COP5615- Distributed Operating System Principles

Project 3

Developed by –
Shaik, Fayaz - UFID: 6326-9935
Mullapudi, Aseesh - UFID: 9175-1971

1) Aim –

The aim of this project is to implement Pastry protocol in FSharp using Akka Actor model and observe the average number of hops for different values of NumberOfNodes and NumberOfRequests.

Input: The input provided (as command line to the project3) will be of the form: project3 numNodes numRequests
Where numNodes is the number of actors involved in the pastry protocol, and numRequests is the total number of requests that each actor requests its connections.

Output: Print the average number of hops that must be traversed to deliver the message.

2) How to Run –

Once Dotnet, Akka.Net and Akka.Fsharp are all installed in the Project environment, extract the zip file, and run the program in the following manner:

Command: dotnet fsi “--langversion:preview” proj3.fsx <numberOfNodes> <numRequests>

Where numberOfNodes (The number of nodes in the network), numRequests (The total number of requests) are the command line arguments.

3) Methodology –

i) Pastry is an overlay protocol where it is a self-organized protocol in which each node routes the message till the message finally reaches its destination.

ii) Using Akka Actor Model in FSharp, we created each actor as a node in the network. These nodes communicate with each other using message passing between actors.

iii) Each node in the network is assigned a nodeID which is unique. The network is created first and then the requests are routed to their respective destinations. Each node puts out "numRequests" requests.

iv) When a node enters the network, its routingTable, leafSet and their neighborhoodSet are updated based on the information that it gets from node that is already in the network.

v) Once the updation is done, it sends its state to all the neighboring nodes so that their tables and sets are updated. Once, this part is done, the master node is sent an acknowledgement which then approves the entry of the new node.

vi) Once a network is created, each node sends "numRequests" requests, as the requests are satisfied, the master node takes care of taking count of numberOfHops needed for all the requests, which finally helps in calculating the average number of hops.

vii) The average number of hops converges to be less than or equal to the $\log_{2^b}$(numberOfNodes) where b in our implementation is 2. That means we took ceil($\log_4$(numberOfNodes)) as the baseValue and our nodeID space is $4^{baseValue}$ (as b = 2).

4) <u>Project Questions Answered</u>:

i) *<u>What is working</u>* –

The pastry protocol is implemented and tested for different values of numberOfNodes and numberOfRequests.

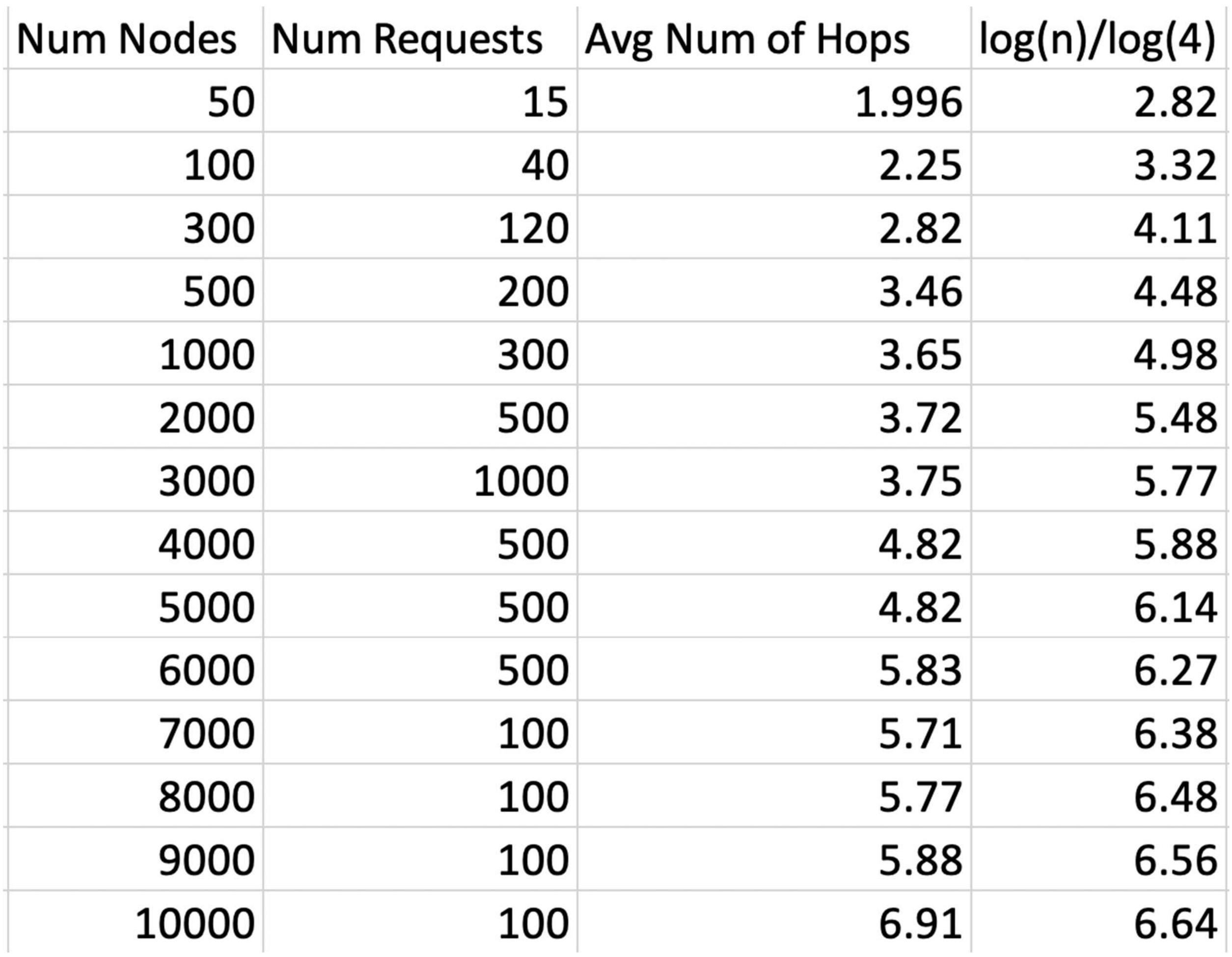
<u>Observations</u> –

a. The convergence of "average number of hops" is always <= $\log_{2^b}$(NumberOfNodes) (b in our implementation is 2).

b. The interesting observation is that, even though we increase the numberOfRequests value, the average number of hops remains almost the same with a very negligible change.

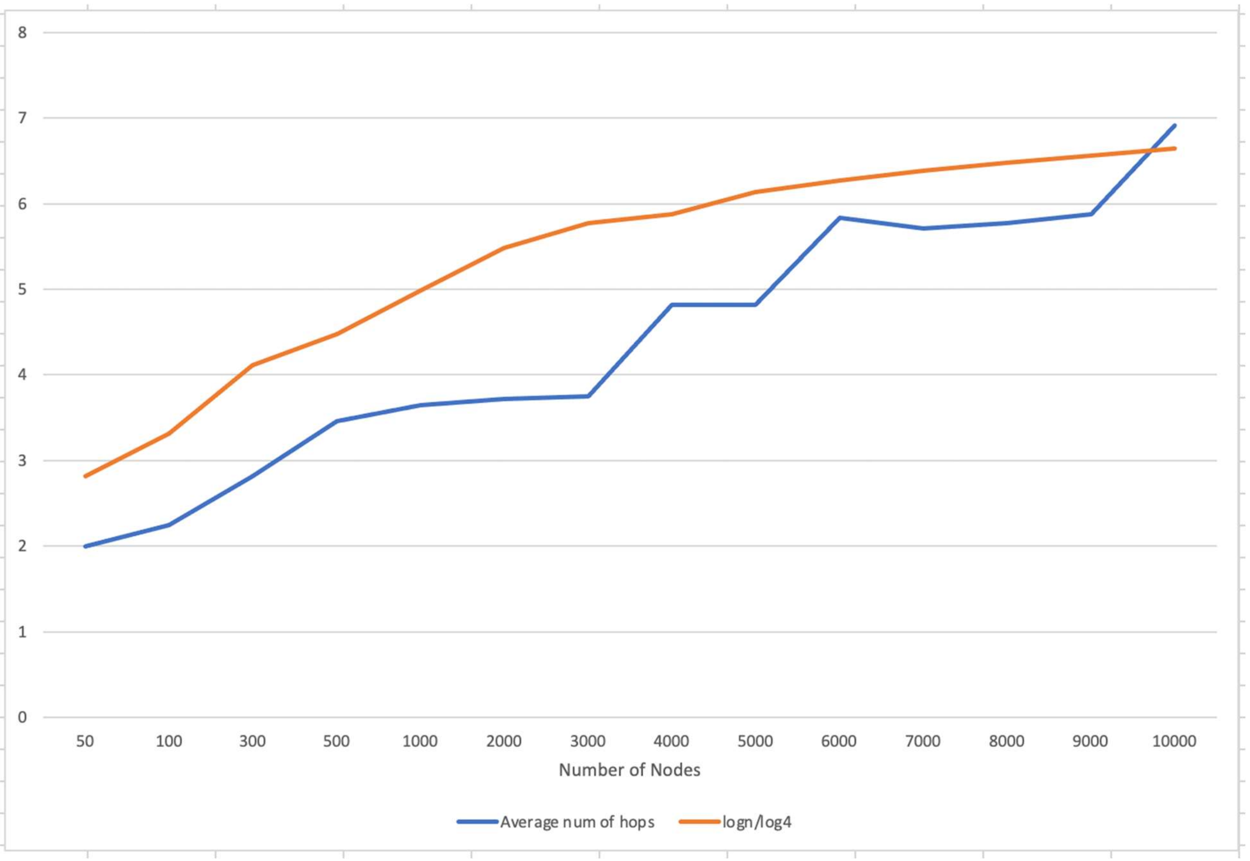
ii) The largest network that we managed to deal with is numberOfNodes= 10000 and numberOfRequests = 100.

Observations Tabular From:



Num Nodes	Num Requests	Avg Num of Hops	log(n)/log(4)
50	15	1.996	2.82
100	40	2.25	3.32
300	120	2.82	4.11
500	200	3.46	4.48
1000	300	3.65	4.98
2000	500	3.72	5.48
3000	1000	3.75	5.77
4000	500	4.82	5.88
5000	500	4.82	6.14
6000	500	5.83	6.27
7000	100	5.71	6.38
8000	100	5.77	6.48
9000	100	5.88	6.56
10000	100	6.91	6.64

These are the values which we have taken and displayed them in the form of a graph below.


8
7
6
5
4
3
2
1
0
50
100
300
500
1000
2000
3000
4000
5000
6000
7000
8000
9000
10000
Number of Nodes
Average num of hops
logn/log4