Natalie Cluck, Megan Fischer

April 7, 2017

CSCE 313-504

MP 6 Report

**Report: –** Once you’ve finished all programming tasks, author a report that answers the

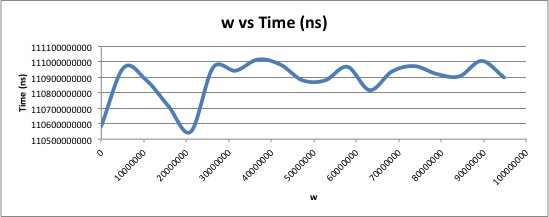
following questions about your code:

1. Describe what your code does and how it differs from the code that was initially given to you.

Our code takes inputs n and w. The user is able to define this at the beginning of the program through the command line. The program then outputs a histogram for three people based on the number of threads and number of requests per person. The main thing that is different is that we had to implement some functions, define variables, and include the time to record the time spent running the program. We had to implement the histogram. We were given a lot of the code for the histogram, but we had to add a few lines to make sure that it printed out the correct amounts. The next thing we added was the timer. We used clock\_gettime() to time the running time of the program. There were several variables that needed to be defined because they were used in the calculation of time.

1. Make a graph that shows how your client program’s running time for n = 10000 varies with the value for w. Include at least 20 data points (more or less evenly spaced) starting at 5 and going to the highest number that will run (*without* reporting some kind of error) on your OS. After making the graph, describe the behavior of the client program as w increases. Is there a point at which the overhead of managing threads in the kernel outweighs the benefits of multithreading? Also compare (quantitatively and qualitatively) your client program’s performance to the code you were originally given.

∗ Note: Timing must be done internally by your program, rather than by the shell it’s running in. A couple of good options for this are clock\_gettime() and gettimeofday() (see corresponding manpages: man 3 clock\_gettime and man 2 gettimeofday).

∗ Note: You may find that the provided test.sh script is a good starting point for gathering data. Feel free to modify it as you find necessary.

The time doesn’t vary that much after w = 30,000,000. The time is around the same. The time varies more between w = 10,000,000 and w = 20,000,000. The larger that w gets the less the process changes. The time difference is very small between all of the threads. There really isn’t that much difference between 5 and 100,000,000. Time is measured in nanoseconds, so the differences are at very small intervals. Our improved code ran a little longer than the code that was given to us, but that’s because we added more code. We also improved the code to make it more efficient, than the code that was given to us.

1. Describe the platform that you gathered your timing data on (I.e. CSCE Linux server, Raspberry Pi, personal computer, etc...). For any device other than the departmental servers, briefly describe the operating system that it runs (and if you used one of the departmental servers, just say which one you used). Also answer the following sub-questions:

I used the terminal on the Vocareum website. The command line says [ccc\_v1\_MmQ0M+95485@terminal\_student\_1 ~]. This was the easiest and fastest way for me to compile the code and gather time information because I have a Mac and can’t compile it on my personal computer. Vocareum is a web based operating system.

∗ What is the maximum number of threads that the host machine will allow your client program to create before reporting an error? What error is reported?

100000000

This is what is printed to the screen:

n == 10000

w == 100000000

CLIENT STARTED:

Establishing control channel... done.

Populating request buffer... done.

Pushing quit requests... Killed

∗ What does the operating system do when your client program tries to create more threads than allowed?

It stops the program and then goes back to the terminal and allows you to change the numbers of your input.

∗ How does your client program behave in response?

The program is killed because it can’t do it, and then goes back to beginning to allow you to change inputs.