**Team-members:**

Kirti Desai, UFID – 04541017, [desai.kirti@ufl.edu](mailto:desai.kirti@ufl.edu)

Pratik Loya, UFID – 31716903, [ploya@ufl.edu](mailto:ploya@ufl.edu)

**What is working:**

* To test fault tolerance we have included one extra input parameters which should be either “true” or “false”
* When False: Program execution will be executed in usual without any removal of Node.
* When true: - Program will execute with creation of routing tables of nodes and dynamic addition of nodes into the tapestry. Post creation of the network, one random process is killed. Even if one node is removed from the network, program is able to route to the destination node and calculate the maximum hop count without failure.

**Run Command:**

* Go to directory desai\_loya/project3\_bonus/
* Run Command to count maximum hops without any failure node: (number of argument - 3)

“mix run project3.exs num\_nodes num\_request false”

* Run Command to count maximum hops with introducing a failure node: (number of argument - 3)

“mix run project3.exs num\_nodes num\_request true”

**Fault-tolerant Analysis:**

When a source node starts routing to a destination node, it looks up in its own table and calculate the nearest node present to the destination and hop into that nearest node and look for the destination or the next nearest one in its table. When it encounters a failure node (which is not present anymore in the network but still present in its routing table), it removes it from its routing table and finds the nearest node to the deleted node and update it in its table. At the same time it hops into this nearest node to the deleted node and search for the destination or next nearest node. Even in the case of failure of a node, it manages to give maximum hop count exhibiting the fault tolerance nature of the tapestry algorithm in a distributed network.