CMSC 818 Project #1 Spring 2011

Structured P2P Protocols **Due: March 10, 2011, 6:00PM**

The purpose of this programming assignment is to learn about the real problems in implementing P2P algorithms and data structures. For this assignment you will implement the peer join, routing, and stabilization algorithms for a distributed hash table (DHT), and also a simple client that makes routing requests into the DHT. You can implement your project in C, C++, Java, or a scripting language (e.g., Python, Ruby, Perl). If you want to use another language, ask Dr. Sussman first.

You must also write a short report (max. 3 pages), describing how you implemented the peer algorithms, and any problems you encountered in creating the peer. Also include information on how to build and run your peer and client.

1 Specifications

Your job is to:

- Implement the Chord, Pasty, or Tapestry peer join, routing and stabilization algorithms, as a standalone peer.
- Use TCP sockets for communication between peers, on a fixed port of your choosing (greater than 1000 and less than 65536).
- A peer joins the system by connecting to a known peer, taken from a command line argument, with a join message. The first peer just starts up as the lone one, and knows it's the only one since no peer is specified from the command line.
- A routing request is also a message on the same socket, but comes from a simple client that just sends routing requests, reading them from a user via stdin (the client takes a peer and port number as command line arguments).
- Write up your results.

1.1 Additional requirements and information

Your peer should print a (short) message for each action it takes onto stdout, including routing a message or a join request. That will help in determining if your peer is working correctly.

You get to decide how to generate node and routing request IDs, but the papers describe standard methods for generating IDs with uniformly distributed values across the large ID space.

For this project you should work in pairs, so please notify Dr. Sussman who you will be working with.

2 Grading Criteria

Your project grade will be determined with the following weights:

Correct node joins	25%
Correct, efficient request routing	25%
Correct stabilization	25%
Writeup	25%

3 Submission

You must submit a single gzipped tar file via email to Dr. Sussman by the project deadline. The file should include all your source code for the peer and client, a README on how to build and run the peer and client, and your writeup. The writeup should include both the source document, and a PDF file.

4 Other Notes

4.1 Using the Bug Cluster

For details about using the bug/hive cluster, see *https://wiki.umiacs.umd.edu/umiacs/index.php/ClusterGuide* . The login node for the cluster is brood00.umiacs.umd.edu, and the cluster nodes are {bug00, bug01, ..., bug63}.umiacs.umd.edu .

For additional, but somewhat out of date, information about using the Maryland cluster PBS/Torque scheduler and other cluster software., see *http://www.umiacs.umd.edu/labs/LPDC/cluster-manual.htm*.

4.2 Academic Integrity

Any evidence of cheating will be referred to the Student Honor Council and may result in a grade of XF in this course. Submissions may be checked with an automated source code comparison tool to look for evidence of cheating. This tool has already proven itself to be very effective, so we advise you to remember that the course syllabus forbids cooperation between students on projects, including either copying **or sharing** source code. You are also further advised to refrain from copying code from external resources. Your projects are to be **your own work**, and **only your work**.

Your instructor is quite serious about this, and will not hesitate to refer cases to the Honor Council should he have sufficient evidence to do so.