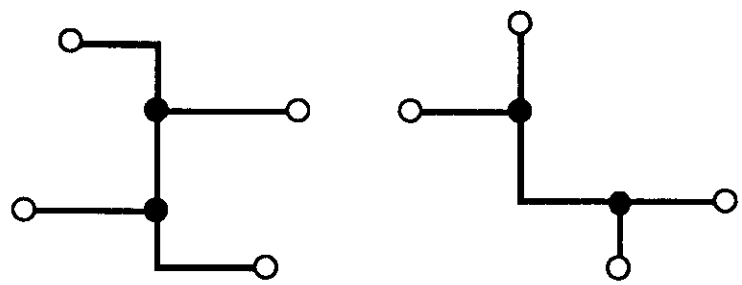

**Figure 5:** The two possible Steiner tree topologies on 4 points [15].

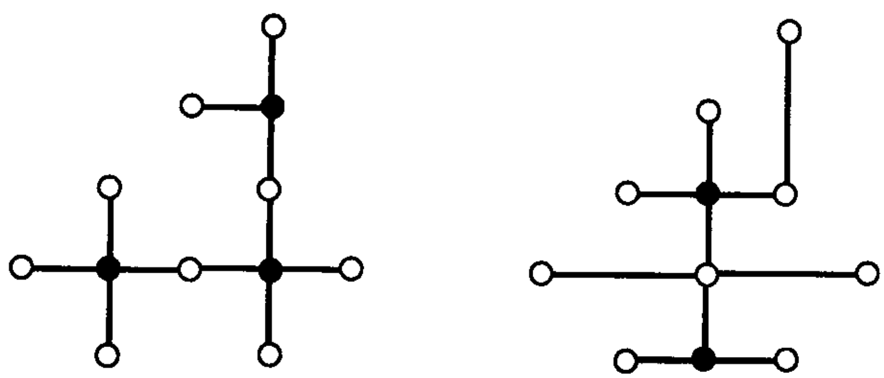

**Figure 6a:** A 9-point example where the Iterated 1-Steiner performance ratio is 13/11; the optimal MRST at left has cost 13 units, while the (possible) heuristic output at right has cost 11 units.

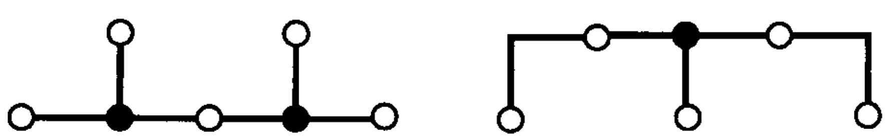

**Figure 6b:** A 5-point example where the Iterated 1-Steiner performance ratio is 7/6. The optimal MRST has cost 6, while a possible heuristic output has cost 7.

### 3.2. On Meta-Heuristics.

For a number of combinatorial problems, the following concept of a *meta-heuristic* is natural. Given an instance of problem P and $n$ different heuristics (algorithms) $H_1, H_2, \ldots, H_n$, we define a meta-heuristic $H(H_1, \ldots, H_n)$ as follows: using $n$ separate processors, execute $H_i$ on the $i^{th}$ processor. Select the best among the $n$ outputs and let it be the output of $H$. If the $H_i$ are algorithms, we can stop all computations as soon as one processor finishes and define the runtime of $H$ analogously. If we run all of the component algorithms in parallel, the total parallel running time would be the maximum of all of the running times of the component algorithms. When there is only one processor available, we can run the heuristics in sequence using $O(\sum_i H_i)$ time; this is asymptotically dominated by the slowest $H_i$.

Intuitively, several methods can trade off in their "areas of expertise", so while the meta-heuristic is of the same time complexity as the slowest component heuristic, the approximation performance is better than the best performance of any single method. In other words, each heuristic has a different set of "bad" instances, and because the set of bad instances for the meta-heuristic is the intersection of these sets, a "malicious oracle" will find it more difficult to contrive examples that will force all of the heuristics to perform badly. A scheme of combining different algorithms to yield a composite algorithm with improved characteristics is sometimes used in parallel algorithm design to reduce the running time complexity of certain algorithms [6], but we are instead advocating the joining of algorithms to improve the average *solution performance.*

To illustrate this phenomenon, we give computational results from implementations of *Corner* (from "corner-flipping"; this method gives results similar to the method of [14]) and *Prim*, a simple analog of Prim's MST heuristic. Table 1a shows that Corner and Prim, when used together, give an average performance of about half a percent better than Corner alone, although the *average* performance of Prim is about two percent worse than that of Corner.

| | | Corner | Prim | Meta |
|---|---|---|---|---|
| # Sets | # Pts | Ave Perf. | Ave Perf. | Ave Perf. |
| 4250 | 5 | 8.022 | 6.162 | 8.580 |
| 4000 | 10 | 8.155 | 6.455 | 8.584 |
| 1000 | 15 | 8.352 | 6.548 | 8.613 |
| 1000 | 20 | 8.240 | 6.392 | 8.424 |

**Table 1a:** Meta(Corner, Prim) outperforms its component heuristics.

In contrast, the meta-heuristic H(Prim, Corner, 1-Steiner) gives essentially the same performance as 1-Steiner alone, implying that 1-Steiner *strictly dominates* the other methods (Table 1b). This is a very important aspect: it suggests that the 1-Steiner method will *universally* give "reasonably good" solutions.

The meta-heuristic is a general algorithmic phenomenon that applies to numerous other problems and subareas of computer science. There is very little evidence in the literature to indicate that this phenomenon, especially for *heuristics*, has received the attention it deserves. Particularly in light of advances in parallel computation and hardware implementation of algorithms, such composite methods should become a highly fertile avenue of research in (practical) optimization.

| | | Corner | Prim | 1-Steiner | Meta |
|---|---|---|---|---|---|
| # Sets | # Pts | Ave Perf. | Ave Perf. | Ave Perf. | Ave Perf. |
| 2000 | 10 | 8.18 | 6.54 | 10.23 | 10.26 |
| 1000 | 12 | 8.16 | 6.30 | 10.25 | 10.28 |
| 500 | 15 | 8.19 | 6.54 | 10.33 | 10.35 |
| 900 | 17 | 8.16 | 6.43 | 10.38 | 10.39 |
| 500 | 18 | 8.25 | 6.48 | 10.51 | 10.52 |
| 250 | 22 | 8.29 | 6.49 | 10.45 | 10.46 |
| 1000 | 25 | 8.38 | 6.53 | 10.65 | 10.66 |
| 50 | 32 | 7.90 | 6.05 | 10.34 | 10.34 |
| 50 | 38 | 8.25 | 6.49 | 10.67 | 10.67 |

**Table 1b:** 1-Steiner dominates both Corner and Prim.

### 3.3. The Number of Steiner Points.

Trivially, there cannot be more than $O(N^2)$ Steiner points produced by our algorithm, since there are only $N^2$ candidate points. In practice, our method requires far fewer Steiner points. Figure 7 shows a four-point example for which three Steiner points are produced by Iterated 1-Steiner, and this example generalizes so that $N - 1$ Steiner points can be produced for a point set of size N.

Thus, our method can generate more Steiner points than would exist in the optimal MRST, although we can easily enforce the $N - 2$ bound by removing degree-2 and degree-1 points as we did in Iterated Random 1-Steiner. (This refinement will also improve the quality of Iterated 1-

Steiner output; results reach 11% better than MST length [18].) A linear bound on the number of *iterations* could follow from examining the manner in which degree-4 Steiner points can "split" as further points are added to the layout.

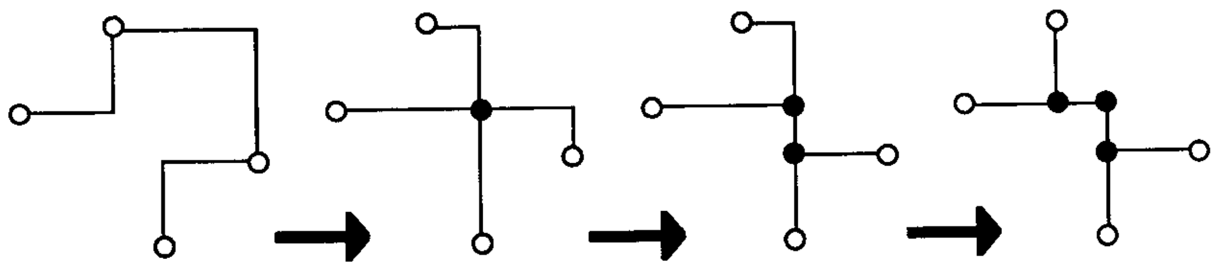

**Figure 7:** An example where Iterated 1-Steiner produces more than N-2 Steiner points: the cost of the MST over the pointset (0,3),(3,0),2,5),(5,2) is reduced from 14 to 12, to 11, and finally to 10 units.

As seen from the results below, the average number of Steiner points grows as approximately N/3, and it is interesting to note that most of the "win" (as a percentage improvement over the MST length) occurs in the first several 1-Steiner iterations. Because of this, it seems reasonable for a layout system to use our method for "k-Steiner point routing"; this will be accomplished in $O(N^2)$ time and the parameter k can reflect via costs, routing congestion, and other design/technology attributes. Similar arguments can be made for a k-round implementation of the Batched 1-Steiner variant, which will take $O(N^2 \log N)$ time. Here the results are dramatic: for 40-point instances, about 95% of the total improvement comes in the *first* round, and over 99% of the improvement comes in the first two rounds of Batched 1-Steiner. Sample results showing this incremental improvement are in Table 3 below.

### 3.4. Computational Results.

We coded the Iterated 1-Steiner Heuristic, the Iterated Random 1-Steiner Heuristic, the Batched 1-Steiner Heuristic, and several existing methods using ANSI C in both Sun-4/UNIX and Apple Macintosh environments (see Figure 8). The code is available from the authors upon request.

Extensive performance comparisons contrasted Iterated 1-Steiner and Random 1-Steiner with the standard Comer and Prim methods described above. For typical values of N, 5000 N-point instances were solved using all methods. The instances were generated randomly from a uniform distribution in a fixed-size grid; we have found that such instances are statistically indistinguishable from the pin locations of actual VLSI layouts, and they are in fact the standard testbed for Steiner tree heuristics [20]. The results are summarized in Table 2, and are depicted graphically in Figure 9. Table 3 gives results demonstrating that Batched 1-Steiner is as effective as Iterated 1-Steiner; the table also shows that even when restricted to a k-point or k-round solution, either method still performs well, with a large portion of the win occurring in the early rounds/iterations.

Table 4 gives preliminary results for 3-dimensional point sets, for which the MRST and MST length functionals grow as $O(N^{2/3})$. To facilitate future research efforts, we also provide tabulations of MST degree statistics in Table 5a (for 2 dimensions) and Table 5b (for 3 dimensions). These are useful not only in estimating the performance of the existing MST-based approaches of [13][14], etc., but also in assessing the practicality of the Georgakopoulos and Papadimitriou type approach. Recall that runtime grows with the degree of a point in the MST: because the average degree in the MST is so low, our method is much faster than the worst-case analysis would indicate.

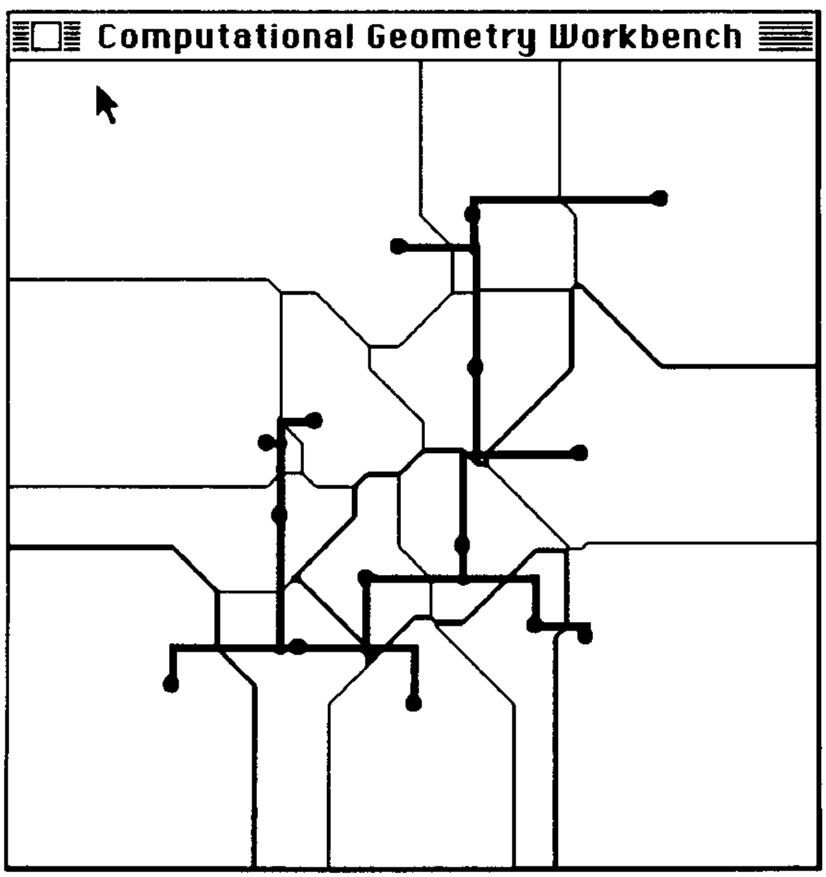


**Figure 8:** An example of Iterated 1-Steiner output for 15 points; the background shows the rectilinear Voronoi diagram for the same pointset.

# Table 2: Steiner Tree Heuristic Statistics

| | | Corner Flipping | | | Prim | | | Iterated 1-Steiner | | | | | | Iterated Random 1-Steiner | | | | | | Meta Heuristic | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # Sets | # Pts | Min Perf. | Ave Perf. | Max Perf. | Min Perf. | Ave Perf. | Max Perf. | Min Perf. | Ave Perf. | Max Perf. | Min #SP | Ave #SP | Max #SP | Min Perf. | Ave Perf. | Max Perf. | Min #SP | Ave #SP | Max #SP | Min Perf. | Ave Perf. | Max Perf. | Min #SP | Ave #SP | Max #SP |
| 5000 | 3 | 0.00 | 6.60 | 24.93 | 0.00 | 4.10 | 24.93 | 0.00 | 6.94 | 24.93 | 0 | 0.66 | 1 | 0.00 | 6.94 | 24.93 | 0 | 0.66 | 1 | 0.00 | 6.94 | 24.93 | 0 | 0.66 | 1 |
| 5000 | 4 | 0.00 | 7.85 | 29.11 | 0.00 | 5.71 | 29.11 | 0.00 | 8.73 | 29.11 | 0 | 1.09 | 3 | 0.00 | 8.70 | 29.11 | 0 | 1.20 | 3 | 0.00 | 8.73 | 29.11 | 0 | 1.09 | 3 |
| 5000 | 5 | 0.00 | 8.09 | 25.07 | 0.00 | 6.20 | 25.07 | 0.00 | 9.41 | 25.07 | 0 | 1.59 | 4 | 0.00 | 9.15 | 25.07 | 0 | 1.72 | 4 | 0.00 | 9.45 | 25.07 | 0 | 1.60 | 4 |
| 5000 | 6 | 0.00 | 8.16 | 27.00 | 0.00 | 6.34 | 22.56 | 0.00 | 9.74 | 27.00 | 0 | 2.03 | 5 | 0.00 | 9.35 | 27.00 | 0 | 2.23 | 4 | 0.00 | 9.80 | 27.00 | 0 | 2.07 | 5 |
| 5000 | 7 | 0.00 | 8.12 | 23.15 | 0.00 | 6.45 | 23.24 | 0.00 | 9.99 | 23.80 | 0 | 2.52 | 5 | 0.00 | 9.42 | 23.80 | 0 | 2.76 | 5 | 0.00 | 10.07 | 23.80 | 0 | 2.57 | 5 |
| 5000 | 8 | 0.00 | 8.20 | 22.57 | 0.00 | 6.52 | 19.97 | 0.31 | 10.10 | 25.46 | 1 | 2.96 | 6 | 0.00 | 9.36 | 25.46 | 1 | 3.22 | 6 | 0.31 | 10.19 | 25.46 | 1 | 3.02 | 6 |
| 5000 | 9 | 0.37 | 8.27 | 20.94 | 0.00 | 6.57 | 19.16 | 0.80 | 10.19 | 22.57 | 1 | 3.42 | 7 | 0.00 | 9.38 | 22.14 | 1 | 3.73 | 7 | 0.80 | 10.28 | 22.57 | 1 | 3.49 | 7 |
| 5000 | 10 | 0.39 | 8.20 | 19.29 | 0.00 | 6.45 | 17.34 | 0.39 | 10.15 | 22.62 | 1 | 3.83 | 7 | 3.30 | 9.24 | 19.99 | 1 | 4.18 | 7 | 0.39 | 10.25 | 22.62 | 1 | 3.94 | 7 |
| 5000 | 12 | 1.02 | 8.20 | 19.27 | 0.00 | 6.44 | 18.40 | 1.50 | 10.26 | 19.93 | 2 | 4.73 | 8 | 1.26 | 9.16 | 19.93 | 1 | 5.11 | 9 | 1.50 | 10.35 | 19.93 | 2 | 4.86 | 9 |
| 2000 | 14 | 1.93 | 8.25 | 18.04 | 0.12 | 6.46 | 16.09 | 2.45 | 10.34 | 21.77 | 2 | 5.60 | 9 | 0.00 | 8.93 | 21.73 | 1 | 5.98 | 11 | 2.45 | 10.42 | 21.77 | 3 | 5.76 | 10 |
| 1000 | 16 | 2.46 | 8.24 | 16.65 | 0.41 | 6.48 | 14.34 | 2.95 | 10.43 | 19.90 | 3 | 6.61 | 10 | 0.00 | 8.85 | 19.35 | 1 | 6.81 | 11 | 3.01 | 10.51 | 19.90 | 3 | 6.78 | 11 |
| 500 | 18 | 3.07 | 8.15 | 16.12 | 0.72 | 6.60 | 13.24 | 3.61 | 10.35 | 17.19 | 4 | 7.42 | 11 | 0.09 | 8.78 | 15.99 | 1 | 7.71 | 12 | 4.09 | 10.43 | 17.19 | 4 | 7.63 | 11 |
| 500 | 20 | 3.26 | 8.23 | 14.21 | 0.84 | 6.31 | 11.89 | 5.40 | 10.52 | 16.25 | 5 | 8.35 | 13 | 0.04 | 8.61 | 15.98 | 1 | 8.50 | 14 | 5.40 | 10.60 | 16.25 | 5 | 8.58 | 14 |
| 500 | 25 | 3.10 | 8.37 | 13.70 | 1.81 | 6.47 | 13.01 | 4.07 | 10.64 | 15.62 | 6 | 10.59 | 15 | 0.42 | 8.21 | 14.39 | 1 | 10.34 | 17 | 4.07 | 10.67 | 15.62 | 6 | 10.83 | 16 |
| 150 | 30 | 3.51 | 8.44 | 12.70 | 2.19 | 6.67 | 11.20 | 5.00 | 10.90 | 15.60 | 9 | 12.80 | 17 | 0.16 | 8.44 | 15.00 | 1 | 12.50 | 19 | 5.00 | 10.90 | 15.60 | 10 | 13.11 | 18 |
| 150 | 35 | 4.98 | 8.35 | 13.18 | 2.85 | 6.53 | 11.32 | 6.46 | 10.74 | 15.32 | 10 | 15.04 | 20 | 0.44 | 7.75 | 14.82 | 1 | 13.56 | 24 | 6.46 | 10.78 | 15.32 | 12 | 15.31 | 22 |
| 150 | 40 | 4.35 | 8.50 | 12.78 | 2.89 | 6.68 | 11.40 | 5.78 | 10.93 | 14.76 | 12 | 17.37 | 22 | 0.22 | 7.90 | 13.64 | 1 | 16.00 | 26 | 5.78 | 10.97 | 14.76 | 12 | 17.78 | 26 |

SP = Steiner points
Performance is expressed as percent improvement over MST cost

Table 3a: average improvement percentages per Steiner point for 1-Steiner

| # | improvement per Steiner point | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| pts | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 3 | 7.66 | 0.00 | | | | | | | | | | | | | | | | | | |
| 4 | 8.39 | 0.62 | 0.00 | | | | | | | | | | | | | | | | | |
| 5 | 7.78 | 1.76 | 0.06 | 0.00 | | | | | | | | | | | | | | | | |
| 6 | 7.06 | 2.47 | 0.34 | 0.01 | 0.00 | | | | | | | | | | | | | | | |
| 7 | 6.34 | 2.83 | 0.77 | 0.07 | 0.00 | | | | | | | | | | | | | | | |
| 8 | 5.90 | 2.96 | 1.16 | 0.21 | 0.01 | 0.00 | | | | | | | | | | | | | | |
| 9 | 5.38 | 2.98 | 1.43 | 0.40 | 0.05 | 0.00 | | | | | | | | | | | | | | |
| 10 | 5.01 | 2.91 | 1.61 | 0.65 | 0.14 | 0.01 | 0.00 | | | | | | | | | | | | | |
| 12 | 4.33 | 2.77 | 1.75 | 0.97 | 0.40 | 0.08 | 0.01 | 0.00 | | | | | | | | | | | | |
| 14 | 3.87 | 2.60 | 1.79 | 1.18 | 0.65 | 0.28 | 0.07 | 0.01 | 0.00 | | | | | | | | | | | |
| 16 | 3.44 | 2.41 | 1.77 | 1.27 | 0.83 | 0.48 | 0.20 | 0.06 | 0.01 | 0.00 | | | | | | | | | | |
| 18 | 3.16 | 2.26 | 1.71 | 1.27 | 0.92 | 0.60 | 0.34 | 0.14 | 0.04 | 0.00 | | | | | | | | | | |
| 20 | 2.88 | 2.10 | 1.63 | 1.27 | 0.97 | 0.72 | 0.46 | 0.26 | 0.10 | 0.04 | 0.01 | 0.01 | 0.00 | | | | | | | |
| 25 | 2.45 | 1.85 | 1.49 | 1.21 | 0.98 | 0.80 | 0.63 | 0.46 | 0.32 | 0.19 | 0.09 | 0.04 | 0.01 | 0.00 | | | | | | |
| 30 | 2.17 | 1.69 | 1.38 | 1.15 | 0.98 | 0.83 | 0.69 | 0.57 | 0.46 | 0.36 | 0.27 | 0.17 | 0.09 | 0.04 | 0.01 | 0.00 | | | | |
| 35 | 1.82 | 1.45 | 1.23 | 1.07 | 0.94 | 0.81 | 0.70 | 0.58 | 0.49 | 0.42 | 0.35 | 0.28 | 0.21 | 0.15 | 0.09 | 0.05 | 0.02 | 0.01 | 0.00 | |
| 40 | 1.60 | 1.28 | 1.10 | 0.95 | 0.83 | 0.75 | 0.67 | 0.60 | 0.53 | 0.46 | 0.39 | 0.33 | 0.27 | 0.21 | 0.15 | 0.10 | 0.06 | 0.03 | 0.01 | 0.00 |

Table 3b: average improvement percentages per-Steiner-point and per-round for Batched 1-Steiner

| # | # of Rounds | | | improvement per round | | | | improvement per Steiner point | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| pts | min | ave | max | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
| 3 | 0 | 0.67 | 1 | 7.66 | 0.00 | 0.00 | 0.00 | 7.66 | 0.00 | | | | | | | | | | | | | | | | | | | | |
| 4 | 0 | 0.96 | 2 | 8.83 | 0.18 | 0.00 | 0.00 | 8.39 | 0.62 | 0.00 | | | | | | | | | | | | | | | | | | | |
| 5 | 0 | 1.07 | 4 | 9.30 | 0.30 | 0.00 | 0.00 | 7.78 | 1.72 | 0.10 | 0.00 | | | | | | | | | | | | | | | | | | |
| 6 | 0 | 1.15 | 4 | 9.48 | 0.38 | 0.01 | 0.00 | 7.06 | 2.37 | 0.42 | 0.03 | 0.00 | | | | | | | | | | | | | | | | | |
| 7 | 0 | 1.19 | 5 | 9.61 | 0.40 | 0.01 | 0.00 | 6.34 | 2.73 | 0.82 | 0.11 | 0.01 | 0.00 | | | | | | | | | | | | | | | | |
| 8 | 0 | 1.24 | 4 | 9.75 | 0.45 | 0.01 | 0.00 | 5.90 | 2.83 | 1.15 | 0.30 | 0.04 | 0.00 | | | | | | | | | | | | | | | | |
| 9 | 1 | 1.28 | 5 | 9.75 | 0.47 | 0.02 | 0.00 | 5.38 | 2.87 | 1.36 | 0.50 | 0.12 | 0.01 | 0.00 | | | | | | | | | | | | | | | |
| 10 | 1 | 1.33 | 6 | 9.82 | 0.49 | 0.02 | 0.00 | 5.01 | 2.83 | 1.51 | 0.69 | 0.24 | 0.04 | 0.00 | | | | | | | | | | | | | | | |
| 12 | 1 | 1.40 | 4 | 9.79 | 0.48 | 0.02 | 0.00 | 4.33 | 2.71 | 1.65 | 0.91 | 0.46 | 0.20 | 0.04 | 0.00 | | | | | | | | | | | | | | |
| 14 | 1 | 1.48 | 5 | 9.87 | 0.53 | 0.02 | 0.00 | 3.87 | 2.56 | 1.70 | 1.07 | 0.63 | 0.37 | 0.17 | 0.05 | 0.01 | 0.00 | | | | | | | | | | | | |
| 16 | 1 | 1.56 | 4 | 9.90 | 0.54 | 0.02 | 0.00 | 3.44 | 2.39 | 1.71 | 1.17 | 0.77 | 0.50 | 0.31 | 0.15 | 0.04 | 0.01 | 0.00 | | | | | | | | | | | |
| 18 | 1 | 1.61 | 4 | 9.85 | 0.54 | 0.03 | 0.00 | 3.16 | 2.23 | 1.65 | 1.20 | 0.83 | 0.56 | 0.39 | 0.26 | 0.11 | 0.03 | 0.00 | | | | | | | | | | | |
| 20 | 1 | 1.65 | 4 | 9.81 | 0.58 | 0.03 | 0.00 | 2.88 | 2.09 | 1.59 | 1.22 | 0.89 | 0.62 | 0.43 | 0.25 | 0.21 | 0.12 | 0.05 | 0.01 | 0.00 | | | | | | | | | |
| 25 | 1 | 1.77 | 5 | 9.97 | 0.52 | 0.03 | 0.00 | 2.45 | 1.84 | 1.46 | 1.18 | 0.94 | 0.74 | 0.57 | 0.41 | 0.30 | 0.24 | 0.20 | 0.11 | 0.06 | 0.02 | 0.01 | 0.00 | | | | | | |
| 30 | 1 | 1.93 | 4 | 10.14 | 0.69 | 0.03 | 0.01 | 2.17 | 1.68 | 1.36 | 1.12 | 0.93 | 0.78 | 0.62 | 0.51 | 0.39 | 0.31 | 0.24 | 0.24 | 0.21 | 0.18 | 0.12 | 0.05 | 0.01 | 0.00 | | | | |
| 35 | 1 | 2.00 | 4 | 10.09 | 0.58 | 0.02 | 0.00 | 1.82 | 1.44 | 1.23 | 1.07 | 0.92 | 0.79 | 0.67 | 0.54 | 0.45 | 0.38 | 0.31 | 0.24 | 0.20 | 0.20 | 0.20 | 0.12 | 0.09 | 0.04 | 0.02 | 0.01 | 0.00 | |
| 40 | 1 | 2.05 | 4 | 9.80 | 0.55 | 0.04 | 0.01 | 1.74 | 1.12 | 0.94 | 0.81 | 0.71 | 0.62 | 0.56 | 0.48 | 0.43 | 0.37 | 0.31 | 0.25 | 0.20 | 0.14 | 0.11 | 0.13 | 0.16 | 0.12 | 0.06 | 0.03 | 0.02 | 0.00 |

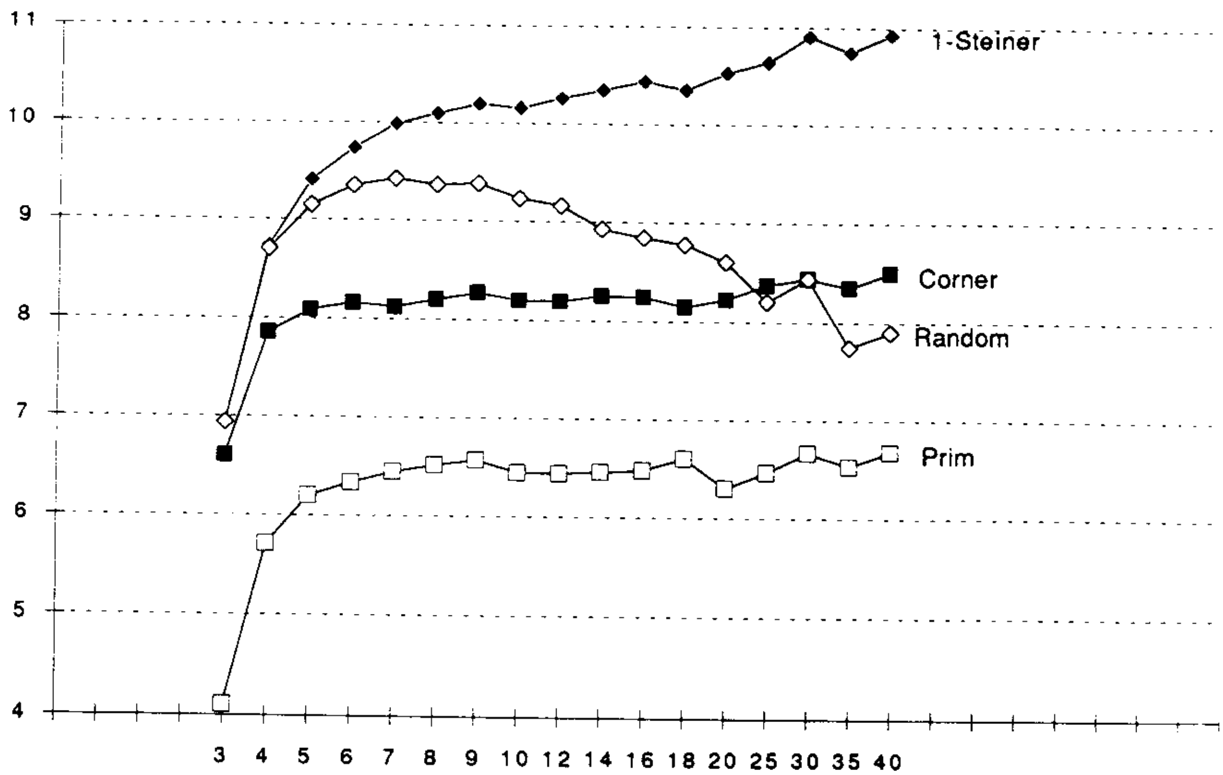


**Figure 9:** Performance comparison of the heuristics; the horizontal axis represents the number of points per set, while the vertical axis represents performance improvement over MST in percent.

| | | 1-Steiner | | | | | | Random 1-Steiner | | | | | | Meta | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # Cases | # Pts | Min Perf. | Ave Perf. | Max Perf. | Min Sp | Ave SP | Max SP | Min Perf. | Ave Perf. | Max Perf. | Min Sp | Ave SP | Max SP | Min Perf. | Ave Perf. | Max Perf. | Min Sp | Ave SP | Max SP |
| 1500 | 10 | 3.78 | 13.11 | 22.91 | 2 | 5.44 | 11 | 2.16 | 10.60 | 21.23 | 1 | 4.93 | 9 | 4.33 | 13.27 | 22.91 | 2 | 5.54 | 11 |
| 400 | 13 | 5.38 | 13.65 | 20.29 | 3 | 7.81 | 12 | 0.93 | 9.95 | 20.26 | 1 | 6.36 | 11 | 5.97 | 13.75 | 20.29 | 4 | 7.88 | 12 |
| 60 | 15 | 5.60 | 13.73 | 19.53 | 5 | 9.08 | 14 | 0.25 | 9.40 | 18.87 | 1 | 6.92 | 12 | 5.60 | 13.82 | 19.53 | 5 | 9.18 | 14 |
| 65 | 18 | 10.29 | 13.95 | 19.07 | 8 | 11.05 | 15 | 0.46 | 8.15 | 15.77 | 1 | 7.54 | 14 | 10.29 | 13.97 | 19.07 | 8 | 11.03 | 15 |
| 7 | 25 | 11.60 | 14.25 | 16.26 | 11 | 14.00 | 19 | 1.87 | 7.31 | 13.73 | 4 | 9.86 | 15 | 11.60 | 14.25 | 16.26 | 11 | 14.00 | 19 |
| 2 | 30 | 14.85 | 15.11 | 15.37 | 18 | 20.50 | 23 | 2.21 | 7.09 | 11.97 | 2 | 11.00 | 20 | 14.85 | 15.11 | 15.37 | 18 | 20.50 | 23 |

**Table 4:** Preliminary results for three dimensions

**Table 5a: MST degree statistics (in 2D)**

| # Sets | # Pts | MST Deg1 | MST Deg2 | MST Deg3 | MST Deg4 | MST Deg5 | MST / √N |
|---|---|---|---|---|---|---|---|
| 5000 | 3 | 66.667 | 33.333 | 0.000 | 0.000 | 0.000 | 0.420 |
| 5000 | 4 | 54.280 | 41.440 | 4.280 | 0.000 | 0.000 | 0.469 |
| 5000 | 5 | 47.896 | 44.344 | 7.624 | 0.136 | 0.000 | 0.496 |
| 5000 | 6 | 44.030 | 45.597 | 10.050 | 0.323 | 0.000 | 0.515 |
| 5000 | 7 | 41.017 | 46.934 | 11.651 | 0.397 | 0.000 | 0.528 |
| 5000 | 8 | 38.803 | 47.903 | 12.787 | 0.507 | 0.000 | 0.535 |
| 5000 | 9 | 37.091 | 48.722 | 13.504 | 0.682 | 0.000 | 0.542 |
| 5000 | 10 | 35.812 | 49.016 | 14.532 | 0.640 | 0.000 | 0.548 |
| 5000 | 12 | 33.497 | 50.475 | 15.227 | 0.802 | 0.000 | 0.555 |
| 5000 | 14 | 32.110 | 50.967 | 16.021 | 0.901 | 0.000 | 0.559 |
| 5000 | 16 | 31.032 | 51.391 | 16.620 | 0.956 | 0.000 | 0.560 |
| 5000 | 18 | 30.129 | 51.812 | 17.101 | 0.957 | 0.001 | 0.563 |
| 5000 | 20 | 29.463 | 52.156 | 17.299 | 1.082 | 0.000 | 0.564 |
| 5000 | 25 | 28.123 | 52.789 | 18.053 | 1.035 | 0.000 | 0.566 |
| 5000 | 30 | 27.238 | 53.317 | 18.320 | 1.123 | 0.002 | 0.566 |
| 5000 | 35 | 26.782 | 53.323 | 18.721 | 1.174 | 0.000 | 0.566 |
| 5000 | 40 | 26.316 | 53.581 | 18.892 | 1.210 | 0.001 | 0.566 |
| 5000 | 45 | 25.956 | 53.724 | 19.128 | 1.192 | 0.000 | 0.565 |
| 5000 | 50 | 25.731 | 53.734 | 19.340 | 1.194 | 0.001 | 0.564 |
| 5000 | 60 | 25.270 | 54.012 | 19.499 | 1.218 | 0.001 | 0.564 |
| 5000 | 70 | 25.043 | 54.029 | 19.671 | 1.256 | 0.001 | 0.563 |
| 5000 | 80 | 24.787 | 54.196 | 19.750 | 1.267 | 0.001 | 0.562 |
| 5000 | 90 | 24.761 | 53.992 | 19.957 | 1.289 | 0.001 | 0.561 |
| 5000 | 100 | 24.443 | 54.395 | 19.882 | 1.279 | 0.001 | 0.560 |
| 5000 | 200 | 23.957 | 54.424 | 20.282 | 1.336 | 0.002 | 0.555 |

**Table 5b: MST degree statistics (in 3D)**

| # Cases | # Pts | MST Deg1 | MST Deg2 | MST Deg3 | MST Deg4 | MST Deg5 | MST Deg6 | MST / N^(2/3) |
|---|---|---|---|---|---|---|---|---|
| 5000 | 3 | 66.667 | 33.333 | 0.000 | 0.000 | 0.000 | 0.000 | 0.542 |
| 5000 | 4 | 55.045 | 39.910 | 5.045 | 0.000 | 0.000 | 0.000 | 0.597 |
| 5000 | 5 | 49.448 | 41.468 | 8.720 | 0.364 | 0.000 | 0.000 | 0.629 |
| 5000 | 6 | 45.900 | 42.370 | 10.907 | 0.810 | 0.013 | 0.000 | 0.646 |
| 5000 | 7 | 43.814 | 42.277 | 12.609 | 1.266 | 0.034 | 0.000 | 0.657 |
| 5000 | 8 | 42.050 | 42.718 | 13.473 | 1.705 | 0.052 | 0.003 | 0.663 |
| 5000 | 9 | 40.951 | 42.351 | 14.740 | 1.884 | 0.073 | 0.000 | 0.671 |
| 5000 | 10 | 39.704 | 42.834 | 15.328 | 2.028 | 0.104 | 0.002 | 0.677 |
| 5000 | 12 | 38.263 | 42.850 | 16.327 | 2.412 | 0.147 | 0.002 | 0.683 |
| 5000 | 14 | 37.010 | 43.237 | 16.964 | 2.609 | 0.177 | 0.003 | 0.685 |
| 5000 | 16 | 36.339 | 43.029 | 17.613 | 2.840 | 0.174 | 0.006 | 0.686 |
| 5000 | 18 | 35.619 | 43.228 | 18.003 | 2.952 | 0.191 | 0.007 | 0.687 |
| 5000 | 20 | 34.910 | 43.594 | 18.283 | 3.014 | 0.197 | 0.002 | 0.688 |
| 5000 | 25 | 33.944 | 43.711 | 18.959 | 3.176 | 0.206 | 0.004 | 0.687 |
| 5000 | 30 | 33.249 | 43.918 | 19.303 | 3.309 | 0.219 | 0.001 | 0.686 |
| 5000 | 35 | 32.645 | 44.250 | 19.516 | 3.359 | 0.225 | 0.006 | 0.688 |
| 5000 | 40 | 32.380 | 44.168 | 19.767 | 3.450 | 0.229 | 0.006 | 0.686 |
| 5000 | 45 | 32.098 | 44.222 | 19.948 | 3.496 | 0.229 | 0.007 | 0.685 |
| 5000 | 50 | 31.792 | 44.419 | 20.028 | 3.526 | 0.228 | 0.007 | 0.683 |
| 5000 | 60 | 31.409 | 44.575 | 20.205 | 3.567 | 0.237 | 0.006 | 0.681 |
| 5000 | 70 | 31.152 | 44.681 | 20.301 | 3.612 | 0.247 | 0.007 | 0.680 |
| 5000 | 80 | 30.939 | 44.755 | 20.455 | 3.579 | 0.267 | 0.006 | 0.677 |
| 5000 | 90 | 30.746 | 44.851 | 20.548 | 3.595 | 0.251 | 0.008 | 0.676 |
| 5000 | 100 | 30.706 | 44.796 | 20.555 | 3.686 | 0.250 | 0.007 | 0.675 |
| 5000 | 200 | 30.143 | 45.009 | 20.838 | 3.731 | 0.271 | 0.007 | 0.666 |

## 3.5. Extensions.

There are several important extensions and generalizations of the work reported here.

### 3.5.1. Lower Bounds.

The 1-Steiner approach may yield new techniques for non-trivial *lower bounds* for MRST length. Currently there is no good lower bound other than that implied by Hwang's theorem, i.e, $2/3 \cdot \mathrm{MST} \le \mathrm{MRST}$, and lower bound techniques for other functionals of the plane (e.g., TSP) do not seem to extend to the Steiner problem. A tighter bound would be of tremendous practical significance in VLSI layout, affording immediate improvements in wiring estimation, models for congestion, etc. By examining the maximum effect of individual candidate Steiner points on the MST length, and using the fact that there are a linear number of Steiner points in the optimal MRST, it is likely that such a bound can be established. Quite possibly, the lower bound LB would not always be better than $2/3 \cdot \mathrm{MST}$, but asymptotically the value $\max(LB\ , 2/3 \cdot \mathrm{MST})$ would grow faster than $2/3 \cdot \mathrm{MST}$.

### 3.5.2. Higher Dimensions.

We observe that multiple-layer wiring, two-sided PCB design, and three-dimensional VLSI technologies are proliferating. Thus, we briefly mention several advantages of our approach in this higher dimension. It is not difficult to see that Hanan's theorem still holds in all higher dimensions [24], and we have conjectured [17] that the obvious generalization of Hwang's theorem holds in $d$-space, i.e.,

$$\mathrm{MST} \le \frac{2d-1}{d} \cdot \mathrm{MRST}.$$

Figure 10 illustrates (in three dimensions) an infinite family of higher-dimensional pointsets for which our 1-Steiner scheme performs *optimally* yet all other MST-based heuristics perform as badly as $(2d-1)/d$ times optimal in $d$ dimensions, which is no better than the MST length for the same pointsets.

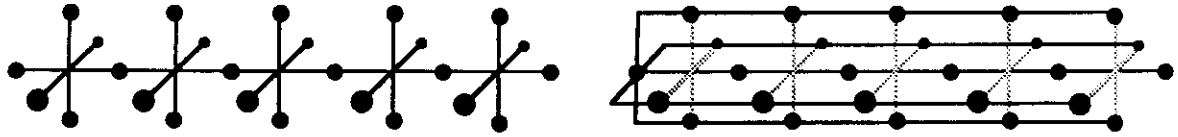

**Figure 10:** An example for D = 3 where the ratio MST-derived-RST/optimal-MRST is arbitrarily close to 5/3. The 3-dimensional MRST on the left has a cost of (6/5)(N-1), while any MRST derivable from the MST on the right has a cost of 2(N-3).

Notice that the previous standard approach, i.e., improving an initial MST solution, becomes much harder in three dimensions since there are more orientations for each edge; this suggests that the benefit of using a constructive 1-Steiner strategy increases in this higher dimension. We are currently studying the complexity of the 1-Steiner approach in higher dimensional geometries. Preliminary empirical results for three-dimensional problem instances seem favorable and are illustrated in Table 3. The 1-Steiner approach also succeeds in the presence of non-orthogonal wiring directions [21], and Hanan's result also generalizes for such geometries.

### 3.5.3. Implementation Refinements.

The 1-Steiner method is readily parallelizable: we can compute a single iteration of 1-Steiner by examining (in parallel) the effect on the MST of adding each one of $O(N^2)$ SP candidates to the original pointset. An MST of a pointset can be computed in time $O(N \log N)$ by one processor [19], or in time $O(\log N \cdot \log\log N \cdot \log\log\log N)$ using $O((N+M) / (\log N \cdot \log\log N))$ processors on a concurrent read, concurrent write parallel random-access machine (CRCW PRAM) model [6]; here $M$ is actually $O(N^2)$ so the number of processors required is $O(N^2 / (\log N \cdot \log\log N))$. Therefore, a single iteration of 1-Steiner can be accomplished in time $O(N \log N)$ with $O(N^2)$ processors, or in time $O(\log N \cdot \log\log N \cdot \log\log\log N)$ with $O(N^4 / (\log N \cdot \log\log N))$ processors, depending on which of the two methods above is used to construct an MST.

In the single-processor case a near-linear expected-time algorithm may be used to construct the MST [5], bringing the total parallel expected-time also to near-linear. Alternatively, an MST may be obtained in linear time from the Voronoi diagram, and the latter can be constructed in expected linear time in all dimensions [7], thus serving to reduce the expected time of each 1-Steiner iteration.

Since the MST over a pointset is a subgraph of the Voronoi dual [19], a precomputation of the Voronoi diagram will reduce $M$ to $O(N)$ and hence $O(N / (\log N \cdot \log\log N))$ processors will suffice for the MST computation; $O(N^3 / (\log N \cdot \log\log N))$ processors will suffice for one iteration of the 1-Steiner method. However, the fastest known parallel Voronoi diagram computation requires time $O(\log^3 N)$ (and $N$ processors) [1], and this term will dominate the overall parallel running time for one iteration of 1-Steiner (of course, a linear number of iterations would multiply all of these running times by a factor of $N$).

This places the problem of computing one iteration of 1-Steiner in the well-known complexity class NC [26] of problems amenable to solutions via parallel algorithms utilizing a polynomial number of processors and terminating within polylogarithmic time. It is an open question whether the complete 1-Steiner algorithm can be placed within NC. Furthermore, none of the above parallelization schemes is an optimal speedup of our implementation (i.e., where the product of the number of processors times the parallel time is equal to the sequential time), and we ask if such a speedup exists.

## 4. Conclusion.

In this paper, we have presented a fast new approach to the rectilinear Steiner problem. The algorithm is practical due to an elegant implementation which uses methods from computational geometry; it parallelizes readily; and it yields results that reduce wirelength by several percent over the best previously methods. A randomized variant of the algorithm, along with a "batched" variant, have also proved successful, and the approach extends readily to routing with non-orthogonal wiring directions and three-dimensional VLSI layout technologies.

## 5. References.

[1] A. Aggrawal, B. Chazelle, L. Guibas, C. O'Dunlaing and C. Yap, "Parallel Computational Geometry", *Proc. IEEE Symp. on Foundations of Computer Science*, 1985, pp. 468-477.

[2] J. Beardwood, H. J. Halton and J. M. Hammersley, "The Shortest Path Through Many Points", *Proc. Cambridge Philos. Soc.* 55 (1959), pp. 299-327.

[3] M. W. Bern, *Personal Communications*, January 1990 and March 1990.

[4] M. W. Bern, "Two Probabilistic Results on Rectilinear Steiner Trees", *Algorithmica* 3 (1988), pp. 191-204.

[5] K. Clarkson, "Fast Expected-Time and Approximation Algorithms for Geometric Minimum Spanning Trees", *Proc. of the ACM Symposium on Theoretical Computer Science* (1984), pp. 342-348.

[6] R. Cole and U. Vishkin, "Deterministic Coin Tossing and Accelerating Cascades: Micro and Macro Techniques for Designing Parallel Algorithms", *Proc. ACM Symp. on Theory of Computation*, 1986, pp. 206-219.

[7] R. Dwyer, "Higher Dimensional Voronoi Diagrams in Linear Expected Time", *Proc. of the Symposium on Computational Geometry*, ACM, 1989, pp. 326-333.

[8] D. Eppstein, G. Italiano, R. Tamassia, R. E. Tarjan, J. Westbrook and M. Yung, "Maintenance of a Minimum Spanning Forest in a Dynamic Planar Graph", *Proc. ACM-SIAM Symp. on Discrete Algorithms*, 1990, pp. 1-11.

[9] M. Garey and D. S. Johnson, "The Rectilinear Steiner Problem is NP-Complete", *SIAM J. of Applied Math.* 32(4) (1977), pp. 826-834.

[10] G. Georgakopoulos and C. H. Papadimitriou, "The 1-Steiner Tree Problem", *J. Algorithms* 8 (1987), pp. 122-130.

[11] E. N. Gilbert and H. O. Pollak, "Steiner Minimal Trees", *SIAM J. of Applied Math.* 16 (1968), pp. 1-29.

[12] M. Hanan, "On Steiner's Problem With Rectilinear Distance", *SIAM J. of Applied Math.* 14 (1966), pp. 255-265.

[13] N. Hasan, G. Vijayan and C. K. Wong, "A Neighborhood Improvement Algorithm for Rectilinear Steiner Trees", Proc. of ISCAS, 1990.

[14] J.-M. Ho, G. Vijayan and C. K. Wong, "New Algorithms for the Rectilinear Steiner Tree Problem", *IEEE Transactions on Computer-Aided Design*, 9(2), 1990, pp. 185-193.

[15] F. K. Hwang, "On Steiner Minimal Trees with Rectilinear Distance", *SIAM J. of Applied Math.* 30(1) (1976), pp. 104-114.

[16] F. K. Hwang, "The Rectilinear Steiner Problem", *J. Design Automation and Fault-Tolerant Computing* (1978), pp. 303-310.

[17] A. Kahng and G. Robins, "On Performance Bounds for Two Rectilinear Steiner Tree Heuristics in Arbitrary Dimension", 1990, submitted to *IEEE Trans. on Computer-Assisted Design.*

[18] A. Kahng and G. Robins, "A New Family of Steiner Tree Heuristics With Good Performance: The Iterated 1-Steiner Approach", *Technical Report*, UCLA CS Department, #CSD-900014., April 1990.

[19] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, New York, Springer-Verlag, 1985.

[20] D. Richards, "Fast Heuristic Algorithms for Rectilinear Steiner Trees", *Algorithmica* 4 (1989), pp. 191-207.

[21] M. Sarrafzadeh and C. K. Wong, "Hierarchical Steiner Tree Construction in Uniform Orientations", *draft*, 1990.

[22] J. M. Smith and J. S. Liebman, "Steiner Trees, Steiner Circuits and the Interference Problem in Building Design", *Engineering Optimization* 4 (1979), pp. 15-36.

[23] J. M. Smith, D. T. Lee and J. S. Liebman, "An O(N log N) Heuristic Algorithm for the Rectilinear Steiner Minimal Tree Problem", *Engineering Optimization* 4 (1980), pp. 179-192.

[24] T. L. Snyder, "A Simple and Faster Algorithm for the Rectilinear Steiner Problem in General Dimension", *Proc. ACM Symp. on Computational Geometry*, 1990, *to appear.*

[25] J. M. Steele, "Growth Rates of Euclidean Minimal Spanning Trees With Power Weighted Edges", *The Annals of Probability* 16(4) (1988), pp. 1767-1787.

[26] U. Vishkin, "Synchronous Parallel Computation: A Survey", *Technical Report* 71 (1983), Courant Institute, NYU.

[27] P. Winter, "Steiner Problem in Networks: A Survey", *Networks* 17 (1987), pp. 129-167.

**Appendix I -- Degree Bounds and MRST Decomposition.**

*Theorem 1*: In any finite two-colored tree such that the degree of every red node must be $d$ or more, the ratio of blue to red nodes is at least $d-2$.

*Proof*: Assume the theorem true for all trees with $k$ or less nodes, and consider a two-colored tree $T$ of $k+1$ nodes. If $T$ contains no red nodes then we are done. Otherwise, there is a red node $r$ in $T$ which is adjacent to at least $d-2$ blue nodes, $b_1, \ldots, b_{d-2}$. This can be seen by repeatedly removing from the tree all blue leaf nodes until none remain; each of the resulting red leaves in this stripped tree has exactly $d-1$ blue sons in the original tree, and there must be at least one such red leaf in the stripped tree if the original tree had any red nodes. Remove $r$ and $b_1, \ldots, b_{d-2}$, yielding a forest of at most $d$ subtrees; now connect the root node of one of the subtrees to the root nodes of all the others, so as to produce a tree of size $k-d+2$ that still satisfies the color constraint, namely that red nodes must have degree $d$ or more; this is true because the degree of any node in the new tree is not any smaller than what it was in the original tree. By the induction hypothesis, the theorem holds for this smaller tree and therefore the ratio of blue to red nodes in the original tree was at least $d-2$. □

The theorem is sharp in the sense that for any fixed integer $d > 2$ and for every positive $\varepsilon$, there is a two-colored tree wherein red nodes all have degree $\geq d$ and blue nodes outnumber red nodes by a ratio of less than $d - 2 + \varepsilon : 1$ for any positive real number $\varepsilon$.

*Corollary*: A point set $P$ has cardinality greater than the number of Steiner points in the MRST on $P$.

*Proof*: Let $T$ be a MRST for an arbitrary pointset $P$. Recall that all Steiner points have degree 3 or greater. Now color all the original points of $P$ blue, and all the Steiner points of $T$ red. By the previous theorem, where now $d=3$, the ratio of blue nodes (original points) to red nodes

(Steiner points) is at least d–2=1; that is, there are more points than Steiner points. □

Note that if the number of red nodes is two or more, there must be at least two red nodes each of which has at least two blue neighbors. Thus in the first step of the proof of Theorem 1, we may eliminate these two red and four (or more) blue nodes, yielding $|\{\text{red points}\}| \le |\{\text{blue points}\}| - 2$, which is sharp.

*Theorem 2*: (MRST Decomposition) An MRST for an arbitrary pointset in an arbitrary metric can be incrementally constructed by starting with one of its points, and adding a constant number of points at each step, such that the topologies of the sequence of intermediate MRST's change only monotonically (i.e., each tree is a subgraph of the next tree in the sequence).

*Proof*: We start with an MRST T for the pointset P, and at each step find and eliminate a subtree consisting of a Steiner point whose sons are all either points in P or leaves in T (as in the proof of the previous theorem), or else strip away a leaf from P whose father is also in P. This process eventually exhausts T while removing O(1) points at each step (since the MRST degree is bounded by a constant, from Lemma 1). The time-reversal of this process will construct T from scratch, keeping the topology monotonic. □

This theorem essentially transforms the MRST problem to the problem of finding a certain permutation of a set of points, such that the MRST's of initial subsequences of this permutation change only monotonically. In other words, MRST's can be built incrementally, by including only O(1) points at each step. It is possible that this scheme can be used to construct a new simple (albeit exponential) *exact* algorithm for the MRST problem.

**Appendix II -- Theorems of Georgakopoulos and Papadimitriou [10].**

*Definition*: Given a point set P, an *isodendral* region R of the plane is a maximal region such that for any point $x \in R$, the topology of the MST on $P \cup \{x\}$ is constant.

We will form the isodendral regions by overlapping eight separate *oriented Dirichlet partitions* of the plane.

*Definition*: Let $\theta_1$ and $\theta_2$ be two directions in the plane; for every point $x \in P$, let $C(\theta_1,\theta_2,p)$ define the obvious cone with vertex x. The *oriented Dirichlet cell* (ODC) of $p \in P$ is the locus of points x such that p is the closest point of P to x among points in $C(\theta_1,\theta_2,x)$. For given $\theta_1$ and $\theta_2$, the ODCs of N points define an *oriented Dirichlet partition* (ODP) of the plane.

*Theorem 4*: The following statements hold: (i) each ODC is a connected region; (ii) the boundary between two adjacent ODCs consists of the union of at most O(1) segments; (iii) computing all ODCs of N points can be done in $O(N^2)$ time; (iv) k ODPs can be overlapped in $O(k^2N^2)$ time; (v) the result of this overlap defines $O(N^2)$ *isodendral* regions. □

We will form the ODP's by using the eight $\theta_1$ - $\theta_2$ pairs which bound the eight octants of the plane. The central idea is as follows. Within each isodendral region, the optimum 1-Steiner point can be adjacent only to some subset of the (at most eight) points of P whose ODCs intersect in that region. Since the size of the subset must be at least three (no Steiner point can have degree less than three), we must examine at most $\sum_{i=3}^{8} C(8,i)$ (which is a constant) subsets of points. For each subset, we can determine the optimum location of the 1-Steiner point in constant time, as this entails simply optimizing a convex function. By Theorem 5 below, since the degree of the 1-Steiner point is $\leq 8$, we can check the cost-savings (over the MST) associated with the optimal 1-Steiner point from each isodendral region within O(1) time. Since there are only $O(N^2)$ regions, we obtain the desired complexity bound for the entire 1-Steiner computation.

*Theorem 5*: With $O(N^2)$ preprocessing, we can compute the new minimum spanning tree, after the lengths of k edges incident to a point p are decreased, in $O(k^2)$ time. □

This is done by precomputing for each pair of nodes i and j in the MST, the length of the shortest edge along the path from i to j in the MST; this may be accomplished in a total of $O(N^2)$ time via N applications of depth-first search, one starting from each node in the MST. When a new SP is to be considered for addition to the pointset (at a particular isodendral region), all the subsets of its O(1) possible neighbors in the new MST are considered, and their distances to each other (via the new point) are recomputed and compared to precomputed shortest edge on the path connecting each pair in the old MST; if the cycles formed in the MST by the connection of the new point to some subset of its neighbors, are such so that the sum of the precomputed previously shortest edges around these cycles is greater than the sum of the new edges added, the old edges are discarded in favor of the new ones, reducing the total new MST cost by the determined amount. All this is accomplished within O(1) time per each new point to be considered for addition into the original pointset.