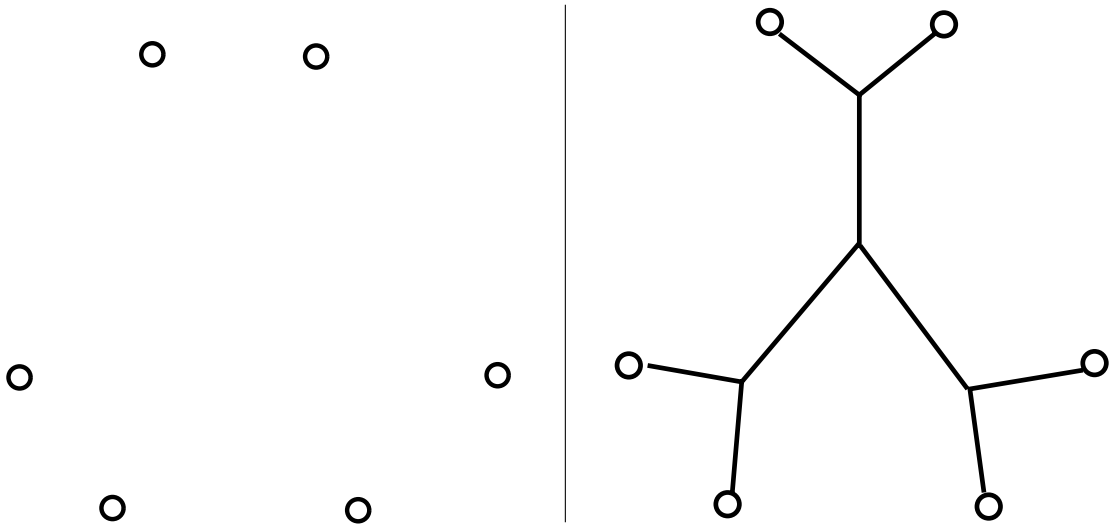

INPUT OUTPUT

16.10 Steiner Tree

Input description: A graph $G = (V, E)$. A subset of vertices $T \in V$.

Problem description: Find the smallest tree connecting all the vertices of T.

Discussion: Steiner trees arise often in network design problems, since the minimum Steiner tree describes how to connect a given set of sites using the smallest amount of wire. Analogous problems occur when designing networks of water pipes or heating ducts and in VLSI circuit design. Typical Steiner tree problems in VLSI are to connect a set of sites to (say) ground under constraints such as material cost, signal propagation time, or reducing capacitance.

The Steiner tree problem is distinguished from the minimum spanning tree (MST) problem (see Section 15.3 (page 484)) in that we are permitted to construct or select intermediate connection points to reduce the cost of the tree. Issues in Steiner tree construction include:

- *How many points do you have to connect?* – The Steiner tree of a pair of vertices is simply the shortest path between them (see Section 15.4 (page 489)). The Steiner tree of all the vertices, when $S = V$, simply defines the MST of G. The general minimum Steiner tree problem is NP-hard despite these special cases, and remains so under a broad range of restrictions.

- *Is the input a set of geometric points or a distance graph?* – Geometric versions of Steiner tree take a set of points as input, typically in the plane, and seek the smallest tree connecting the points. A complication is that the set of possible intermediate points is not given as part of the input but must be deduced from the set of points. These possible Steiner points must satisfy several geometric properties, which can be used to reduce the set of candidates down to a finite number. For example, every Steiner point will have a degree of exactly three in a minimum Steiner tree, and the angles formed between any two of these edges must be exactly 120 degrees.

- *Are there constraints on the edges we can use?* – Many wiring problems correspond to geometric versions of the problem, where all edges are restricted to being either horizontal or vertical. This is the so-called *rectilinear Steiner problem*. A different set of angular and degree conditions apply for rectilinear Steiner trees than for Euclidean trees. In particular, all angles must be multiples of 90 degrees, and each vertex is of a degree up to four.

- *Do I really need an optimal tree?* – Certain Steiner tree applications (e.g., circuit design and communications networks) justify investing large amounts of computation to find the best possible Steiner tree. This implies an exhaustive search technique such as backtracking or branch-and-bound. There are many opportunities for pruning search based on geometric and graph-theoretic constraints.

  Still, Steiner tree remains a hard problem. We recommend experimenting with the implementations described below before attempting your own.

- *How can I reconstruct Steiner vertices I never knew about?* – A very special type of Steiner tree arises in classification and evolution. A *phylogenic tree* illustrates the relative similarity between different objects or species. Each object represents (typically) a leaf/terminal vertex of the tree, with intermediate vertices representing branching points between classes of objects. For example, an evolutionary tree of animal species might have leaf nodes of *human, dog, snake* and internal nodes corresponding to taxa *(animal, mammal, reptile)*. A tree rooted at *animal* with *dog* and *human* classified under *mammal* implies that humans are closer to dogs than to snakes.

  Many different phylogenic tree construction algorithms have been developed that vary in (1) the data they attempt to model, and (2) the desired optimization criterion. Each combination of reconstruction algorithm and distance measure is likely to give a different answer, so identifying the "right" method for any given application is somewhat a question of faith. A reasonable procedure is to acquire a standard package of implementations, discussed below, and then see what happens to your data under all of them.

Fortunately, there is a good, efficient heuristic for finding Steiner trees that works well on all versions of the problem. Construct a graph modeling your input,

setting the weight of edge (i, j) equal to the distance from point i to point j. Find an MST of this graph. You are guaranteed a provably good approximation for both Euclidean and rectilinear Steiner trees.

The worst case for a MST approximation of the Euclidean Steiner tree is three points forming an equilateral triangle. The MST will contain two of the sides (for a length of 2), whereas the minimum Steiner tree will connect the three points using an interior point, for a total length of $\sqrt{3}$. This ratio of $\sqrt{3}/2 \approx 0.866$ is always achieved, and in practice the easily-computed MST is usually within a few percent of the optimal Steiner tree. The rectilinear Steiner tree / MST ratio is always $\geq 2/3 \approx 0.667$.

Such an MST can be refined by inserting a Steiner point whenever the edges of the minimum spanning tree incident on a vertex form an angle of less than 120 degrees between them. Inserting these points and locally readjusting the tree edges can move the solution a few more percent towards the optimum. Similar optimizations are possible for rectilinear spanning trees.

Note that we are only interested in the subtree connecting the terminal vertices. We may need to trim the MST if we add nonterminal vertices to the input of the problem. We retain only the tree edges which lie on the (unique) path between some pair of terminal nodes. The complete set of these can be found in $O(n)$ time by performing a BFS on the full tree starting from any single terminal node.

An alternative heuristic for graphs is based on shortest path. Start with a tree consisting of the shortest path between two terminals. For each remaining terminal t, find the shortest path to a vertex v in the tree and add this path to the tree. The time complexity and quality of this heuristic depend upon the insertion order of the terminals and how the shortest-path computations are performed, but something simple and fairly effective is likely to result.

Implementations: GeoSteiner is a package for solving both Euclidean and rectilinear Steiner tree problems in the plane by Warme, Winter, and Zachariasen [WWZ00]. It also solves the related problem of MSTs in hypergraphs, and claims to have solved problems as large as 10,000 points to optimality. It is available from *http://www.diku.dk/geosteiner/*. This is almost certainly the best code for geometric instances of Steiner tree.

FLUTE (*http://home.eng.iastate.edu/~cnchu/flute.html*) computes rectilinear Steiner trees, emphasizing speed. It contains a user-defined parameter to control the tradeoff between solution quality and run time.

GOBLIN (*http://www.math.uni-augsburg.de/~fremuth/goblin.html*) includes both heuristics and search methods for finding Steiner trees in graphs.

The programs PHYLIP (*http://evolution.genetics.washington.edu/phylip.html*) and PAUP (*http://paup.csit.fsu.edu/*) are widely-used packages for inferring phylogenic trees. Both contain over 20 different algorithms for constructing phylogenic trees from data. Although many of them are designed to work with molecular sequence data, several general methods accept arbitrary distance matrices as input.

Notes: Recent monographs on the Steiner tree problem include Hwang, Richards, and Winter [HRW92] and Prömel and Steger [PS02]. Du, et al. [DSR00] is a collection of recent surveys on all aspects of Steiner trees. Older surveys on the problem include [Kuh75]. Empirical results on Steiner tree heuristics include [SFG82, Vos92].

The Euclidean Steiner problem dates back to Fermat, who asked how to find a point p in the plane minimizing the sum of the distances to three given points. This was solved by Torricelli before 1640. Steiner was apparently one of several mathematicians who worked the general problem for n points, and was mistakenly credited with the problem. An interesting, more detailed history appears in [HRW92].

Gilbert and Pollak [GP68] first conjectured that the ratio of the length of the minimum Steiner tree over the MST is always $\geq \sqrt{3}/2 \approx 0.866$. After twenty years of active research, the Gilbert-Pollak ratio was finally proven by Du and Hwang [DH92]. The Euclidean MST for n points in the plane can be constructed in $O(n \lg n)$ time [PS85].

Arora [Aro98] gave a polynomial-time approximation scheme (PTAS) for Steiner trees in k-dimensional Euclidian space. A 1.55-factor approximation for Steiner trees on graphs is due to Robins and Zelikovsky [RZ05].

Expositions on the proof that the Steiner tree problem for graphs is hard [Kar72] include [Eve79a]. Expositions on exact algorithms for Steiner trees in graphs include [Law76]. The hardness of Steiner tree for Euclidean and rectilinear metrics was established in [GGJ77, GJ77]. Euclidean Steiner tree is not known to be in NP, because of numerical issues in representing distances.

Analogies can be drawn between minimum Steiner trees and minimum energy configurations in certain physical systems. The case that such analog systems—including the behavior of soap films over wire frames—"solve" the Steiner tree problem is discussed in [Mie58].

Related Problems: Minimum spanning tree (see page 484), shortest path (see page 489).