CS425
Xiaobin Zheng
MP3 Report

Chord Spec

The mp3 contains three files: 1) Makefile. 2) chord_sim.h 3) chord_dim.cpp
Makefile: Makefile is created for easier compile and clean up the old executable programs. Run "make" under the directory to compile the code.
Chord_sim.h: contains the functions.
Chord_sim.cpp: C++ code contains all function call.

Brief instruction for running the code:

1) make
2) run ./chord_sim config
   config contains the min and max delay and base port number for chord node.

Main tasks are join, find, show, show all and crash. The program is implemented as thread, in which each thread represents one node in the chord.

Main function:

Initializing Node 0:

Main initially creates a thread node 0, it initializes every element in finger table to 0, and key from 0 to 255. Once the node 0 is completed initialization, it sends back "Ready for next command" via message passing to client thread. Now main thread takes user standin.

Join P:

Join implementation is similar to initialization of node 0. In addition, it connects to immediate successor for requesting keys. Its immediate successor passes the keys to newly join node along with its node number. In addition, it asks all nodes to update the backup key list. The newly joined node updates its local key and predecessor. When all operations are done, it sends back "Ready for next command" to client via message passing. Now the client is ready for next user's input.

Find P K:

It is pretty straightforward. For example, if p0 doesn't contain the k, it looks its finger table and find next node that is closest to the k and then connects to it. Repeat this process till key is found. Since the implementation is using TCP socket, the node containing the key "write" back to request node with its node number till the message reaches to P. Now P writes back to client to indicates which node contains the key.

Show P:

It is pretty self-explanatory. The requesting packs all his finger table and key into a message and passes back to client.

Show all:

It is implemented on top of Show P. The client calls Show P N times where N is number of active nodes in the chord.

CS425
Xiaobin Zheng
MP3 Report

Crash P:
The crash itself basically exits the thread so the node no longer exists in the chord. However, to maintain the key and functionality of chord, I add another layer on top of preexisting find and show command implementation. Now Show and Find first check to see if this successor or predecessor has crashed.
1) If its successor is crashed, it informs its new successor to combine the key with backup key and then broadcast all nodes to update this finger table and back_up key. It is O(N) operation since all node has to do the updates. Once it is completed, it resume show or find operation as usual.
2) If its predecessor is crashed, it combined this key with backup key. And then broadcast all node to update the finger table and update the backup list. It is O(N) operation since all node has to do the updates. Once it is completed, it resume show or find operation as usual.

Note: the message passing is evaluated when crashed disable.

All Trails F= 128	Phase1 Avg. # of message	Phase2 Avg. # of message
P = 4, N = 5	~9	~4
P = 8, N = 5	~17	~6
P = 20, N = 5	~41	~8
P = 30, N = 5	~61	~10

Due to implementation method, the join takes 2N+1 and Find takes ~2Log2(N), ,where N is number of node in chord.