Chord and Pastry

Pratik Karia - 2019MCS2568

March 2020

1 Pastry and Chord

1.1 Pastry Routing Table

Pastry routing table can be explicitly printed using a printAllNodes function

1.2 Path taken up by a look up request

On using –log-enable during running of chord, the path is generated in a log file for all search queries.

1.3 Evaluation

1.3.1 Chord

Number of Nodes	Average Number of Hops
100	5.3
500	5.9
1000	6.3

1.3.2 Chord

Number of Nodes	Average Number of Hops
100	2.7
500	3.1
1000	3.5

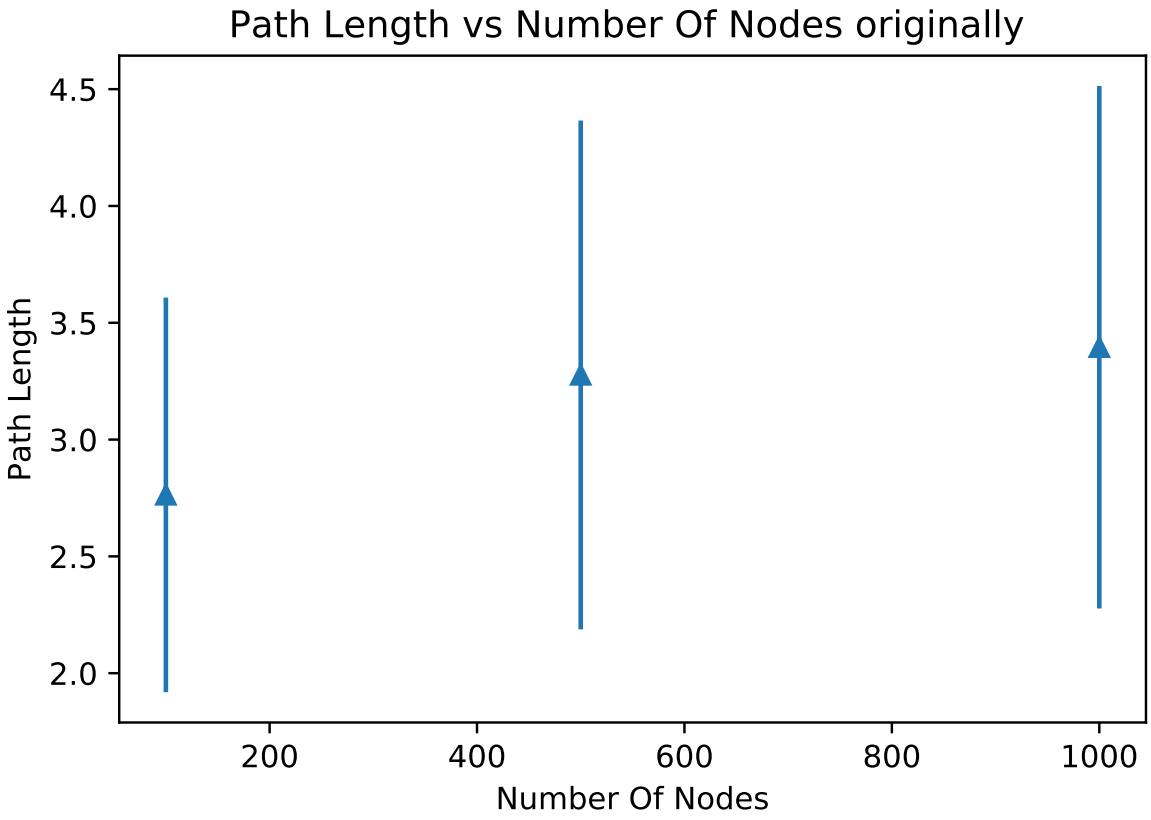
2 Report and Insights

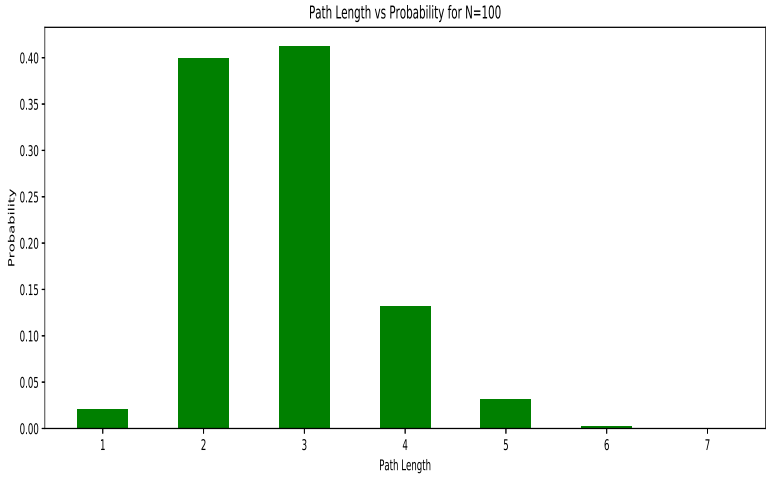
2.1 Experiment Settings

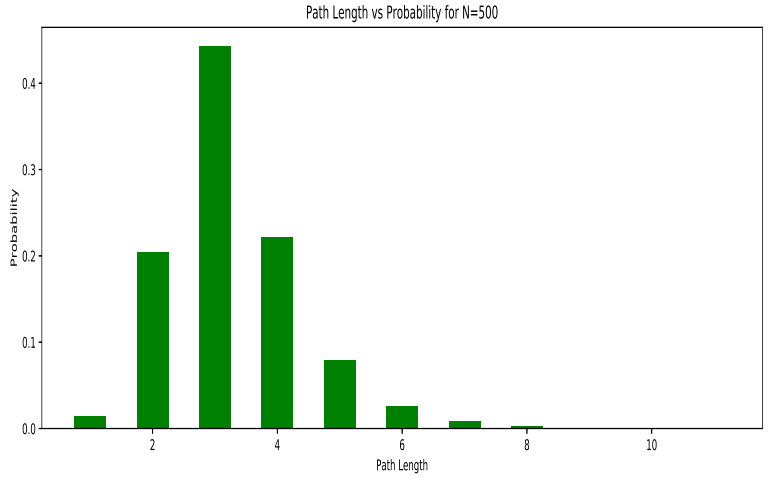
State	Number Of Nodes	Number Of Data Points	Search Queries
Before Deletion of 50% nodes	100	10000	1000000
After Deletion of 50% nodes	50	10000	1000000
Before Deletion of 50% nodes	500	10000	1000000
After Deletion of 50% nodes	250	10000	1000000
Before Deletion of 50% nodes	1000	10000	1000000
After Deletion of 50% nodes	500	10000	1000000

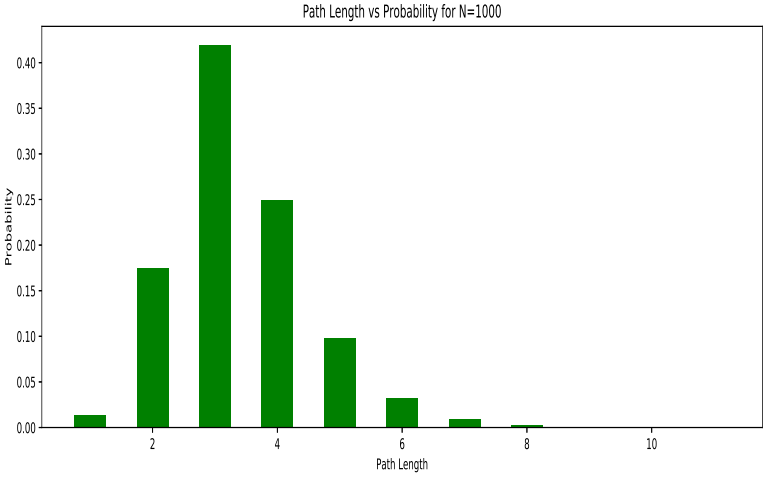
2.2 Plots

2.2.1 Pastry[1]

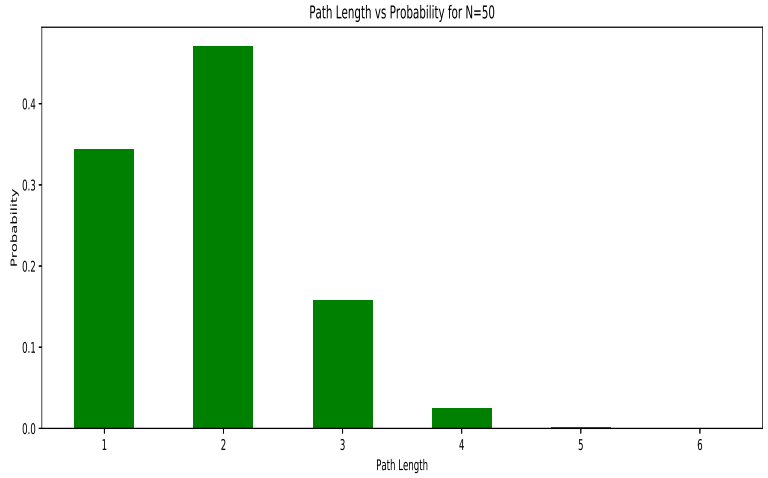




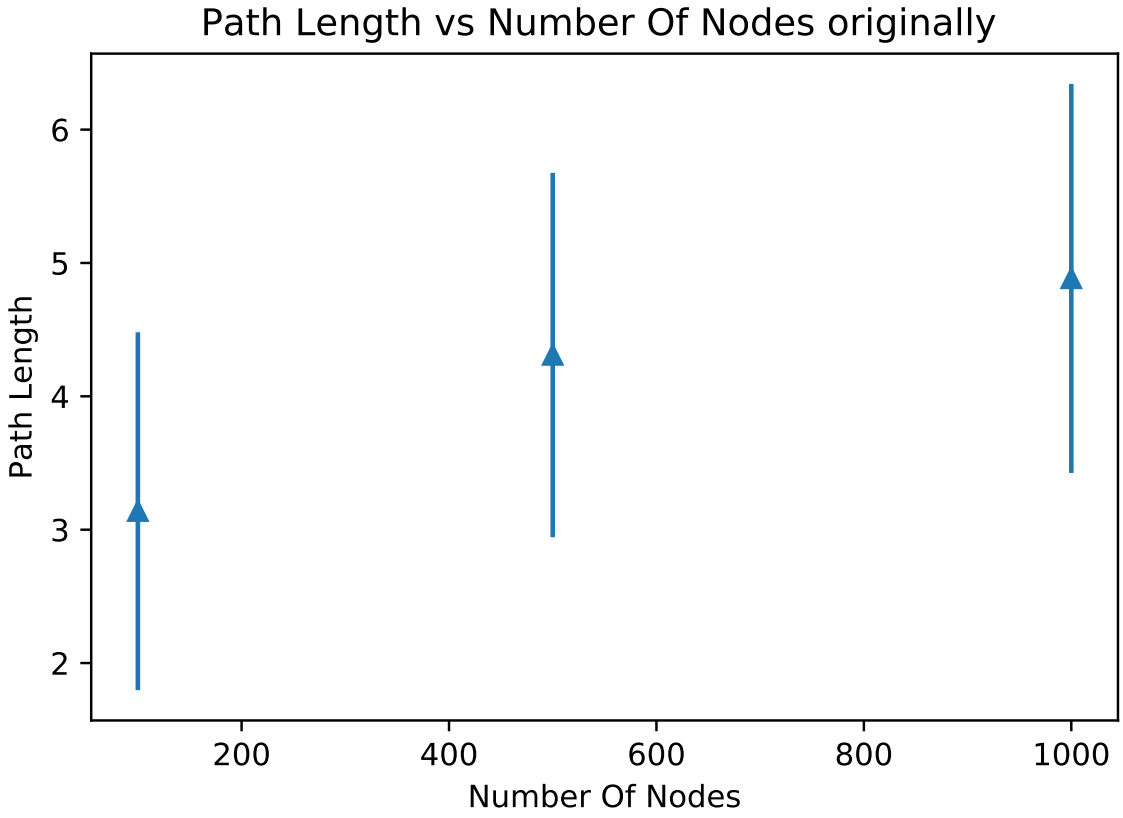


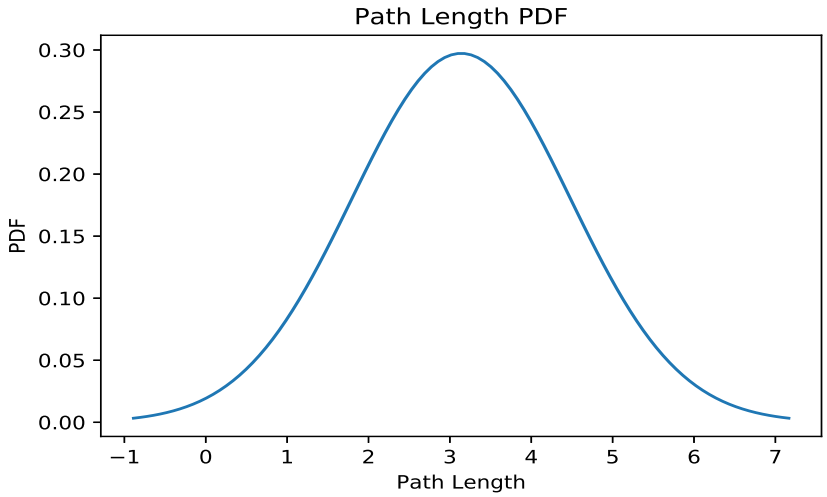


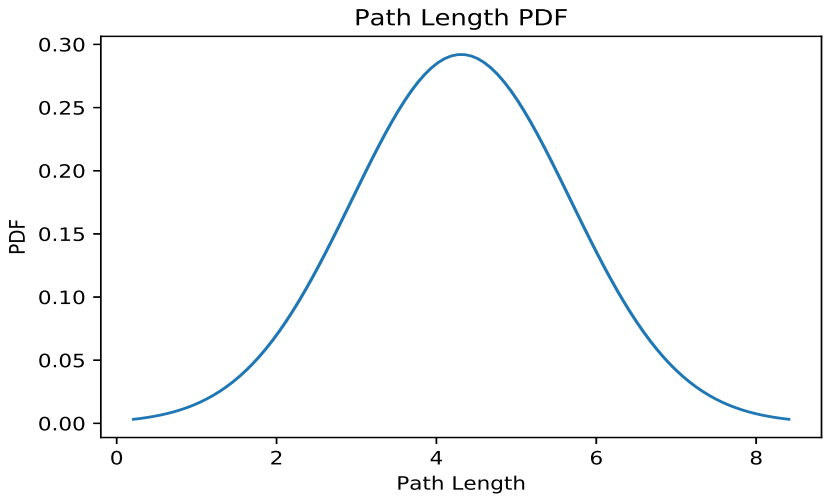
After Deletion

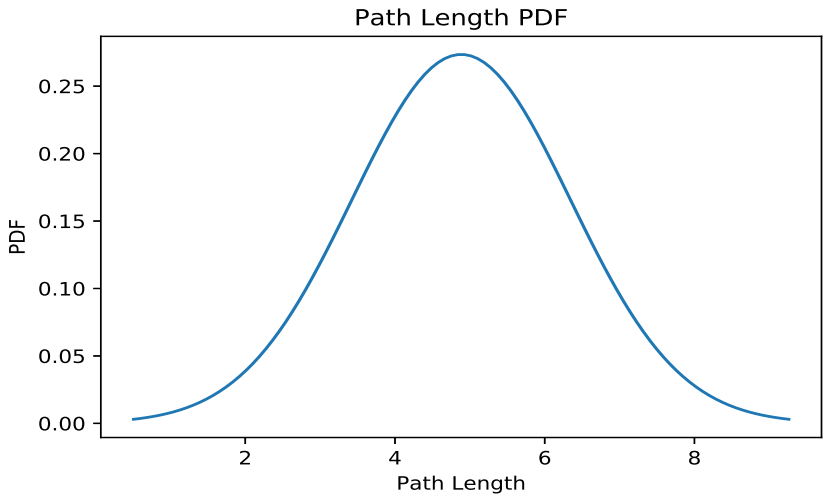


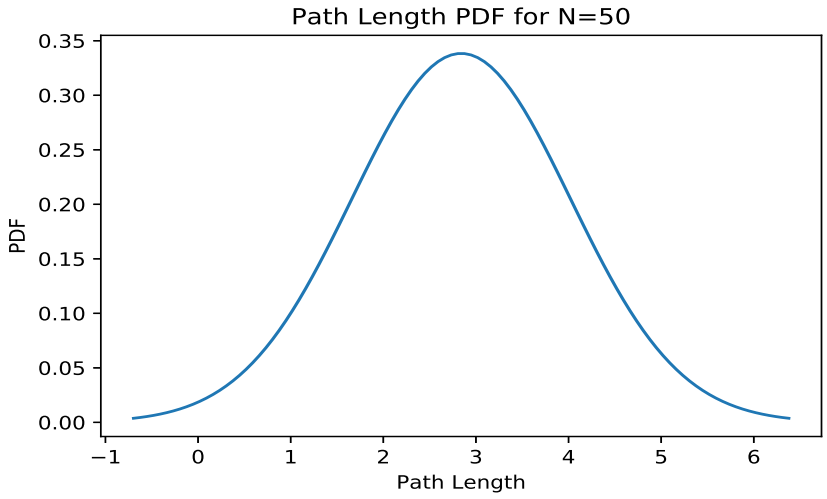
2.2.2 Chord[2]

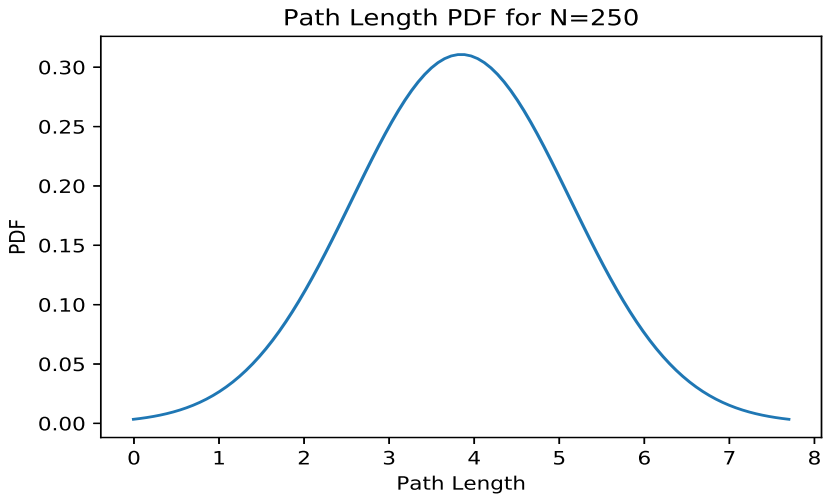

Path Length vs Number Of Nodes originally
Path Length
Number Of Nodes
2
3
4
5
6
200
400
600
800
1000

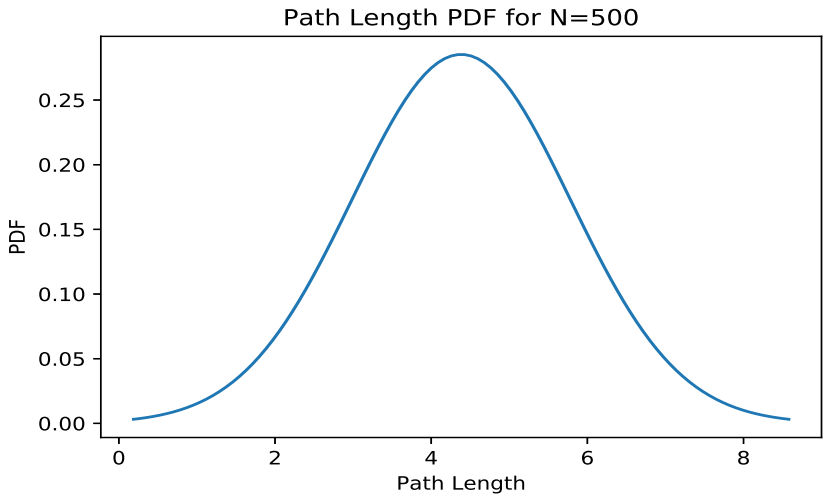

Path Length PDF
PDF
Path Length
0.00
0.05
0.10
0.15
0.20
0.25
0.30
−1
0
1
2
3
4
5
6
7


Path Length PDF
PDF
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
2
4
6
8
Path Length


Path Length PDF
PDF
0.25
0.20
0.15
0.10
0.05
0.00
2
4
6
8
Path Length


Path Length PDF for N=50
PDF
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−1
0
1
2
3
4
5
6
Path Length





2.2.3 Behavior Observed

The behavior was seen that as we delete nodes, the search time reduces but the density of data on each node increases drastically.

2.2.4 Justification of Trends above

As number of nodes increases so path length also increases. This is evident as the data can be stored further away from the start point of search.

Sample Finger Table(Few entries)

Start	Node
730995094934889621632647 2	733395113718878529707031 4
365619685602163892081726 0	367687818515731204817771 0
182931980935801027306265 4	185255304241428627399765 3
915881286026195949185351	935304437806193988521127 4

Note: Sample routing table of pastry was huge and so it is possible to view it explicitly code itself.

2.3 Performance Comparison

It is clearly justified that pastry performs with less number of hops than chord. But chord has better load balancing

2.4 How to run the code

To run Chord: python3 Chord.py –help to get all details To run Pastry: python3 Pastry.py –help to get all details

References

[1] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing*, pages 329–350. Springer, 2001.

[2] Ion Stoica, Robert Morris, David Karger, M Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review*, 31(4):149–160, 2001.