Lab 19 :: Queues and Routing Data Packets

The goal of this project is straightforward enough: Design the Router so that packets aren’t dropped and packets get to their destinations as quickly and fairly as possible.

There are two parts to the assignment

1.) Write an implementation of AP Queue
2.) Modify the Router class to better manage network traffic (hint: queues will help)

Requirements:
You are required to do all of part I and at least the first two steps of part II.

Part I:
Write an implementation of AP Queue. You may use java.util.ArrayList or java.util.LinkedList to do so, if you like. Test it sufficiently to make sure it works.
You should also add two methods to your implementation that will be helpful for the QueueNet project that follows:
1. add a constructor that accepts an int that represents the maximum size of the queue. The no-args constructor should set this value to something indicating no max size has been set, like -1.
2. Add a method boolean isFull() which returns true if the max size has been reached. In the case where no max size has been set this method will always return true.

Part II:
The second part will take more time, but it is more interesting. I recommend taking each of the following steps and testing it out after each one.

1. Make a simple router
Make a queue to store all incoming packets. In the run() method, dequeue a packet, determine the DataWire it needs to go out on, wait for the wire to be available, and then put the packet on the wire. At this point you will also have to set a limit on the size of your queue. That limit should be somewhere shy of the point when you run out of memory to handle all of the packets. You could also fix the size to something smaller in order to observe the behavior. If the packet arrives and the queue is full, you should drop the packet.

2. A slightly better router – implementing utilization
The problem with the simple Router is that some DataWires may not be fully utilized since there may be packets that could go out on other data wires but they’re blocked by the packet at the front waiting for its line to be open.
Fix this by making a separate Queue for each DataWire. When a packet arrives place it in a Queue with all other packets bound for the same destination. In the run method, attempt to put a packet on each of the data wires.

3. Even better router – implementing fairness
The problem with the 2nd version of the router is that one data source could hog one of the outgoing lines by flooding it with packets. Imagine host A sends 1000 packets to Host F. Meanwhile, poor old Host B who just wants to send 1 packet to Host F might have to wait for 1000 packets to clear the queue. This isn’t fair. To fix this, make separate queues for each outgoing wire that differentiate between source hosts. That is to say, we’re talking about a set of queues for each data wire. Continuing with the example, the DataWire going to host F would have 2 queues associated with it. One for packets arriving from Host A and another with packets from Host B. Packets are placed on the wire by rotating through all of the queues for the wire.

4. Improvements from this point will involve modifying and improving the simulation code which you are free to do. You also may wish to investigate some question; design an experiment and collect data. For example:

- What’s your router’s rate of throughput? This can be calculated as the number of packets sent out per second (or minute in this slowed down world). Does your router achieve the maximum theoretical throughput? The theoretical max throughput would be equal to the number of packets per second forwarded by your router IF it could send packets at the maximum transmission rate of each wire. How close to the max are you?
- Can you give priority to packets coming from certain sources or going to certain destinations? How does this affect the throughput from the hosts perspective who has priority vs. hosts without priority?
- Make a graphical representation of the queue(s) inside your router.
- Make a bigger network. Add hosts. Add routers. If you design a network that has multiple paths between any pair of hosts, how will the routers determine which path to take?
- In real life DataWires also have a maximum amount of data that can be sent in any one packet. Add this limitation to the DataWire class. If packet arrives a payload that’s too big, break it up into multiple smaller packets that can be sent.
- Etc.

You may also consult Mr. Franke with your ideas for improvements.

**Turning it in:**
YOU MUST WRITE something into the README file describing how far you got with the project. What did you implement? Did you try to do anything beyond the requirements? If so, what? Is there anything I (Mr. Franke) should do, or should look for, while messing with your simulation?