## Project 3 Bonus

## Failure Implementation in Pastry

0

### Submission by :

### Sree Pardha Saradhi Gunnam (UFID : 70304545) (pardhug@ufl.edu)

### Nitin Agrahara Ravikumar (UFID : 73714398) ([arnitin@ufl.edu](mailto:arnitin@ufl.edu))

The Pastry implementation was done as described in the paper Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. by A. Rowstron and P. Druschel.

We have a running network on which a pastry network is built. An equivalent of the Join as described in the paper gets the pastry network up and running.

The failure is introduced as such: Before the message passing starts, we force kill a number of actors(pastry nodes). The number of nodes to kill is a input parameter. Default is 30. Thus an arbitrary number of nodes are killed before message passing starts. Now, it is important to note that the routing tables and leaf tables are not affected by the kill. This is to ensure that failure is forcibly encountered.

Once the nodes are killed, we enforce a delay of 5 seconds just to make sure that all nodes have exited.

Then we start the message passing. Our implementation follows the requirements specified for the first part. The number of requests is an input parameter using which the program initiates the message passing.

In case the route to the destination from a source contains nodes that are all alive, then well and good, the routing proceeds without glitches. In case the route to destination involves a dead node, then upon knowing that the node to go next is dead, rerouting is done. The nearest node to the dead node is calculated and the routing is done through that node. This is because, when the message reached the current node, it did so because a certain number of bits were matching. Then it is safe to assume that the next nearest node will also have a certain number of bits matching. This is the part that affects the hops. Since we waver from the original path some extra hopping may be required to fall back on the path to the destination.

We observed that number of nodes killed is proportional to the number of hops that the message needs to reach the destination. As more and more nodes are killed, the number of hops required to reach the destination increases but the message does finally reach the destination within log 2bN hops.

Our hashing function is a binary method. Hence the number of bits involved is two (0 & 1) making the log base 2 for all practical purposes in this project.

Hence a pastry network containing 1024 nodes will have mandate that a message be passed in 10 hops. Similarly 600 pastry network will mandate 9 hops or less

We observed that for following:

The left column is the number of nodes and the middle column is the number of nodes killed and the right most column is the average number of hops. The number of requests being sent by all nodes is 10.

With Failure

|  |  |  |
| --- | --- | --- |
| Pastry Nodes | Nodes Killed | Avg. No. Hops |
| 16 | 10 | 2.5 |
| 32 | 16 | 3.2 |
| 64 | 30 | 3.5 |
| 128 | 40 | 4.2 |
| 256 | 100 | 5.0 |
| 512 | 200 | 5.4 |
| 1024 | 400 | 5.6 |
| 2048 | 8000 | 6.5 |
| 4096 | 1600 | 6.8 |

Without Failure

|  |  |  |
| --- | --- | --- |
| Pastry Nodes | Num Requests | Avg. No. Hops |
| 16 | 10 | 2.417 |
| 32 | 10 | 3.24 |
| 64 | 10 | 3.8 |
| 128 | 10 | 4.45 |
| 256 | 10 | 4.98 |
| 512 | 10 | 5.4 |
| 1024 | 10 | 6.01 |
| 2048 | 10 | 6.48 |
| 4096 | 10 | 6.97 |

The results show that with our without failure, the number of hops is almost the same.