Geometric Steiner Trees

From the book: "Optimal Interconnection Trees in the Plane"

By Marcus Brazil and Martin Zachariasen

Part 5: Rectilinear Steiner Trees

Marcus Brazil

2015

Part 5: Rectilinear Steiner Trees

The Rectilinear Steiner Tree problem

In this Section we consider the problem of constructing a network of minimum length interconnecting a given set of points in the Euclidean plane, where each edge of the network is composed of *horizontal* and *vertical* line segments. This problem has applications in chip design, in particular in the physical design of very-large-scale integration (VLSI) circuits.

For any two points p and q in the plane, the minimum length of a path between them composed of horizontal and vertical line segments defines a norm, known as the ℓ_1 *norm* (or ℓ_1 *distance*). If $p = (p_x, p_y)$ and $q = (q_x, q_y)$ in the plane, their ℓ_1 *distance* is $|pq|_1 = |p_x - q_x| + |p_y - q_y|$, that is, the sum of distances in each of the two dimensions. The ℓ_1 distance is also called the *rectilinear* or *Manhattan* or *taxicab* distance.

The Rectilinear Steiner Tree problem

Rectilinear Steiner tree problem in the plane

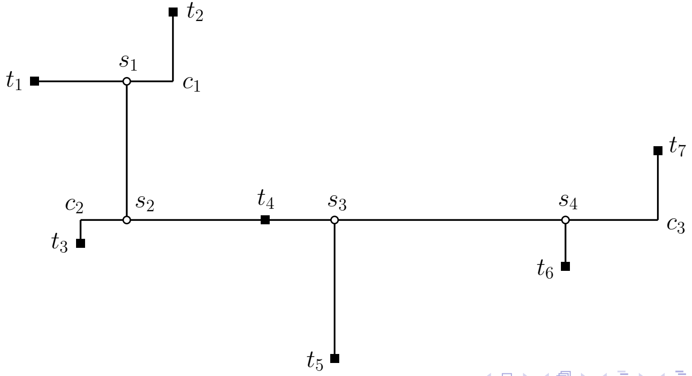
Given: A set of points $N = \{t_1, \ldots, t_n\}$ lying in the plane.
Find: A geometric network $T = (V(T), E(T))$, such that $N \subseteq V(T)$, and such that $|T|_1 := \sum_{e \in E(T)} |e|_1$ is minimised.

A solution to this problem is always a tree, and is referred to as a *minimum rectilinear Steiner tree*. As before, the given points in N are denoted *terminals*, and the possible points in $V(T) \setminus N$ are called *Steiner points*.

In this Section we always consider a minimum rectilinear Steiner tree as being embedded in the Euclidean plane using line segments that are horizontal or vertical only — these are called the *legal directions* for the tree. This means that all lengths can be measured using the Euclidean metric.

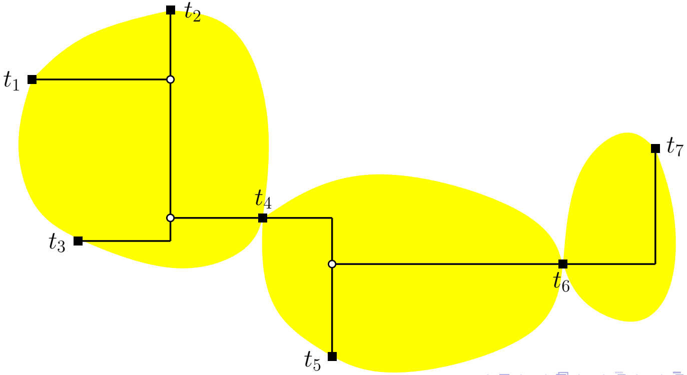
Some basic definitions

Consider a minimum rectilinear Steiner tree $T = (V(T), E(T))$ for a given terminal set N. The node set $V(T)$ contains all elements of N and some additional Steiner points. We can assume (by Theorem 30, for example) that all Steiner points have degree 3 or 4. A Steiner point of degree 3 is called a *T-point*, and a Steiner point of degree 4 is called a *cross*.



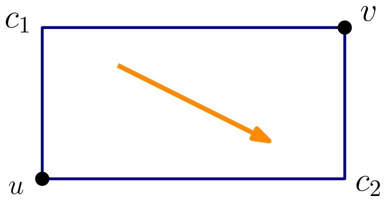
Some basic definitions - full components

As before, a rectilinear Steiner tree in which every terminal has degree 1 is called a *full* rectilinear Steiner tree. Every rectilinear Steiner tree is a union of *full components* meeting only at terminals. Such a tree is said to be *fulsome* if it has the maximum possible number of full components amongst all rectilinear Steiner trees with the same length.



Some basic definitions - edges

The edge set $E(T)$ consists of edges that connect pairs of nodes u and v by shortest rectilinear paths. The edge (u, v) is a *straight edge* if uv is either a horizontal or a vertical line segment; otherwise, (u, v) is a *bent edge*. A bent edge can be assumed to consist of a horizontal line segment and a vertical line segment that meet at a *corner point*. For any bent edge (u, v) there are two possible minimum length embeddings that contain a single corner point. We describe the process of moving from one of these embeddings to the other as a *flip*.



Some basic definitions - complete corner

A *line of segments* is a sequence of one or more adjacent, collinear segments with no terminal nodes sharing two adjacent segments (however, the endpoints of the line may be terminals).

Definitions: Complete line, complete corner

A *complete line* is a line of segments of maximal length; it is not properly contained in any other line of segments. Any corner point c is an endpoint of two complete lines, one in each of the two perpendicular directions given by the incident segments. Let t and t' be the other endpoints of the complete lines incident to c. The pair of complete lines (ct, ct') is called a *complete corner* located at c; ct and ct' are the *legs* of the complete corner.

An example of a complete corner is (c_3t_4, c_3t_7) in the figure on Slide 5.

Properties of Steiner points

Lemma 35

Let s be a Steiner point in a minimum rectilinear Steiner tree T. Then the following properties are true:

- The edges incident to s cannot overlap with each other for any embedding of the edges.
- If s is a cross, then all edges incident to s are straight edges.
- If s is a T-point, then at most one edge incident to s is a bent edge.
- For any straight edge (s, u) incident to s there exists another straight edge (s, v) incident to s and perpendicular to (s, u).

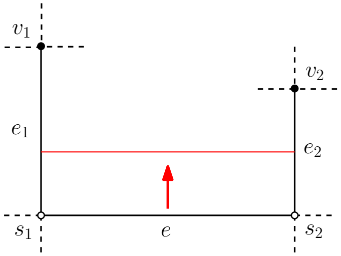
The proof of this lemma is left as an exercise (see the Problem Sheet).

Properties of Steiner points

Lemma 36: Rectilinear Sliding lemma

Let $e = (s_1, s_2)$ be a straight edge connecting two Steiner points s_1 and s_2 in a minimum rectilinear Steiner tree T. Let $e_1 = (s_1, v_1)$ be the next edge incident to s_1 travelling counter-clockwise from e, and let $e_2 = (s_2, v_2)$ be the next edge incident to s_2 travelling clockwise from e. Suppose e_1 and e_2 are straight edges, perpendicular to e, and located on the same side of the line through e. Then T is not fulsome.

This result is a direct corollary of Lemma 33, the Sliding Lemma.



Properties of Steiner points

There are three useful and straightforward consequences of the previous two lemmas.

Lemma 37: A cross has only terminals as neighbours

Let s be a cross in a fulsome minimum rectilinear Steiner tree T. Then the neighbours of s are terminals.

Proof: The four edges incident to s must be straight edges (by Lemma 35). Assume that one of the neighbours of s, denoted by u, is a Steiner point. By Lemma 35, Steiner point u has an incident straight edge (u, x) that is perpendicular to (s, u). Now, edge (s, u) fulfils the conditions of Lemma 36, contradicting the fulsomeness of T. **QED**

Properties of Steiner points

Lemma 38: A T-point ends in a terminal

Let s be a T-point in a fulsome minimum rectilinear Steiner tree T, and let u, v and w be the three neighbouring nodes of s. Suppose the edges (s, v) and (s, w) are collinear and straight edges. Then (s, u) is a straight edge and u is a terminal.

Proof: First we observe that (s, u) must be a straight edge. Assume that u is a Steiner point. Then we can use the same arguments as in the proof of Lemma 37 to show that T is not fulsome. **QED**

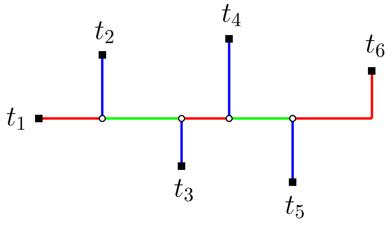
Lemma 39: A complete corner ends in terminals

Let c be a corner point for a bent edge in a fulsome minimum rectilinear Steiner tree T, and let (ct, ct') be the complete corner located at c. Then t and t' are both terminals of T.

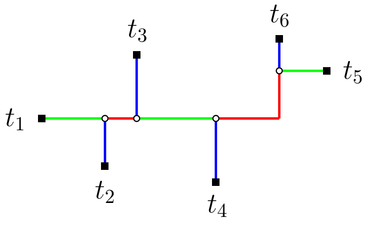
The proof of Lemma 39 is an exercise on the Problem sheet.

Canonical forms for full components

In general there may be infinitely many minimum rectilinear Steiner trees for a given set of terminals. It is therefore important from an algorithmic point of view to devise canonical forms for such trees that can be constructed efficiently. In this section we show that full and fulsome minimum rectilinear Steiner trees can be assumed to have a simple canonical form called the *Hwang form*.



Type (i)



Type (ii)

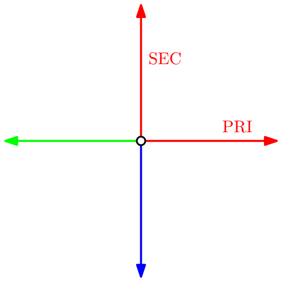
Direction sets

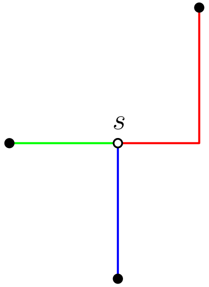
Definitions: Maximal Steiner configuration, direction set

For any Steiner configuration of degree 3 there is an associated set of legal directions, namely the legal directions used by all edges in the star (where directions are considered as oriented outward from the centre). A Steiner configuration $\mathcal{S}$ is said to be *maximal* if there exists no other Steiner configuration (for any set of terminals) that uses a strict superset of the legal directions used by $\mathcal{S}$. We define a *direction set* to be a set of legal directions used by a maximal Steiner configuration, listed in counter-clockwise order around the centre.

For the rectilinear metric, a Steiner point of degree 3 has at most one incident bent edge (Lemma 35). Therefore, a direction set has four directions: two *red* directions corresponding to the (possibly) bent edge, one *green* and one *blue* direction.

Direction sets





The two red directions are labelled the *exclusively primary* and *exclusively secondary* direction, respectively, in counter-clockwise order around the Steiner point; the blue and green edges can be considered to be both primary and secondary.

Properties of a full component

Let T be a full and fulsome minimum rectilinear Steiner tree. From the theory of fixed orientation Steiner trees (not covered in this course) we have the following results:

1. T uses a single direction set.
2. There exists a minimum rectilinear Steiner tree with the same terminals and topology as T that has at most one bent edge.

Definition: Caterpillar tree

Define a *caterpillar tree* to be a tree that has a central path $\mathcal{P}$ such that every node in the tree is either on $\mathcal{P}$ or is connected directly to $\mathcal{P}$.

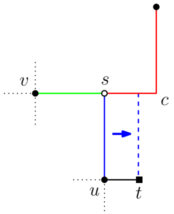
In other words, T is a caterpillar tree if and only if the subtree induced by the Steiner points of T is a path. Note that being a caterpillar is a property of the topology of the Steiner tree.

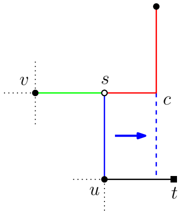
Rectilinear full components are caterpillar trees

Lemma 40

Let T be a full and fulsome minimum rectilinear Steiner tree spanning at least 3 terminals. Then the topology of T is a caterpillar tree where the central path is formed by all the Steiner points in T.

The key to the proof is to show that each Steiner point s in T is adjacent to at least one terminal. The only case that poses any difficulty is when s has an incident bent edge.





Clean subtrees of a full component.

Definition: Clean subtrees

A subtree of rectilinear Steiner tree is a *primary* subtree (or *secondary* subtree), if all edges are primary (respectively, secondary) edges. A subtree that is either primary or secondary is denoted a *clean* subtree.

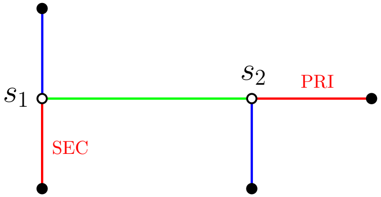
The importance of this definition comes from the following lemma.

Lemma 41: Subtrees consisting of straight edges only are clean

Let T be a full and fulsome minimum rectilinear Steiner tree. Consider any subtree T' of T that consists of straight edges only. Then T' is a clean subtree.

Proof of Lemma 41

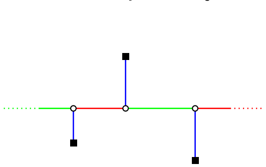
Assume that T' is not clean; that is, there exist both a primary red edge and a secondary red edge in T'. Then there must exist a pair of neighbouring Steiner points s_1 and s_2 in T', such that s_1 has an incident primary red edge and s_2 has an incident secondary red edge. It follows that exactly one of these red edges is collinear with (s_1, s_2); hence, either s_1 or s_2 is a T-point where the non-collinear incident edge does not end in a terminal.

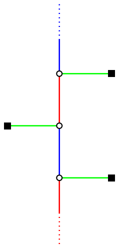


By Lemma 38, this gives a contradiction. **QED**

Canonical forms

It follows that for any given terminal set there exists a fulsome minimum rectilinear Steiner tree T such that any full component T' of T is either composed only of straight edges, and hence is clean, or contains a single bent edge e_ϕ such that $T' - e_\phi$ consists of two subtrees, T_1 and T_2, each of which is clean. Each of these subtrees has one of the following forms, depending on whether it is primary or secondary.





Furthermore, it is not difficult to show that we can assume one of the subtrees of $T' - e_\phi$, say T_1, is primary and the other is secondary.

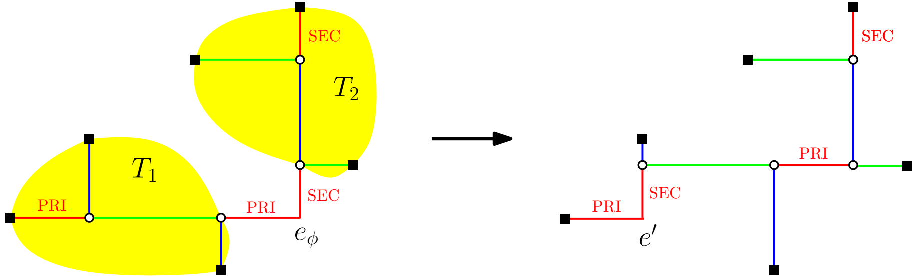
The Hwang Form

To show that this implies the Hwang form, we require the following.

Claim

One of the subtrees T_1 and T_2 spans at most 2 terminals.

The claim follows by contradiction, using Lemma 41.



The Hwang Form

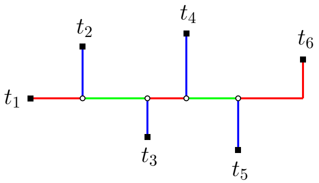
Theorem 42

There exists a minimum rectilinear Steiner tree for any terminal set such that every full component has the so-called *Hwang form*. In this form, every full component spanning k terminals consists of a complete corner with terminal endpoints referred to as the *root* t_1 and the *tip* t_k. The leg containing the root is called the *long* leg and the leg containing the tip is called the *short* leg of the complete corner. There are two main types (i) and (ii) (and two degenerate cases):

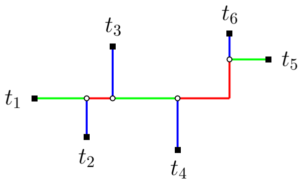
- Type (i) has $k-2$ alternating segments incident to the long leg and no segment incident to the short leg.
- Type (ii) has $k-3$ alternating segments incident to the long leg and one segment incident to the short leg.

The Hwang Form

Examples of the main types of the *Hwang form*.

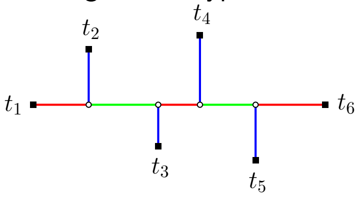


Type (i)

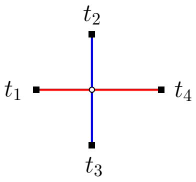


Type (ii)

Examples of the degenerate types.



Case (i')



Case (i")

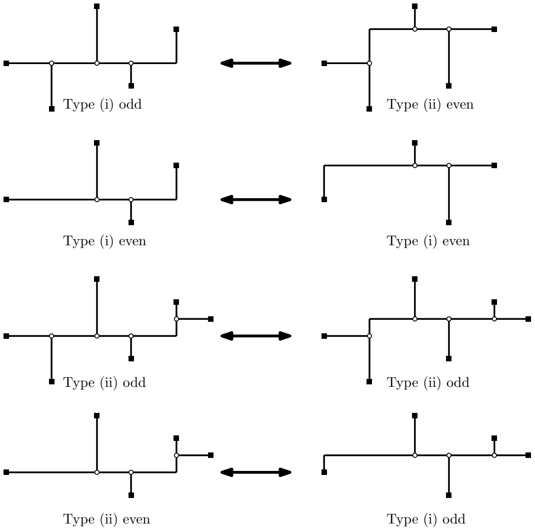
Corner-flipped Hwang forms

A non-degenerate full rectilinear Steiner tree T in Hwang form can be transformed into a *corner-flipped* version of itself. The corner-flipped version is obtained through a series of flips and slides.

In the corner-flipped version the direction of the long leg from the corner point becomes the opposite to what it is in the original tree (east versus west or north versus south). This observation implies that we only need to consider *two* rather than four directions of the long leg when enumerating Hwang form trees.

This is a useful observation when developing a GeoSteiner algorithm for the Rectilinear Steiner tree problem.

Corner-flipped Hwang forms



Empty regions

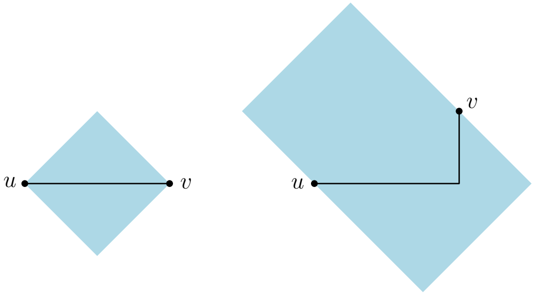
In this section we first study some necessary geometric conditions that must be satisfied by minimum rectilinear Steiner trees for a given terminal set N, independently of the topology of the tree. The conditions presented here are so-called *empty region* properties.

Recall that an empty region is a region in the plane that can be shown to be free of Steiner points and/or terminals if certain conditions are fulfilled. All the empty regions can be efficiently computed without having to first compute the complete minimum rectilinear Steiner tree, and they are therefore useful as efficient pruning conditions for eliminating non-feasible full components. As such they form an important part of the GeoSteiner algorithm.

Empty regions are also useful from a theoretical viewpoint for helping bound the number of candidate full components.

The lune property

Recall from Part 2 that a lune $\mathcal{L}(u, v)$ is defined as the set of points that are strictly within distance $|uv|_1$ of both u and v (where distance here is given by the rectilinear metric). Geometrically, a lune for edge (u, v) is the intersection of the interiors of the two ℓ_1 circles with radius $|uv|_1$ centred at u and v, respectively.



If (u, v) is an edge in a minimum rectilinear Steiner tree, then $\mathcal{L}(u, v)$ does not contain any points of the tree that do not lie on (u, v) (by Lemma 8).

The disjoint lunes property

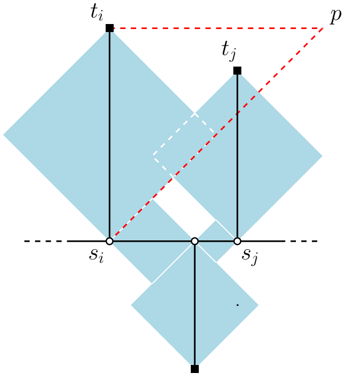
Consider a full component T in a minimum rectilinear Steiner tree. Not only are the lunes empty; they are also pairwise geometrically disjoint:

Lemma 43: Disjoint lunes property

Let T be a full and fulsome minimum rectilinear Steiner tree with Hwang form. For any pair of distinct segments uv and ws in T, we have $\mathcal{L}(u,v) \cap \mathcal{L}(w,z) = \emptyset$.

Proof: Consider any pair of distinct edges from T. If one of the edges is part of the backbone of T, then the corresponding lunes are clearly disjoint. If the two edges are not on the same side of the backbone, then they are also disjoint.
The only remaining case is when the edges are on the same side of the backbone. Let $s_i t_i$ and $s_j t_j$ be a pair of such incident segments; we assume without loss of generality that $s_i t_i$ and $s_j t_j$ are both vertical segments and that $|s_i t_i| \geq |s_j t_j|$.

The disjoint lunes property

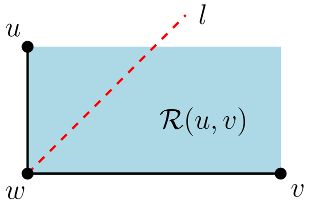


Suppose that $\mathcal{L}(s_i, t_i) \cap \mathcal{L}(s_j, t_j) \neq \emptyset$. let p be the point on the horizontal line through t_i on the same side of $s_i t_i$ as t_j such that $|t_i p| = |t_i s_i|$. Since the lunes overlap, it follows that t_j is in the interior of triangle $\triangle s_i t_i p$, from which a contradiction to minimality follows. **QED**

The above proof implies that incident segments on the same side of the backbone of a Hwang form full component cannot be too close to each other; ie, if $s_i t_i$ and $s_j t_j$ are two incident segments, then $|s_i s_j| \geq \min(|s_i t_i|, |s_j t_j|)$.

The empty rectangle property

Consider two perpendicular segments uw and wv meeting at a node w. Let $\mathcal{R}(u,v)$ be the interior of the axis-aligned rectangle with sides uw and wv.



Lemma 44: Empty Rectangle property

If uw and wv are perpendicular segments in a minimum rectilinear Steiner tree T, then $\mathcal{R}(u,v)$ contains no point of T.

Assume on the contrary that T contains a point $p \in \mathcal{R}(u,v)$. Let l be the line through w which bisects the perpendicular angle. The proof involves showing that there is a contradiction to minimality if p lies above, below or on l.

GeoSteiner Algorithm

We conclude this section with a high-level discussion of the GeoSteiner algorithm for the rectilinear metric.

Recall that the main idea of the GeoSteiner approach is to enumerate full components — or full Steiner trees (FSTs) — and then choose a subset of the generated FSTs to form a minimum rectilinear Steiner tree. The first phase is called *FST generation* and the second *FST concatenation*.

Compared to other metrics, FST generation for the rectilinear problem appears to be particularly fast in practice, due to the existence of the Hwang form for full components. As noted in Part 2, the FST concatenation phase of the algorithm is independent of the underlying metric, so is identical to that discussed in the last section of Part 2.

Hence, here we only need to consider the FST generation algorithm.

FST generation algorithm

Consider a Hwang form FST T in a minimum rectilinear Steiner tree, where T spans k terminals $t_1, t_2, \ldots, t_k$. T consists of a complete corner (or backbone) given by a root t_1 and a tip t_k; all other terminals spanned by T are connected directly to the backbone with straight line segments, and at most one is connected to the short leg.

Let c be the corner point of the single bent edge pq of T; if T has no bent edge, then let c be the midpoint of the straight edge pq incident to the tip t_k of T. Recall the definition of *branches* and *branch trees* from Part 2. If we cut edge pq at c, we obtain two branch trees having *straight edges only*: one with its root at p and the other with its root at q.

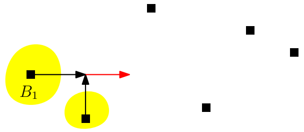
In the construction below, it suffices to consider the generation of individual branch trees rather than branches. For a branch of size greater than 1 with a given root and a given direction for the long leg we will see that under the construction scheme below the branch has a uniquely determined branch tree.

FST generation algorithm – combining branch trees

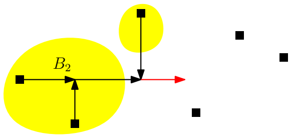
Branch trees of size 1 consist of a single terminal having a stem leaving in one of the legal directions. The Hwang form for rectilinear FSTs implies that we only need to consider combinations where one of the branch trees has size 1. Any full component T can be obtained by starting with a branch tree B_1 consisting of the root of T with a stem in the direction of the long leg. Then each of the terminals — in alternating fashion along the long leg — is iteratively added by combining the current branch tree with a size 1 branch tree that spans the added terminal. This results in a series of increasingly larger branch trees $B_1, B_2, \ldots$ as shown on the next slide.

A type (i) full component is obtained by combining a branch tree B_i with a size 1 branch tree that spans the tip of the full component; finally, a type (ii) full component can be obtained by attaching a terminal to the short leg of the constructed type (i) full component.

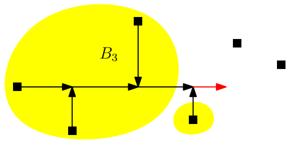
FST generation algorithm – combining branch trees



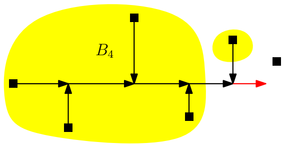
Combining B_1 with a size 1 branch tree



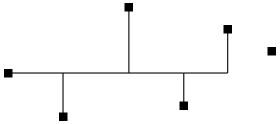
Combining B_2 with a size 1 branch tree



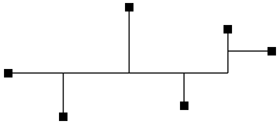
Combining B_3 with a size 1 branch tree



Combining B_4 with a size 1 branch tree



Type (i) full component



Type (ii) full component

FST generation algorithm

The simplified construction of branch trees for the rectilinear Steiner tree problem makes it possible to design a particularly efficient version of the general FST generation algorithm (from Part 2, slide 20).

- Instead of enumerating branch trees by increasing size, the main loop of the rectilinear FST generation algorithm iterates through all terminals $t \in N$. For a given terminal t, all feasible FSTs that have t as their *root* (in the Hwang form) are constructed. Each possible direction of the long leg is tried in turn; however, due to the existence of corner-flipped topologies only two perpendicular directions need to be tried (say, north and east).
- Branches and FSTs are pruned using the empty region properties and the BSD bound (which is defined as in the Euclidean case).
- A short list of FST terminal candidates for each FST root t can be constructed (using these pruning properties) as a preprocessing step.

FST generation algorithm – performance

For most randomly generated instances, the preprocessing phase (running time about $O(n \log n)$) dominates the total running time – the running time of the FST generation algorithm is close to being linear in n. This can be explained by the fact that very few terminals (less than 6 terminals for randomly generated problem instances with 10000 terminals) are added to the short list of terminals considered for a given root and direction.

A well-tuned implementation of this algorithm generates the FSTs for a randomly generated 1000-terminal instance in less than 0.1 second; for 10000 terminals the running time is less than 2 seconds, and more than half of the time is used for preprocessing. The number of FSTs surviving all tests is approximately $4n$.

The bottleneck of the GeoSteiner algorithm for the rectilinear Steiner tree problem is therefore the concatenation of FSTs.