CSE 344 – System Programming – Midterm Report

Muhammet Fikret ATAR – 1801042693

The main idea of this project is to simulate the aforementioned paradigm with 2 process-pooled servers: Y and Z executing on the same system as the clients. There are 3 programs to be implemented: client X, server Y and server Z. There will be a single instance of Y and of Z, and there can be an arbitrary number of client processes X, all running concurrently. Each client will submit a matrix to the server Y, and receive a response (from Y or Z) about whether it is invertible or not.

Design Plan And Implementation:

1) Client X

The square matrix was read from the csv file and converted into a message to be sent. I've stored all the data as an integer in a one-dimensional array.

It communicates via client x and servery fifo. Therefore, client x opens the fifo created by the server and places the requests in it. At the same time, a fifo name is sent to each child so that it can respond. In this way, the children will know which client to send the response to. Therefore, I created a fifo on the path containing the program's own id in the client program.

2) Server Y

In this section, I first take measures against double instantiation. When the program is opened, it runs as a deamon. and cannot be controlled by terminal. Once a request arrives through the serverFifo, it will send it (using a pipe) to any available worker process. . But how will it know which worker is available?

I solved this problem with shared memory. The information has to go through the kernel.

I have a pool size * 2 array in shared memory . In half of this array, the ids of the children are written in the other half whether they are busy or not(0,1). In this way, the parent process will be able to know the status of the children.

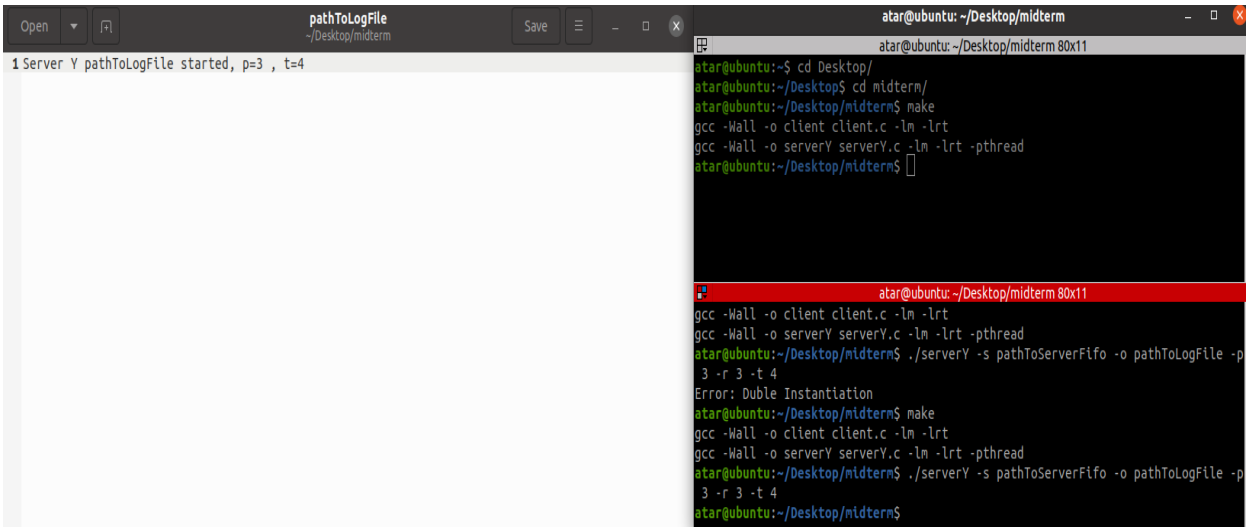
The pool workers will simulate intensive processing by sleeping for t seconds. Afterwards, the worker will wait for the next request, In this part, I called the sleep() function as much as t taken in the argument of the child process that finished its calculation. Also, the child communicates with the parent pip here, and each child has its own fd, so the read/write pipe function works perfectly.

3)Server Z

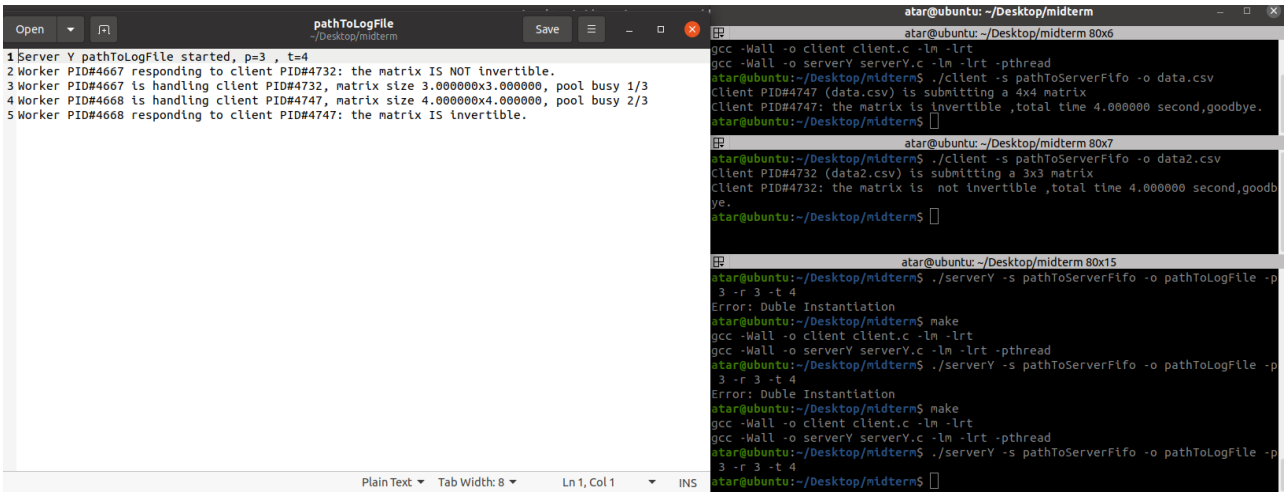
However, if all worker processes of serverY are already busy handling other requests, then serverY could forward the incoming request to serverS (using a pipe). I couldn't complete this part ,also my sigint call for all programs is not working properly.

Test And Result:

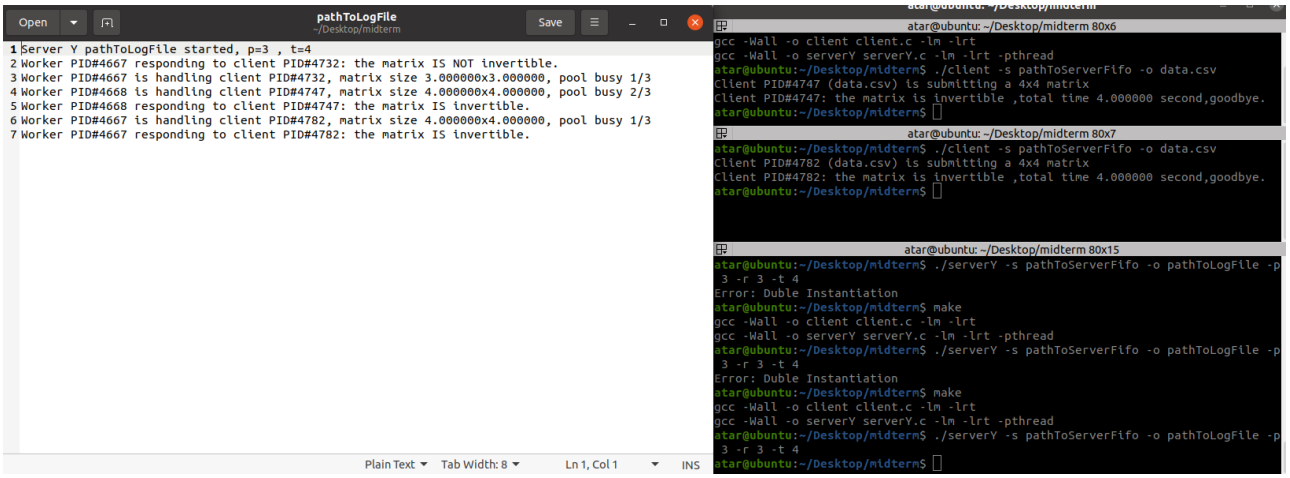
1)



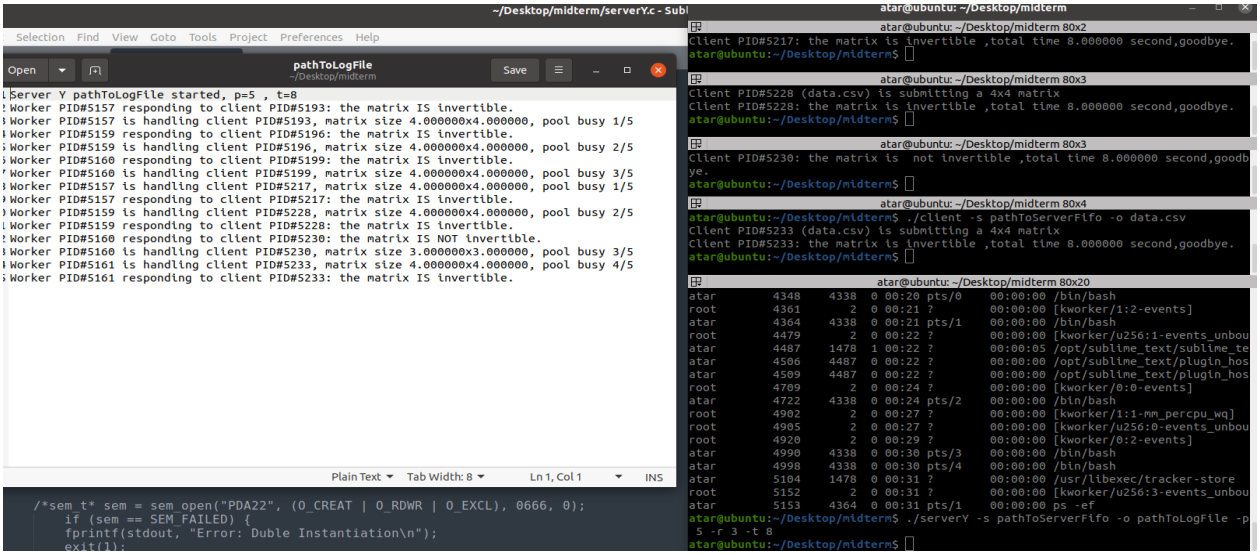
2)



3)


pathToLogFile
~/Desktop/midterm
Open
Save
Server Y pathToLogFile started, p=3 , t=4
Worker PID#4667 responding to client PID#4732: the matrix IS NOT invertible.
Worker PID#4667 is handling client PID#4732, matrix size 3.000000x3.000000, pool busy 1/3
Worker PID#4668 is handling client PID#4747, matrix size 4.000000x4.000000, pool busy 2/3
Worker PID#4668 responding to client PID#4747: the matrix IS invertible.
Worker PID#4667 is handling client PID#4782, matrix size 4.000000x4.000000, pool busy 1/3
Worker PID#4667 responding to client PID#4782: the matrix IS invertible.
Plain Text
Tab Width: 8
Ln 1, Col 1
INS
atar@ubuntu: ~/Desktop/midterm 80x6
gcc -Wall -o client client.c -lm -lrt
gcc -Wall -o serverY serverY.c -lm -lrt -pthread
atar@ubuntu:~/Desktop/midterm$ ./client -s pathToServerFifo -o data.csv
Client PID#4747 (data.csv) is submitting a 4x4 matrix
Client PID#4747: the matrix is invertible ,total time 4.000000 second,goodbye.
atar@ubuntu:~/Desktop/midterm$
atar@ubuntu: ~/Desktop/midterm 80x7
atar@ubuntu:~/Desktop/midterm$ ./client -s pathToServerFifo -o data.csv
Client PID#4782 (data.csv) is submitting a 4x4 matrix
Client PID#4782: the matrix is invertible ,total time 4.000000 second,goodbye.
atar@ubuntu:~/Desktop/midterm$
atar@ubuntu: ~/Desktop/midterm 80x15
atar@ubuntu:~/Desktop/midterm$ ./serverY -s pathToServerFifo -o pathToLogFile -p 3 -r 3 -t 4
Error: Duble Instantiation
atar@ubuntu:~/Desktop/midterm$ make
gcc -Wall -o client client.c -lm -lrt
gcc -Wall -o serverY serverY.c -lm -lrt -pthread
atar@ubuntu:~/Desktop/midterm$ ./serverY -s pathToServerFifo -o pathToLogFile -p 3 -r 3 -t 4
Error: Duble Instantiation
atar@ubuntu:~/Desktop/midterm$ make
gcc -Wall -o client client.c -lm -lrt
gcc -Wall -o serverY serverY.c -lm -lrt -pthread
atar@ubuntu:~/Desktop/midterm$ ./serverY -s pathToServerFifo -o pathToLogFile -p 3 -r 3 -t 4
atar@ubuntu:~/Desktop/midterm$

4)


~/Desktop/midterm/serverY.c - Subl
Selection Find View Goto Tools Project Preferences Help
pathToLogFile
~/Desktop/midterm
Open
Save
Server Y pathToLogFile started, p=5 , t=8
Worker PID#5157 responding to client PID#5193: the matrix IS invertible.
Worker PID#5157 is handling client PID#5193, matrix size 4.000000x4.000000, pool busy 1/5
Worker PID#5159 responding to client PID#5196: the matrix IS invertible.
Worker PID#5159 is handling client PID#5196, matrix size 4.000000x4.000000, pool busy 2/5
Worker PID#5160 responding to client PID#5199: the matrix IS invertible.
Worker PID#5160 is handling client PID#5199, matrix size 4.000000x4.000000, pool busy 3/5
Worker PID#5157 is handling client PID#5217, matrix size 4.000000x4.000000, pool busy 1/5
Worker PID#5157 responding to client PID#5217: the matrix IS invertible.
Worker PID#5159 is handling client PID#5228, matrix size 4.000000x4.000000, pool busy 2/5
Worker PID#5159 responding to client PID#5228: the matrix IS invertible.
Worker PID#5160 responding to client PID#5230: the matrix IS NOT invertible.
Worker PID#5160 is handling client PID#5230, matrix size 3.000000x3.000000, pool busy 3/5
Worker PID#5161 is handling client PID#5233, matrix size 4.000000x4.000000, pool busy 4/5
Worker PID#5161 responding to client PID#5233: the matrix IS invertible.
Plain Text
Tab Width: 8
Ln 1, Col 1
INS
/*sem_t* sem = sem_open("PDA22", (O_CREAT | O_RDWR | O_EXCL), 0666, 0);
if (sem == SEM_FAILED) {
fprintf(stdout, "Error: Duble Instantiation\n");
exit(1);
atar@ubuntu: ~/Desktop/midterm
atar@ubuntu: ~/Desktop/midterm 80x2
Client PID#5217: the matrix is invertible ,total time 8.000000 second,goodbye.
atar@ubuntu:~/Desktop/midterm$
atar@ubuntu: ~/Desktop/midterm 80x3
Client PID#5228 (data.csv) is submitting a 4x4 matrix
Client PID#5228: the matrix is invertible ,total time 8.000000 second,goodbye.
atar@ubuntu:~/Desktop/midterm$
atar@ubuntu: ~/Desktop/midterm 80x3
Client PID#5230: the matrix is not invertible ,total time 8.000000 second,goodbye.
atar@ubuntu:~/Desktop/midterm$
atar@ubuntu: ~/Desktop/midterm 80x4
atar@ubuntu:~/Desktop/midterm$ ./client -s pathToServerFifo -o data.csv
Client PID#5233 (data.csv) is submitting a 4x4 matrix
Client PID#5233: the matrix is invertible ,total time 8.000000 second,goodbye.
atar@ubuntu:~/Desktop/midterm$
atar@ubuntu: ~/Desktop/midterm 80x20
atar 4348 4338 0 00:20 pts/0 00:00:00 /bin/bash
root 4361 2 0 00:21 ? 00:00:00 [kworker/1:2-events]
atar 4364 4338 0 00:21 pts/1 00:00:00 /bin/bash
root 4479 2 0 00:22 ? 00:00:00 [kworker/u256:1-events_unbou
atar 4487 1478 1 00:22 ? 00:00:05 /opt/sublime_text/sublime_te
atar 4506 4487 0 00:22 ? 00:00:00 /opt/sublime_text/plugin_hos
atar 4509 4487 0 00:22 ? 00:00:00 /opt/sublime_text/plugin_hos
root 4709 2 0 00:24 ? 00:00:00 [kworker/0:0-events]
atar 4722 4338 0 00:24 pts/2 00:00:00 /bin/bash
root 4902 2 0 00:27 ? 00:00:00 [kworker/1:1-mm_percpu_wq]
root 4905 2 0 00:27 ? 00:00:00 [kworker/u256:0-events_unbou
root 4920 2 0 00:29 ? 00:00:00 [kworker/0:2-events]
atar 4990 4338 0 00:30 pts/3 00:00:00 /bin/bash
atar 4998 4338 0 00:30 pts/4 00:00:00 /bin/bash
atar 5104 1478 0 00:31 ? 00:00:00 /usr/libexec/tracker-store
root 5152 2 0 00:31 ? 00:00:00 [kworker/u256:3-events_unbou
atar 5153 4364 0 00:31 pts/1 00:00:00 ps -ef
atar@ubuntu:~/Desktop/midterm$ ./serverY -s pathToServerFifo -o pathToLogFile -p 5 -r 3 -t 8
atar@ubuntu:~/Desktop/midterm$