CSCI 4061 Discussion 13

4/23/18

Overview

- PA4 Updates
- Exercise - Should help with PA4

Changes

- Poll changes no longer need to propagate upwards.
- There is no longer a maximum number of concurrent clients.

Changes

- You can assume that the correct size of a message sent between client and server is 256 bytes.
- When your receive a response on the client, print not only the response code, but also the data, as the example server does.

Errata

- Return_Winner does not take a region as input.
- The instructions for Subtract_Votes are slightly incorrect. If a candidate is in the command of votes to remove, but not in the region's current list of candidates, this is an illegal subtraction. An illegal subtraction also occurs if the subtraction would push any candidate below zero votes.

Hints

- You can use a single mutex lock to synchronize the entire tree, or have one lock per node.

- The votes files specified in the input.req files are assumed to be in the same directory as the input.req file, which may or may not be in the same directory as the binaries.

Hints

- You can assume that region names are at most 15 characters, each region has at most five children.
- Despite the fact that the maximum candidate name is 100 bytes, 256 byte messages will be fine.
- There will always be a root node in the DAG. This root node could have any name, not just Who_Won.

Hints

- The recv call will indicate when the socket it is receiving from has undergone an orderly shutdown by returning zero.
- Polls have 3 states: initial (unopened), open, and closed.
- Aggregation can happen with open or closed polls.

Upcoming

- Updated PA4 document coming tonight, along with a list of all changes.
- Solution to this week's lab will be released soon.
- Updated test cases and client/server executables also coming soon.

Questions About PA4

- Read the FAQ document and add new questions.
  - It is updated multiple times a day.

- Email fulto081 AT umn.edu with other questions.

Exercise

- Finish the concurrent server code in the file rec13.c
- server.o takes 2 arguments
  - the number of accounts
  - the max number of concurrent connections.
- client.o takes 3 arguments
  - the address of the server
  - the index of the account to alter
  - the change in the account's value

ToDo

- Fill in the locations where the TODO comments are.
- The server should send back the new account balance after the change. The client should print this.
- **YOU MAY ADD/ALTER ANY CODE!**
- You can assume:
  - The client will never send an index in excess of the number given as the server's first argument.
  - The server never receives more than the maximum number of requests given as the second server argument.

Correctness

- You will know your code is correct by running:
  - ./test.o client.o 127.0.0.1 <index> <change> <num>
  - The highest (not necessarily last) number printed out should be <change> * <num>
  - **WARNING:** Don't run test.o many times quickly, the network security may block your account from connecting.
    - If this happens, you may have to open new terminals or log out and back in.