Final Report – Lab 5

Koppu Sai Kiran

CSU ID: 830448879

Adarsh Anand Kulkarni

CSU ID: 830082226

Design:

The key(ID) for the node is obtained by using MD5 hashing of IP and port ,and we took last 6 hex digits as it is convenient to store in Long format. For hashing the files, if the file contained only one word we obtained the key directly but if the file had more than one word then we got keys for each word separately and one key as whole file name, these keys are store beside to whole full name only this is to facilitate that even a part of file is searched ,then whole file name is returned.

Algorithm:

- The network size is hard coded into the code.
- Initial finger table is formed, by having its own ID as successor. Finger table contains start, End Intervals, successor of that interval and the successor's IP ,port.
- Initial key table is formed, the key values stored are the keys and files present at its node. Our key table contains file name , key ,IP and port.
- Command for registration is sent to the BS server. Depending on the reply from BS that is if BS replies with any nodes already present in network ,then finger table is updated and "UPFIN 0 " is sent to each of the node sent by BS, then GETKY command issued to the successor ,and the returned keys are stored in the key table.
- When we receive "UPFIN 0" from other nodes- this happens when a new node is added to the network, then finger table should be updated.
- When GETKY is received then the key which are eligible to be sent to the asking node is sent .The keys to be sent is found out from Chord protocol.
- Whenever a node leaves, that should transfer its keys to the successor and unregister from the BS.
- When a query is received ,then the successor of the key(present in the query) is found, if the received node is the successor then it should search in its key table and answer back to the queering node else it should forward to the successor after incrementing the hop count.

Statistics for 20 Nodes:

The readings for 20 nodes across five trials are as follows:

Average Readings for 20 Nodes across 5 trials				
Avg Hops	Avg Latency	Avg Msgs/node	Avg Fingertable size	Avg Keytable Size
3	0	22	24	39
1	1	14	24	16
1	2	12	24	36
2	2	14	24	26
2	1	4	24	33
2	0	2	24	22
1	1	0	24	90
3	2	16	24	24
1	0	30	24	35
0	1	18	24	24
2	1	0	24	25
2	0	0	24	38
1	0	12	24	19
1	0	0	24	14
0	1	6	24	24
3	1	22	24	15
2	0	20	24	28
0	0	14	24	24
1	1	10	24	16
0	0	0	24	24

Hops
Minimum Hop Count = 0
Maximum Hop Count = 3
Average Hop Count = 1.5
Standard Deviation = 2.12132

Messages per node
Minimum Messages per Node=0
Maximum Messages per Node=30
Average Messages per Node=10.8
Standard Deviation=9.023945

Latency
Minimum Latency=0 Seconds
Maximum Latency = 2 Seconds
Average Latency=0.7 Seconds
Standard Deviation=0.732695

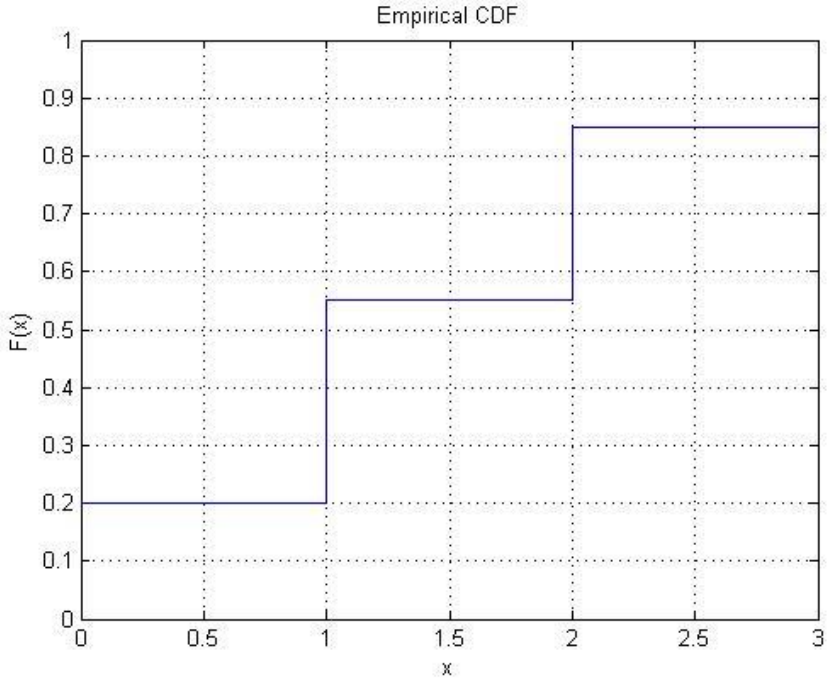
Fingertable
Minimum Fingertable size = 24
Maximum Fingertable size = 24
Average Fingertable size = 24
Standard Deviation = 0

Keytable
Minimum Keytable size = 14
Maximum Keytable size = 90
Average Keytable size = 28.6
Standard Deviation = 16.31725

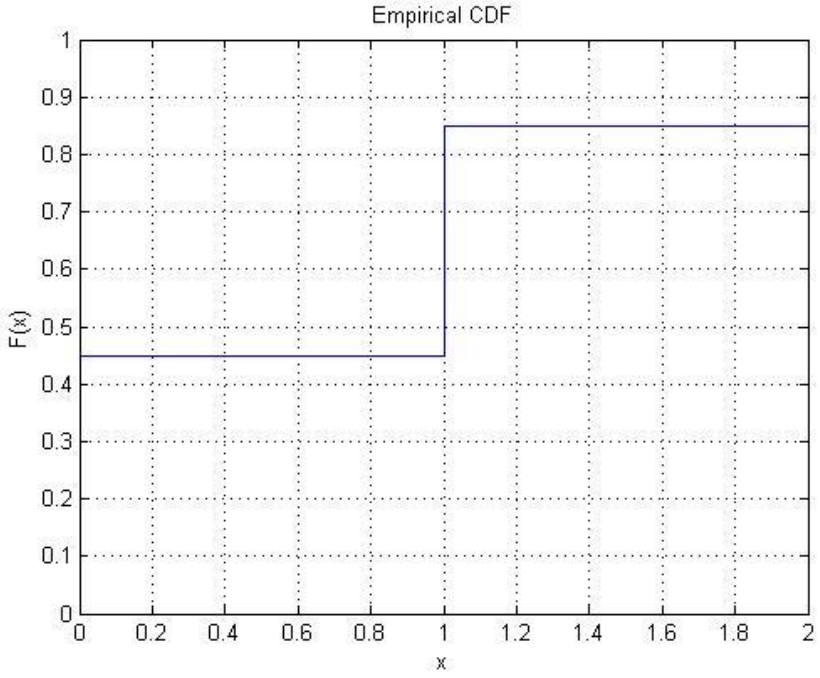
Per Node cost = Messages/Node = 10.8

Per Query cost = Messages/Query = 3.6

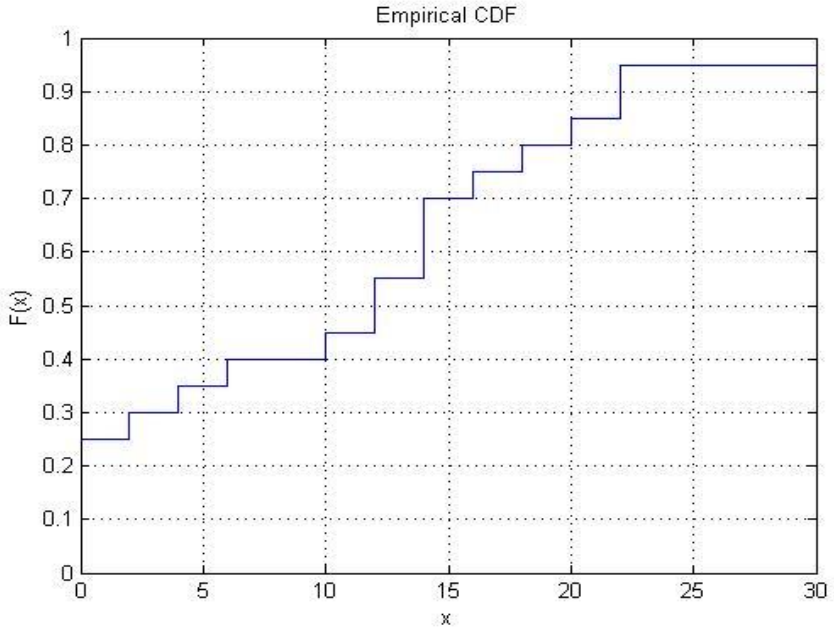
Cumulative Distribution Function plot for Hops:



Cumulative Distribution Function plot for Latency:



Cumulative Distribution Function plot for Messages per Node:



Statistics for 40 Nodes:

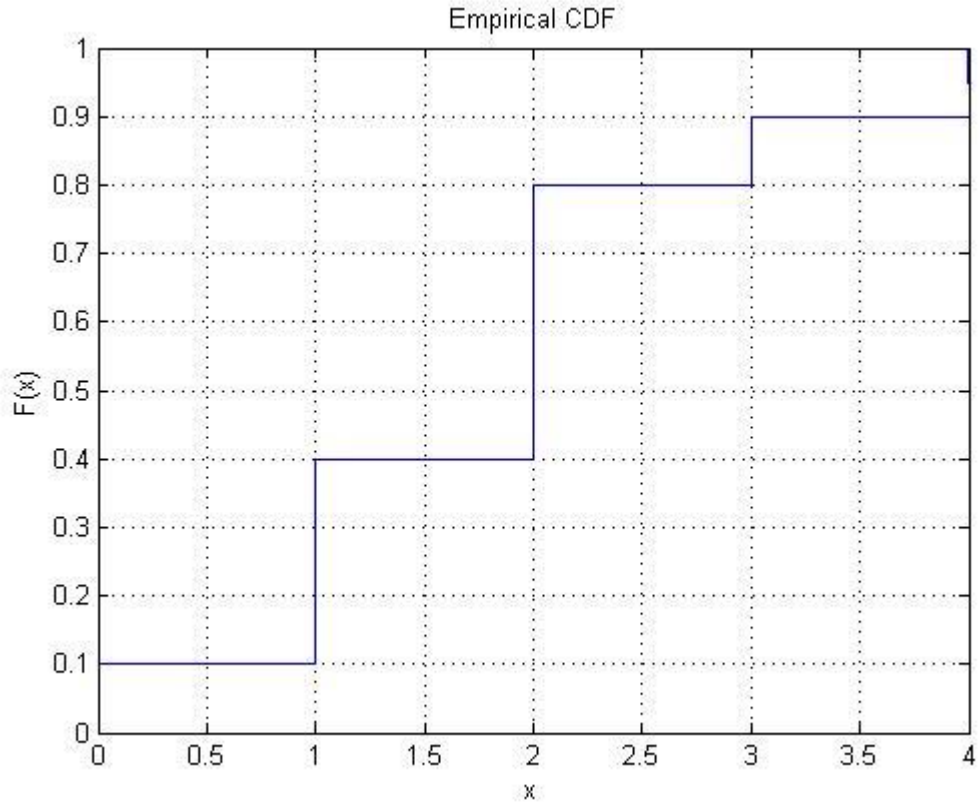
The readings for 40 nodes across five trials are as follows:

Average Readings for 40 Nodes across 5 trials				
Avg Hops	Avg Latency	Avg Msgs/node	Avg Fingertable size	Avg Keytable Size
4	1	16	24	45
2	2	12	24	33
0	2	10	24	25
1	2	14	24	30
3	1	6	24	18
2	1	4	24	24
1	0	0	24	140
2	0	36	24	30
1	2	0	24	38
1	2	18	24	28
2	1	8	24	31
4	1	0	24	38
0	2	14	24	24
1	1	0	24	20
2	1	4	24	26
2	2	24	24	18
2	1	20	24	26
2	0	16	24	34
3	2	10	24	21
1	1	14	24	24

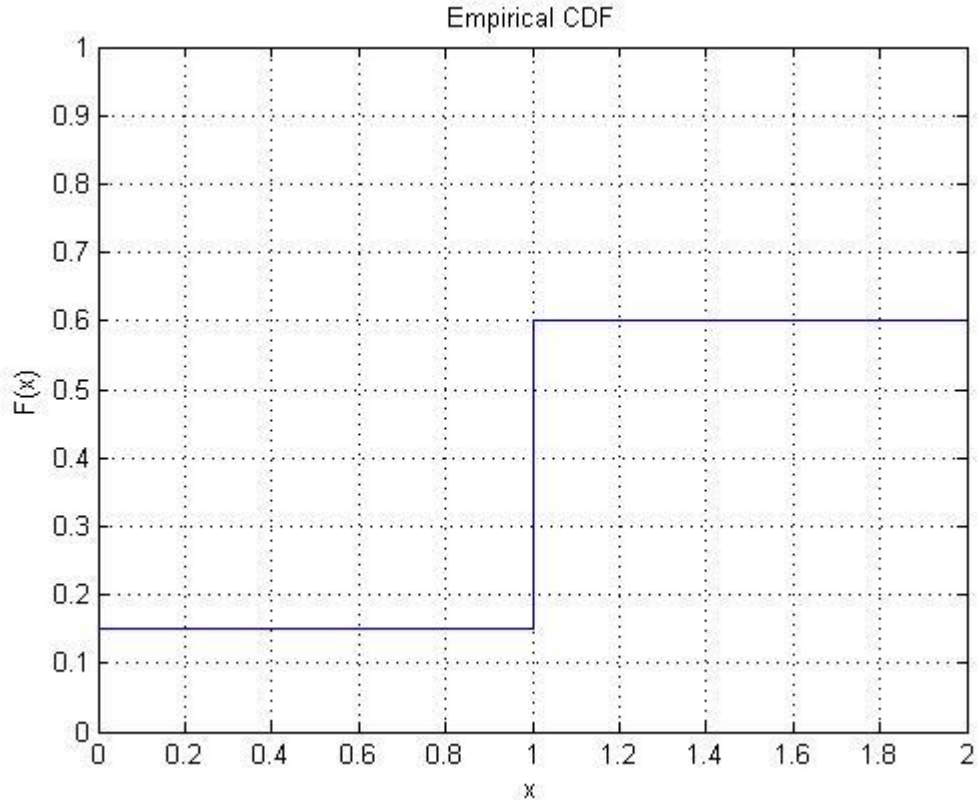
Per Node cost = Messages/Node = 11.25

Per Query cost = Messages/Query = 3.76

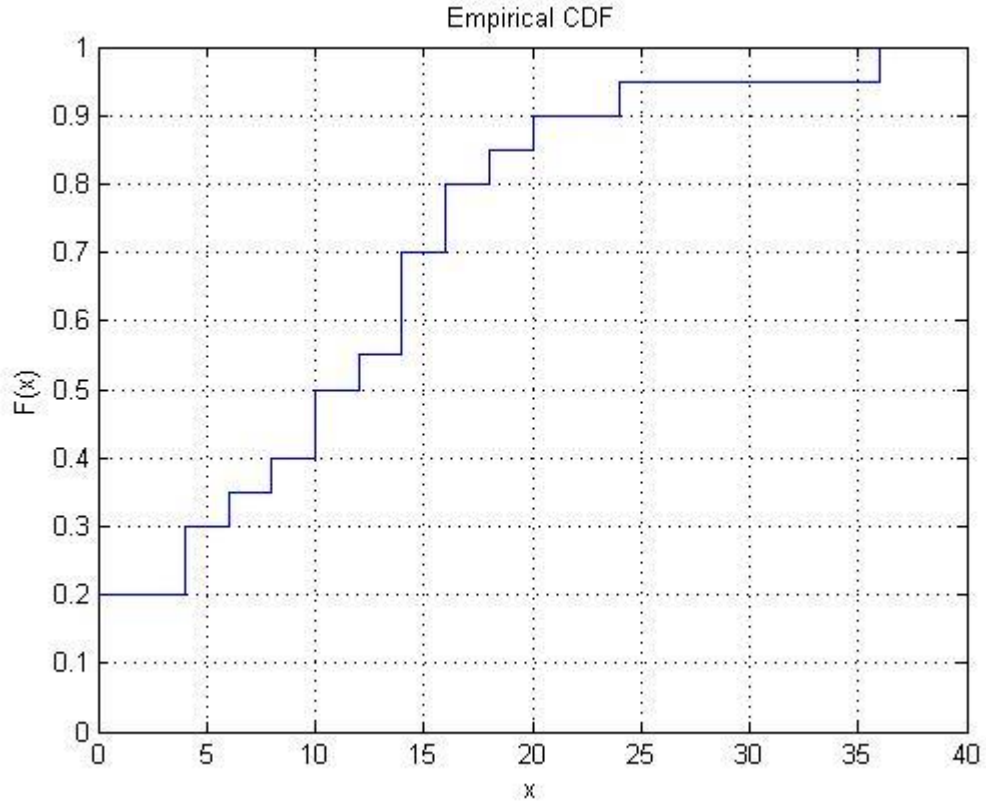
Cumulative Distribution Function plot for Hops:



Cumulative Distribution Function plot for Latency:



Cumulative Distribution Function plot for Messages per Node:



Effects of Scaling Network Size: If the network size is increased then the number of messages, hop count and the latency for resolving the query also increases. However, if the network size

increases it also means that there are more files present in the network which increases the probability of query resolution.

Techniques for Improving Query Resolution: Instead of increasing the number of nodes in the network, we should increase the number of files present in node that is increasing the memory size at the node. This inturn will help in reducing the number of messages, hop count and the latency.