Routing and Forwarding Packets with C++ Simulation

Vanessa E. Sanders

Kent State University Honors College

Computer Communication Networks

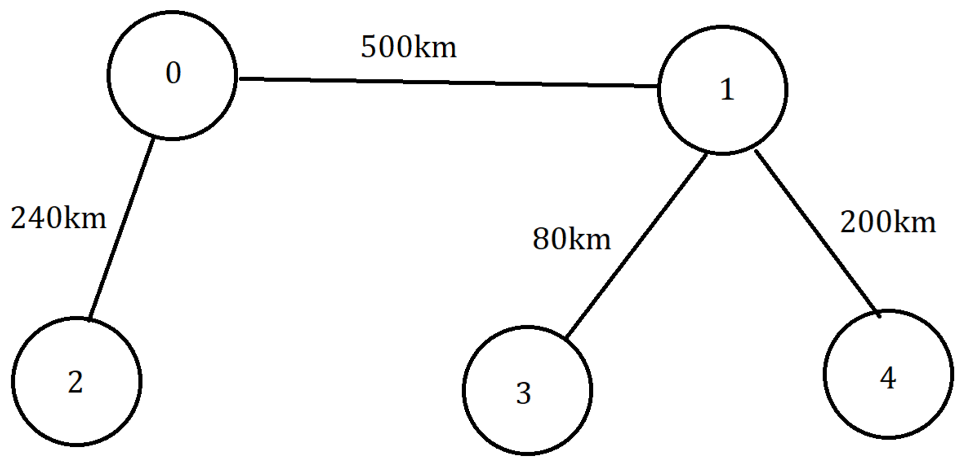
Dr. Hassan Peyravi

Routing and Forwarding Packets with C++ Simulation

The network layer, otherwise known as the “Internet Layer” of the Open Systems Interconnection Model (OSI Model) is responsible for transferring data from one node to another with the help of various networks. This layer is also present in the TCP/IP model, but their location in the sequence of layers differs between the two models. My project demonstrates a subset of the functionality of the network layer. The purpose of my code is to initialize a system which consists of nodes within a network, route packets between these nodes, and calculate the overall delay.

The first step I took in creating this program was deciding what classes I would need to implement. The classes that were necessary in creating this simulation were: Packet, Router, Event, Network. The Packet class contains a subset of the data that is stored in a TCP/IP packet header. For the sake of simplicity in my project, my Packet class contained information such as, but not limited to, id number, source node, destination node, and delay. The nodes within my network are represented by the Router class. Each Router object has a queue that stores the packets that arrive and a routing table that allows it to determine the node in which it needs to forward a packet in order for it to reach its destination, along with its own id. For the sake of my project there are no loops that complicate the routing tables. The Event class is extremely important and composes the driving force of the simulation. Events hold information regarding the time in which something will occur, where it will occur, to which packet it will occur and what the event type is. The two event types are Arrival and Departure. A packet reaching its destination is handled within the function for packet departure, rather than having its own Event. The Network class is composed of the other classes and has its own interface that handles the arrivals and departures on the network level. The Network contains the overall driving force, which is a priority queue of events. This priority queue allows events to occur in a sequence that the delay determines.

The main program is where I initialized the routing tables for each network to determine what the network would look like, which can be visualized as such:



The nodes are numbered 0 through 4 and the distances are an approximation that allowed for me to determine the appropriate delay range that would occur between a departure and an arrival between the different nodes. On top of that, each node has its own delay related to processing the packet. For instance, a departure from node 0 to node 1 may have a delay range of 30 to 40 milliseconds, whereas the delay from node 0 to node 2 might be from 10 to 17 milliseconds. I also made sure that the delay going in one direction would not be the same for another.

Because the delay range is predetermined by the programmer, you should be able to estimate the delay for each packet. For example, in my program I send a packet from node 2 to node 4. I know that this will make 3 departures and 3 arrivals. The departure from node 2 will add approximately 15 ms the arrival at node 0 will add approximately 5 ms the departure from node 0 will add approximately 39 ms the arrival at 1 will add approximately 7 ms the departure at 1 will add approximately 38 ms and the arrival at node 4 will add approximately 6 ms. Overall adding them together will give you 110 ms as the total packet delay. When running the program 5 different times the packet delay for the packet sent from node 2 to node 4 was:

Execution Number	Delay For TestPacket4
1	92ms
2	131ms
3	69ms
4	148 ms
5	102 ms
Average	108 ms

The estimate that was roughly calculated from adding the average number of delays for each departure and arrival that would occur on the route from node 2 to node 4 was 110ms and the actual average delay was 108.4ms. The percent error in the estimation using the averages is 1.81%. This percent error is acceptable, although I believe that it would have been more precise given the usage of doubles instead of integers.

For this project, I learned more in depth about the functionality of routers, especially the use of a routing table. I also learned a lot more about the importance of creating simulations and building a model of a network. It was easy to see the value of modeling the network in order to estimate delay. If the network were larger and consisted of loops, a well constructed simulation would be extremely useful in understanding the paths and their delays. Before arriving at the final program, I was working with the idea of dropping packets, and although it was not used in the end product, it was extremely helpful in my understanding of the queuing system and how packets get lost. While I was researching, I also learned about the structure of TCP/IP packet headers which was important for this project. My favorite part of the project to implement was the Event class and the priority queue / Event list that drove the order of packet departures and arrivals. This structure was new to me, and I really enjoyed learning about its use in simulations. To conclude, the program was successful in routing packets and accumulating the delay over the time the packet spent in the network. I learned a lot throughout the process and I would love to continue researching and building on to this project given more time.