Minimum Spanning Tree

The minimum spanning tree problem is to choose edges from an undirected connected network so that every two nodes are connected by a path. The objective is to choose edges with minimum total distance. It is intuitively clear that such a set will not contain a cycle. Therefore, the set will be a tree. This tree will connect (*span*) all the nodes.

This problem has a number of important practical applications. For example, it can sometimes be helpful in planning *transportation networks* that will not be used much. There, the primary consideration is to provide some link between all pairs of nodes in the most economical way. Other examples include the planning of large-scale communication models and distribution networks.

The minimum spanning tree problem can be solved in a very straightforward way because it is one of the few problems where being *greedy* at each stage of a solution procedure still results in an optimal solution at the end! The algorithm is as follows:

Greedy algorithm

1. *Initialization* The algorithm starts with no edge in the tree.

2. *Iterative Step* The edges are considered in increasing order of their length. If an edge does not form a cycle with edges already in the tree, add it to the tree. Otherwise, discard it.

3. *Stopping Criterion:* Stop when all nodes are connected.

Proof that the greedy algorithm finds a minimum spanning tree: Let T^* denote the tree found by the greedy algorithm and suppose some other tree is shorter. We will show that this leads to a contradiction. Among the trees which are shorter than the greedy solution T^*, denote by **T** one which has the largest number of common edges with T^*. Let **a** be a shortest edge in T^* but not in **T** and let **C** be the unique cycle formed when edge **a** is added to the edges of **T**. The cycle **C** contains at least one edge not in T^*, say **b**, since T^* contains no cycle. The length of **a** is less than or equal to that of **b**, since edge **a** must have been considered before **b** by the greedy algorithm. Now the tree T' obtained from **T** by adding edge **a** and removing edge **b** is of the same length as or shorter than **T**. Therefore T' is shorter than T^*. But it has one more edge in common with T^*, a contradiction. This completes the proof.

Here's a different algorithm for finding a minimum spanning tree:

1. Select a node arbitrarily, and connect it to a node closest to it.

2. Identify an unconnected node that is closest to a connected node and add the edge between them.

Repeat until all nodes have been connected.

It may seem that it makes a difference which node to start at. It turns out it doesn't (except for possibly finding alternative minimum spanning trees). We omit the proof that this algorithm also gives an optimum solution to the minimum spanning tree problem. In the homework and exam, you can use either of these two algorithms to solve the problem.

Exercise 3 *A bank is hooking up computer terminals at each of its branch offices using special phone lines. The phone line from a branch does not need to be directly connected to the main office, but there must be a sequence of phone lines from any office to the main office.*

The charge for the special phone lines is directly proportional to the mileage involved. The distance between the offices are given in the following table.

	Main	*B1*	*B2*	*B3*	*B4*	*B5*
Main Office	—	*190*	*70*	*115*	*270*	*160*
Branch 1	*190*	—	*100*	*240*	*215*	*50*
Branch 2	*70*	*100*	—	*140*	*120*	*220*
Branch 3	*115*	*240*	*140*	—	*175*	*80*
Branch 4	*270*	*215*	*120*	*175*	—	*310*
Branch 5	*160*	*50*	*220*	*80*	*310*	—

(a) Explain how this problem fits in with the definition of the minimum spanning tree problem.

(b) Solve this problem.

Michael A. Trick
Wed Nov 8 14:14:52 EST 1995