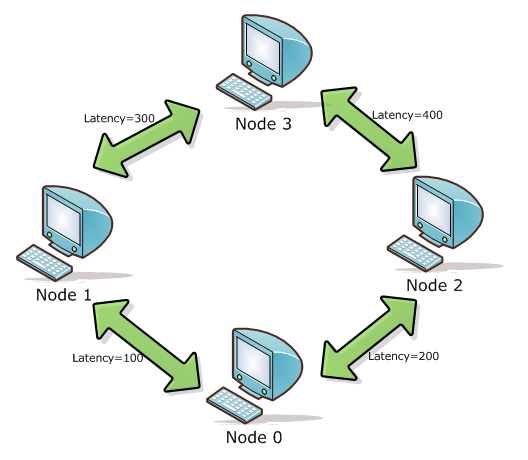
Homework1

Question1: Bidirectional links and no loss

The first topology we build upon our implementation is illustrated in the figure below.



Four nodes are connected in the way to form a ring network. The links are with no loss rate and has various latencies in milliseconds.

We obtain the following results by printing the current time when handling the FloodDoneEvent in the application component on each node:

Node 0 -> Done flooding message: "hello" at 1202054597421

Node 1 -> Done flooding message: "hello" at 1202054600508

Node 2 -> Done flooding message: "hello" at 1202054601501

Node 3 -> Done flooding message: "hello" at 1202054599421

One can see that node 0 finishes first, followed by node 3, node 1 and node 2 are among the last respectively, the order in which the food is done at each process is within our expectation.

Node 0 initializes the flooding by sending the flood message to node 1 and node 2, when node 1 sees the message it floods forward to node 0 and node 3, so does node 2. As soon as node 0 receives message back from node 1 and node 2, it finishes the job. The time spent in the transmission for node 0 to complete is 200\*2 = 400ms.

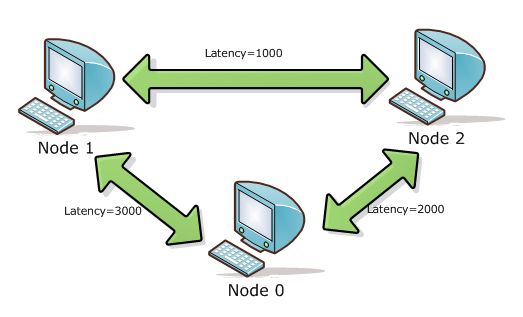
Node 3 sees the message from node 1 first (and flood it to node 1 and node 2), immediately followed by the same message from node 2, then it is done, the time spent in the transmission for node 3 to complete is 200+400 = 600ms.

Then node 1 gets the message from node 3 just before node 2 does, and they are done. The time spent in the transmission for node 1 to complete is 100+300\*2 = 700ms, whilst for node 2 is 200+400\*2 = 1000ms.

Notice that time calculated above counts from the start of the initial flooding by node 0, and the sequence of the time matches the order of termination.

If we assume the calculation and processing time on each node is more or less the same, than the interval between transmission times also matches the interval between system-times printed out.

The second topology is shown as following, three nodes are connected together, and each node has two links to the other two nodes respectively.



The result is as follows:

Node 0 -> Done flooding message: "hello" at 1202071564765

Node 1 -> Done flooding message: "hello" at 1202065595091

Node 2 -> Done flooding message: "hello" at 1202065596049

The order can be sorted as node 1, node 2, node 0, from top to down, earliest to latest.

The initial message flooded by node 0 arrives at node 2 1000ms earlier than at node 1, as soon as node 2 sees the message it floods the message to node 0 and node 1.

Meanwhile, node 1 gets the message and floods it to node 2 and node 0.

Because the latency of link between node 1 and node 2 is much smaller than the links between node 2 and node 0 as well as node 1 and node 0, node 1 and node 2 will end their job much earlier than node 0.

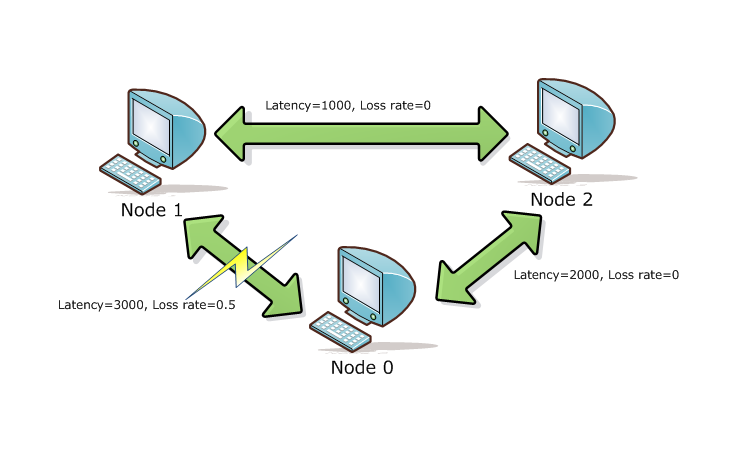
The time spent in the transmission for node 1 to complete is 3000ms approximately.

The time spent in the transmission for node 2 to complete is 2000 + 1000\*2 = 4000ms.

The time spent in the transmission for node 0 to complete is 3000\*2 = 6000ms

Question 2: Bidirectional links and loss rate = 50%

The topology with loss rate is shown as following, three nodes are connected together, and each node has two links to the other two nodes respectively, and all links have loss rate which is 50%.



**The results are as follows:**

|  |
| --- |
| Scenario A: |
| Node 0 -> Start flooding message: "A" at 1202071558437 |
| Node 0 -> Done flooding message: "A" at 1202071564765  Node 1 -> No flooding Done  Node 2 -> Done flooding message: "A" at 1202071562749 |
| Package Lost:  Node 0 -> 3313 INFO [DropComponentFactory:DropComponent] {DropComponent} Message FloodMessage to 1 dropped |

|  |
| --- |
| Scenario B: |
| Node 0 -> Start flooding message: "B" at 1202071978624 |
| Node 0 -> No flooding Done  Node 1 -> Done flooding message: "B" at 1202071981843  Node 2 -> Done flooding message: "B" at 1202071982734 |
| Package Lost:  Node 1 -> 9860 INFO [DropComponentFactory:DropComponent] {DropComponent} Message FloodMessage to 0 dropped |

|  |
| --- |
| Scenario C: |
| Node 0 -> Start flooding message: "C" at 1202072847890 |
| Node 0 -> Done flooding message: "C" at 1202072854015  Node 1 -> Done flooding message: "C" at 1202072851109  Node 2 -> Done flooding message: "C" at 1202072852031 |
| Package Lost:  No Package Lost! |

In this question we focus on the loss rate. From the topology picture, we know that there is a loss rate 50% between Node 0 and Node 1. In other words, message from *node 0* to *node 1* may drop as well as the opposite direction.

In Scenario A, *node 0* loses a package to *node 1*, since each message will be send only once by each node, *node 1* will never get message “A” from *node 1*. So *node 1* will never finish its flood. Although *node 0* failed to send message to *node 1*, *node 1* can still get the same message from *node 2*. And *node 1* will send the message to both *node 2* and *node 0*. Finally *node 0* gets a message from *node 1* and finishes flooding.

In Scenario B, when *node 1* sends the message to *node 0*, the message drops. And node 0 will never get the message from *node 1*. So node 0 cannot finish flooding.

In Scenario C, no package lost. So all node finish flooding.