Project 3 Tapestry Bonus

Input

For the bonus part, we included and extra argument in our input. We added a third argument which will be the percentage of Total nodes suffering from failure or fault.

Working

We randomly selected the given percentage of nodes as Inactive or failed nodes from the namespace. We basically try to find an active neighbor if a target or hop is inactive. When a hop is about to happen, there is a call to next hop target for whether it is active or not. In a more real world scenario this would ascertain that if a positive reply is returned from the next node then that means it is in Active state otherwise a no reply or Inactive state reply would tell that it is Inactive.
In our implementation, we request the node for its status, whether Active/ Inactive. If the next hop node is Active then we continue with hopping otherwise, we find the closest active node of that inactive node.
We have handled certain corner cases like when the next active node crosses the maximum number of node allowed, we send it to the lowest numbered node in order to handle index out of bound error. We also had to handle an infinite loop condition where due to the target node being inactive we keep going in a loop to find the next active node near the target.
Performance wise, finding the next active node along with handleCall method increase the execution time by slight amount as we have to use a Call to get the reply back as Active/ Inactive status.

Findings

We noticed that in a faulty network, without even updating the routing points, it is possible to do the routing with the help of finding the next active node. For example - An inactive node 4000 can be used to find next active node 4001 or 4002 which can further help find target node 420A. Therefore, we can observe that Tapestry is resilient to faulty nodes.