Group Project Survivable Least Cost Path Design

<Purpose>

In this project, you will learn and simulate how to find two link disjoint paths in the network so that the successful delivery from the source to the destination will be guaranteed when a single link is failure.

<Algorithm description and reference>

In the network routing problem, one of the most important problems is to find two paths from the same source to the same destination. These two paths do not share any link in the network so that when a single path fails, the destination can still get the data from the source.

We assume the network can be represented as the bidirectional graph with the positive cost on each link. The algorithm can be described as follows.

Step 1. Find the shortest path from the source s to the destination d by running Dijkstra's algorithm. Let P1 denote this shortest path from s to d unidirectional.

Step 2. Create a residual graph as follows: all the edge along the path P1 will reverse the direction with the same cost.

Step 3. Find the shortest path from the source s to the destination d by running Dijkstra's algorithm again on the graph in Step 2. Let P2 denote this shortest path from s to d unidirectional.

Step 4. Discard the reversed edges of P2 from both paths. The remaining edges of P1 and P2 form a sub-graph with two outgoing edges at s, two incoming edges at d, and one incoming and one outgoing edge at each remaining vertex. Therefore, this sub-graph consists of two link-disjoint paths from s to d.

This algorithm is named Suurballe's algorithm since Suurballe first proposed this algorithm in 1974. You may check this paper for the detailed algorithm description and fast implementation at http://www.eecs.yorku.ca/course_archive/2007-08/F/6590/Notes/surballe_alg.pdf

<Simulation Requirements>

You should first understand the whole algorithm before you start implementation. In your simulation, you are required to run the algorithm on both small-scale and large-scale networks (shown in Figs. 1 and 2). You are required to randomly generate the cost on each link (do not put the same cost) and find the two paths for each pair of nodes in the network.

<Submission Requirements>

You are required to give a demo at the last week of the class (date and location will be decided later). Also you should submit the source code and paths for each pair of nodes. Please also provide the code explanation and indicate any help from the website or other persons except instructors.

<Additional Points>

In the above simulation, we only guarantee the link-disjoint which means the two paths may go to the same node through different incoming and outgoing edges. If two paths go through the same node and this node fails, the destination still cannot get the data from the source. Now you are required to find two node-disjoint paths

so that these two paths can tolerant any single link or node failure. If you can always find node-disjoint paths, your team will earn 20 additional points for the group project.

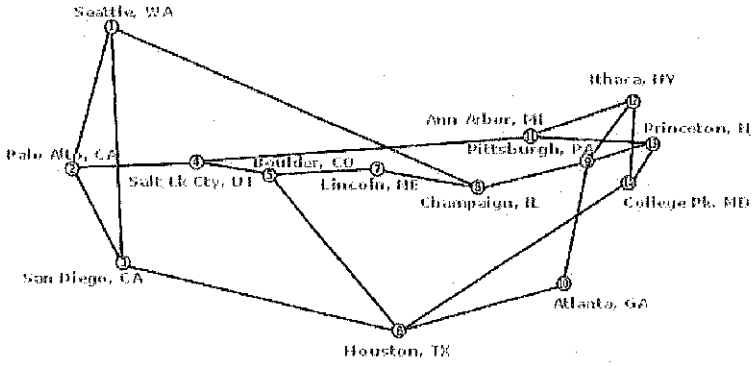

Figure 1. 14-Node small scale network (NSFNET)

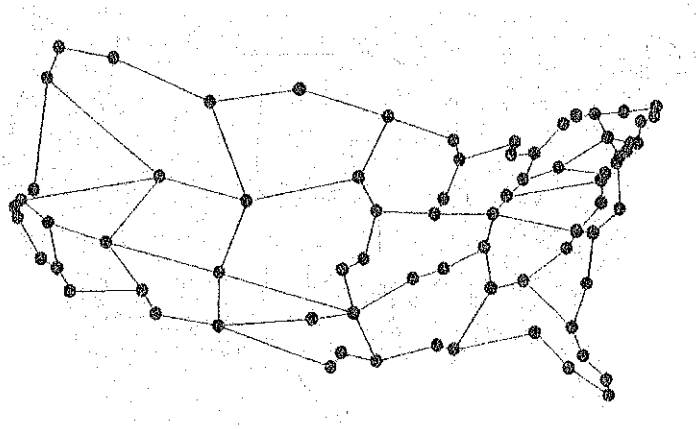

Figure 2. 75-node large scale network (CORONET)