**Project Report #3: Distributed Procedures for Asynchronous Vector Routing**

Daniel Hanna

Computer Networking (CSC 360)

The College of New Jersey

Department of Computer Science



This project involved developing routines for distance vector routing given a network. It involved around writing two specific routines for each node: rtinit0() and rtupdate(struct rtpkt \*rcvdpkt). The network topology is shown below.

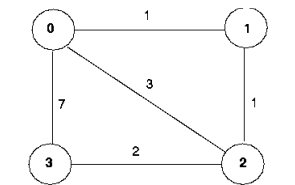


Figure 1: Network Topology

Students were given 5 C-files to be utilized for this project, 4 of which needed to be modified – node0.c, node1.c, node2.c, node3.c. These four files maintain extremely similar code, and thus in explaining the details of one of the files, understanding can be obtained of how the remaining files operate.

Within the node0.c file, the two aforementioned functions rtinit0() and rtupdate(struct rtpkt \*rcvdpkt) were developed. Within rtinit0() counter variables *i* and *j* were defined, along with pkt0. The source of pkt0 is the current node, which is node 0. Clocktime is also utilized to keep track of the timing of events as they occur.

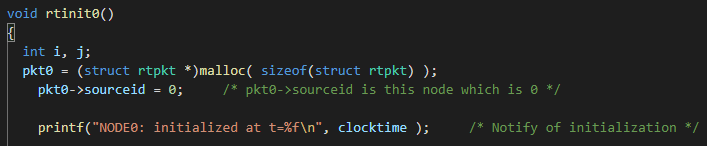


Figure 2: Defining pkt0->sourceid

Additionally, known costs were defined.

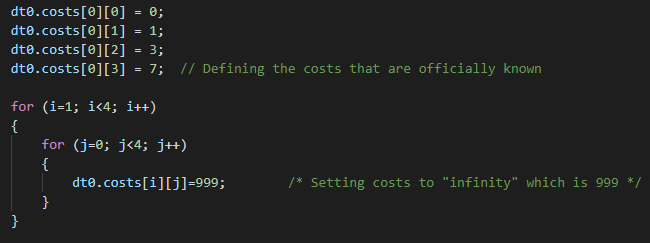


Figure 3: Known Costs

The minimum costs between nodes was then defined. The cost of a node to itself was defined as 0, and costs between non-direct nodes were assigned a value of 999. Thereafter, the resulting distance table was printed, and the source id was defined. The pre-provided function tolayer2(\*pkt0) was then utilized for sending the packet to all directly attached nodes.

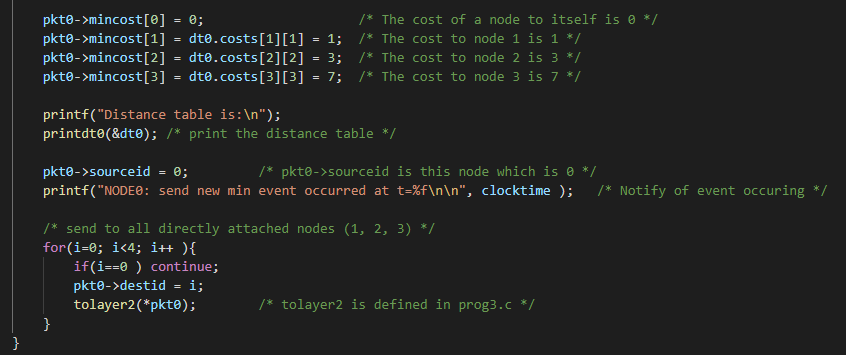


Figure 4: Defining Min Costs and Sending to Other Nodes

The rtupdate0() function was then developed, wherein updates for a change are implemented and the distance table is updated.

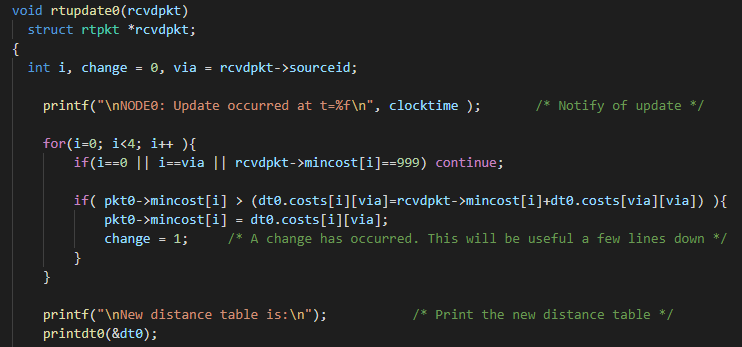


Figure 5: Implementing Change/Update

In the advent that a change has occurred, the packet is sent to all directly attached nodes.

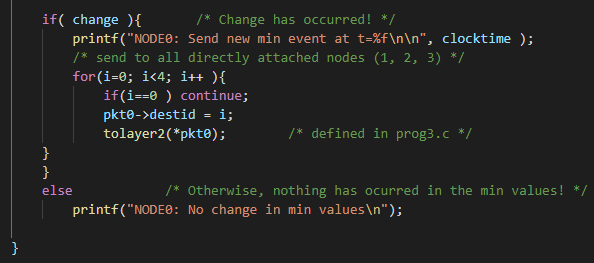
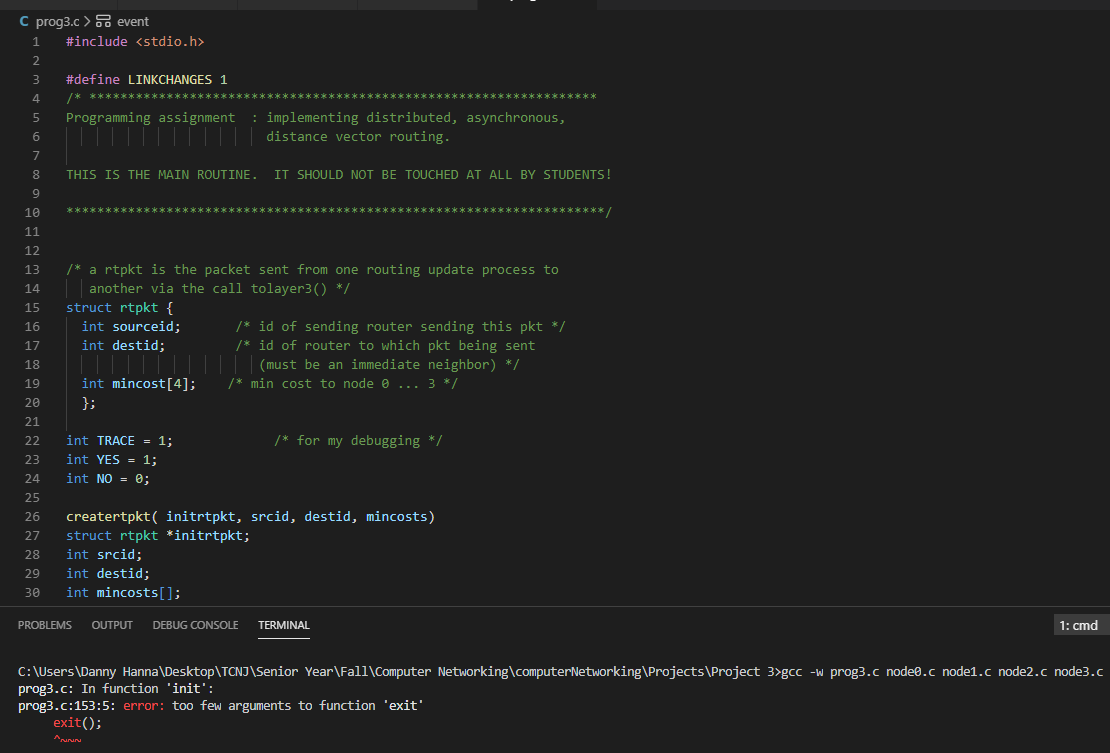
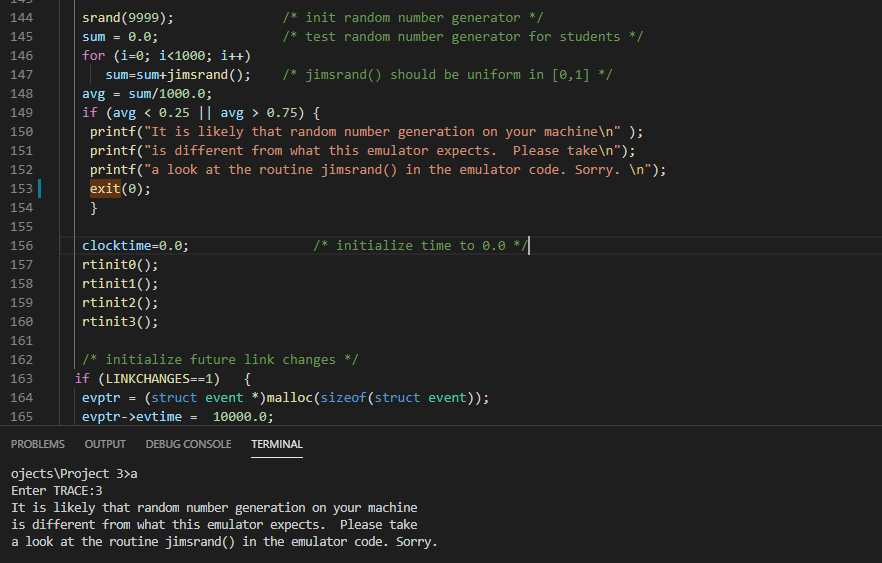


Figure 6: Sending to Nodes

The program is compiled in the terminal for testing. It should be noted that the pre-provided prog3.c file had several issues present therein. For example, invalid syntax was present for a line which contained exit(). Additionally, after this problem was fixed, a logic error was present. These issues are seen in the figures below.





Figures 7 + 8: Errors in Given Code

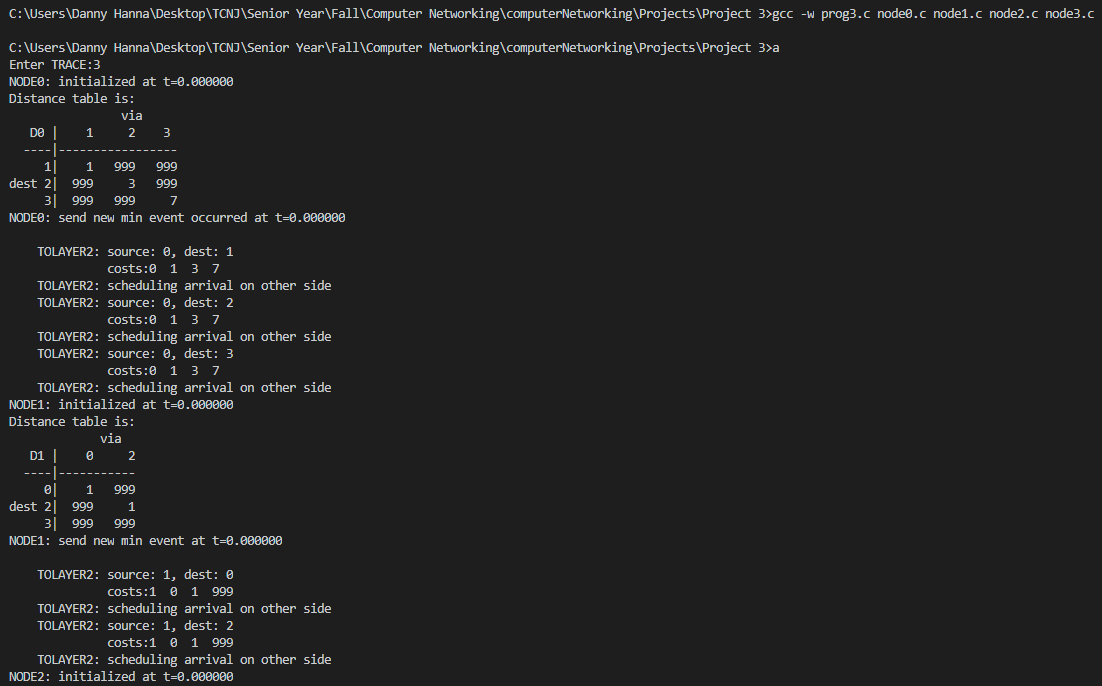
After addressing these issues, the code could be properly tested. A portion of an example output is shown below. As a result, the project was completed successfully.

Figure 9: Successful Output