Appears in Parallel Algorithms
and Architectures, Ed. Albrecht
et al., Akademie-Verlag
Berlin, 1987

Computing a Rectilinear Steiner Minimal Tree in $n^{O(\sqrt{n})}$ Time

*Clark D. Thomborson (a.k.a. Thompson)**
Linda L. Deneen
Gary M. Shute

Computer Science Department
University of Minnesota
Duluth, Minnesota 55812
U.S.A.

Abstract

We propose an algorithm for computing a rectilinear Steiner minimal tree on n points in $2^{O(\sqrt{n}\log n)}$ time and $O(n^2)$ space. This is an asymptotic improvement on the $2^{O(n)}$ time required by current algorithms. If the points are distributed uniformly at random on the unit square, the Steiner tree calculated by our algorithm is minimal with high probability. The "constant factors" of our algorithm are such that it should be feasible to obtain exact solutions for n-point problems whenever $n \leq 25$. Previously, only problems of size $n \leq 20$ were feasible.

1 Introduction

The earliest formulation of Steiner's Problem is attributed to Fermat in the early 17th century, but a generalization is the problem we know today: given as input a set V_{in} of vertices in the Euclidean plane, find a set S of points, called *Steiner points*, and a set E of edges on $V_{in} \cup S$ such that E forms a tree on $V_{in} \cup S$ and the sum of the lengths of the edges in E is minimal.

Kuhn's excellent survey paper [14] traces the development of work on this Euclidean Steiner Tree Problem. It has important applications, such as the design of telephone networks, and has captured the attention of computer scientists interested in its computational aspects. In 1977 Graham, Garey, and Johnson showed that the Euclidean Steiner Tree Problem is NP-Complete [7], dashing hopes of optimal algorithms for large sets V_{in}, but legitimizing interest in heuristics [4,10,13,19,20,24] and easier variants of the problem [1,22].

A variant of the Euclidean Steiner problem has important applications to circuit design. The Euclidean problem can be translated to the rectilinear (L_1) metric by simply changing the distance function. The resulting *Rectilinear Steiner Problem*, surveyed by Hwang in [11], asks for a minimal Steiner tree with edges parallel to the x- and y-axes. Such a tree is called a *Rectilinear Steiner Minimal Tree* (RSMT).

Any VLSI design system must cope with RSMT problems (usually in suboptimal ways) in order to connect the terminals of subcircuits. The rectilinear restriction arises because of technological difficulties in creating non-rectilinear wiring paths on the surface of a semiconductor substrate. RSMT problems also arise at another level of circuit design, when one deposits metal on a printed circuit board in order to connect one chip's pins to another's. In both applications, it is important to have a minimal or near-minimal length wiring path to reduce stray capacitance, and thus to increase circuit speed. A constraint of secondary importance is to minimize the number of bends in the RSMT. This makes it easier to perform the "detailed routing" that assigns slightly different positions to parallel edges in unconnected RSMTs.

*Supported by the National Science Foundation, through its Design, Tools and Test Program under grant number MIP 84-06408.

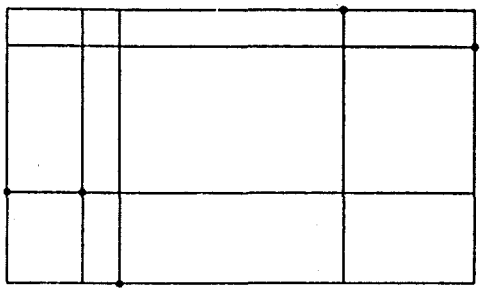

Figure 1: A grid graph on five points.

Most RSMT algorithms find Steiner trees that are subgraphs of a "grid graph" formed by drawing horizontal and vertical line segments through the vertices, where the segments are limited to the rectangle enclosing the vertices [9]. See Figure 1. Despite this restricted search space, the RSMT problem is NP-complete [8].

The published literature contains two exact algorithms for finding a RSMT on a set of points. Both algorithms are exponential and can only be used if the set is small. Yang and Wing describe a branch-and-bound algorithm running in $2^{O(n^2)}$ time and $O(n^2)$ space [23]. Apparently, Yang and Wing's algorithm is feasible for vertex set sizes up to $n = 20$. Dreyfus and Wagner use dynamic programming to solve the more general problem of computing a minimal Steiner tree for a graph [6]. Their algorithm is easily adapted to solve the Rectilinear Steiner Problem in $O(n^2 3^n)$ time, where n is the number of vertices. Unfortunately, the Dreyfus and Wagner algorithm requires $O(n^2 3^n)$ space, making it infeasible for $n \geq 18$.

In this algorithm we begin by dividing the vertex set of size n in half by a vertical cut line. Any RSMT on the vertex set has $k \geq 1$ edges that cross the cut line. (Our preliminary analysis shows that if $n \geq 20$ points are uniformly distributed at random on the unit square, then with probability $p > 0.9$ there is a RSMT for which $k \leq 2\lceil\sqrt{n}\rceil$.) For each possible value of k, then, we choose k edges of the associated grid graph where the RSMT is assumed to cross the cut line. Next we find all pairs of forests that can join together at the k edges to form a Steiner tree. The forest pairs define "terminals" on the edge of two Steiner tree subproblems. By solving all subproblems, and choosing the pair with minimal total length, we find a minimum Steiner tree with high probability.

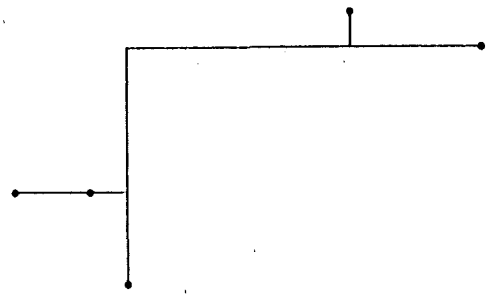

Figure 2: A minimum Steiner tree on the graph of Figure 1.

We have developed a new algorithm using a technique called *nonserial dynamic programming* [5]. When calculations on each half are done recursively with cut lines alternating between vertical and horizontal, straightforward analysis shows that our algorithm runs in $2^{O(\sqrt{n}\log n)}$ time. This is a clear improvement on the $2^{O(n)}$ performance of the best previous algorithm, at least for large n. We believe it is also a significant improvement for feasibly-small n, say $n \leq 25$.

2 Formal Description of the Problem

Given a set V_{in} of $|V_{in}| = n$ points in the plane, it is trivial to construct a grid graph like the one shown in Figure 1. Formally, the vertex set V_{gg} of the grid graph G_{gg} is $X \times Y$, where X and Y are the sets of x- and y-coordinates appearing in the input point set V_{in}. The edge set E_{gg} of the grid graph is defined by the 4-neighborhoods of the points in V_{gg}. If there are no degeneracies, there are $2n(n-1)$ edges in the grid graph on n points. The grid graph of Figure 1 has one y-degeneracy, so it has only $(2n-1)(n-1)$ edges.

To find a rectilinear Steiner minimal tree (RSMT) for V_{in}, it is sufficient to find a minimal-length connected subgraph of G_{gg} containing all the points in V_{in} [9]. Figure 2 shows one of the two RSMTs that can be obtained in this way for the points of Figure 1. Note that the RSMT would not be unique even if the y-degeneracy were removed. Another trivial observation is that there are in general many RSMTs that are not subgraphs of a grid graph (because extra jogs can be inserted anywhere in an "L" shaped wire). In some applications [17], it is important to produce several RSMTs or near-minimal rectilinear Steiner trees for a given set of points. Such algorithms are beyond the scope of this paper.

A trivial algorithm for finding a RSMT is to search the space of subgraphs of V_{gg}, picking a shortest one that satisfies the connectedness and point inclusion properties. A naive implementation of this algorithm would take $2^{O(n^2)}$ time. (Yang and Wing's algorithm, mentioned in the introduction, is a branch-and-bound improvement on the trivial algorithm. Unfortunately, they neither prove nor conjecture an asymptotic decrease in time [23].)

3 Description of the New Algorithm

As outlined in the introduction, the new algorithm searches the space of subgraphs of V_{gg} in a recursive fashion, solving RSMT subproblems on nearly-square subgrids. A typical subproblem is shown in Figure 3. The desired solution to a subproblem is actually a forest of RSMTs, one Steiner tree for each star graph in the subproblem. (Figure 3 has three star graphs. The one on the top right has degree one, the one on the bottom right has degree two, and the one on the left has degree four.) We seek a forest of RSMTs of minimal total length, connecting the terminals on the boundary of the subproblem in the same way as the star graphs. Additionally, we require the solution forest to contain every vertex in the input set V_{in} that lies within the boundary of the subproblem.

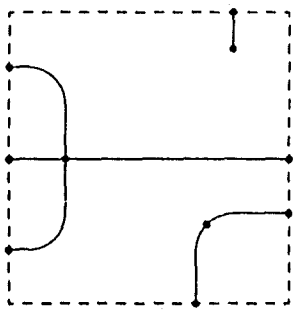

Figure 3: A 7-terminal subproblem with three stars.

A typical recursive subdivision of Figure 3 is shown in Figure 4. Here, we show how Figure 3 can be solved by two subproblems joined by three cut-edges. The left-hand subproblem has three stars and six terminals; the right-hand subproblem has three stars and seven terminals. The gap shown between the subproblems is one "column" of the grid graph. It contains no vertices.

To obtain an optimal solution to a (sub)problem, we must examine all possible subdivisions. A convenient way to generate all subdivisions is outlined below. Before doing this, we note that, without loss of generality, no two star graphs intersect. If they did, their solution RSMTs would also intersect, forming a cycle; and a shorter composite RSMT could be formed by omitting one of the edges in that cycle. Thus the star graphs divide the region inside a RSMT problem into a set of planar faces.

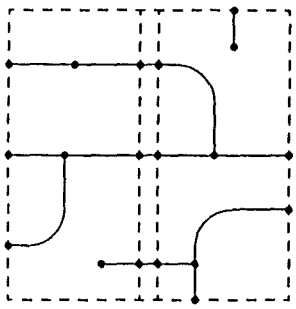

Figure 4: A vertical subdivision of Figure 3 into a 6-terminal problem and a 7-terminal problem.

Graphically, a recursive subdivision of a RSMT problem is a pair of dashed lines running from mid-bottom to mid-top of the problem region. See Figure 4. We start the subdivision process by enumerating the star graphs on the boundary of the planar face containing the mid-bottom "start" point of our subdividing line. After choosing (exhaustively) one of these star graphs, we have several options of how to generate a cut-edge.

The simplest case is that of a degree-1 star. Figure 5 shows how we must always split such a star into a degree-1 star and a degree-2 star, where the degree-1 star may be either on the left (Figure 5b) or the right (Figure 5c) of the cut-edge. After choosing one of these options, the cut-edge can be assigned a y-coordinate to locate it precisely in the grid graph. If there are n_r rows in this problem's region of the grid graph, then, there are $2n_r$ ways of creating a cut-edge by splitting a degree-1 star.

Similar reasoning gives an exhaustive analysis for generating cut-edges from a degree-2 star. Figure 6 shows the three possibilities. We may form two degree-2 stars, one on each side of the cut-edge, as in Figure 6b. Alternatively, we may form one degree-1 star and one degree-3 star, as in Figures 6c and 6d.

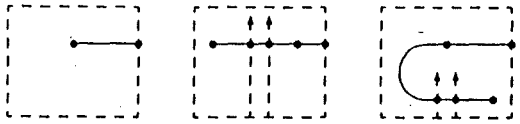

Figure 5a. Figure 5b. Figure 5c.

Figure 5: Two options for splitting the degree-1 star of Figure 5a.

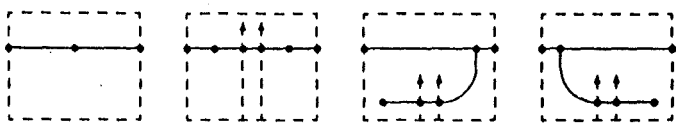

Figure 6a. Figure 6b. Figure 6c. Figure 6d.

Figure 6: Three options for splitting the degree-2 star of Figure 6a.

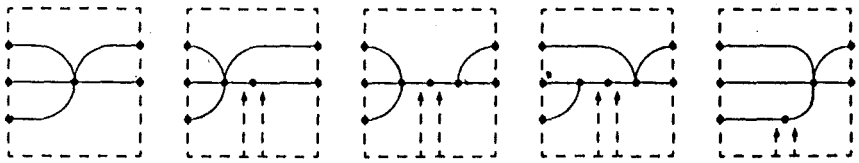

Figure 7a. Figure 7b. Figure 7c. Figure 7d. Figure 7e.

Figure 7: Four ways of getting a degree-2 star from the degree-5 star of Figure 7a.

Higher-degree stars are a little more complicated. It seems best to start by enumerating all $d-1$ planar partitions of a degree-d star into a degree-c star on the left (where $0 \le c \le d-2$), a degree-2 star in the middle, and a degree-$(d-c-2)$ star on the right. See, for example, Figure 7. The middle degree-2 star can then be split in three ways, as in Figure 6. In all, then, there are $3(d-1)$ topologically distinct ways of generating a cut-edge from a degree-d star, for all $d \ge 2$.

Lemma 1 *There are fewer than $k(3t+5k)^k\binom{n_r}{k}$ distinct ways to subdivide a n_r-row problem with t terminals into two subproblems joined by k cut-edges.*

Proof sketch. Each terminal gives rise to 2 topological options if it is attached to a degree-1 star, and d terminals give rise to $3(d-1)$ options if attached to a degree-d star. The number of options per terminal per cutedge is thus upper-bounded by 3. The operations of Figures 5b, 5c, and 6b add a degree-2 star to the problem, increasing the number of options for subsequent cuts by at most 3. (Note that fewer options are available, in general, because not all stars are be on the boundary of the planar face containing the current endpoints of the dashed lines.) The operations of Figure 6c and 6d add a degree-1 star and increase the degree of an adjoining star by one. This increases the number of options for subsequent cuts by at most 5.

The factor of $\binom{n_r}{k}$ arises because we must assign each cut-edge to one of the n_r rows in the current region of the grid graph. The leading factor of k is a (crude) allowance for the possibility of subdividing the problem with fewer than k cut-edges. □

It remains to set up the basis and starting-point for the recursion. The most convenient basis for the recursion is formed by a zero-height, zero-width subproblem. Such a subproblem is solved by an RSMT with no edges.

The RSMT problem we wish to solve at the outset has zero terminals and thus no stars. Figure 8 shows how to generate the first cut-edge in this situation. Successive cut-edges can then be generated in the usual fashion, by splitting stars.

One last difficulty: due to degeneracies, one may encounter a subproblem of zero width but non-zero height. Such subproblems should be rotated by 90 degrees before being subdivided. This

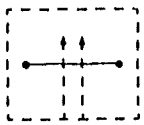

Figure 8: First cut-edge through a 0-terminal problem.

"rotation before subdivision" operation can be put to more general use to obtain the best solution time: one should always rotate a problem if there are more rows than columns in its region of the grid graph.

We can now write a recurrence for the time to run our algorithm on a grid graph with n_r rows and n_c columns ($n_r \le n_c$), with at most $f(n_r)$ cut-edges bridging the two sub-problems. We write the recurrence in terms of the perimeter $m = 2n_r + 2n_c$:

$$T(m) \le \begin{cases} 2f(m)(3m+5f(m))^{f(m)}\binom{m}{f(m)}T(\lfloor m/2 \rfloor), & \text{if } m > 1; \\ 1, & \text{otherwise.} \end{cases}$$

In the next section, we argue that $f(n) = 2\lceil\sqrt{n}\rceil$ will yield a tree that is a RSMT with high probability. Solving the recurrence above, we find that the time to solve an n-point RSMT, $T(4n)$, is $n^{O(\sqrt{n})}$.

4 How Many Cut-Edges?

We present here the preliminary results of our probabilistic analysis.

Lemma 2 *[9] Let V_{in} be a set of points in the plane, and let T be any RSMT on V_{in}. Without changing the length of T, T can be transformed to a RSMT T' with the property that each edge of T' has at least one endpoint in V_{in}.*

We say a RSMT T is in normal form if it obeys the end-point property of T' in Lemma 2.

Lemma 3 *Let S be a 1×1 square, and let V_{in} be any set of n points in S. Let C be a vertical line through X, and let R be the $2w+1$ rectangle consisting of all points of S within distance w of C. Let V_R be the points of V_{in} lying in R. Finally, let T be a normal-form RSMT on V_{in}, and let E be the set of edges of T which cross C. Then*

$$|E| \leq |V_R| + 2\left\lceil \frac{1}{w} \right\rceil + 2.$$

Proof sketch. The key observation here is that the edges which cross C must either have length greater than w or have an endpoint in V_R. Long edges crossing C must be at least w units apart, otherwise we can develop a contradiction to T being an RSMT. □

By elementary probability theory, if n points are chosen uniformly at random in the unit square, the probability of seeing at most j points in a predetermined $1 \times 1/(2\sqrt{n})$ rectangle is

$$\sum_{0 \leq i \leq j} \binom{n}{i} \left(\frac{1}{\sqrt{n}}\right)^i \left(1 - \frac{1}{\sqrt{n}}\right)^{n-i}.$$

This sum is lower-bounded by $1 - \epsilon$ if we set

$$j = \sqrt{n}/2 + (e-1)\sqrt{\sqrt{n}/4 \ln \frac{1}{\epsilon}},$$

for any $0 < \epsilon < 1$ [16,18]. (Here, $e = 2.718...$, the natural base for logarithms.)

For our algorithm to find a RSMT, we must have a high-probability upper bound on the number of points near our $n^2 - 1$ recursive cuts (one of length 1, six of length 1/2, twenty-four of length 1/4, etc.). We find that our algorithm will succeed with probability p if we use up to $f(n_r)$ cut-edges when subdividing a problem on n_r grid-graph rows, where

$$f(n) = \sqrt{n} + (e-1)\sqrt{\sqrt{n} \ln \frac{n^2 - 1}{1-p}}.$$

A simpler formula that gives a probability of success greater than 0.9 for all $n \geq 20$ is

$$f(n) = 2\left\lceil \sqrt{n} \right\rceil.$$

We note, in passing, that we can use Lemma 3 to bound the number of cut-edges to allow at each stage of the recursion. We would then obtain an exact algorithm for the RSMT that runs in $O(n^{\sqrt{n}})$ time with high probability.

5 Experimental Results

We have not yet completed coding the general case of our algorithm. We do, however, have experimental results for the simple case of $f(n) = 1$, in which at most one cut-edge separates subproblems. This algorithm runs in $n^{O(\log n)}$ time, and is feasible for $n \leq 50$ on a Sun-3/50 workstation. See Figure 9 for the result of a 20-hour computation, in which 841 million subproblems were solved. Possibly this algorithm will be of value as a heuristic, although the competition is fierce: an $O(n \log n)$ time minimum spanning tree algorithm, suitably modified, is an excellent heuristic [4,13].

6 Future Work

We have described an exact algorithm for computing a rectilinear minimal Steiner tree (RMST) on n points that runs with high probability in time $n^{O(\sqrt{n})}$, as well as an algorithm that runs in $n^{O(\sqrt{n})}$ deterministic time that with high probability produces a RMST.

The results reported here are preliminary in nature. We intend to improve our algorithm in several ways.

- We can reduce the size of the grid graph over which the algorithm must search for solutions [24,12].

- We can tabulate the results of frequently-occurring intermediate computations (e.g., those with one terminal) so that these results need not be recomputed.
- We can short-circuit the algorithm when it reaches small cases that can be solved more quickly, using algorithms for point sets satisfying special constraints [1,3].
- We can terminate the algorithm before examining all cut sets, by proving that a previously obtained solution is in fact optimal. Such termination criteria are well established for the Euclidean TSP [15], but no one has applied this idea to the RSMT (but see [2,21]).
- Most importantly, we don't have to cut every point set at the same x-coordinate. By positioning the cut line appropriately, or possibly by applying the planar separator theorem, we can reduce either the expected or maximum number of cut edges.

Using the techniques listed above, we expect to develop faster approximate and exact algorithms for calculating an RSMT.

References

[1] A. V. Aho, M. R. Garey, and F. K. Hwang. Rectilinear Steiner trees: efficient special-case algorithms. *Networks*, 7:37–58, 1977.

[2] Y. P. Aneja. An integer linear programming approach to the Steiner problem in graphs. *Networks*, 10:167–178, 1980.

[3] Marshall W. Bern. A more general special case of the Steiner tree problem. 1986. Manuscript submitted to STOC–19.

[4] Marshall W. Bern. Two probabilistic results on rectilinear Steiner trees. In *Proceedings of the 18th Annual ACM Symposium on the Theory of Computing*, pages 433–441, May 1986.

[5] U. Bertele and F. Briosche. *Nonserial Dynamic Programming.* Academic Press, New York, 1972.

[6] S. E. Dreyfus and R. A. Wagner. The Steiner problem in graphs. *Networks*, 1:195–207, 1972.

[7] M. R. Garey, R. L. Graham, and D. S. Johnson. The complexity of computing Steiner minimal trees. *SIAM Journal of Applied Mathematics*, 32(4):835–859, 1977.

[8] M. R. Garey and D. S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM Journal of Applied Mathematics*, 32(4):826–834, June 1977.

[9] M. Hanan. On Steiner's problem with rectilinear distance. *SIAM Journal of Applied Mathematics*, 14(2):255–265, March 1966.

[10] F. K. Hwang. An $O(n \log n)$ algorithm for suboptimal rectilinear Steiner trees. *IEEE Transactions on Circuits and Systems*, CAS–26(1):75–77, January 1979.

[11] F. K. Hwang. The rectilinear Steiner problem. *Design Automation and Fault Tolerant Computing*, 2:303–310, 1978.

[12] Alfred Iwainsky, Enrico Canuto, Oleg Taraszow, and Agostino Villa. Network decomposition for the optimization of connection structures. *Networks*, 16:205–235, 1986.

[13] Janos Komlos and M. T. Shing. Probabilistic partitioning algorithms for the rectilinear Steiner tree problem. *Networks*, 15:413–423, 1985.

[14] Harold W. Kuhn. "Steiner's" problem revisited. In G. B. Dantzig and B. C. Eaves, editors, *Studies in Mathematics, Vol. 10: Studies in Optimization*, pages 52–70, Mathematical Association of America, 1974.

[15] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, editors. *The Traveling Salesman Problem: A Guided Tour of Optimization.* John Wiley and Sons Ltd., 1985.

[16] Prabhakar Raghavan. *Randomized Rounding and Discrete Ham-Sandwich Theorems: Provably Good Algorithms for Routing and Packing Problems.* PhD thesis, U. C. Berkeley, Computer Science Division, August 1986.

[17] Prabhakar Raghavan and Clark D. Thompson. Provably good routing in graphs: regular arrays. In *Proceedings of the 17th Annual ACM Symposium on the Theory of Computing*, pages 79–87, ACM, May 1985.

[18] Prabhakar Raghavan and Clark D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, to appear, 1987.

[19] Michal Servit. Heuristic algorithms for rectilinear Steiner trees. *Digital Processes*, 7:21–32, 1981.

[20] J. MacGregor Smith, D. T. Lee, and Judith S. Liebman. An $O(n \log n)$ heuristic algorithm for the rectilinear Steiner minimal tree problem. *Engineering Optimization*, 4:179–192, 1980.

[21] David I. Steinberg. The fixed charge problem. *Naval Research Logistics Quarterly*, 17:217–225, 1970.

[22] V. A. Trubin. Subclass of the Steiner problems on a plane with rectilinear metric. *Cybernetics (U.S.A.)*, 21(3):320–325, 1985. translated from *Kibernetika 21:3*, 37–40, May-June 1985.

[23] Y. Y. Yang and Omar Wing. Optimal and suboptimal solution algorithms for the wiring problem. In *IEEE International Symposium on Circuit Theory*, pages 154–158, 1972.

[24] Y. Y. Yang and Omar Wing. Suboptimal algorithm for a wire routing problem. *IEEE Transactions on Circuit Theory*, 508–510, September 1972.

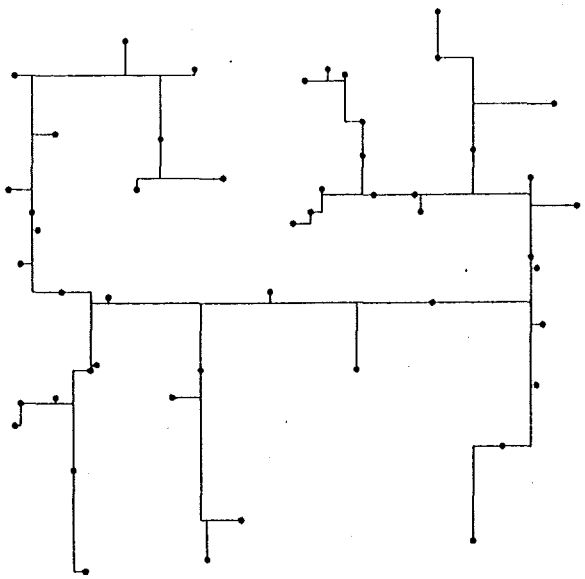

Figure 9: Rectilinear Steiner tree, minimal for cut size 1, on 50 random points.