Exercise 3

Peer-to-Peer Networks, Fall 2008

Guidelines

- Group work **encouraged**, groups of up to 3 people allowed
- Due date: 25.11.2008[1] at 16:14
- Exercise is worth 18 points
- What to return: One zip- or tar-file which contains 3 files which contain the object-to-node mappings for each of the three parts *in the format specified below.*
- Don't forget to write all the names and student numbers on the answer!
- Return by email to pervila@cs.helsinki.fi.

General Information

In this exercise, you will take a look at how different DHTs map objects to nodes, and compare them. We will look at 3 different DHTs, Chord, CAN, and Tapestry. For each one of them, you need to report the distribution of objects to nodes, and compare them in terms of how well they distribute the objects.

On the web site you will find a zip-file exercise3.zip which contains files for this exercise, two for each of the parts. For each part, one of the two files will contain node IDs and the other will contain the object IDs.

File Formats

The files on the web site contain simply a column of numbers (either node or object IDs). The files with the word "node" in the name are node IDs and the others, with the word "objects" in the name are object IDs. For CAN, the numbers give 2-dimensional coordinates separated by a comma (e.g., 23756,37696 means the point with x-coordinate 23756 and y-coordinate 37696).

[1] There will be a Q&A session during the normal exercise hours on 18.11.2008

For your answers, use the following file format. Put one line per node, start the line with the node ID, followed by a colon (:), and then all the objects that were mapped to that node. Use plain text. For example: node1: o1 o2 o3 o4. The answer file should have one line for each node. Keep the leading zeros in the node and object identifiers. The order of nodes in the file and objects in the line of a node do not matter.

If your returned files do not follow the above format, you will not get any points for those parts of the exercise.

Name your files in such a way that the filename tells which DHT is in that file. For example, call them chord.txt, can.txt, and tapestry.txt.

Question 1: Chord, 2 points

The files for Chord are called chord-node.txt and chord-objects.txt. The IDs are distributed between 0 and 99999. Map the objects to the correct nodes and provide your answer in the above format.

Hint: Think this one over with a smaller address space at first. You can reach the correct solution with only a handful of lines of code.

Question 2: CAN, 3 points

The files for CAN are can-node.txt and can-objects.txt. We use a 2-dimensional CAN, so each line contains two coordinates separated by a comma. Assume that each node is the center of the zone for which that node is responsible. Use the standard Euclidean distance as a metric and map the objects to their closest nodes. Nodes are inserted consecutively from top to bottom of the .txt file. If a zone is square shaped and needs to be split, the zone will be divided vertically. If a zone is a rectangle, it will be split horizontally and the result will be two squares.

Hint: Be very thorough with the Euclidian distances and remember CAN's signature wraparound at the edges.

Question 3: Tapestry, 5 points

The files for Tapestry are called tapestry-node.txt and tapestry-objects.txt. The IDs in these files are all 5-digit numbers, i.e., distributed between 00000 and 99999. This means that your Tapestry network uses base 10 numbers and has 5 digits in the namespace, i.e., 10 columns and 5 levels in the neighbor map.

You need to accomplish the following steps:

- Create neighbor maps for all of the nodes. When several nodes are possible candidates for an entry in the neighbor table, pick the node

with the smallest ID among them.

- Map each object to its responsible node. For each object, pick a starting node (at random) and route the object according to the neighbor maps all the way to its responsible node.

  When there is no entry in the neighbor map for the next hop, pick the closest entry in the neighbor map for that level which has an entry (can be the node itself). In case of ties, pick the smaller of the two.

Return the object-to-node mapping in the above format.

Hint: Precision is everything. Start early, work in pairs, and ask your questions long before the deadline approaches.