Approaching the 5/4–Approximation for Rectilinear Steiner Trees

Piotr Berman[1] **Ulrich Fößmeier**[2] **Marek Karpinski**[3]
Michael Kaufmann[4] **Alexander Zelikovsky**[5]

TR–94–041

August, 1994

Abstract

The rectilinear Steiner tree problem requires to find a shortest tree connecting a given set of terminal points in the plane with rectilinear distance. We show that the performance ratios of Zelikovsky's[17] heuristic is between 1.3 and 1.3125 (before it was only bounded from above by 1.375), while the performance ratio of the heuristic of Berman and Ramaiyer[1] is at most 1.271 (while the previous bound was 1.347). Moreover, we provide $O(n \cdot \log^2 n)$-time algorithms that satisfy these performance ratios.

---

[1]Computer Science Department, Pennsylvania, State University, University Park, PA 16802, USA. Email: `berman@cse.psu.edu`.

[2]Wilhelm–Schickard–Institut für Informatik, Universität Tübingen, Sand 13, 72076 Tübingen. Email: `foessmei@informatik.uni-tuebingen.de`.

[3]Department of Computer Science, Universität Bonn, Römerstraße 164, 53117 Bonn. Email: `marek@cs.uni-bonn.de`. Research partially supported by the International Computer Science Institute, Berkeley, California, by the DFG Grant KA 67314, by the ESPRIT BR Grant 7097 and by ECUS030.

[4]Wilhelm–Schickard–Institut für Informatik, Universität Tübingen, Sand 13, 72076 Tübingen. Email: `mk@informatik.uni-tuebingen.de`.

[5]Institute of Mathematics, Akademiei 5, Kishinev, 277028, Moldova. Email: `17azz@mathem.moldova.su`. Research partially supported by Volkswagen Stiftung

1 Introduction

Consider a metric space with distance function d. For any set of *terminal* points S one can efficiently find MST(S), a minimum spanning tree of S. Let mst(S, d) be the cost of this tree in metric d. A Steiner tree is a spanning tree of a superset of the terminal points (the extra points are called Steiner points). It was already observed by Pierre Fermat that the cost of a Steiner tree of S may be smaller than mst(S, d). Thus it is natural to look for the Steiner minimum tree, that is, for the least cost Steiner tree. However, finding such a tree is NP-hard for almost all interesting metrics, like Euclidean, rectilinear, Hamming distance, shortest-path distance in a graph etc. [7, 12]. Because these problems have many applications, they were subject of extensive research [3, 7, 8, 9, 15, 11].

In the last two decades many approximation algorithms for finding Steiner minimum trees appeared. The quality of an approximation algorithm is measured by its performance ratio: an upper bound of the ratio between the achieved length and the optimal length.

In the *rectilinear* metric, the distance between two points is the sum of the differences of their $x-$ and $y-$coordinates. The rectilinear Steiner tree problem (RSP) got recently new importance in the development of techniques for VLSI routing [13, 14].

The most obvious heuristic for the Steiner tree problem approximates a Steiner minimum tree of S with MST(S). While in all metric spaces the performance ratio of this heuristic is at most 2 [16], Hwang [10] proved that in the rectilinear plane the performance ratio of this heuristic equals exactly 1.5.

Zelikovsky [17] and Berman/Ramaiyer [1] gave two better heuristics for RSP. Now we give a more precise analysis of the performance ratio of these heuristics. Our results are the following:

1. Zelikovsky's algorithm has a performance ratio between 1.3 and 1.3125.
2. The Berman/Ramaiyer algorithm has a performance ratio of at most $\frac{61}{48} \approx 1.271$.
3. The now best approximation of factor 1.271 can be found in time $O(n \log^2 n)$ considerably improving the previous approximation [1] not only in quality, but also in efficiency. The previous time bound was $O(n^{3.5})$.

In the next section we provide a synopsis of the two approaches of Zelikovsky and Berman/Ramaiyer. In Sections 3, 4 and 5, we derive new estimates for the performance ratios. In particular, in Section 3 we introduce a new technical tool to study those ratios, so called *double covers*. Section 4 provides upper and lower bounds for approximations based on 3-restricted Steiner trees. Finally in Section 5, we improve the approximation further by considering 4-restricted trees.

Section 6 and 7 show how to obtain these approximation efficiently. We prove that for a heuristic that builds a 4-restricted tree it suffices to consider only linearly many quadruples of terminal points and demonstrate how to find those quadruples in time $O(n \log^2 n)$. This improves the running time of the approximation algorithm by a factor of $O(n^2)$ to $O(n^{1.5})$. Section 8 describes a slightly worse heuristic which comes arbitrarily close to the new performance ratio and which runs in time $O(n \log^2 n)$. We conclude with some final remarks and directions for further research.

Note that many proofs in the text are omitted; the details can be found in [2].

2 Berman/ Ramaiyer's and Zelikovsky's Heuristics

A Steiner tree T of a set of terminals S is *full* if every internal node of T is a Steiner point, i.e., not a terminal. If T is not full, it can be decomposed into full Steiner trees for subsets of terminals that overlap only at leaves. Such subtrees are called *full Steiner components* of T [8]. T is called *k-restricted* if every full component of T has at most k terminals. The length of the shortest $k-$restricted Steiner tree of S is denoted by $t_k = t_k(S)$, and $s = t_\infty$ denotes the length of the optimal Steiner tree.

So, $t_2(S)$ is the length of the min. spanning tree of S. $t_2 \leq \frac{3}{2}s$ [10], $t_3 \leq \frac{5}{4}s$ [17] and generally $t_k \leq \frac{2k-1}{2k-2}s$ [1]. These bounds are tight for $k = 2, 3$ but in general $t_k \leq \frac{2k}{2k-1}s$ for $k \geq 4$. The main idea of the new heuristics is to obtain good $k-$restricted Steiner trees and to show how they approximate a Steiner minimum tree.

The method described here can be applied with an arbitrary metric d. Without loss of generality, we may assume that the metric d on the set of terminals S is the shortest-path distance for the weighted edges D connecting S. This way, MST(d) is the minimum spanning tree of the graph $< S, D >$, we denote this tree with

MST(D), and its cost with mst(D). If we increase the set of edges D by some extra edges, say forming a set E, the shortest-path distance will decrease; MST($D \cup E$) is the minimum spanning tree for the modified metric.

Let z be a triple of terminals. Let $T(z)$ be the Steiner minimum tree of z, $d(z)$ is the cost of $T(z)$ and $Z(z)$ is a spanning tree of z consisting of zero-cost edges.

If we decide to use $T(z)$ as a part of that tree, the remaining part can be computed optimally as MST($D \cup Z(z)$), from which we remove zero-cost edges of $Z(z)$. The improvement of the tree cost due to this decision is the *gain* of z, denoted $g(z, D)$. It is easy to see that $g(z, D)$ = mst(D)-mst($D \cup Z(z)$)-$d(z)$. Now, if we have already decided to use some set of triples, so that the zero-cost edges of their $Z(z)$'s form set Z, the gain of a subsequent triple z_0 can be expressed as $g(z_0, D \cup Z)$ = mst($S, D \cup Z$)-mst($S, D \cup Z \cup Z(z_0)$)-$d(z_0)$.

In Zelikovsky's greedy approach (GA), Z is initially empty. In an iteration step, we choose a triple z that maximizes the gain $g(z, D \cup Z)$. If this gain is positive, we use $T(z)$ and replace Z with $Z + Z(z)$; otherwise we exit the loop. At the end, we remove zero-cost edges from MST($D \cup Z$) and replace them with the chosen $T(z)$'s. The output of this heuristic has length of at most $\frac{t_2+t_3}{2}$ [17].

Before we describe the Berman/Ramaiyer heuristic (BR) [1], we have to look closer at the way how to obtain MST($D \cup Z(z)$) from M = MST(D). Say that $Z(z)$ = $\{e_1, e_2\}$. When e_1 is inserted, the longest edge $H(e_1, D)$ in the path joining the ends of e_1 with cost $h(e_1, D)$ is removed from M. Then we do the same with e_2.

The idea of BR is to make the initial choices (performed in the *Evaluation Phase*) tentative, and to check later (in the *Selection Phase*) for better alternatives.

Evaluation Phase. Initially, M = MST(D). For every triple z considered, find $g = g(z, M)$. If $g \leq 0$, z is simply discarded. Otherwise we do the following for every edge e of some spanning tree of z: find $e' = H(e, M)$ and $c = h(e, M)$, make the cost of e equal to $c - g$, replace in M edge e' with e, put e in a set B_{new} and e' in B_{old}. Once this spanning tree of z is processed, we place the tuple $< z, B_{new}, B_{old} >$ on a Stack (for the future inspection in the second phase). Repeat this while there are triples with positive gain. For later analysis, we define t_3^* to be the length of M at this point, continue the process with quadruples and get t_4^* as the final length of M.

Selection Phase. We initialize $D = M$. Then we repeatedly pop $< z, B_{new}, B_{old} >$ from the *Stack*, and insert B_{old} to D. If $B_{new} \subseteq$MST(D), then such z is a 'good'

one (i.e. we use $T(z)$ in the final output), otherwise we remove all edges of B_{new} from D.

All 'good' quadruples and triples with the rest of MST-edges form the output Steiner tree of the Berman/Ramaiyer heuristic. Its length is at most

$$\frac{t_2}{2} + \frac{t_3^*}{6} + \frac{t_4^*}{3}. \tag{1}$$

We call this version of BR, that considers triples and quadruples, BR4, to distinguish from the shorter version, BR3, that considers only triples. It is easy to see that any estimate for BR3 holds for GA. The length of the output of BR3 is at most

$$\frac{t_2}{2} + \frac{t_3^*}{2} \tag{2}$$

How can we bound the values (1) and (2)? Obviously $t_3^* \leq t_3$ and $t_4^* \leq t_4$. So (1) and (2) is at most $\frac{11}{8}s$ and $\frac{97}{72}s$, respectively [1]. In the next section we develop tools that provide bounds for the linear combinations of t_2, t_3^* and t_4^* that are far better than the linear combinations of the bounds above. For example, while bounds $t_2 \leq \frac{3}{2}s$, $t_3 \leq \frac{5}{4}$ and $t_4 \leq \frac{8}{7}s$ are tight, we will show that $t_2 + t_3^* \leq \frac{21}{16}s$ and $t_2 + t_4 \leq \frac{5}{2}s$.

3 Double Covers and Spanning Trees

Let C be a union of pairs $C2$ of terminals (edges) and of triples $C3$ of terminals respectively. We say that a set of edges E is implied by C if it contains all edges of $C2$ and for each triple $z \in C3$ it contains two distinct edges contained in z. Both $C3$ and E are multisets, where some elements may belong "twice". We say that C is a *double cover* of the set of terminals S if every set of edges implied by C is a multiset union of two spanning trees of S (i.e., if an edge belongs to both trees, it has to belong to E twice). The total length of C is $d(C) = \sum_{x \in C} d(x)$, where $d(x)$ denotes the length of the Steiner minimum tree of x.

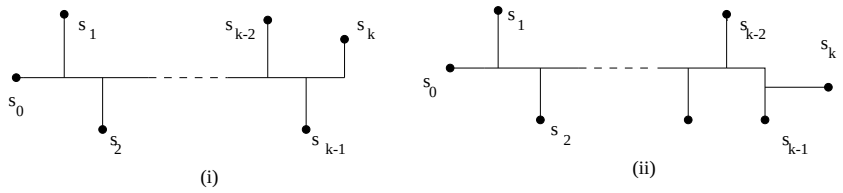

Fig. 1. Two shapes of a full Steiner component

Lemma 1 *If C is a double cover of S, then $d(C) \geq 2t_3^*$. [2]*

Our next tool is a pair of sufficiently short spanning trees. Hwang [10] proved that there is a Steiner minimum tree where every full component has one of the shapes shown in Fig. 1. Let $a_1, \ldots, a_k$ and $b_0 = 0, b_1, \ldots, b_k$ be the lengths of horizontal and vertical lines of a full Steiner component K with terminals $s_0, \ldots, s_k$. The horizontal lines form a *spine* of K. Moreover, in case (i) $b_k < b_{k-2}$ holds. In case (ii) assume that $b_k = 0$. Consider the sequences $b_0, b_1, b_3, \ldots, b_{2i+1}, \ldots$ and $b_0, b_2, \ldots, b_{2i}, \ldots$. Let

$$b_{h(0)} = b_0, b_{h(1)}, \ldots, b_{h(p+1)} = b_k \tag{3}$$

be the sequence of local minima of these sequences, i.e. $b_{h(j)-2} \geq b_{h(j)} < b_{h(j)+2}$. If $h(p) = k - 1$, we exclude the member $b_{h(p)}$ from (3). For the case of $h(j+1) = h(j) + 1$, $(j = 1, \ldots, p-1)$, we exclude arbitrarily either $b_{h(j+1)}$ or $b_{h(j)}$. So, we get $h(j+1) - h(j) \geq 3$. The elements of the refined sequence (3) are called *hooks*. Further we assume that a full Steiner tree K nontrivially contains at least 4 terminals ($k \geq 4$). A *Steiner segment* K_j is a part of a full Steiner component bounded by two sequential hook terminals $s_{h(j)}$, $s_{h(j+1)}$. So two neighbouring Steiner segments have a common hook. K_j contains the two *furthest* terminals below and above the spine. Denote the index of the *first* of them by $f(j)$ and the last by $l(j)$, $(f(j) < l(j))$.

Lemma 2 *Let K_j be a Steiner segment. For the terminals $s_{h(j)}, \ldots, s_{h(j+1)}$ there are two spanning trees Top_j and Bot_j such that*

$$d(Top_j) + d(Bot_j) = 3d(K_j) - b_{h(j)} - b_{h(j+1)} - Rest_j; \tag{4}$$

$Rest_j$ sums the lengths of the thin drawn Steiner tree edges of both trees in Fig. 2.

Proof. Equation (4) can be easily drawn from Fig. 2. The Steiner segment edges are partitioned into three parts: the lengths of thick lines are counted twice, thin lines are counted once, and some hooks (dashed) do not appear at all.

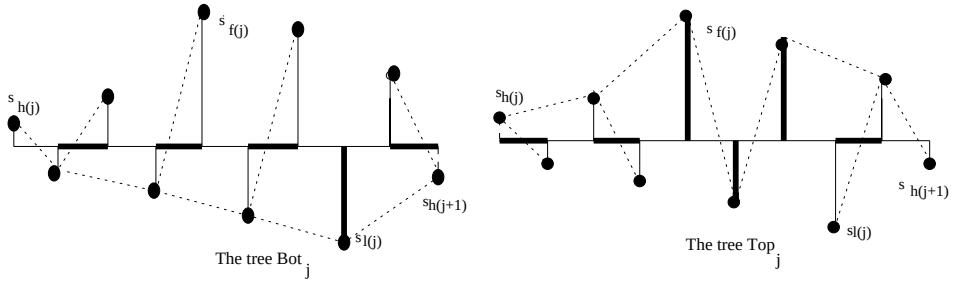

Fig. 2. Two spanning trees

Lemma 3 *For terminals $s_0, \ldots, s_k$, there are two spanning trees Top and Bot s.t.*

$$d(Top) + d(Bot) \leq 3d(K) - \sum_{j=1}^{p} b_{h(j)} - \sum_{j=1}^{p} Rest_j. \tag{5}$$

Proof. Since $d(K) = \sum_{i=0}^{p} d(K_j) - \sum_{j=1}^{p} b_{h(j)}$ (5) is satisfied. $\diamondsuit$

Corollary 1 *(Hwang [10]) For any instance of RSP, $2t_2 \leq 3s$.* $\diamondsuit$

4 A Performance Ratio of BR3 and GA

Theorem 1 *For any instance of the rectilinear Steiner tree problem,*

$$6t_2 + 8t_3^* \leq 18s. \tag{6}$$

Proof. (Sketch.) It is sufficient to prove inequality (6) for a full Steiner tree K. We will use three pairs of spanning trees *Top* and *Bot* and construct four double covers C^α, $\alpha = 1, \ldots, 4$ such that

$$3d(Top) + 3d(Bot) + \sum_{\alpha=1}^{4} d(C^\alpha) \leq 18s. \tag{7}$$

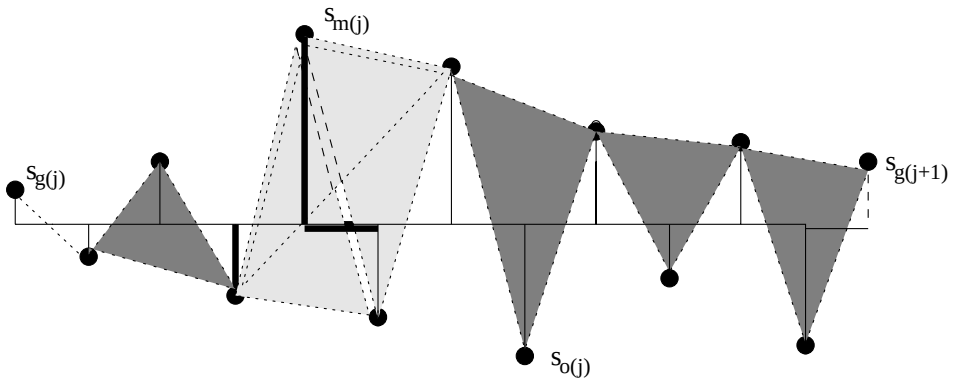

Fig. 3. The double cover for a segment B_j

At first we partition K into some other segments $B_j, j = 0, \ldots, r$, each a union of some Steiner segments. In this partition we must save the right hook of each segment. The first segment B_1 of K is bounded by $s_{g(0)} = s_0$ and $s_{g(1)}$ which is the first minimum at the same side of the spine as $s_{m(0)} = s_{f(0)}$, i.e. $s_{m(1)} \geq \ldots \geq s_{g(1)-2} \geq s_{g(1)} < s_{g(1)+2}$. Similarly, let $s_{m(2)}$ be the first furthest terminal after $s_{g(1)}$. Then B_2 is bounded by $s_{g(1)}$ and $s_{g(2)}$ which is the first minimum at the same side of the spine as $s_{m(2)}$, and so on. Only the last time, if the first minimum at the same side of the spine has index $k-1$, we bound the last segment B_r by s_k instead of s_{k-1}. So $g(r+1) = b_k$. Note that $g(j+1) - m(j)$ is even for $j = 0, \ldots, r$.

If a set of triples and pairs C has a cut terminal and both parts are double covers of their terminal sets, then C is a double cover of the whole terminal set. Therefore, we may construct double covers C_j for each segment B_j separately. Moreover, a set of three triples is a double cover of four terminals. Each C_j consists of a set of three triples for some four terminals, a set of doubled triples (i.e. pairs of the same triple) and may be a double edge $(s_{g(j)}, s_{g(j)+1})$ (if $m(j) - g(j)$ is even).

The double cover C_1 is shown on Fig. 3. Dark triples are doubled and light triples cover four terminals; thick Steiner tree edges are those which lengths participate in $d(C_1)$ three times, thin Steiner tree edges participate in $d(C_1)$ only twice and the dashed hook does not participate in $d(C_1)$ at all. The double cover C_2 differs only in light triples: they have the common end in $s_{m(j)-1}$ instead of $s_{m(j)}$. Similarly for every B_j we construct the next pair of double covers around the opposite furthest terminal, called $s_{o(j)}$ of a Steiner segment which begins at the same terminal as B_j.

The main point of the proof is that the sets of thick edges for different double covers intersect in edges defining the term $Rest_j$ in (5). $\diamondsuit$

Corollary 2 *For any instance of RSP,* $8t_2 + 8t_3^* \leq 21s$. $\diamondsuit$

Corollary 3 *The performance guarantee of GA and BR3 is at most* $\frac{21}{16} = 1.3125$.$\diamondsuit$

To get a lower bound for the performance ratio, we construct a series of instances of the rectilinear Steiner tree problem for which the approximation ratio comes arbitrarily close to 1.3. The terminals are placed on the following coordinates:

$a_i = (4^i, 0)$, $i = 0, \ldots, k$; $b_i = (2 \cdot 4^{i-1}, -4^{i-1})$, $c_i = (3 \cdot 4^{i-1}, 4^{i-1})$, $i = 1, \ldots, k$;

$a'_i = (0, 4^i)$, $i = 0, \ldots, k$; $b'_i = (-4^{i-1}, 2 \cdot 4^{i-1})$, $c'_i = (4^{i-1}, 3 \cdot 4^{i-1})$, $i = 1, \ldots, k$.

The length of the Steiner minimum tree $s(k)$ is $\frac{10}{3}(4^k - 1) + 2$, $t_2(k) = 14\frac{4^k-1}{3} + 2$ and the total gain obtained by GA is $G(k) = \frac{1}{3}(4^k - 1)$. So, the length of the output tree of GA is $Gr(k) = t_2(k) - G(k) = \frac{13}{3}(4^k - 1) + 2$ and $\lim_{k\to\infty} Gr(k)/s(k) = 1.3$.

Remark 1 *The performance guarantee of GA and BR3 cannot be less than 1.3.*

5 A Performance Ratio of BR4

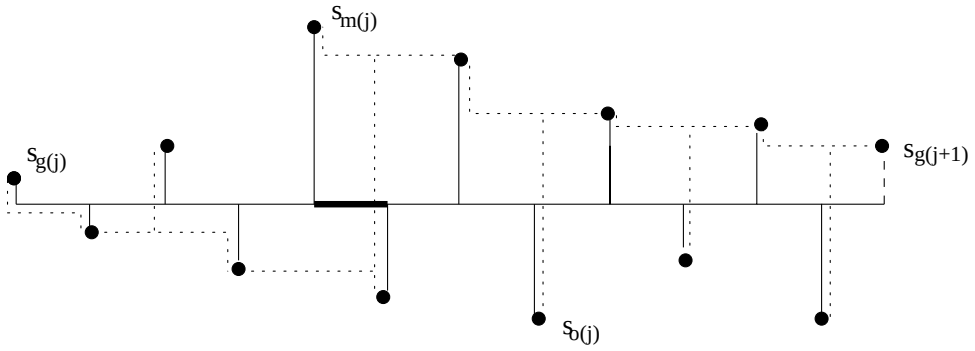

Fig. 4. The 4-restricted tree F for a segment B_j

Theorem 2 *For any instance of RSP,* $2t_2 + 2t_4 \leq 5s$.

Proof.(Sketch.) It is sufficient to prove Theorem 2 for a full Steiner component K. Similarly to the proof of Theorem 1, we construct two 4-restricted Steiner trees F and L such that $d(Top) + d(Bot) + d(F) + d(L) \leq 5d(K)$. We use the same

partition of K into the segments $B_j, j = 0, \ldots, r$. Recall that $g(j+1) - m(j)$ is even and the set of terminals $\{s_{m(j)}, j = 1, \ldots, r\}$ is a subset of the set $\{s_{f(j)}, j = 0, \ldots, p\}$ of the first furthest terminals in Steiner segments.

The 4-restricted Steiner tree F_j is constructed for each segment B_j (Fig. 4). It consists of a quadruple for $s_{m(j)}$, triples and a possible edge at the left end. Dotted lines denote the edges of F_j and, as above, thick Steiner tree edges are counted twice and thin Steiner tree edges only once in the length of F_j. We are confident that the right (dashed) hook is saved. Similarly, the tree L_j has a quadruple for $s_{o(j)}$.

The crucial point of the proof is that the sets of thick edges for both trees F and L belong to $Rest_j$ from (5).$\diamondsuit$

Let *quadruple* mean a Steiner minimum tree for a set of four terminals. From Fig. 1 we know that there are only two different shapes for quadruples:*normal* (i) and *cross* (ii). Denote the terminals of the quadruple with u, v, w and z from the left to the right. We call a cross quadruple *long*, if $a2 \geq 2\min\{a1, a3, a4, a5\}$, (*short* otherwise), where $a2$ is the rectilinear distance between the Steiner points and $a1, a3, a4, a5$ are the distances between the terminals and their adjacent Steiner points.

Lemma 4 *If a Steiner minimum tree for four points is a long cross quadruple, then*

$$2t_2 + 2t_3 \leq 5s.$$

Proof. Let $a_1 = \min\{a1, a3, a4, a5\}$, then a Steiner tree for the set $\{u, w, z\}$ together with the edge (u, v) have the length $s + a1 \leq t_3$. Note, that $t_2 \leq s + a1 + a4$ and $t_2 \leq s + a3 + a5$. Therefore, $2t_3 + 2t_2 = 2s + 2a1 + 2s + a1 + a3 + a4 + a5 \leq 5s$. $\diamondsuit$

Let t'_4 be the length of the shortest 4-restricted Steiner tree without any long cross quadruples. Note that in the proof of Theorem 2 the quadruples of 4-restricted trees are normal (Fig. 4). Therefore, a cross quadruple may appear only as a full component. Thus, bound (1), Theorem 2, Corollary 2, and Lemma 4 imply

Theorem 3 *BR4 without long cross quadruples has a performance ratio of* $61/48 \approx 1.271$.$\diamondsuit$

6 How Many Quadruples Have to be Considered?

To keep the time complexity small we have to restrict the number of candidates where we look for our triples and quadruples. We show that it is sufficient to regard only a linear number of each triples and quadruples. In [5] this fact is shown for triples and there is given an algorithm to construct these triples in time $O(n \cdot \log^2 n)$. So it remains to show that a linear number of quadruples is enough for our algorithm.

For a point p, p.x and p.y denote its x- and y-coordinate respectively. At first we only consider normal quadruples. The segment containing the Steiner points is called *Steiner chain*, the unique terminal on the Steiner chain is the *root* of the quadruple. A quadruple is called a *left rooted quadruple*, if the Steiner chain is a horizontal line and the root is its left end. Otherwise we call a quadruple top-, right- or bottom rooted. So we have eight kinds of normal stars: For each possible root position there are two possibilities at which side of the Steiner chain is only one point (w) and at which side are the other two (v,z). Since left- and right rooted quadruples have the same length it suffices to consider only left rooted quadruples, for which the condition $v.x - u.x < z.x - w.x$ holds. For a left rooted quadruple, v is called the *top point*, w the *bottom point* and z the *right point*. Also note that $z.y < v.y$ (cf. Fig. 1).

A quadruple is called a *tree quadruple* if the following condition holds: An MST for the point set $V \cup \{c_1, c_2\}$ where c_1 and c_2 are the Steiner points of the quadruple, contains the edges $(u, c_1), (v, c_1), (c_1, c_2), (w, c_2)$ and (z, c_2). ¿From results in [1] and [5] we know that we only have to consider tree quadruples where the rectangles defined by the point sets {u,v}, {v,w} and {w,z} do not contain any other points.

Lemma 5 *For a given left root u and a top point v there is at most one bottom point w. If the bottom point w is also given, there are at most two right points z.* $\diamondsuit$

For a point v the union of the triangles T_1, T_2 and T_3 is called *butterfly(v)*, where these triangles are defined in Fig. 5. Triangle T_4 is the *right wing* of the butterfly.

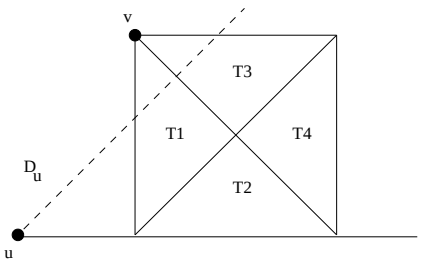

Fig. 5: The butterfly of the top point v

Lemma 6 *There is a 4-restricted Steiner minimum tree T where at each used quadruple the top point has an empty butterfly.*

Proof. Let Δ_1 be $T_1 \cup T_2$, $\Delta_2 = T_1 \cup T_3$. Δ_1 has to be empty, because otherwise (v, c_1) could not be part of an MST of $V \cup \{c_1, c_2\}$; so let p be a point in $\Delta_2 \backslash \Delta_1$

(i.e. T_3); let $x \in \{u, v, w, z\}$ be the first point on the path from p to the quadruple in T. If $x \in \{u, w, z\}$ we can add the edge (v, p), delete the edge (v, c_1) and get a 4-restricted Steiner tree with smaller cost, a contradiction to the optimality of T. So $x = v$: Case(i): $z.x < v.x$: Then $dist(p, z) < dist(v, c_1)$, because z does not lie in Δ_1. So we get a better 4-restricted Steiner tree by adding the edge (p, z) and deleting (v, c_1). Case(ii): $z.x > v.x$: Here we add a vertical edge from p down to the Steiner chain, delete the edge (v, c_1) and again we get a better 4-restricted Steiner tree. $\diamondsuit$

¿From Lemma 5 follows that for given u and v the number of tree quadruples is at most four. So we try to bound the number of top points with empty butterflies. Let D_u be the 45°-diagonal through u with positive gradient (Fig. 5).

Lemma 7 *For a given u there is at most one top point v with empty butterfly at the left side of D_u.*

Proof. Let v and v' two such points with $v.x < v'.x$. Then v' has to lie below v, because the rectangle defined by u and v' has to be empty. But then v' lies in the butterfly of v, so this butterfly is not empty. $\diamondsuit$

So we can restrict our search for other top points to the region at the right side of D_u. Let v be the leftmost candidate, i.e. v is at the right side of D_u, the rectangle defined by u and v, and the butterfly of v are empty and v is the leftmost point with these characteristics. The tree star condition for quadruples implies that all other candidates for top points have to be in the right wing of the butterfly of v.

Lemma 8 *There are at most two possible top points in the right wing of the butterfly of v.*

So there are at most four top points for the left root u: one at the left of D_u, and at its right v, and two claimed in Lemma 8. For a given pair (u, v) the bottom point is unique and we have at most two right points (Lemma 5), so we have eight tree quadruples with root u of normal shape and since there are eight possibilities for the state of the quadruple, the number of normal quadruples is at most $8 \times 8n = 64n$.

Lemma 9 *At most $64n$ normal quadruples are necessary for the approximation claimed in Theorem 3.*

The number of necessary cross quadruples can be bounded by $32n$ using similar techniques and the tree star property [2].

Theorem 4 *At most $96n$ quadruples are necessary for the approximation claimed in Theorem 3.*

7 Computation of Quadruples

To compute the normal quadruples we need a fast method to find top points v with empty butterfly. But the size of the butterfly of a top point depends of its left root, so checking the butterflies of all points for every root would require quadratic time. Therefore we use the following construction: For every point v, let p_v be the highest point (maximal y-coordinate) in the region at the right of the vertical line through v and at the left of the 45°-diagonal through v with negative gradient. The horizontal line through p_v together with this region defines the triangle Δ_1. Let q_v be the point with smallest rectilinear distance to v lying right below v (i.e. $v.x < q_v.x$ and $v.y > q_v.y$). Then Δ_2, the triangle defined by the vertical and horizontal lines through v and the 45°-diagonal through q_v with positive gradient is empty and it is maximal among all such triangles. The length of the leg of the smaller triangle denotes the size of the largest empty butterfly of v.
Now we draw for every point v a vertical line segment, starting at v in the bottom direction, with the length of the largest empty butterfly of v. Regarding a left root u, we only have to compute intersections of the horizontal line through u with such vertical line segments; the corresponding points are those having an empty butterfly with respect to u.

With a similar construction we can handle empty rectangles. Following the proof of Lemma 8, we can find the corresponding four top point candidates for a given left root u. The implementation of these operations can be performed in a quite standard way in time $O(n \log^2 n)$ using priority search trees to find the maximal empty butterflies and segment trees to compute the candidates for top points.

The necessary cross quadruples can be found in time $O(n \log^2 n)$ [2]. After computing the triples and quadruples, we can apply the algorithm of Berman/Ramaiyer for 4-restricted Steiner trees and get an approximation as stated in Theorem 3. The operations of the algorithm itself are mainly update-operations on a minimum spanning tree, which can be performed in time $O(\sqrt{n})$ each [6]. We conclude:

Theorem 5 *For any rectilinear Steiner tree problem an 1.271-approximation can be found in time* $O(n^{3/2})$.

8 A Faster Approximation

In this section we give a generalization of the fast parameterized version of BR presented in [5]. The main idea is, not to update the whole configuration after each step, but to sort triples and later quadruples according to their gain, and to evaluate those stars according to this sorting, even if it changed meanwhile. Stars are considered to be good, if their gain lies close to the optimal gain (in the range of $m/(m+1)$). We refuse to take triples (quadruples) with smaller gains, or if they would destroy planarity or if they contain artificial edges.

We use the following notations: g_i is the gain of a star (triple or quadruple) at the beginning of a phase, a_i its actual gain at the time of its treatment and a_i' its actual gain without using artificial edges, that means edges which were created in the same phase. B_{old} and B_{new} are the sets of edges from Section 2. Importantly, we have several possibilities to connect the terminals of a quadruple, i.e. e_1 and e_2 are not strictly fixed. We choose these new edges such that the structure remains planar.

The main difference to the algorithm described in [5] is that we now have to run a Selection Phase which we could avoid in the case of considering only triples. For the running time the only problem is maintaining an MST in the Selection Phase.

But since the structure always stays planar we can use the data structure introduced in [4] that allows maintaining an MST in time $O(\log n)$ per step. So we run BR4 with an involved version of the Evaluation Phase:

Phase 1: (triple insertion)
  $E := \{d(u,v) : u, v \in S\}$; $M := \mathrm{MST}(E)$;
  Compute the triples τ_i^3 and store them;
  repeat $r \cdot m \cdot \log n$ times:
    Sort the triples due to their gains in decreasing order;
    compute j: $g_j \geq g_1 \frac{m}{m+1}, g_{j+1} < g_1 \frac{m}{m+1}$;
    for $i := 1$ **to** j **do**
      if $a_i' \geq g_1 \frac{m}{m+1}$ **and** τ_i^3 is planar
      then $M := M \setminus B_{old}(\tau_i^3) \cup B_{new}(\tau_i^3)$;
        push $(\tau_i^3, B_{old}(\tau_i^3), B_{new}(\tau_i^3))$ on a stack;
Phase 2: (quadruple insertion)
  Repeat Phase 1 for quadruples τ_i^4;
  all edges in M at the beginning of Phase 2 are said to be original for this phase.

The rest of this section is devoted to the proof of

Theorem 6 *The algorithm computes in time $O(r \cdot m \cdot n \log^2 n)$ a Steiner tree for a given set S of points, $|S| = n$. Its approximation ratio is at most $\frac{61}{48} + \frac{7}{4m} + \frac{1}{(2 \cdot 32^{r-2})}$.*

Now we have to count how much we lose compared to the analysis of section 5 by

a) finishing the Evaluation Phases after $O(\log n)$ rounds, i.e. although there might still be some stars with positive gain.

b) ignoring stars which would use artificial edges.

c) ignoring non-planar stars.

Phase 1 is the same as in the algorithm in [5], so we only analyse Phase 2.

The loss due to a) and b) is easily counted: With the same arguments as in the proofs of Lemma 7 and 8 in [5] we can state that the total loss here can be bounded by the fraction $\frac{1}{96^{r-2}} + \frac{3}{m}$ of the total gain. The constant 3 comes from the fact that every quadruple ignored because of artificial edges would decrease the total cost by three times its gain by decreasing the length of three edges.

Let *tuple* mean a triple or a quadruple. The planarity restriction requires to ignore every quadruple that would cross an original edge or a tuple already lying on the stack. Two tuples 'cross' or 'intersect' each other if there is a crossing between an artificial edge created by one tuple and an artificial edge created earlier by the other, and the tuples have at most one common point. If a quadruple and a triple have more than one point in common, we can avoid intersections by placing the new edges in such a way that they do not cross artificial edges of the triple. This can be done by never adding the 'diagonals' of the quadruple which are the only edges that may cross. Two quadruples having more than one point in common cause no problems, because the second one always uses an artificial edge created by the other.

Lemma 10 *There is a 4-restricted Steiner tree without intersections.*

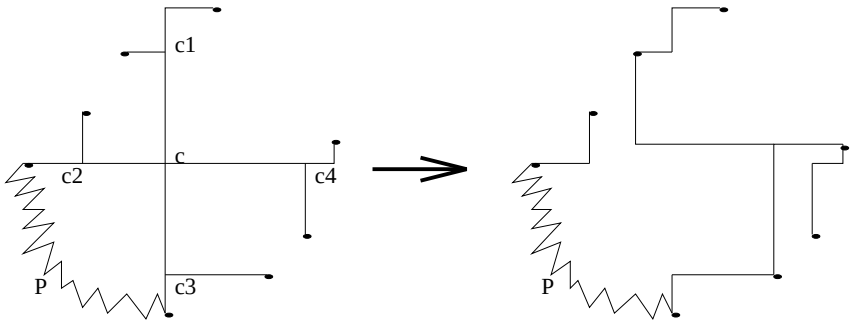

Fig. 6. Proof of Lemma 10: Triples are better than crossing quadruples

So the gain of all the quadruples ignored because of non-planarity can be bounded by the gains of all triples after the triple insertion phase.

Corollary 2 *After the quadruple insertion phase all quadruples with positive gain that were refused because of non-planarity together have a gain of at most* $\frac{1}{32^{r-2}} + \frac{4}{m}$.

Note that the planarity test can be done in time $O(\log n)$ by maintaining a planar map of the actual edge set where the edges incident to the same point are ordered in

lists according to their direction. Since every pair of edges that ever appear in the course of the algorithm has no crossing, we can maintain an MST in the Selection Phase in time $O(\log n)$ per step [4]. Thus the time bound of Theorem 6 is proved.

The total gain that could be realized is

$$(\frac{t_2}{2} - \frac{t_3^*}{6} - \frac{t_4}{3})(1 - (\frac{1}{32^{r-2}} + \frac{4}{m} + \frac{1}{96^{r-2}} + \frac{3}{m}))$$

and the performance ratio is at least

$$\frac{61}{48} + (\frac{t_2}{2} - \frac{t_3^*}{6} - \frac{t_4}{3})(\frac{7}{m} + \frac{2}{32^{r-2}}) \leq \frac{61}{48} + \frac{7}{4m} + \frac{1}{2 \cdot 32^{r-2}}.$$

Corollary 3 *We can achieve in time $O(n \log^3 n)$ an approximation ratio $61/48 + \log\log n / \log n$ and in time $O(n \log^2 n)$ a ratio $61/48 + \varepsilon$ for any $\varepsilon > 0$.*

9 Conclusion

In this paper refined analysis of the new approximation algorithms for rectilinear Steiner trees of Zelikovsky and Berman/Ramaiyer were presented. The introduction of new techniques for estimating the length of different classes of Steiner trees enables to considerably improve the best known approximation factors. Somewhat surprisingly, the same time bounds of $O(n \log^2 n)$ can be achieved as before [5], although we have to use much more involved methods.

This paper almost closes the first approach to improve the approximation ratio for rectilinear Steiner trees, bringing the ratio down from 3/2 [10], via 11/8 [17, 1] now to 61/48, which is very close to 5/4. The next task is to jump below the 5/4-barrier.

References

[1] Berman, P., Ramaiyer, V., *Improved approximations for the Steiner tree problem*, in *Proc. of 3rd ACM–SIAM Symp. on Discrete Algorithms (1992)*, pp. 325–334.

[2] Berman, P. Fößmeier, U., Karpinski, M., Kaufmann, M., and Zelikovsky, A., *Approaching the 5/4–Approximations for Rectilinear Steiner Trees*, Techn. Rep. WSI-94-6, Tübingen.

[3] Du, D. Zhang, Y., and Feng, Q., *On better heuristic for Euclidean Steiner minimum trees*, in *Proc. 32nd IEEE Symp. on Found. of Comp. Science (1991)*, pp. 431–439.

[4] Eppstein, D. et al., *Maintaining of a Minimum Spanning Forest in a Dynamic Planar Graph*, in *Proceedings 1st ACM–SIAM Symp. on Discrete Algorithms (1990)*, pp. 1–11.

[5] Fößmeier, F., Kaufmann, M., and Zelikovsky, A., *Faster Approximation Algorithms for the Rectilinear Steiner Tree Problem*, in *LNCS 762 (1993)*, p. 533–542.

[6] Frederickson, G., *Data Structures for On–Line Updating of Minimum Spanning Trees, with Applications*, in *SIAM J. Comp. 14 (1985)*, pp. 781-789.

[7] Garey, M.R., Johnson, D.S., *The Rectilinear Steiner Problem is NP–Complete*, in *SIAM J. Appl. Math. 32 (1977)*, pp. 826–834.

[8] Gilbert, E.N.,, Pollak, H.O., *Steiner Minimal Trees*, in *SIAM Appl. Math. 16 (1968)*, pp. 1–29.

[9] Hanan, M., *On Steiner's Problem with Rectilinear Distance*, in *SIAM J. Appl. Math. 14 (1966)*, pp. 255–265.

[10] Hwang, F.K., *On Steiner Minimal Trees with Rectilinear Distance*, in *SIAM J. Appl. Math. 30 (1976)*, pp. 104–114.

[11] Hwang, F.K., Richards, D.S., Winter, P., *The Steiner Tree Problem*, in *Annals of Disc. Math. 53 (1992)*.

[12] Karp, R.M., *Reducibility among combinatorial problems*, in *Complexity of Computer Computations*, (Miller and Thatcher (Eds.), Plenum Press, pp. 85–103, 1972.

[13] Korte, B., Prömel, H.J., and Steger, A., *Steiner Trees in VLSI–Layouts*, in *Paths, Flows and VLSI-Layout*, (Korte et al. (Eds.)), Springer–Verlag, 1990.

[14] Lengauer, Th., *Combinatorial Algorithms for Integrated Circuit Layout*, John Wiley, 1990.

[15] Richards, D., *Fast Heuristic Algorithms for Rectilinear Steiner Trees*, in *Algorithmica 4 (1989)*, pp. 191–207.

[16] Takahashi, H., Matsuyama, A., *An approximate solution for the Steiner problem in graphs*, in *Math. Japonica 24 (1980)*, pp. 573–577.

[17] Zelikovsky, A.Z., *An 11/8-approximation Algorithm for the Steiner Problem on Networks with Rectilinear Distance*, in *Coll. Math. Soc. J. Bolyai 60 (1992)*, pp. 733–745.

[18] Zelikovsky, A.Z., *A Faster Approximation Algorithm for the Steiner Tree Problem in Graphs*, in *Inf. Process. Lett. 46 (1993)*, pp. 79–83.