**Project 4: network simulation and visualization**

Landen Black, Chris Dumond, and Brandon Henning

**Abstract**

For this project we were asked to create a network simulator composed of routers and links. These routers are to be located anywhere in the network by having their own IP address. For the simulation we are to be able to add routers to the network and creating links to other routers in the network. These routers are to have different properties including an IP address, the number of connected links of each router, a list of adjacent routers, link weights between each adjacent neighbor, and a forwarding table. Finally, we are asked to create a visualization tool so that we can visualize the current state of the network. We should be able to see the current active routers, links, and the link weights.

**Approach**

The project is written entirely in Java, using our custom router object to represent the routers in the network. Information regarding the forwarding tables and the neighbors for each router is propagated across the network simulation. Each router uses the total network information (i.e.they know the entire network graph) when calculating its own forwarding table using Dijkstra’s algorithm, but only uses the forwarding table and its known neighbors to send data about the network structure or to transmit user data from client to client. The visualization program shows the entire network graph that may reside entirely on the local client or partially on multiple clients. Routers are represented by the IP of the machine that is running the simulation the router resides on, followed by a routerNumber that uniquely identifies the routers on that machine (i.e. 172.30.197.132:0 denotes router 0 on the program running on the computer with IP 172.30.197.132.) This allows us to allow for router “processes” running on multiple computers for challenge 2. Router message types are simply denoted at the beginning of

the message (e.g. a “send” message is for sending a file from one computer to another, “request” messages are to request the forwarding table from routers - used when adding new routers to the network, and “update” messages are sent in response to a request for updated forwarding tables.

**Resources used**

To implement this project we used several native Java classes and interfaces. We used the class ArrayList<> to create List<String> objects for identifying neighbors of each router. We also used this to put the address and a new ArrayList<> into the neighbors Map<String, List<Link>>. We used a HashMap to create a new neighbor Map<String, List<Link>>, that lets us create the links for the routers.To create the visualization of the network we used Jpanel with Jbuttons.

**Challenges**

We were able to implement all three challenges. To show the routing table for each router, left click on a router in the visualization window. The routing table will show up in the top left corner. To connect multiple computers, run the program on both, and type in the source routers of both visualizations in one of the computers visualization program to create a router that connects the two computers together. To send a file from one computer to the other, type in the “send” command, along with the file name, a source router, and the destination router, all separated by dashes. The destinations computer’s IDE will display a message that it has received a file, along with the path taken.

**How to run/use**

**To add a router:**

Type the router name (IP:routerNumber) of the routers that the new router should be connected to. For example, to connect the new router to a router represented on your computer, type your own IP, a colon, and the routerNumber(s) on your computer. To connect the new router to multiple routers, separate the addresses with commas. To connect to a router on another client, type their IP and the router number on their computer.

    Ex: (“172.30.197.132:0, 161.6.151.161:0”) to connect the new router to each computer’s respective router #0.

**To see link weights and router’s forwarding table:**

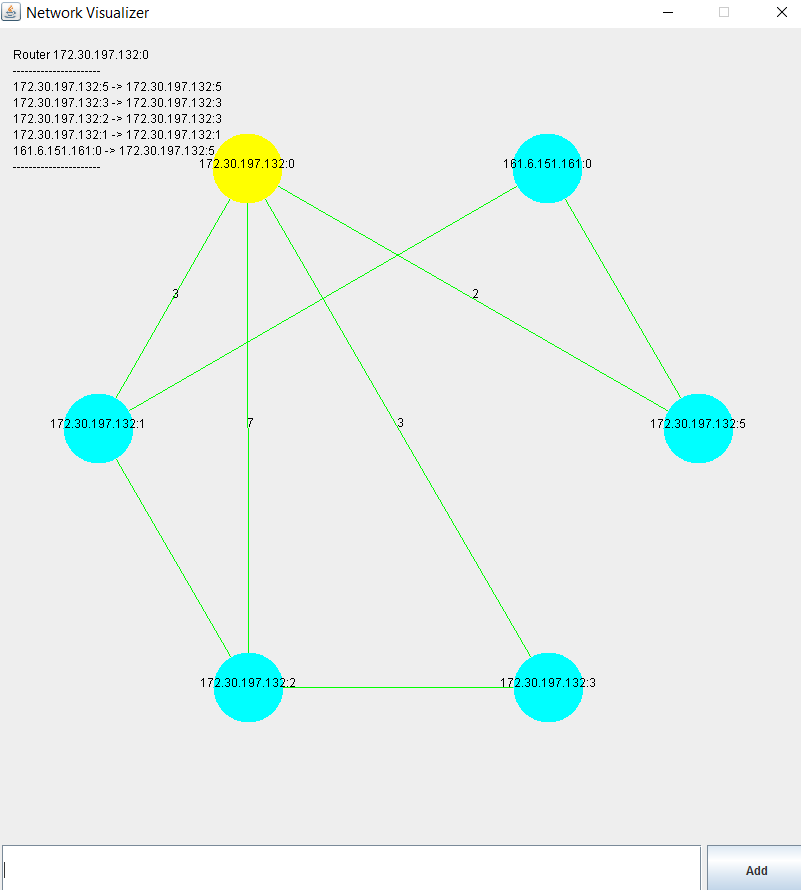
    In order to see the link weights and the routers forwarding table, left-click on the router in the graphical representation. This will display the weights on the links, and will show the routing table of the selected router in the top left corner of the visualization.

**To remove a router:**

    To remove a router from the network, right-click on the router in the graphical representation. This will remove it in real time from the network.

**To send a file:**

    You are able to send a file through our simulated network. To send a file, enter the format “send-{filename}-{sourceRouter}-{destinationRouter}” without the quotes or brackets. Received files are named as the address of the destination router. Path the file takes is also printed in the console to verify that the file took the path with the smallest weight.



**Problems overcome**

A problem we ran into was thread sensitivity was Java Swing not being thread safe. To overcome this we used the final Timer. This combined with the visualizer only allows you to add one router at a time makes it so routers are not created simultaneously.