Writeup: Multithreaded httpserver

November 22, 2019

- Testing Summary

  The way I tested my program to ensure it was working correctly, was by testing commands that were used to grade assignment 1. Then to test multi threading I copied the commands from the assignment 2 with comments at the bottom. with those commands I put them into a script and then ran that script to test if multi threading was working correctly.

  An issue that I ran into was that you can't run GET after two PUT requests. Also I needed the -s flag because it would show the transfer time/speed in the terminal but all jumbled up.

  Commands I used were:

  ```
  PUT :
  curl -s -T localfile http://localhost:8080 --request-target filename_27_character
  > cmd1 &
  ```

  ```
  GET :
  curl -s http://localhost:8080 --request-target filename_27_character > cmd2
  &
  ```

- Is there any difference in performance? what is the bottleneck in the system? how much concurrency is available in various parts and can you increase concurrency in any of these areas and if so how?

  Yes there was a difference in performance. Single-thread took about .451s for 4 requests and multithreads only took .224s for 4 requests. This is probably because we can run threads concurrently to handle requests. I think the bottleneck in this system is when we doGetPut. if we modularize it into seperate functions and call a thread on say Get and Put only when we need them then it would be quicker because once it's done with Get and Put then the thread would be opened up. Where as currently if you call a thread it has to go through the Get AND Put code before releasing the thread. I believe that we can increase concurrency if we modularize certain functions so that we only call the threads when we need them to do work.