PROJECT 3-Bonus
PASTRY

Lavanya Chennupati(UFID: 1129-9429)
Sowmya Duggimpudi(UFID: 1171-2252)

Working

Pastry is self-organizing routing overlay protocol, where each node routes the message until the message reaches the destination.

Implementation

Pastry is implemented mainly in the overlay networks in which each node is assigned a unique identifier (nodeId). The network is created based on the number of the nodes given through the input value numofNodes. The id assigned to a node is 128 bit long and is assigned to its physically closest node based on the proximity metric.

When a single node successfully joins the network, the subsequent nodes are started by the pastry manager. When a new node joins the network, its leafset, routing table and the neighborhood set are updated based on the node which it already knows in the network. After the updation is done, its state is sent to all the nodes in its table so that their entries can be updated. Once it is done the new node sends an acknowledgement to the Manager which approves that the node can join the network. The nodes can be changed dynamically in the network i.e. a node can join or leave the network as needed,

Now the nodes can send the messages to one another and the number of requests is given by the variable numofRequests. Once the messages are sent successfully, the Manager is notified which calculates the average number of hops. This converges to the value log(numofNodes)/log16. We take log16 as the base of the node id is 16 (b=4).

Node Failure

Whenever one or more nodes leave the network, we cannot guarantee the delivery of the message and hence the convergence also can't be attained. In such cases when we try to update the tables for failed nodes, the network may be lead to an in-consistent state.

Run

scalac project3bonus.scala
Case 1 : By default only 1 node fails in the network

scala project3bonus <numofNodes> <numofRequests>
Example : scala project3bonus 100 10

Case2 : Give the numofFailedNodes in command line

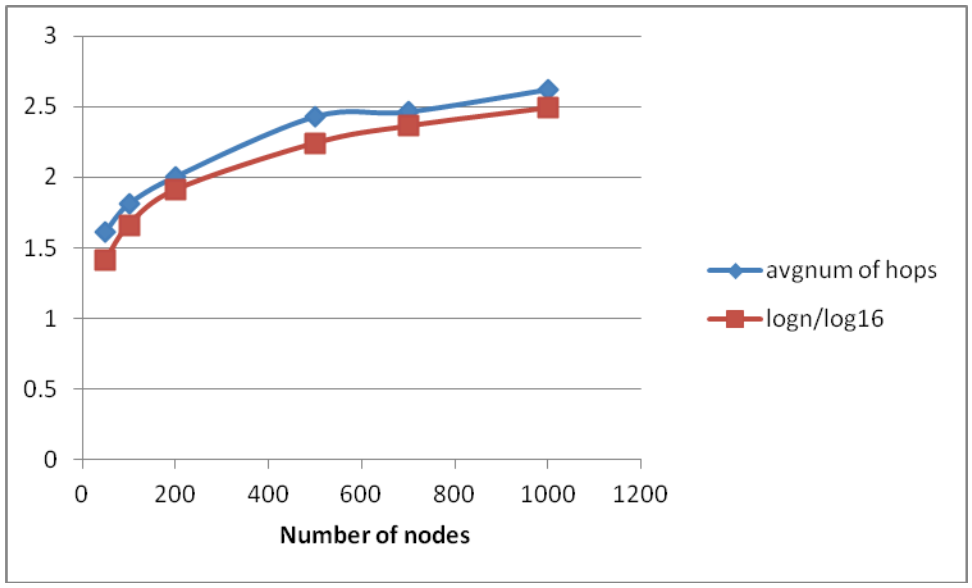
scala project3bonus <numofNodes> <numofRequests><numofFailedNodes>
Example : scala project3bonus 100 10 3

Largest Problem Solved:

Observations:

No.of nodes	No. of failure nodes	Number of Requests	Average No. of Hops	Log(NumNodes)/Log(16)
50	3	10	1.613	1.411
100	10	30	1.810	1.661
200	30	50	2.004	1.911
500	2	10	2.431	2.242
700	3	10	2.465	2.363
1000	3	20	2.622	2.492

Graph :


3
2.5
2
1.5
1
0.5
0
0
200
400
600
800
1000
1200
Number of nodes
avgnum of hops
logn/log16