CS4032 project

Support document

Stephen Dodd

08363030

31/12/13

The objective of this project was to build a peer to peer based search engine using ruby. The peer search spec used was the peer search simplified spec. The spec can be broken up into 4 different sections.

1. Searching
2. Indexing
3. Routing
4. Joining

The first three sections were implement how ever the joining section was not implemented.

Searching

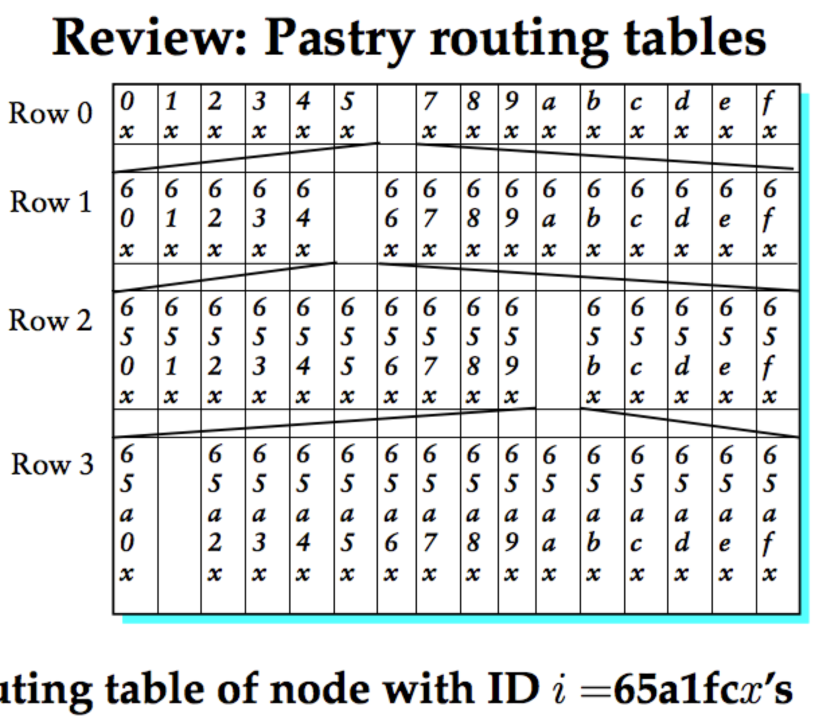
The search is done by applying the hashing algorithm to a search term, assumed to be one word. The result of the hash is then considered to be the target node ID. The information is then packaged in a search message, which is formatted to JSON. This message is then routed through the network until the target node is found. The process then aits fro the target node to send back a search response message contacting all relevant URLS

Indexing

The indexing is done in a similar manner to the search once the target node id is found by hashing the keyword. An index message is sent across the network to the target id containing the keyword and the URL to be stored. The message is then routed across the system. When an index message is received by a target node that node sends back a ack index message to inform the sending node that its data was stored in the search network.

Routing

Each nod in the system has a routing table based on the pastry routing table figure 1 shows a simple pastry routing table. Due to the nature of the peer search simplified spec no nearest neighbour or leaf set of nodes is stored in the pastry table.
When a message is sent to a node and the node is not the target node the node scans its routing table and sends the message on with the next nearest value the node has stored in its routing table. This is doe for the number of hops taken to get to the target node. The spec also assume that the target node always exists hence why there is no leaf set.



Review: Pastry routing tables

Row 0	0x	1x	2x	3x	4x	5x		7x	8x	9x	ax	bx	cx	dx	ex	fx
Row 1	60x	61x	62x	63x	64x		66x	67x	68x	69x	6ax	6bx	6cx	6dx	6ex	6fx
Row 2	650x	651x	652x	653x	654x	655x	656x	657x	658x	659x		65bx	65cx	65dx	65ex	65fx
Row 3	65a0x		65a2x	65a3x	65a4x	65a5x	65a6x	65a7x	65a8x	65a9x	65aax	65abx	65acx	65adx	65aex	65afx

- **Routing table of node with ID i =65a1fc$x$'s**

Figure 1: Pastry Routing Table

Joining

The joining is done by routing a join message through to the bootstrap node. It is assumed that every node knows a node within the network to act as a gateway node. The message is then routed through the network to the bootstrap. This node then return in a routing info message that contains a copy of the nodes routing table this can be used to initialise the joining nodes routing table. When a node is leaving the network it sends out a leaving message to all nodes in its routing table instructing the nodes to delete it from their routing table. The spec has no protocol for unexpected leaving nodes. The least recently used storage method for the routing table should remove any nodes that are not a part of the network. The spec also assumes that each node hold the data for a single keyword this means that if a node leaves its data is lost to the network and no node can store it.