CS342 Operating Systems

Project 1

Experiments Report

Ata Seren

21901575

Section-2

Introduction

First, I implemented histserver.c, histclient.c, histserver_th.c and histclient_th.c. By implementing several message queues, I connected the client and server to send and receive histogram data for both thread-type and process-type files. Then, I created some test files with integer values and evaluated them by file reading structures of C language and my algorithms. For such structures, I got help from the example codes on Moodle. I used these codes to learn and implement the structures of message queues, process and thread creation. At the end, I got the desired results from all files.

After completing all of the implementations, I used gettimeofday function of C, which I also used in Homework 1, to measure the processing time. I started to measure the processing time at the beginning of the child process creation and thread creation. I stopped it at the end of these creations.

Computer Specifications

I have a laptop with Windows 10 Home, Intel i7-9750H CPU @ 2.60GHz with 4.2GHz Turbo and 16,0 GB of RAM.

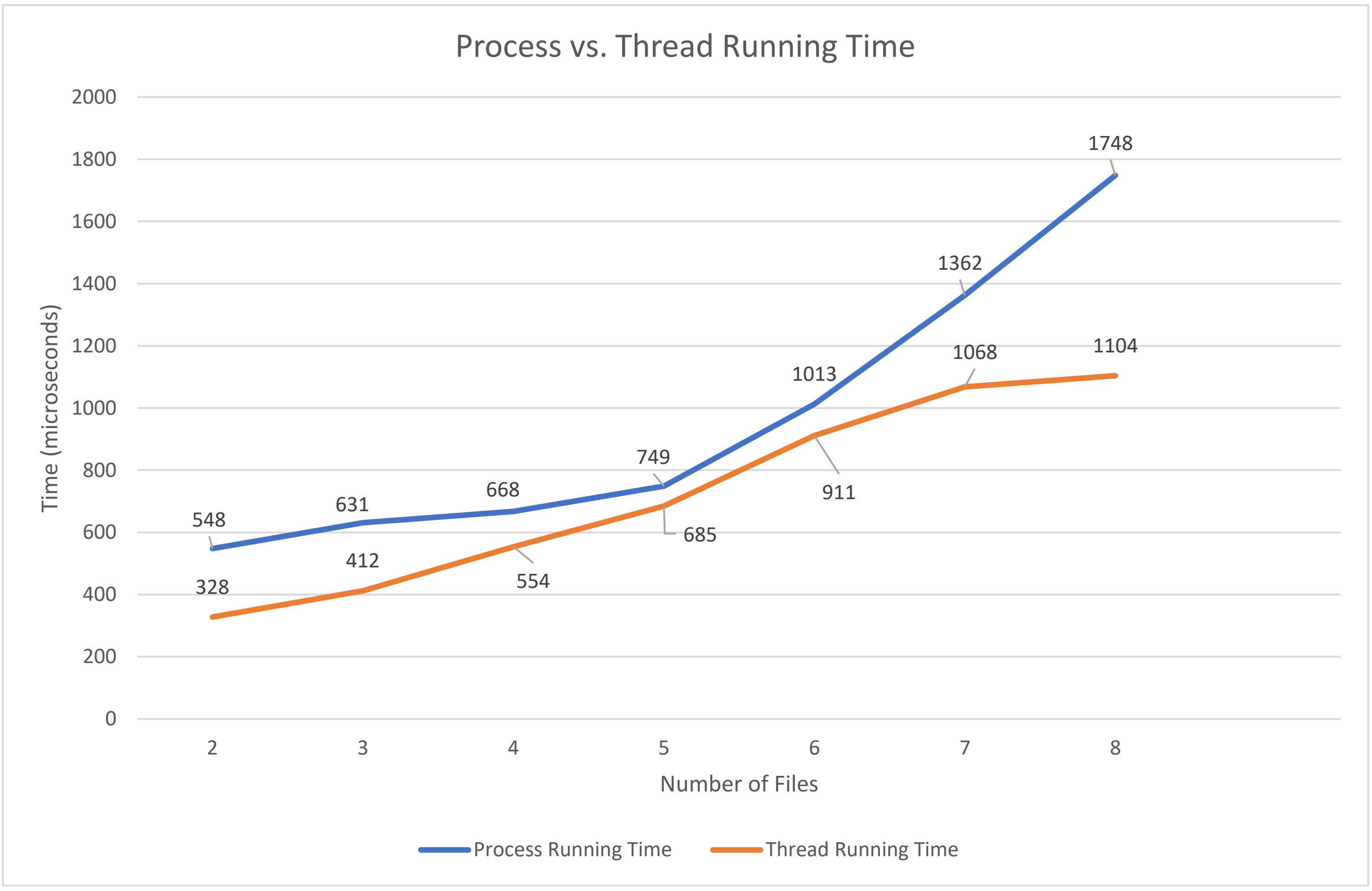
I used Oracle VirtualBox VM to run my code on virtual machine with Ubuntu 20.04. I reserved 5 CPU cores and 10725MB's of RAM for the virtual machine. In the machine, I used Linux terminal to compile and run my code and VS Code to write my code.

Experiments

For the experiments, I created 8 text files with random integers which I know what they result in a histogram with 5 intervals, interval start of 1000 and interval width of 200, just like in the example of the project document. Since the increment of the number of files directly affect the child process and thread creation and usage, I just incremented the number of files used in server for each case and measured and saved the running time. I received correct results from client and time results in microseconds with the function I used.

Table 1: Results of process running time and thread running time in microseconds

Number of Files	2	3	4	5	6	7	8
Process Running Time	548	631	668	749	1013	1362	1748
Thread Running Time	328	412	554	685	911	1068	1104



When I conducted my experiment for the first time, I received inaccurate results from gettimeofday function. For each number of files, I conducted the experiments multiple times to get rid of this inaccuracy. Then, I used the closest time to the other values for my experiment results. As it can be seen from the graph and the table, threads are little bit faster than child processes. I have few comments on this result. According to my research on similar codes and discussions I found on internet, threads should be faster than my results. In my program, threads work slower than normal or processes work faster than normal. I have some theories about this situation. This situation may have happened because of the virtual machine and the fluctuations of processing power between virtual machine and host machine or the reserved CPU's and RAM may have caused this situation or fluctuations. Another theory is that while experimenting, I may have disrupted the message queues by calling the program many times, even I shut them down at the end of the programs. However, this theory is probably wrong since they have been shut down at the end of every running program and I checked if they have been shut down properly. Despite the fluctuations on the graph, we can clearly see that at every number of files, threads gave the result faster than the child processes.

Conclusion

As we can see from the table of the graph, running time of threads is less than running time of child processes. According to these results, it is proven that threads in the program run faster than child processes.