Geometric Steiner Trees

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

By Marcus Brazil and Martin Zachariasen

Part 1: Euclidean Steiner Trees and local properties

Marcus Brazil

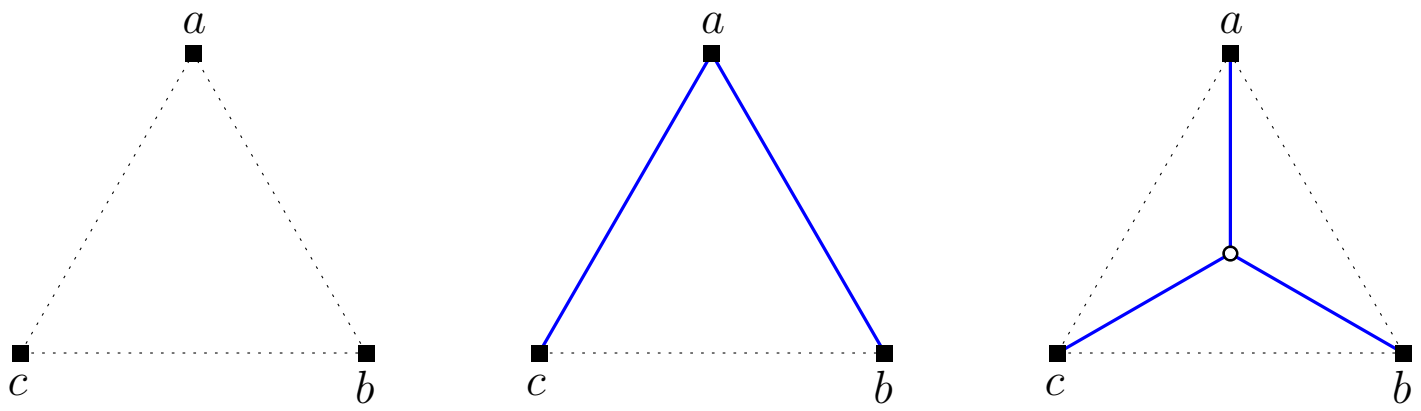
2015

Part 1: Euclidean Steiner Trees and local properties

1. The Fermat-Toricelli problem
2. The Steiner Tree problem
3. Steiner topologies and Steiner trees
4. The Melzak-Hwang algorithm

The Fermat-Toricelli problem

We begin with a simple example. Suppose that a, b and c are the vertices of an equilateral triangle in the plane with side length 1 as shown below.



The network on the right is the minimum length network where the possibility of including extra vertices is allowed.
This is an instance of a more general problem, posed by the 17th century mathematician, Pierre de Fermat, and first solved by the Italian physicist and mathematician Evangelista Torricelli

The Fermat-Toricelli problem

Fermat-Torricelli problem

Given: A set of three points $N = \{a, b, c\}$ lying in the plane.
Find: A point s such that the sum of Euclidean distances from s to a, b and c is minimised.

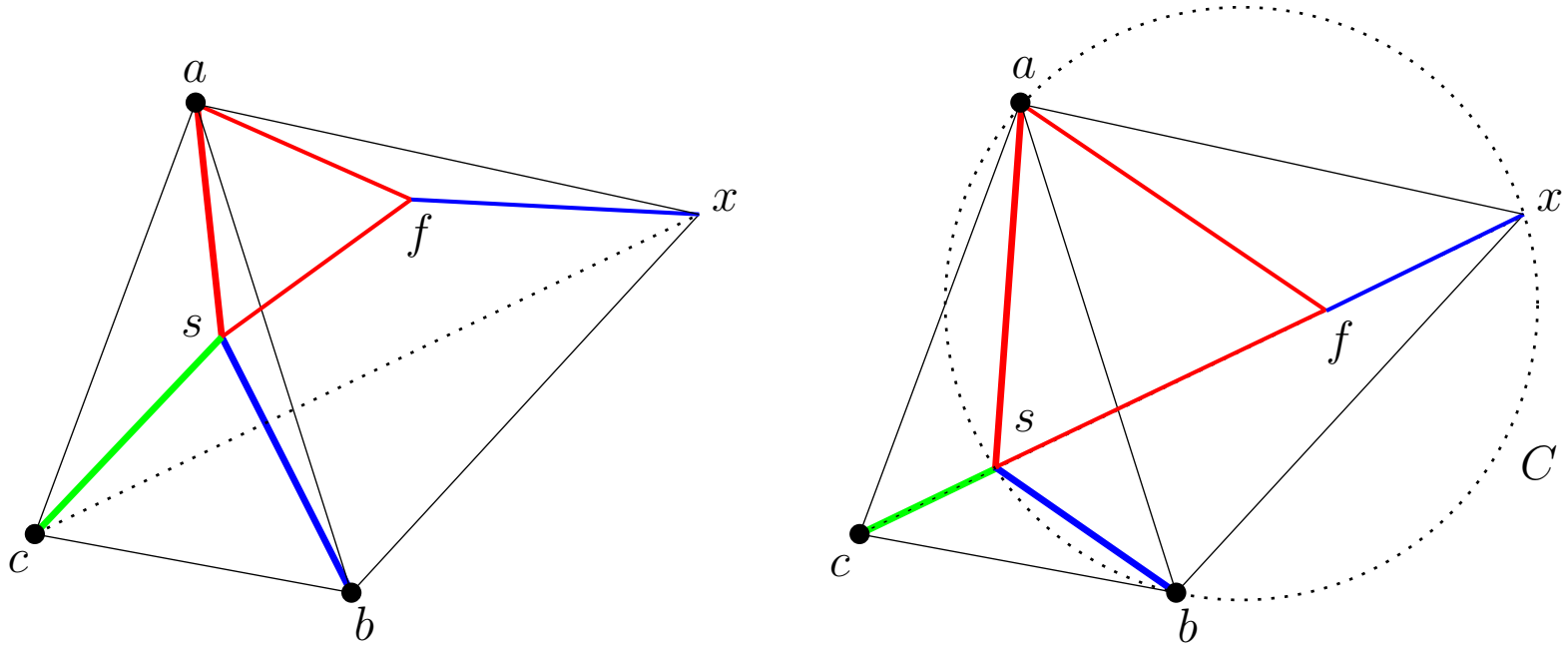
The connections from s to a, b and c form a star T of length $|as| + |bs| + |cs|$. (**Note:** A *star* on $k + 1$ vertices is a tree containing one vertex of degree k and k vertices of degree 1.)
In T, s may coincide with one of the given points in N .
A star T solving the Fermat-Torricelli problem is a shortest network interconnecting the points in N.

Definition: Steiner point

Given points a, b and c in the plane, a point s solving the Fermat-Torricelli problem for the given points will be referred to as a *Steiner point*.

The Fermat-Toricelli problem solution

We now examine a method for solving the Fermat-Torricelli problem using the so-called *rotation proof*.
This is illustrated below for points a, b and c, where the edge ab rotates.



In each diagram, red, green or blue edges of the same colour have the same length. Hence, the sum of distances $|as| + |bs| + |cs|$ is equal to the length of the path $c - s - f - x$.

The Fermat-Toricelli problem solution

The same construction can be applied to the other two sides bc and ca of $\triangle abc$, using the following definitions and notation.

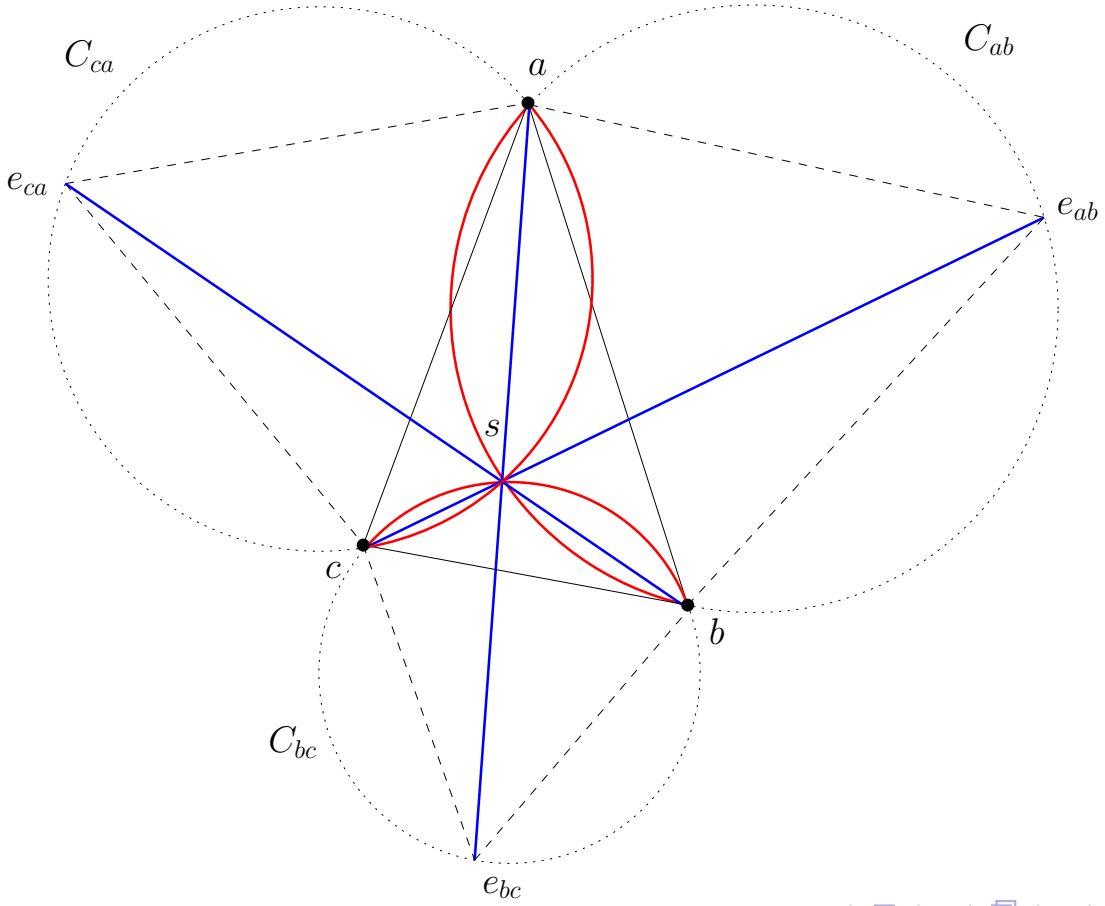
Definitions: Equilateral point, Simpson line

Given distinct points a and b in the plane, define the *equilateral point* e_{ab} for a and b to be the third point of the equilateral triangle with vertices a and b, such that the three vertices are labelled a, b and e_{ab} in counter-clockwise order around the triangle. Given three points a, b and c labelled in clockwise order around $\triangle abc$, the *Simpson line* for c is the line segment ce_{ab}.

Note that for the figures on the previous slide, we have $x = e_{ab}$.

The Fermat-Toricelli problem solution

We can construct the Steiner point by drawing the circumcircles of the three equilateral triangles on the sides of $\triangle abc$. The unique intersection point of these circles is the Steiner point.



The Fermat-Toricelli problem solution

Definition: Steiner arc

Given three points a, b and c labelled in clockwise order around $\triangle abc$, let C_{ab} be the circumcircle of the equilateral triangle abe_{ab}. Then the open arc of C_{ab} between a and b (and not containing e_{ab}) is called the *Steiner arc* of a and b, denoted $\overset{\frown}{ab}$.

The Steiner point can be obtained as the intersection of any pair of Simpson lines and/or Steiner arcs for $\triangle abc$.

Finally, if one of the angles of $\triangle abc$ is $2\pi/3$ or greater, then the Steiner point coincides with the given point having that large angle.

The Fermat-Toricelli problem solution

Theorem 1

Let a, b and c be three distinct points in the plane; let s be the Steiner point minimising $|as| + |bs| + |cs|$.

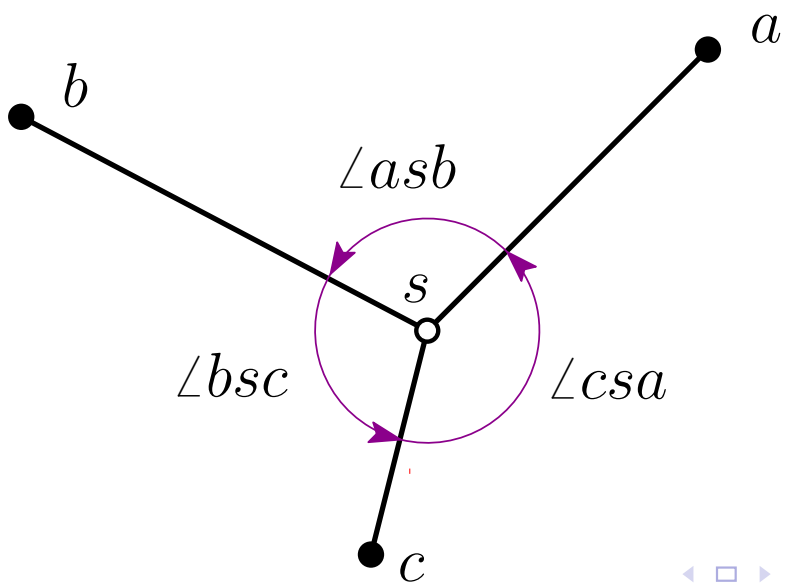
(1) If each angle in the triangle $\triangle abc$ is less than $2\pi/3$, then
   (i) s can be obtained as the intersection between any pair of Simpson lines ae_{bc}, be_{ca} and ce_{ab} and/or Steiner arcs $\widehat{ab}$, $\widehat{bc}$ and $\widehat{ca}$;
   (ii) the length of each Simpson line is $|as| + |bs| + |cs|$; and
   (iii) the three angles around s are each $2\pi/3$.

(2) If some angle in $\triangle abc$ at some vertex is greater than or equal to $2\pi/3$, then s coincides with that vertex.

Hence, a minimum length interconnection of three given points *either* consists of a star connecting a Steiner point to the three given points *or* consists of a pair of line segments connecting one of the given points to the two other points.

Steiner trees: some preliminary definitions

- We define a *geometric network* $G = (V(G), E(G))$ to be a connected graph that is embedded in the plane; the vertices $V(G)$ are points, and the edges $E(G)$ are simple curves.
- We denote by $|e|$ the Euclidean length of the embedding of edge $e \in E(G)$.
- If s is a vertex of G, we define the *meeting angles* of s to be the angles that appear counter-clockwise around s in G.



The Steiner tree problem

Euclidean 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 $|T| := \sum_{e \in E(T)} |e|$ is minimised.

Note: T, considered as a graph, can be assumed to be a tree, and all non-zero edges of T must be embedded as line segments.

Definitions: Minimum Steiner tree, terminals, Steiner points

A network $T = (V(T), E(T))$ representing a solution to the Steiner tree problem is referred to as a *Minimum Steiner tree*. The given points $N \subseteq V(T)$ are called *terminals*; any extra vertices $S = V(T) \setminus N$ are called *Steiner points*.

Basic properties of minimum Steiner trees

Theorem 2

For any finite set of points N in the plane, there exists a minimum Steiner tree $T = (V(T), E(T))$ for terminals $N \subseteq V(T)$, having Steiner points $S = V(T) \setminus N$, satisfying the following four conditions:

(1) Each edge of T is a line segments and does not intersect any other vertex or edge of T.

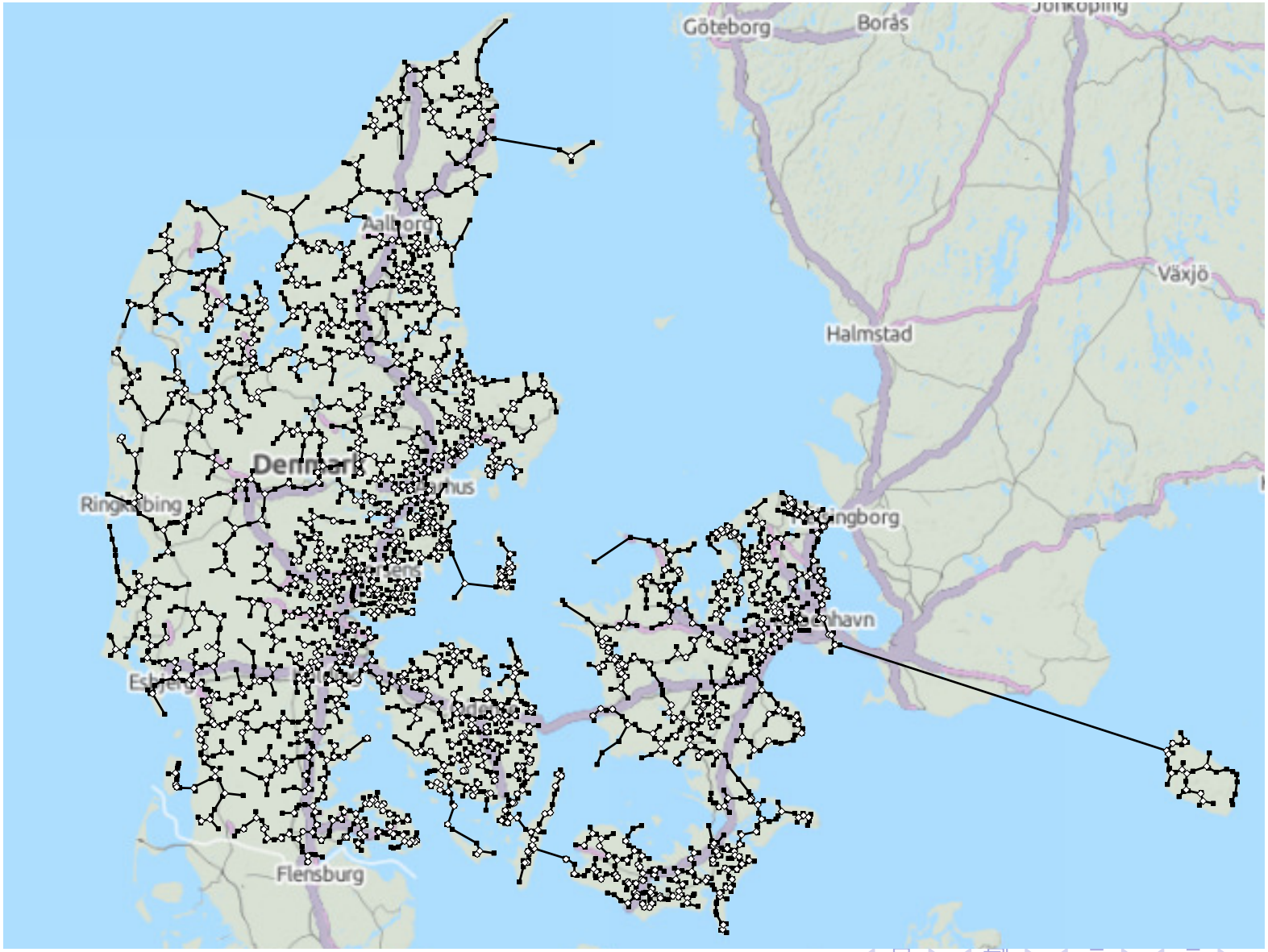
(2) Terminals in N have degree at most 3 in T, and each pair of incident edges meets at an angle of $2\pi/3$ or greater.

(3) Steiner points in S have degree 3 in T, and each pair of incident edges meets at an angle of $2\pi/3$.

(4) T has at most $n-2$ Steiner points and at most $2n-3$ edges, where $n = |N|$ is the number of terminals.

Properties (2) and (3) follow largely from Theorem 1. The proofs of properties (1) and (4) are left as Exercises.

An example of a minimum Steiner tree



Steiner topologies – some definitions

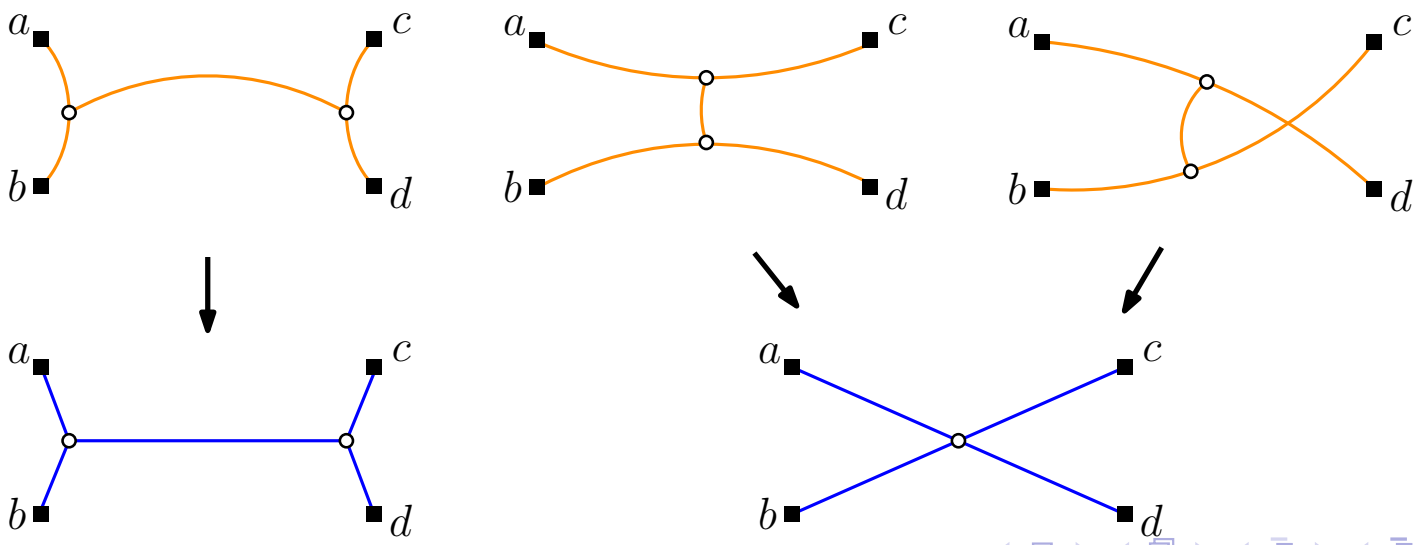
- The underlying graph $\mathcal{T} = (V(\mathcal{T}), E(\mathcal{T}))$ for a geometric network $T = (V(T), E(T))$ that interconnects a set of terminals N is called the *topology* of T.
- **Note:** In a topology, the vertices and edges are are not embedded in the plane, but each vertex is either labelled with the label of its corresponding terminal or labelled as a Steiner point.
- An embedding of a geometric network is called *non-degenerate* if all edges in the embedding have non-zero length — otherwise it is *degenerate*.

Definitions: Steiner topology, full Steiner topology

The topology of a non-degenerate minimum Steiner tree is called a *Steiner topology*. The topology of a non-degenerate minimum Steiner tree in which every terminal has degree 1 is called a *full Steiner topology*.

Relatively minimal trees

- By Theorem 2 we can assume that Steiner vertices have degree exactly 3 and terminals have degree at most 3 in a Steiner topology.
- A shortest possible embedding of a Steiner topology is called a *relatively minimal tree*; this is obtained by locating the Steiner vertices so that the length of the geometric network is minimised.
- A relatively minimal tree may be degenerate, in which case edges can have meeting angles that are less than $2\pi/3$.

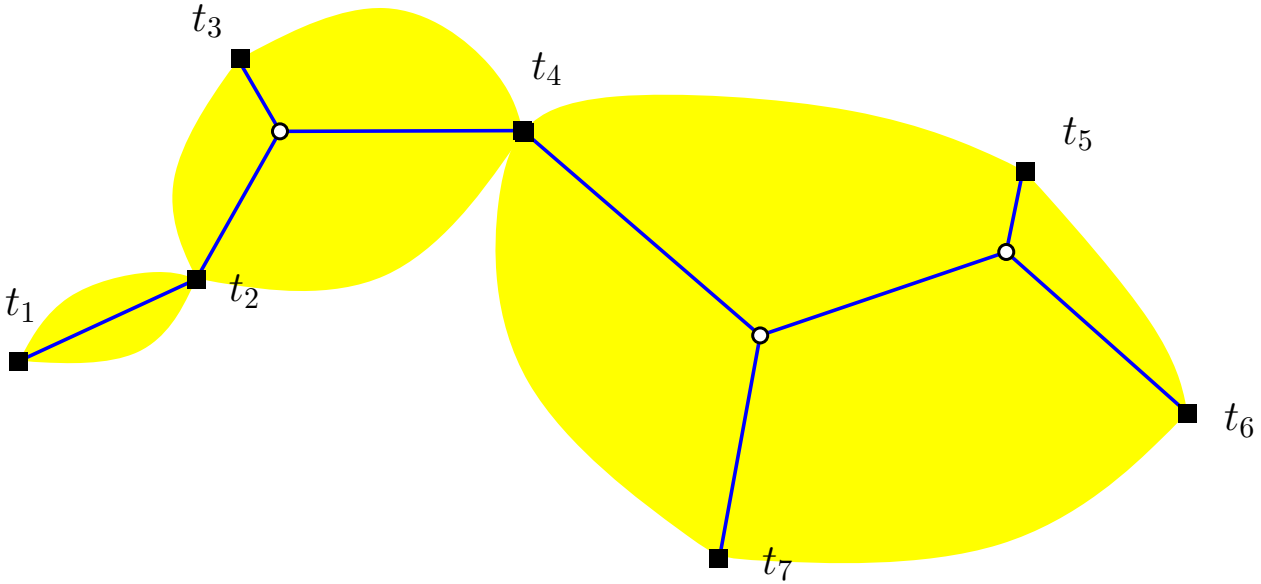


Steiner trees

Definitions: Steiner tree, full Steiner tree

A non-degenerate relatively minimal tree for a (full) Steiner topology $\mathcal{T}$ is called a *(full) Steiner tree* for $\mathcal{T}$.

Note that there exists a Steiner tree that is a minimum Steiner tree. Furthermore, a minimum Steiner tree T that interconnects two or more terminals is a union of full Steiner trees, called the *full components* of T.



Necessary and sufficient properties of Steiner trees

The following lemma shows that Steiner trees satisfy the same conditions as minimum Steiner trees.

Lemma 3

If a non-degenerate embedding T of a Steiner topology $\mathcal{T}$ fulfils the four conditions of Theorem 2, then T is the unique Steiner tree for $\mathcal{T}$.

The proof of this lemma involves two steps.
Step 1 is to show that if T fulfils the conditions of Theorem 2, then T is a Steiner tree.
Step 2 is to show that such a T is unique.

The number of full Steiner topologies

We now focus on full Steiner trees. From an algorithmic point of view, full Steiner topologies and trees are particularly useful, as we can construct full Steiner trees from full Steiner topologies very efficiently; this will be shown in the next section.

The challenge is that the number of full Steiner topologies (and trees) increases rapidly, as the number of terminals increases.

Theorem 4

The number of distinct full Steiner topologies for n terminals, where $n \geq 3$, is:

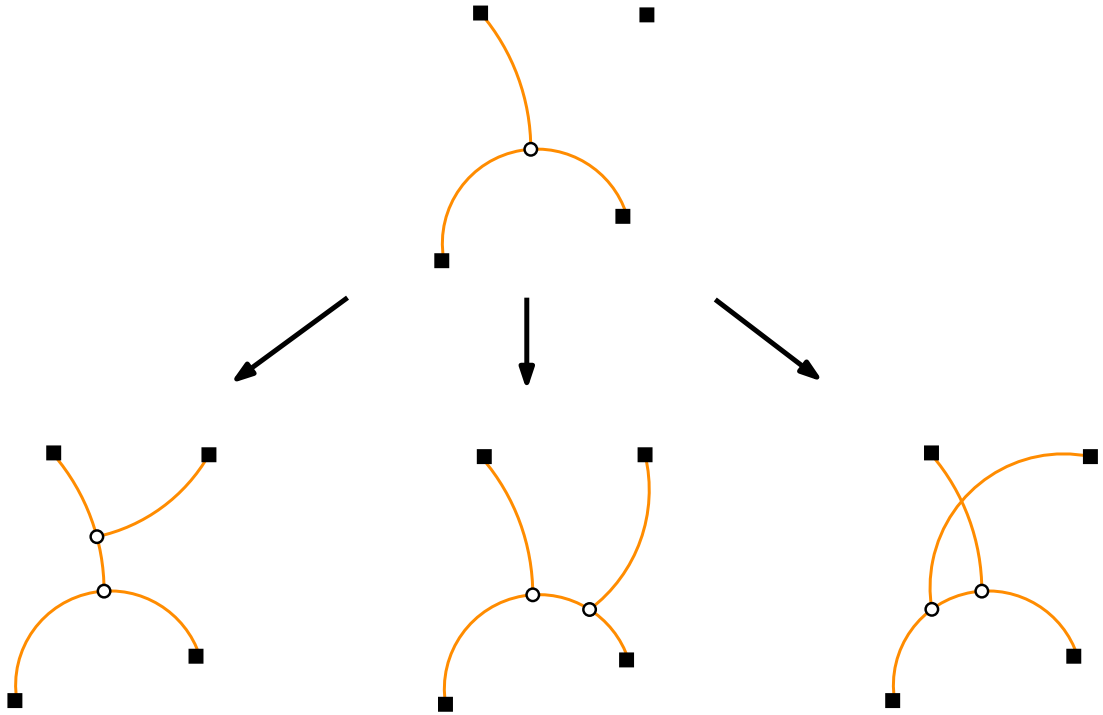
$$f(n) = 1 \cdot 3 \cdot 5 \cdots (2n-7) \cdot (2n-5) = \frac{(2n-4)!}{(n-2)!\ 2^{n-2}}.$$

Furthermore, each of these topologies can be realised as the unique minimum Steiner tree for some set of terminals.

Proof of Theorem 4 – part 1

A full Steiner topology on n terminals has $2n-3$ edges. Hence $f(n)$, the number of full Steiner topologies, satisfies:

$$f(n) = (2(n-1)-3)f(n-1) = (2n-5)f(n-1)$$



The formula for $f(n)$ follows by induction.

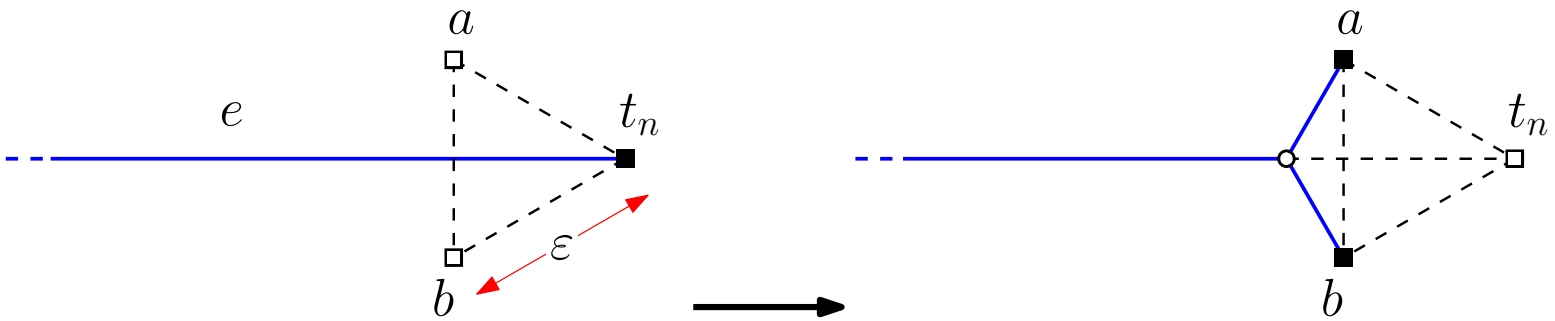
Proof of Theorem 4 – part 2

Part 2 can be proved by replacing any terminal in a minimum Steiner tree by a sufficiently small **cherry**, defined as follows:

Definition: Cherry

A pair of terminals adjacent to a common Steiner point in a Steiner topology is called a *cherry* for the topology.

Note that at least one cherry always exists for a full Steiner topology for $n \geq 3$ terminals.



T_n T_{n+1}

The number of full topologies

The function $f(n)$ from Theorem 4 is super-exponential, so enumerating all possible full Steiner topologies for large values of n is infeasible.

n	3	4	5	6	7	8	10
$f(n)$	1	3	15	105	945	10395	2027025

Many of the full Steiner topologies have, however, no associated full Steiner tree for a given configuration of terminals. Despite this, no general exponential, or slower growing, bound is known for the number of possible full topologies.

This explosion in the number of topologies is one of the main challenges in designing an efficient algorithm to solve the Steiner tree problem.

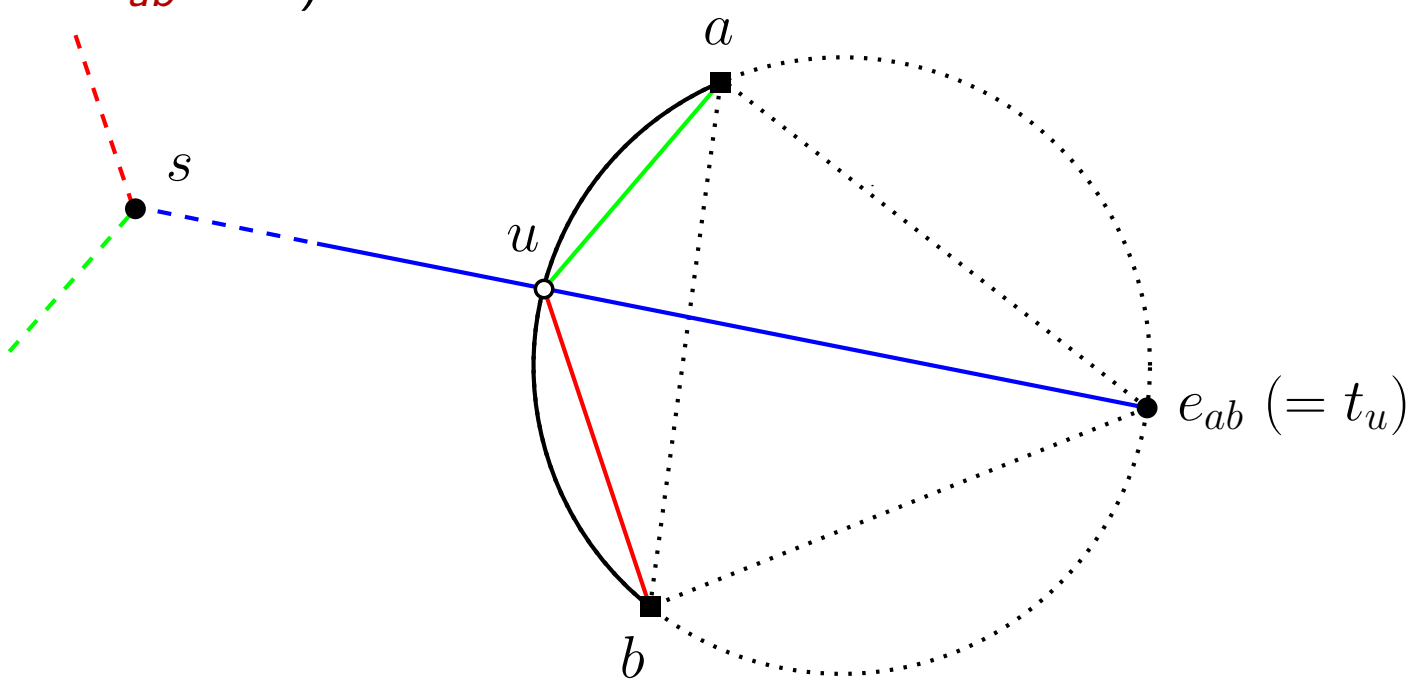
The Melzak-Hwang algorithm

Let $\mathcal{T}$ be a full Steiner topology for $n \geq 3$ terminals. In this section we give an efficient algorithm for computing a full Steiner tree T for $\mathcal{T}$ (or showing that it does not exist).

- First we present a basic recursive (and slow) algorithm.
- Then we show how the running time can be improved.
- Finally, we present an efficient implementation of the algorithm.

Basic recursive algorithm

Let u be a Steiner point in $\mathcal{T}$ with neighbouring vertices a, b and s, where a and b are terminals (forming a cherry). Assume that in the full Steiner tree T for $\mathcal{T}$ the vertex u is located on the opposite side of $\overline{ab}$ to the equilateral point e_{ab}. By Theorem 1, u lies on the Steiner arc $\widehat{ab}$ (on the same side of e_{ab} as s)



Basic recursive algorithm

Also, e_{ab} lies on the Simpson line for T through su and $|e_{ab}u| = |au| + |bu|$. Hence, we may replace a, b and u with a new point $t_u = e_{ab}$ that is connected directly to s. We refer to this new point as a *pseudo-terminal*. This reduction from n terminals to $n-1$ terminals is called a **merging step**.

Let $\mathcal{T}'$ be the full Steiner topology on the new set of $n-1$ terminals (where a and b have been replaced by t_u), and let T' be the full Steiner tree for $\mathcal{T}'$. Then u can be determined as the intersection between $\widehat{ab}$ and t_us in T'. The Steiner tree T is constructed by deleting edge t_us in T', and adding the edges au, bu and us; this is called a **reconstruction step**.

The complete algorithm therefore consists of $n-2$ merging steps (one for each Steiner point), and similarly $n-2$ reconstruction steps; after $n-2$ merging steps the topology consists of two terminals for which the full Steiner tree is a simple line segment.

Notes on the basic recursive algorithm

- This is generally referred to as Melzak's algorithm.
- In each reconstruction step, the new Steiner tree T exists if and only if T' exists and the edge $t_u s$ in T' intersects the Steiner arc $\widehat{ab}$.
- The major difficulty in implementing the above algorithm is that for any pair of terminals a, b forming a cherry we do not know on which side of $\overline{ab}$ to construct the pseudo-terminal t_u in each merging step. There are two possibilities for t_u, corresponding to the two equilateral points e_{ab} and e_{ba}; this creates two subproblems for each merging step — in total a running time of $O(2^n)$.

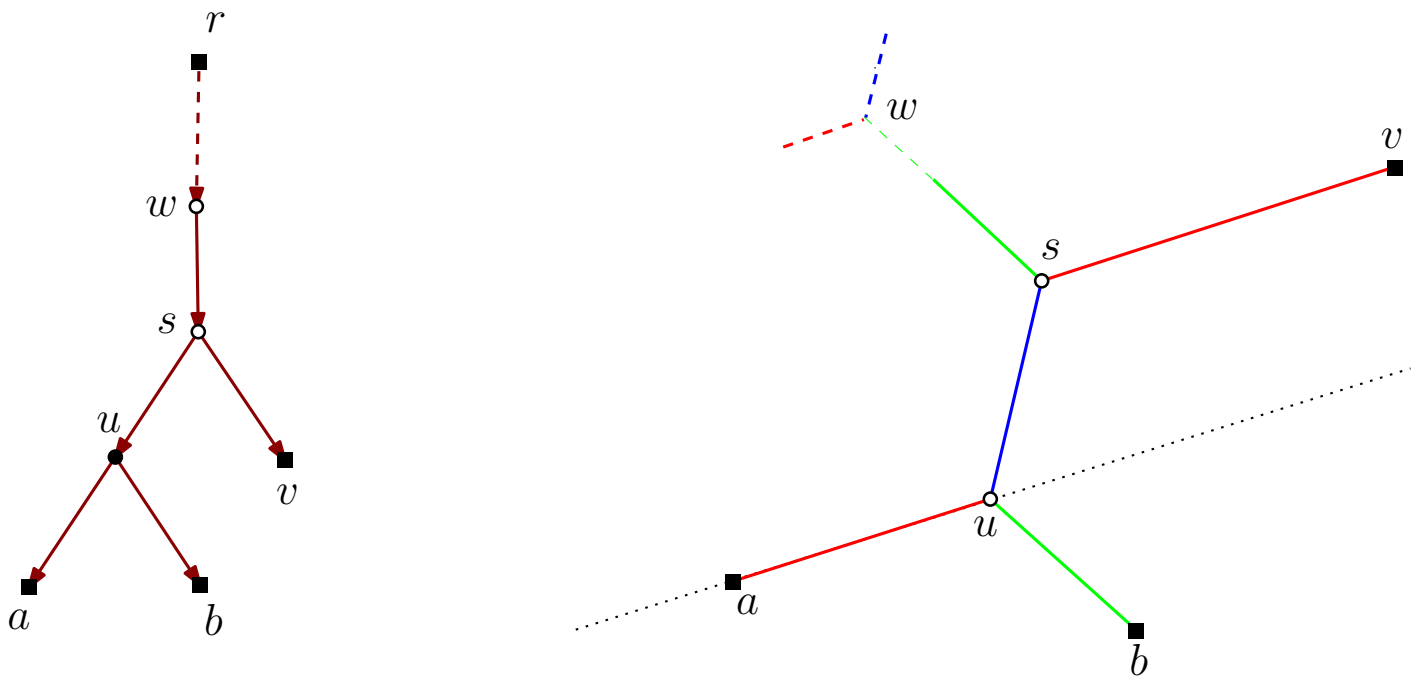
We next show how to implement the algorithm to run in $O(n)$ time. This is achieved by selecting a particular merging order, which makes it possible to correctly identify a single equilateral point in each merging step.

Identifying the side of the equilateral point

Assume that topology $\mathcal{T}$ has five or more terminals. Select an arbitrary terminal r in $\mathcal{T}$, and denote by $\mathcal{T}_r$ the tree $\mathcal{T}$ with r as its root; every Steiner point in $\mathcal{T}_r$ is an internal node with two children. Let u be a Steiner point that is deepest in the rooted tree; therefore, u has two terminals a and b as children. Let s be the parent of u, and let v be the second child of s (or the sibling of u). Finally, let w be the parent of s.

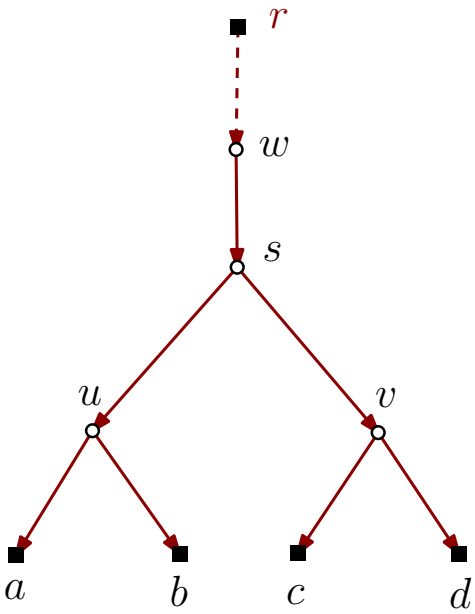
Since u is chosen as a Steiner point of maximal depth in $\mathcal{T}_r$, v is either a terminal (**Case 1**) or a Steiner point with two children c and d that are both terminals (**Case 2**).

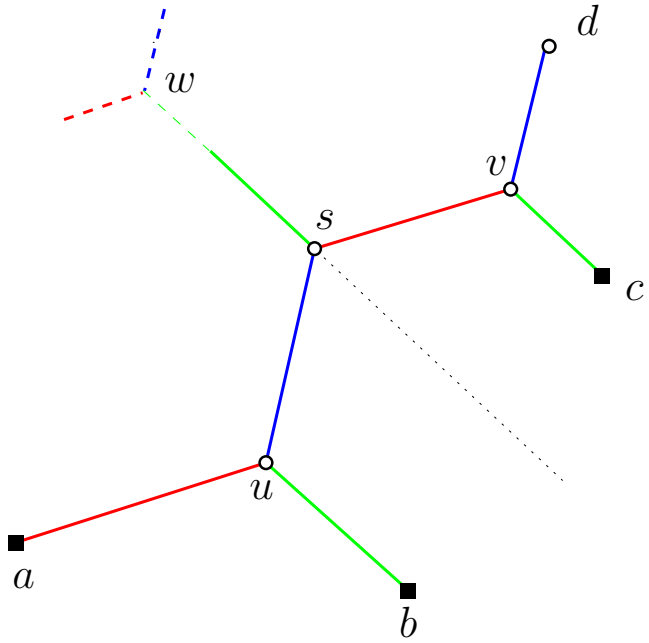
Identifying the side of the equilateral point — Case 1



Assume without loss of generality that au is parallel to sv. Then v and b are on opposite sides of $\overline{au}$; hence, the Steiner point u must be chosen to be on the same side of line $\overline{ab}$ as terminal v.

Identifying the side of the equilateral point — Case 2





Again assume that au is parallel to sv, and bu is parallel to cv (and hence to sw). The line $\overline{sw}$ separates the line segments ab and cd. Therefore, ab and cd cannot intersect each other: either c and d are both on the same side of line $\overline{ab}$ (in which case the Steiner point u is on the same side of $\overline{ab}$ as c and d), or a and b are both on the same side of line $\overline{cd}$ (in which case the Steiner point v is on the same side of $\overline{cd}$ as a and b).

The Melzak-Hwang theorem

Whichever case occurs, we can perform a merging operation that correctly identifies a single equilateral point to choose as our next pseudo-terminal. Identifying the correct side for each equilateral point (and constructing the pseudo-terminal) clearly takes constant time per merging operation.

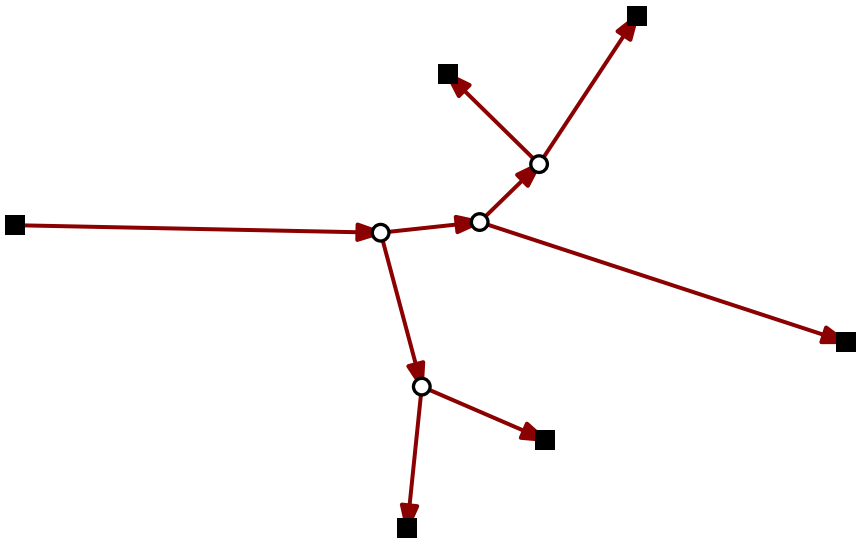
Each of the three main phases of this improved *Melzak-Hwang algorithm* — breadth-first search, merging and reconstruction — clearly takes $O(n)$ time, resulting in the following theorem:

Theorem 5

Given a full Steiner topology $\mathcal{T}$ with n terminals, there is an $O(n)$ time algorithm to either compute a full Steiner tree for $\mathcal{T}$ or decide that no such tree exists.

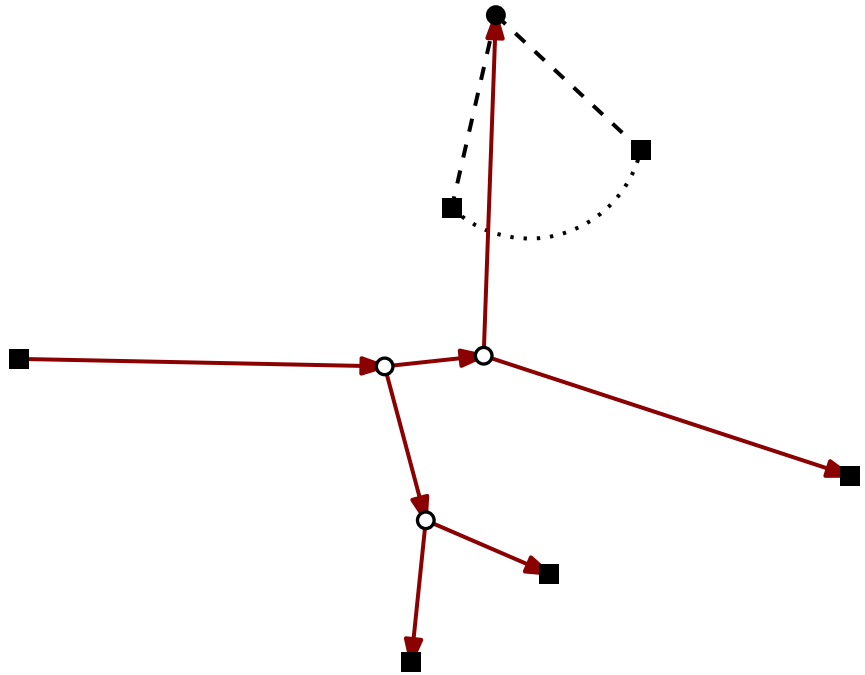
A detailed description of the algorithm is given on the subject website.

An example

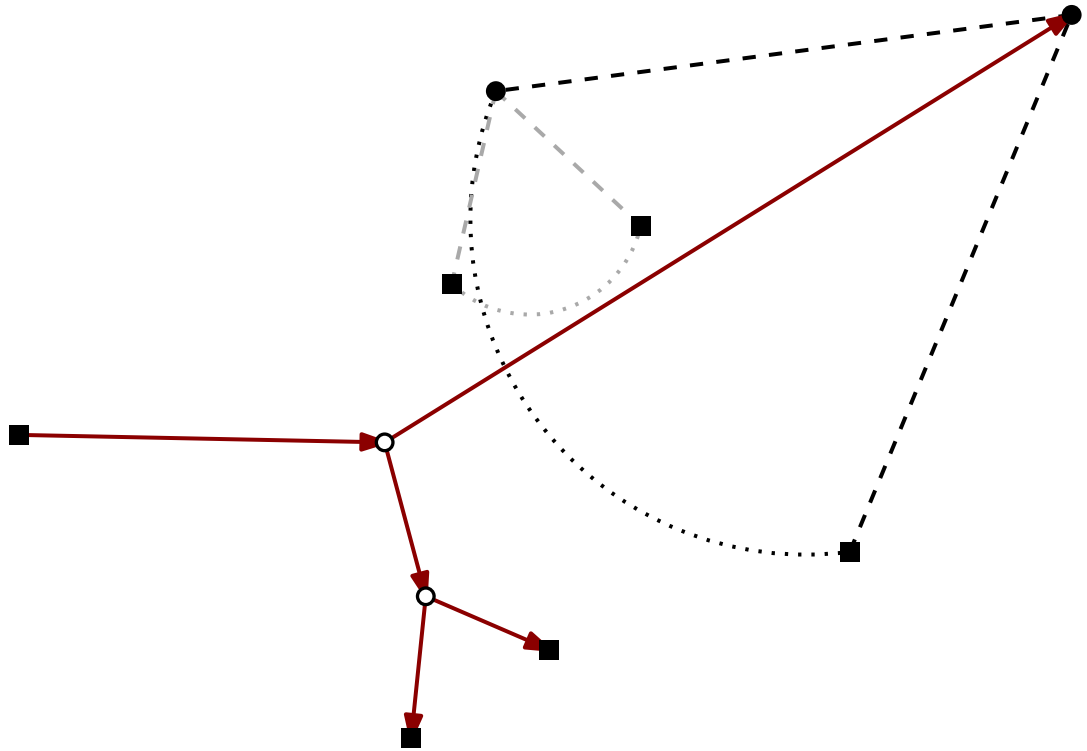


Here is an example of how the Melzak-Hwang algorithm works for a rooted full Steiner topology with 6 terminals. The diagram indicates the given location of each terminal.

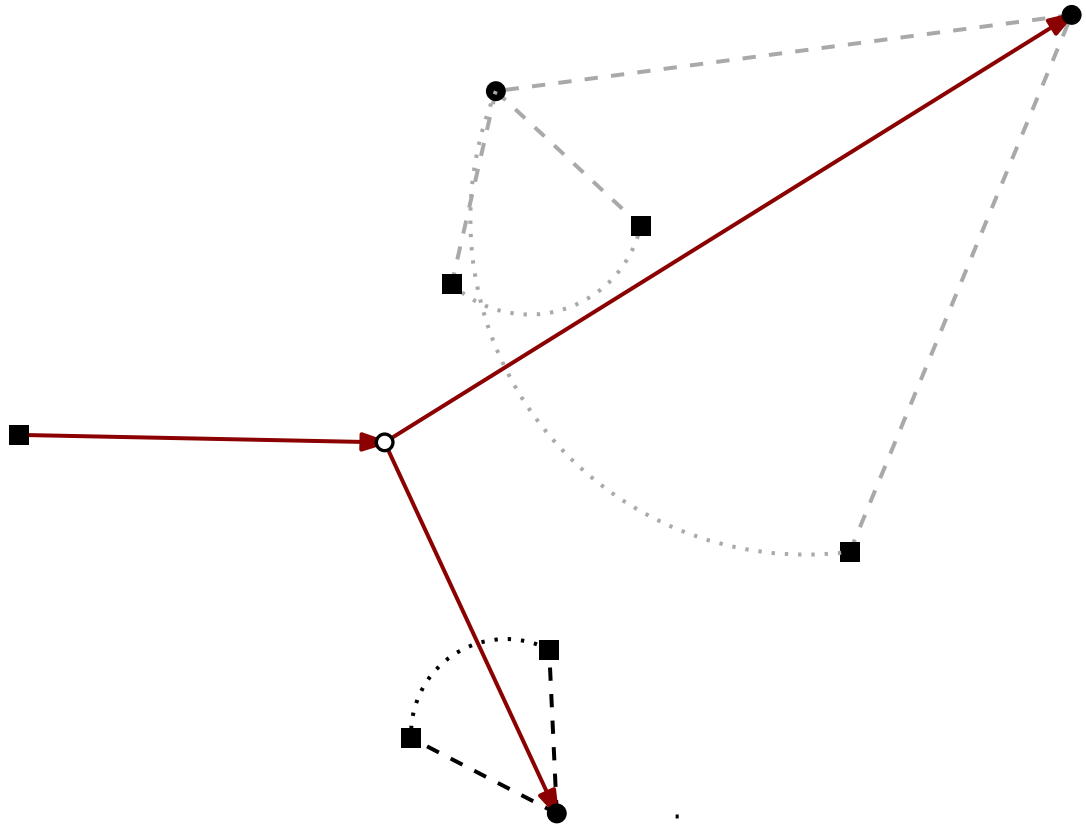
An example — Merging steps



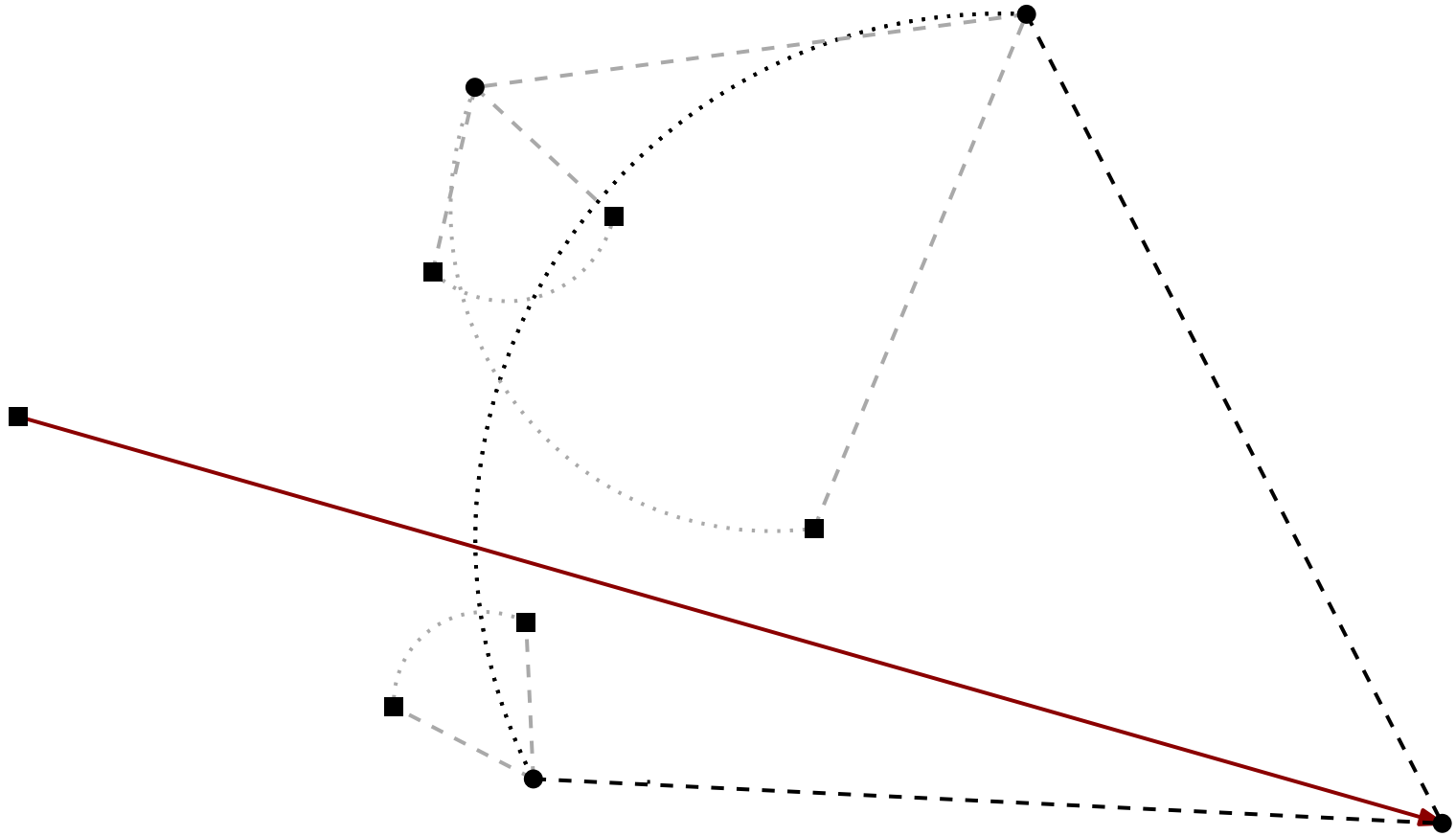
An example — Merging steps



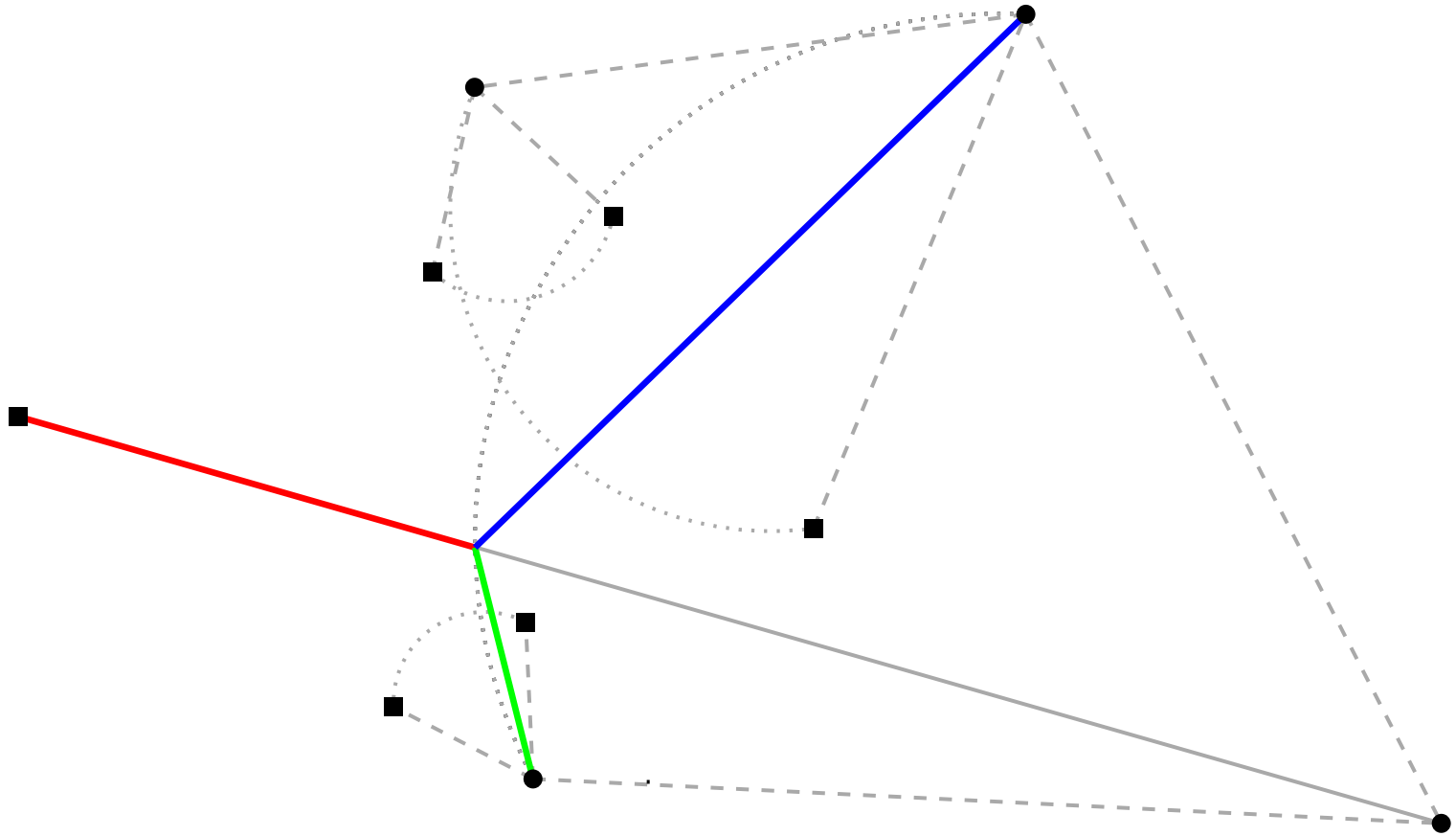
An example — Merging steps



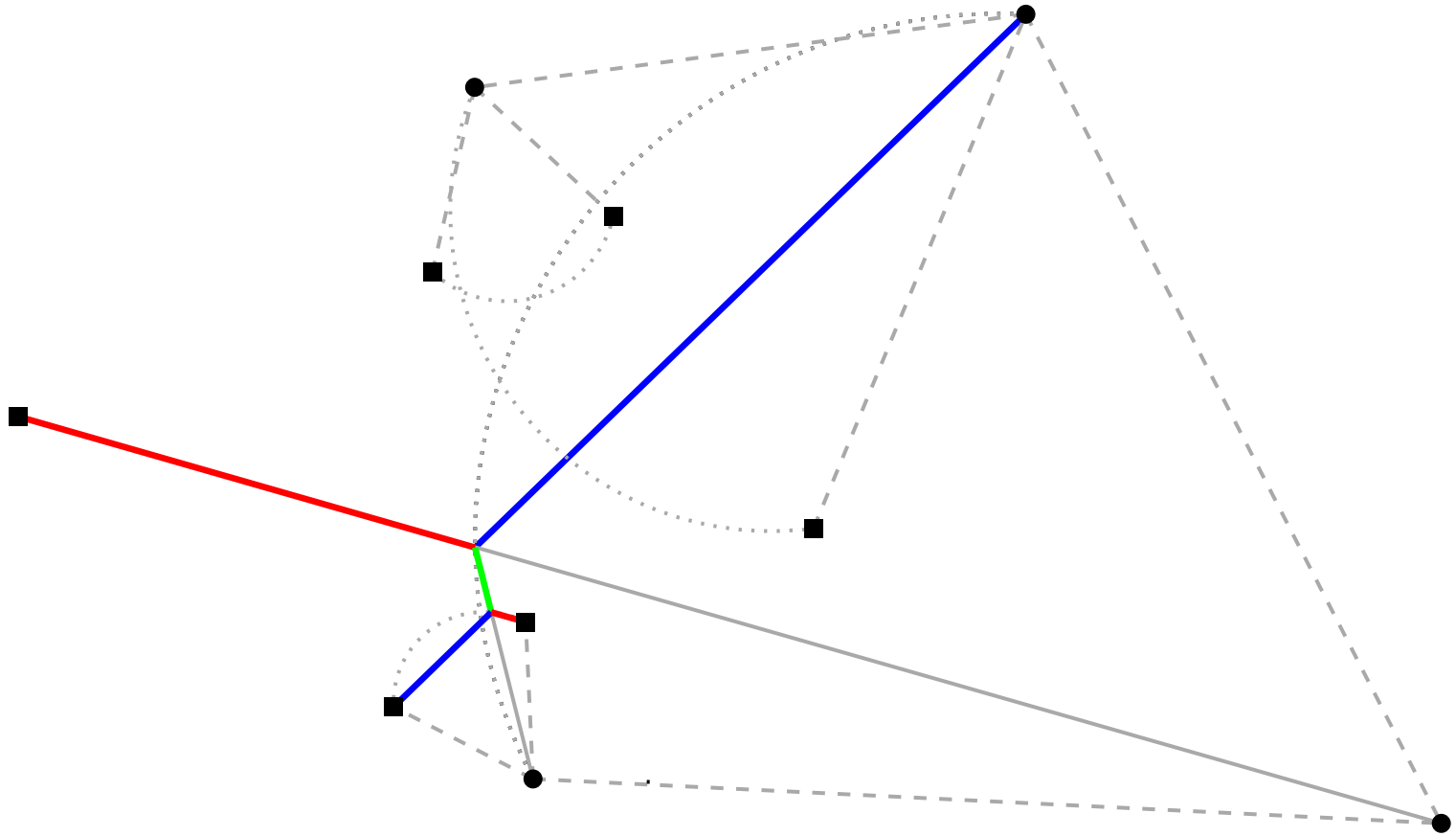
An example — Merging steps



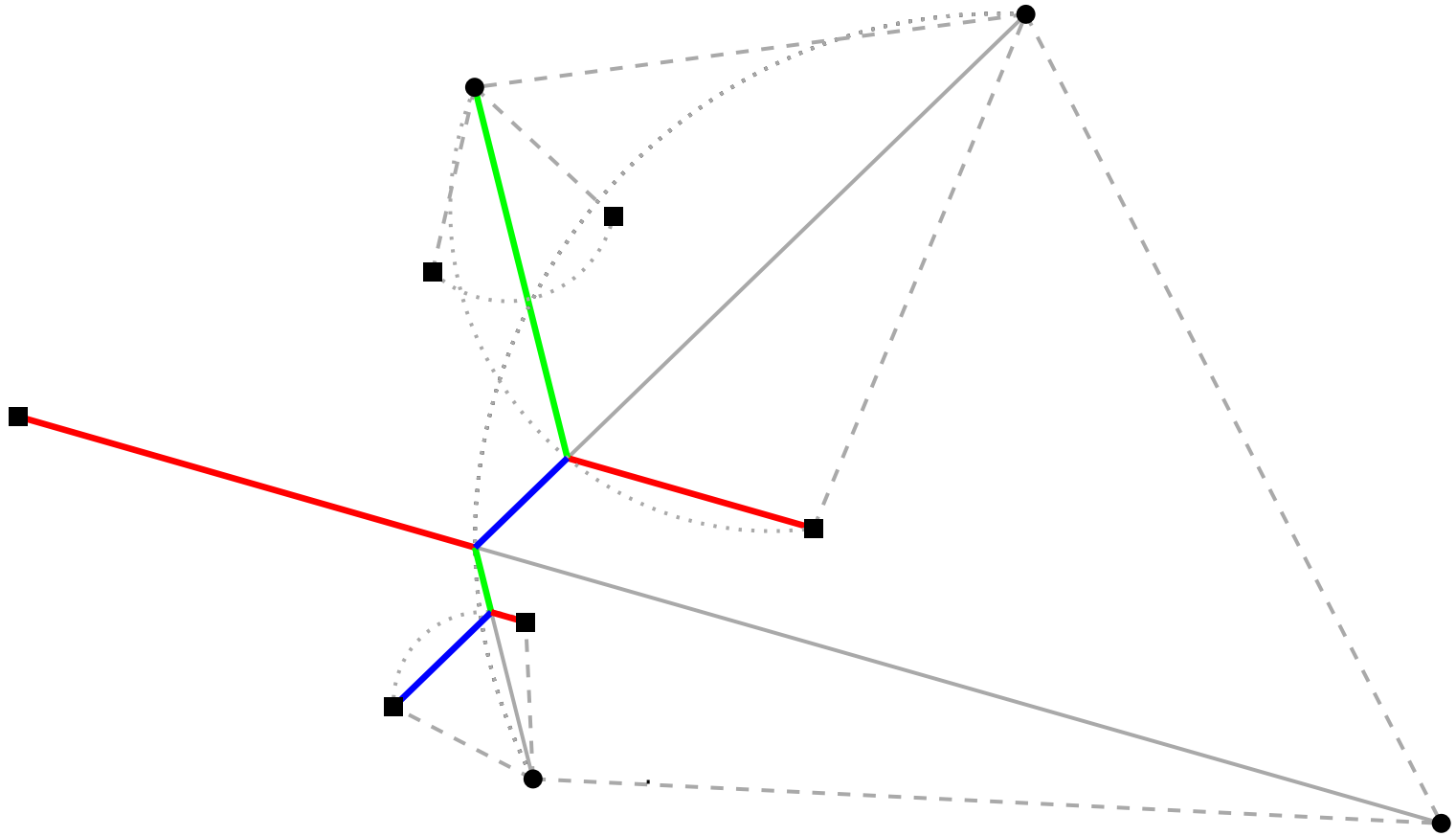
An example — Reconstruction steps



An example — Reconstruction steps



An example — Reconstruction steps



An example — Reconstruction steps