Team Members

Akhil Jain
UFID : 29806485
akhilj@ufl.edu

Vinay Kini
UFID : 38161319
kinivinay@yahoo.com

Working

The following describes the implementation of the Pastry API described in section 2.3 of the pastry overlay protocol described in the paper 'Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. by A. Rowstron and P. Druschel'.

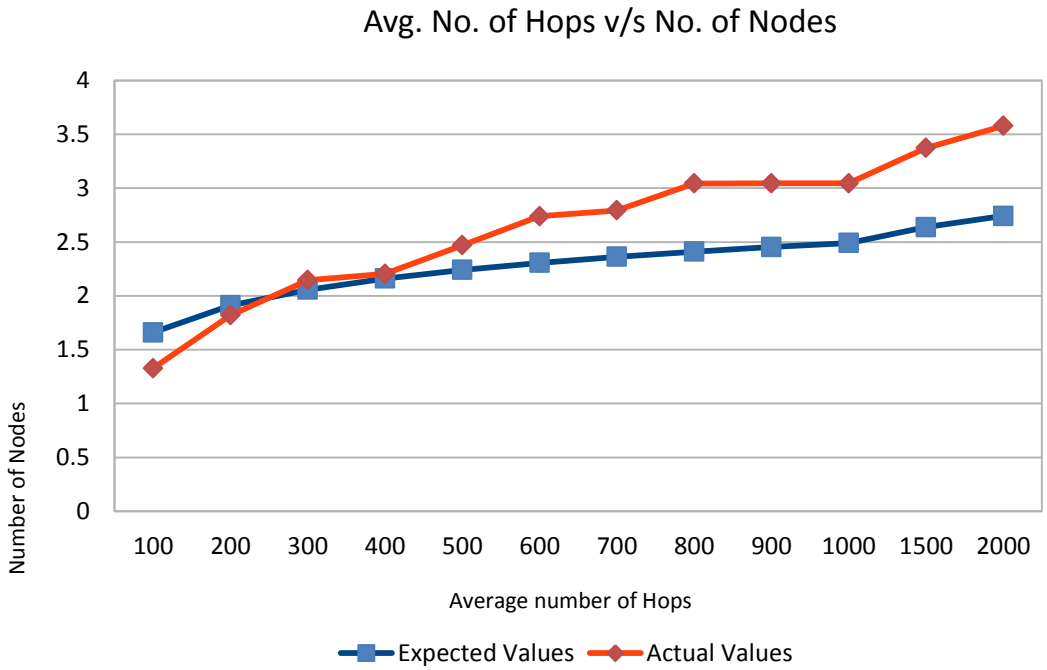
The program takes as input two parameters, the number of Nodes to be added to the pastry network and the total number of requests each node has to perform.
Scala project3 <number of nodes> <number of requests>

We ran the program for sample values in the range of 100 to 2000 nodes, keeping the number of requests sent by each node constant at 10. The following are the values recorded and a graph of the Average number of hops against the number of nodes inserted in the pastry network. Also included is a graph of the expected values, $x = Log_{2^b}N$ where x is the Average number of hops, N is the number of nodes in the network and b is kept at 4. Simplifying, we get, $x = Log_{16}N$.

Number of Nodes	Expected Values	Actual Values
100	1.6609	1.3269
200	1.9109	1.8195
300	2.0572	2.1453
400	2.1609	2.2035
500	2.2414	2.4696
600	2.3072	2.7395
700	2.3628	2.7941
800	2.4109	3.0433
900	2.4534	3.0447
1000	2.4914	3.0442
1500	2.6376	3.3728
2000	2.7414	3.5776

Example:
Scala project3 300 10
Average Number of Outputs: 2.1453
Time taken for sending all requests = 306 ms

Scala project3 500 10
Average Number of Outputs: 2.4696
Time taken for sending all requests = 859 ms



The program takes the number of nodes to be initiated and number of requests to be sent by each node as arguments and terminates when all the nodes have sent the requests the stipulated number of times. It outputs the average number of hops required by the program for all nodes in that network to finish transmitting the message the required number of times. There are two parts to this program, namely, Node Addition and Message Routing. The program uses the Node Arrival mechanism as described in the pastry protocol paper to add a new node to the network. The routing of messages assumes the network is in consistent state i.e. all the nodes have correct entries of neighborhood set, routing table and leafset, the main actor instructs all the node actors to send requests/messages in the network. The network tries to route this message to the node, which has the numerically closest nodeId for this message. Once, the message reaches the destination node, the main actor and sender is informed of the message delivery to the correct location and the number of hops used was recorded thereby allowing us to calculate the average number of hops later.

As we notice from the graph, if the value of b is kept constant at b = 4, the routing protocol takes close to **$\log_2^b N$ hops** to reach destination from the source. Also the leafset is of fixed size, that is set to 4. If the node finds the destination set in its neighborhood set, then the message is directly routed to the destination node without consulting the routing table.

The choice of the value of b involves a trade-off between the size of the populated portion of the routing table and the maximum number of hops required to route between any pair of nodes i.e. $(\log_2^b N) \times (2^b - 1)$ entries v/s $\log_2^b N$ hops. Thus for a 10000 node network, with a value of b = 4, we

will need around 50 entries and 3.321 average number of hops, while for a value of b = 2, and number of nodes remaining the same, the number of entries will reduce to around 20 but the average number of hops required will become 6.6438. Thus depending on space or time complexity, an appropriate value of b needs to be selected.

Largest Network

The largest network we managed was 2000 nodes. Beyond that, it takes too long for the tables to be created and for all nodes to transmit the message to their destination. Also, it was seen that this implementation of the routing protocol was fast while the number of nodes remained less than 500, in the range of 30 seconds to a minute. However above that value, the time taken to initialize all the nodes and terminate the program takes a while. Also the maximum number of requests sent were seen to be 100 per node.