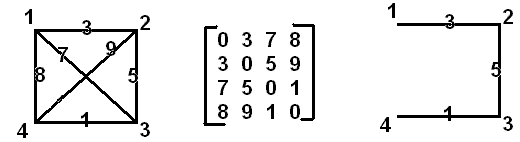
EECS 560 Lab 11 Week of April 11

A minimum spanning tree for a graph, as the name implies, is the spanning tree with the smallest cost (i.e. the sum of the weights of the edges in the tree is a minimum). If your graph has 10 vertices the MST for it will contain nine edges. Since you are finding a tree, what you end up with must, of course, be connected.

In this lab you are to find the minimum spanning tree of a graph in two ways—by breaking cycles and by using Prim’s algorithm. For the first method, identify cycles in the graph, and for each such cycle, remove the edge with the highest cost. (Do this one cycle at a time, don’t try to find all cycles and then start removing edges.) The easiest way to detect the cycles is use depth first search (See section 9.6 of the text or refer to lab 6.). Your input will be the cost matrix of a graph. Assume vertices are numbered beginning with 1.



In the diagram above, the graph is on the left, the cost matrix is in the middle and the minimum spanning tree is on the right. The tree was obtained as follows:

1. Remove the edge between 1 and 3 from the cycle 1,2,3,1

2. Remove the edge between 2 and 4 from the cycle 2,3,4,2

3. Remove the edge between 1 and 4 from the cycle 1,2,3,4,1

At each step the largest edge in the cycle was removed, and the tree that remains has cost 9.

Prim’s algorithm can be found in the class notes or in the Weiss text.

You are to find the minimum spanning trees for the graphs contained in files MMST1.txt and MMST2.txt (for Monday), TMST1.txt and TMST2.txt (for Tuesday) and RMST1.txt and RMST2.txt (for Thursday). Each file contains the cost matrix for one graph. Obviously, your two methods must produce the same results. For each graph, output the edges in the spanning tree. For the example above, your output would be (1, 2), (2, 3), (3, 4). For each ordered pair (u, v), u must be smaller than v. (Note: vertices are always numbered starting with vertex 1.)

The lab is due by midnight Saturday.