For each node (H1, R1, R2, R3, R4, H2), I developed one tcp client and one tcp server.

In the project, I create 4 .java file:

-server.java(server for each node)

-client.java(client for each node)

-Graph.java(for the Bellman-Ford algorithm and write result to the .txt file)

-runBellman.java(for output the result)

Also, I create 6.txt file, Each of them has the weight of the edge. They are “H1R1.txt”, “R1R2.txt”, “R1R3.txt”, “R2R4.txt”, “R3R4.txt”, “R4H2.txt”.

For running the project, first we need to compile the .java file.

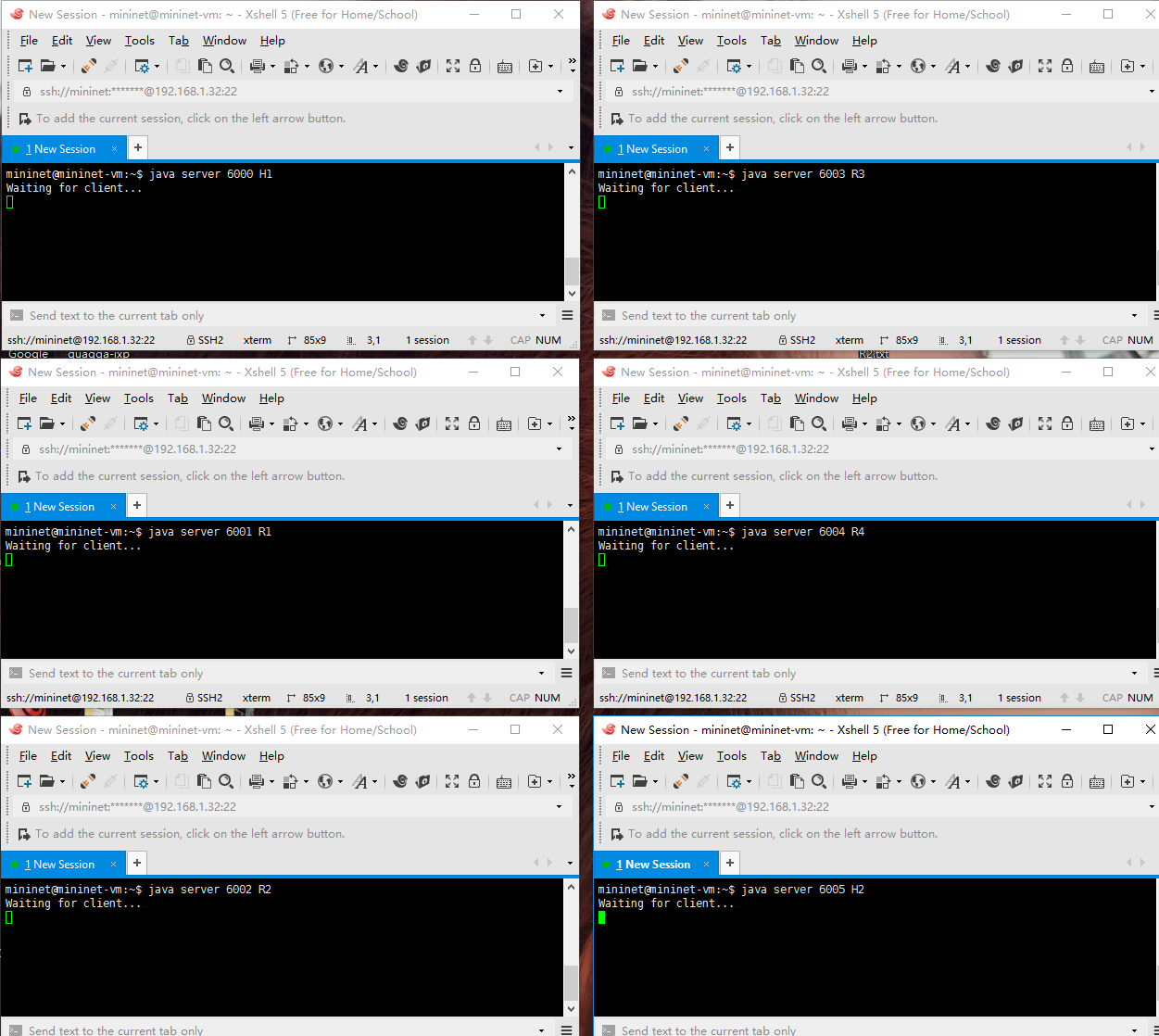
-javac server.java

-javac client.java

-javac runBellman.java

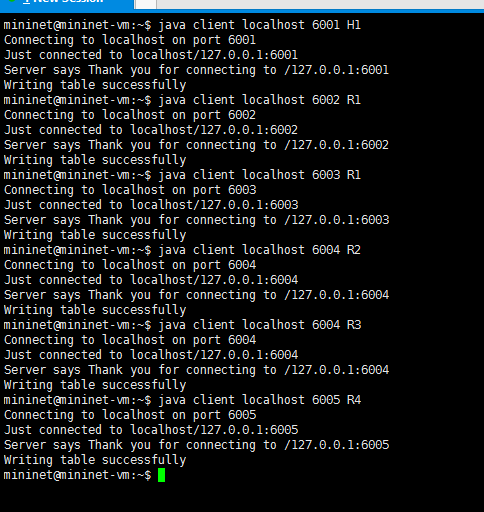
First, we need to start all six nodes’ server, the command is

--java server <localhost post number> <Node>



Then, we will run client. The command is

--java client < next hop neighbor address> <next hop neighbor port> <Node>



Each time the server get the request, server will collect “fromNode” and “Node” to read the .txt file to get the weight. And update the topology into the “beginning.txt”. After we run all the client, we will collect all the edge weight for the topology.

After doing this, we collect information about the topology, and write them into a .txt file, named “beginning.txt”.

Then we run command:

-java runBellman

Then we will get result for each node, which is wrote into six “node.txt” file

Bellman-Ford Algorithm:

The program takes 2 arguments. First is the graph(topolygy: build by beginning.txt), the second is the source node as a String. The output will be wrote in a .txt file, which means the cost to each node from the source node. In the algorithm, we keep an array to save the weight to each node. if the weight is smaller, it is written to the array to cover the previous data. Totally run at most (V-1)E times. Where V is the total number of nodes and E is the total number of paths between nodes. If a negative cycle is found after these many steps, that means the presence of a negative loop, In such case the algorithm prints error. In other words, It works with negative edge but prints error if it finds a negative cycle. I got the algorithm idea from the following website:

<http://www.geeksforgeeks.org/dynamicprogrammingset23bellmanfordalgorithm>

ps:

I treat the topology as a directly connected link.