Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 00-012

Polynomial Time Approximation Scheme for the Rectilinear Steiner Arborescence Problem

Bing Lu and Lu Ruan

February 14, 2000

Polynomial Time Approximation Scheme for the Rectilinear Steiner Arborescence Problem

Bing Lu *　　　Lu Ruan*

Abstract

Given a set N of n terminals in the first quadrant of the Euclidean plane E^2, find a minimum length directed tree rooted at the origin o, connecting to all terminals in N, and consisting of only horizontal and vertical arcs oriented from left to right or from bottom to top. This problem is called *rectilinear Steiner arborescence problem.* which has been proved to be NP-complete recently[1]. In this paper, we present a polynomial time approximation scheme for this problem.

Keywords: PATS, Arborescence, Steiner terminals, VLSI design, Approximation Algorithm

*Computer Science Department, University of Minnesota, Minneapolis, MN55455, USA. Email:{blu,ruan}@cs.umn.edu

1 Introduction

Given a set N of n terminals in the first quadrant of the Euclidean plane E^2, find a minimum length directed tree rooted at the origin o, connecting to all terminals in N, and consisting of only horizontal and vertical arcs oriented from left to right or from bottom to top. This problem is called *rectilinear Steiner arborescence problem* (RSAP). Each feasible solution is called a *rectilinear Steiner arborescence*(RSA). RSAP has been proved to be NP-complete recently[1]. Rao et al.[7] presented the first polynomial-time($O(nlogn)$) approximation to the RSAP with performance ratio 2. Córdova et al.[4] generalized the RSA approximation to the all-quadrant RSAP with the same time complexity and performance ratio. Ladeira De Matos [8] proposed a dynamic programming algorithm which takes exponential time to solve the problem. Nastansky et al.[3] applied an integer programming formulation to the d dimensional version of the RSAP which also has exponential running time complexity. Leung et al.[2] presented two optimal algorithms recursive branch-and-bound and dynamic programming for solving the RSAP. Although these algorithms outperformed previous exact methods on average, the worst case running time is still exponential. In this paper, we present a new polynomial-time approximation scheme (PTAS) for the RSAP by combining with the general framework proposed by Arora [5]. That is, for every fixed $c > 1$, a randomized algorithm computes a $(1 + 1/c)$-approximation of the RSMA in time $O(n^{O(c)} \cdot logn)$ and a deterministic version obtained from derandomization computes a $(1 + 1/c)$-approximation of the RSMA also in time $O(n^{O(c)} \cdot logn)$.

With the advent of the VLSI design and process enters sub-micron range, the RSAP plays more important role in the field of performance-driven VLSI design. Cong et al.[6] showed that rectilinear Steiner Arborescence significantly outperforms the traditional Steiner tree approach in delay optimization. The RSAP also has various applications in multicast network communication and supercomputer message routing.

2 Preliminaries

We consider only the case that all terminals lie in the first quadrant of E^2. The general RSAP can be efficiently decomposed into four problems. One in each quadrant. The distances are measured in the L_1 metric. Furthermore, all terminals in N are called *sinks* and $G(N)$ represents the grid that includes a vertical line segment and a horizontal line segment through every sink, and is bounded by the horizontal line segment through the highest sink, the vertical line segment through the rightmost sink, the x-*axis* and the y-*axis*. A *rectilinear Steiner minimum arborescence*(RSMA) for the set N is an RSA π with minimum $l(\pi)$ which denotes the total length of arcs in π. From Rao et al. [7], we have the following basic properties of RSAs:

(a1) An RSA is a directed arboresence rooted at the origin and contains all the terminals of N. Furthermore, all leaves are terminals of N.

(a2) Each *Steiner* terminal of an RSA has degree three (one indegree and two outdegree).

(a3) All edges point "northeast". That is, if an edge e joins p to q, then $x_p \leq x_q$ and $y_p \leq y_q$.

(a4) There exists an RSMA which uses only arcs of $G(N)$.

(a5) An RSMA contains at most $n-2$ Steiner terminals.

(a6) Only the origin of RSA has indegree zero.

Let π denote an RSA and L denote a line segment (parallel to x-*axis* or y-*axis*) within $G(N)$. Let M_1, M_2, $\cdots$, M_t be the points on which π crosses L. Break π at those points, π will become a forest. We need two copies of each M_i, one for each side of L. Let $M_i^{'}$ and $M_i^{''}$ denote these copies.

Lemma 1 *The vertical(horizontal) line segment L will always break an RSA π into forest F over the terminals set $N \cup \{M_1^{'}, \cdots, M_t^{'}, M_1^{''}, \cdots, M_t^{''}\}$. Let $\pi^{'}$ and $\pi^{''}$ denote a partition* *of the trees in the forest F(that is,* $\pi^{'} \cup \pi^{''} = F$ *and* $\pi^{'} \cap \pi^{''} = \emptyset$*). If* $\pi^{'}$ *contains all points in* $M_1^{'}, \cdots, M_t^{'}$ *and* $\pi^{''}$ *contains all points in* $M_1^{''}, \cdots, M_t^{''}$*, then* $\pi^{'}$ *is an RSA and* $\pi^{''}$ *is forest.*

proof. It is obviously that $\pi^{''}$ is forest. Suppose $\pi^{'}$ is not an RSA. Then $\pi^{'}$ is also forest. Let $T_1^{'}$ and $T_2^{'}$ be two distinct trees in $\pi^{'}$. That means there are two terminals(Steiner terminals or terminals of N) with indegree zero in the original RSA. It is impossible. Therefore, $\pi^{'}$ must be an RSA. □

3 Main Result

Theorem 1 *There exists an approximation algorithm computing a* $(1+1/c)$*-approximation of the RSMA in time* $O(n^{O(c)} \log n)$ *where* $c > 1$*.*

We prove the theorem by showing that we can construct a polynomial-time algorithm which finds a $(1 + 1/c)$-approximation of the RSMA. The algorithm applies the approach recently initiated by Arora[5].

The algorithm consists of three steps: perturbation and rounding of RSAP–to perturb an instance of RSAP to make all coordinates well-rounded; shifted quadtree construction–to construct a (a, b)-shifted quadtree; dynamic programming–to find an optimal r-light RSA. The running time of this algorithm is $O(n^{O(c)} \log n)$.

Definition 1 *An RSA is "r-light" with respect to the shifted dissection if it crosses each boundary of each square in the dissection at most r times.*

3.1 Perturbation and Rounding of RSAP

The first step of the algorithm is to perturb an instance of RSAP to make all coordinates integral, and the minimum interterminal distance at least 2 (*well-rounded*). Let the *bounding box* of the set N denote the smallest axis-aligned square that contains the set, and let L be the size of the box where L is the length of the longer side of the box and OPT be an optimum RSA. Then we have $OPT \geq L$. In order to perturb the instance, we place a grid of granularity $L/8nc$ and move each terminal(Steiner terminal or original terminal) to its nearest grid point(more than one terminal may map to the same grid point). Since OPT contains at most $n-2$ Steiner terminals, only $(2n-2)$ terminals are moved. Because each terminal is moved by at most $2 \cdot L/8nc$, the cost of the OPT is changed by at most $2(2n-2) \cdot L/8nc \leq L/2c$, which is at most $OPT/2c$. Rescaling distance by $L/16nc$, the instance becomes well-rounded, that is, the minimum interterminal distance at least 2. As a result, the size of the bounding box, L, is $O(nc) = O(n)$. Thus we now need to compute a $(1+1/2c)$-approximation in this new instance. But since $c > 1$ can be an arbitrary constant, that doesn't matter.

By requiring the Steiner terminals to lie on the grid, the precision issues which will arise when the algorithm "guesses" the location of a Steiner terminal can be avoided.

Lemma 2 *The perturbation and rounding of a RSA π results in another RSA π'.*

proof. Although the perturbation may map more than one terminal to the same grid point, it will not create cycles and edges pointing from right to left or top to bottom. □

3.2 Shifted Quadtree construction

We construct the dissection and quadtree with shift (a, b) in the same way as [5]. For convenience of the reader, we sketch the construction as follows:

The dissection of the bounding box is constructed by first partitioning the bounding box into four equal smaller squares. Then each square is recursively partitioned into four equal squares until the size of resulting squares is ≤ 1. The quadtree is constructed in the similar way except we stop the partitioning when the square has at most one node.

Choosing two integers a,b in $[0, L)$, (a, b)-shift of the dissection and quadtree is defined as follows: the x-coordinate of the left line of the dissection or quadtree is moved to x-coordinate a and the y-coordinate of the lower line of the dissection or quadtree is moved to y-coordinate b. Then the rest of the dissection or quadtree is "wrapped-around".

Since the size of the well-rounded bounding box is $O(n)$, the depth of the shifted quadtree is $O(logn)$, and the number of squares in it is $S = O(n \cdot logn)$.

Lemma 3 *(Structure Lemma)*

Let the minimum nonzero interterminal(including Steiner terminals) distance in an RSA instance be 2 *and* L *be the size of its bounding box(*$L = O(n)$*). shift* (a, b) *is randomly chosen, where* $0 \leq a, b \leq L$*. Then there exists an RSA* π *of cost at most* $(1 + 1/c) - OPT$ *which is r-light with respect to the dissection with shift* (a, b)*, where* $r = O(c), c > 1$*.*

3.3 Dynamic Programming

Assuming the truth of the Structure Lemma, we apply dynamic programming to find the optimal r-light RSA with respect to the randomly shifted quadtree. The running time of the dynamic programming is $O(n^{O(c)} \cdot logn)$. This optimal r-light RSA is a $(1 + 1/c)$-approximation for the RSMA.

Suppose s is a square of the randomly shifted quadtree. Then the optimal r-light RSA crosses the boundaries of s a total of $\leq 4r$ times. Then the portion of the optimal RSA inside s is forest such that (a) a set P containing $\leq r$ points on each of the four boundaries of the square, (b) a partition $(P_1, P_2, \cdots, P_t)$ of P where $t \leq 2r$. The goal is to find an optimum r-light collection of t directed Steiner trees such that all edges point "northeast" and there is no intersection between any pair of trees, where the ith tree contains all the points in P_i, and the trees together contain every terminal inside s. For those squares that do not contain any input terminal, the algorithm has to "guess" the Steiner terminals inside them. However, since each square is entered and left only at most $4r$ times, we can solve it in constant time by using exact algorithms proposed by [2][3][8] as an instance of the RSAP of size at most $4r$(r is a constant). The number of leaves in the shifted quadtree is $O(n)$ and the size of the shifted quadtree is $O(n \cdot logn)$.

The dynamic programming puts costs of the optimal solutions to all instances of the r-light directed Steiner forest(all edges point "northeast") problems into a lookup table. The algorithm is done when this table is completely built up. The total number of entries in the lookup table is the total number of different instances of the r-light directed Steiner forest problems in the shifted quadtree. In a quadtree with S squares, this number is at most $O(S \cdot \binom{O(n)}{r}^4 \cdot 2^{4r} \cdot (4r)!)$.

The table is built up in a bottom-up fashion. The algorithm optimally solves instances at the leaves of the quadtree in $O(r)$ time since they contain at most 1 terminal and $O(r)$ selected points. Inductively, suppose all r-light directed Steiner forest problem for squares at depth $> i$ have been solved and let s be a square at depth i. Let s_1, s_2, s_3, s_4 be its four children of s. For every choice in (a), (b) for s, the algorithm enumerates all possible ways in which an r-light directed Steiner forest could cross the boundaries of $s_1, \cdots, s_4$. Thus all choices for the following are enumerated: a set of $\leq r$ points on the four inner edges of the $s_1, \cdots, s_4$; The number of choices is at most $(\binom{O(n)}{r}^4 \cdot 2^{4r})$. Thus the running time of the dynamic programming algorithm is $O(S \cdot \binom{O(n)}{r}^8 \cdot (2)^{8r} \cdot (4r)!)$, which is $O(n^{O(c)} \cdot (logn))$.

A deterministic approximation algorithm can be obtained by derandomizing the above algorithm by runing the dynamic programming part of the algorithm for all choices of the pair (a, b). Therefore, the running time will be increased by a factor of $O(n^2)$.

3.4 Proof of the Structure Lemma

Lemma 4 *(Patching Lemma)*

Let L be any line segment of length l and π be an RSA that crosses L at least twice. Then there exists a new RSA π' with total additional length at most l and crosses the line segment once.

Proof. We only prove when the line segment is vertical (with the same argument, we can prove the case in which the line segment is horizontal).

Suppose π crosses L a total of t times. Let $M_1, M_2, \cdots, M_t$ be these crossing points. Assuming without loss of generality that the y-coordinates of M_i are non-decreasing. Breaking π at those points will break π into forest F. Let M_i' and M_i'' denote two copies of M_i, one on each side of L, respectively. Furthermore, let π_L and π_R be a partition of trees of F such that π_L contains $M_1', \cdots, M_t'$ and π_R contains $M_1'', \cdots, M_t''$. From *Lemma 1*, we know that π_L is an RSA and π_R is a forest. If we use a vertical line segment L' which is a sub-segment of infinitesimally shifted version of L to connect $M_1^{''}, M_2^{''}, \cdots, M_t^{''}$, we can get a new tree $\pi_R^{'}$. Connecting $M_1^{'}$ and $M_1^{''}$ will combine π_L and $\pi_R^{'}$ into a new RSA $\pi^{'}$ which crosses L only once and $l(\pi^{'}) \geq l(\pi)$ because $l(L^{'}) \leq l(L)$. Furthermore, $l(\pi^{'}) - l(\pi) \leq l(L)$. □

We use a sequence of horizontal and vertical line segments at unit distance from each other to grid the bounding box of the set N. Let l be one of the grid lines, π is an RSA with total length of A and $t(\pi, l)$ denote the total number of times that π crosses l. The next lemma relates $t(\pi, l)$ to A.

Lemma 5 *If the minimum internode dinstance is at least* 2*, then*

$$\sum_{l:vertical} t(\pi, l) + \sum_{l:horizontal} t(\pi, l) \leq 2A \tag{1}$$

Proof. Let e be an edge of π and has length E. Suppose a and b are the lengths of the horizontal and vertical projections of the edge($a + b = E$). Then it contributes at most $(a+1)+(b+1)$ to the left hand side. Since $(a+1)+(b+1) = a+b+2 = E+2$ and $E \geq 2$, we have $E + 2 \leq 2E$. □

Proof:(Structure Lemma) Suppose π is an optimum RSA with total length of OPT and shift$(a, b)(0 \leq a, b \leq L)$ is randomly generated. We use a deterministic procedure to prove the Structure Lemma by modifying π over many steps into an RSA $\pi^{'}$ which is r-light with respect to the randomly-shifted dissection. We upperbound the increasing cost(slight) in expectation as follows. We use the same accounting methods as[5] and "charge" any cost increase to some (horizontal or vertical) line of the grid. We will show that for each line l of the grid,

$$E_{a,b}[charge\ to\ line\ l\ when\ shift\ is\ (a, b)] \leq \frac{2t(\pi, l)}{r-1}, \tag{2}$$

By linearity of expectations it then follows that the expected increase in the cost of the RSA is

$$\sum_{l:vertical} \frac{2t(\pi, l)}{r-1} + \sum_{l:horizontal} \frac{2t(\pi, l)}{r-1}, \tag{3}$$

which is $\leq \frac{4OPT}{r-1}$ by Lemma 5. Let $r \geq 8c+1$, the expected increase in the RSA cost is at most $OPT/2c$. Then with probability at least $1/2$, the cost of of the optimum r-light RSA for the shifted dissection is at most $(1+1/c)\cdot OPT$ (Markov's inequality).

Assume without loss of generality, that the size of the bounding box L is a power of 2. Therefore, all lines used in the dissection are grid lines. The *maximal level* defined in [5]: the maximal level of a line is highest level it is at. For example, the bounding box is at level 0.

For each vertical line l in the grid we have,

$$\Pr_a[l\ is\ at\ level\ i] = \frac{2^i}{L}. \tag{4}$$

where each $i \leq logL$ and a is randomly picked. A similar statement is true for horizontal lines.

We call the procedure MODIFY(l,i,b) defined in [5] to modify the optimum RSA to an r-light RSA by patching "bottom up" for all levels $j \geq i$(with slight change).

MODIFY(l,i,b)

(l is a vertical grid line, b is the vertical shift of the dissection,

and i is the maximal level of l)

For $j = logL$ downto i do:

For $p = 0, 1, \cdots, 2^j - 1$, if the segment of l between the

y-coordinates $(b + p \cdot \frac{L}{2^j}\ mod\ L)$ and

$(b + (p+1) \cdot \frac{L}{2^j}\ mod\ L)$ is crossed by the

current RSA more than r times, then use the patching lemma to

reduce the number of croosings to 2.

Because the line segment could be "wrapped-around" and the patching need to be done for its two parts separately, the number of crossings after patching is 2.

Let $c_{l,j}(b)(j \geq i)$ denote the number of times we apply the patching lemma in the iteration corresponding to j in the "for" loop in MODIFY(l,i,b). Because the optimum

RSA π crossed line l only $t(\pi, l)$ times, and each call of the MODIFY(l, i, b) replaces at least $r+1$ crossings by at most 2. We have that

$$\sum_{j \geq 1} c_{l,j}(b) \leq \frac{t(\pi, l)}{r-1} \tag{5}$$

Moreover, the cost increased can be estimated as follows:

$$\textit{Increase in RSA cost due to MODIFY}(l, i, b) \leq \sum_{j \geq i} c_{l,j}(b) \cdot \frac{L}{2^j} \tag{6}$$

We charge this cost to l, and only when i is the maximal level of line l this charge occurs. By equation (4) that happens with probability at most $\frac{2^i}{L}$ (over the choice of the horizontal shift a). Thus,

$$\begin{aligned} E_a[\textit{charge to } l \textit{ when horizontal shift is } a] &= \sum_{i \geq 1} \frac{2^i}{L} \cdot \textit{cost increase due to MODIFY}(l, i, b) \\ &\leq \sum_{i \geq 1} \frac{2^i}{\cdot} \sum_{j \geq i} c_{l,j}(b) \cdot \frac{L}{2^j} \\ &= \sum_{j \geq 1} \frac{c_{l,j}(b)}{2^j} \cdot \sum_{i \leq j} 2^i \\ &\leq \sum_{j \geq 1} 2 \cdot c_{l,j}(b) \\ &\leq \frac{2t(\pi, l)}{r-1} \end{aligned}$$

where every vertical line l and every $0 \leq b \leq L-1$. □

References

[1] W. Shi and C. Su, "The rectilinear Steiner arborescence problem is NP-complete", to appear *11th ACM-SIAM Symposium on Discrete Algorithms(SODA),January 2000.*

[2] K. S. Leung and J. Cong, "Fast Optimal Algorithms for the Minimum Rectilinear Steiner Arborescence Problem", *Proc. Intl Symposium on Circuits and Systems*, May 1997, pp. 1568-1571.

[3] L. Nastansky, S. M. Selkow, and N. F. Stewart, "Cost-minimal trees in directed acyclic graphs", Z. Oper. Res. 18(1974), 59-67.

[4] J. Córdova, Y. H. Lee, "A Heuristic Algorithm for the Rectilinear Steiner Arborescence Problem", University of Florida CIS Department Tech. Report TR-94-025, 1994.

[5] S. Arora, "Polynomial Time Approximation Schemes for Euclidean Traveling Salesman and other Geometric Problems", to appear in *Journal of the ACM.*

[6] J. Cong, K. S. Leung and D. Zhou, "Performance Driven Interconnect Design Based on Distributed RC Delay Model", *Proc. ACM/IEEE Design Automation Conf.* 1993. pp. 606-611.

[7] S. K. Rao, P. Sadayappan, F. K. Hwang, P. W. Shor, "The Rectiliear Steiner Arborescence Problem". *Algorithmica*, 7(1992), pp.277-288.

[8] R. R. Ladeira de Matos, "A Rectilinear Arborescence Problem", Dissertation, University of Alabama, 1979.