CSE 344 SYSTEM PROGRAMMING
MIDTERM PROJECT REPORT

MURAT BEYAZ

151044004

Problem Definition

Servers consist commonly of very powerful hardware in terms of both processing and storage capacity, so as to be able to handle the high number of often concurrent client requests. However, centralization of such tasks can lead to financial disadvantages.

An inefficient approach to solve this problem is to organize multiple mid or even low end machines serially. For example client X connects to server Y. If server Y is not busy then it processes the request of X and responds to X. Otherwise if Y is busy handling other clients then Y forwards X's request to another server Z who processes it and responds to client X.

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 were 3 programs to be implemented client X, server Y and server Z. There will be a single Y and Z and there can be arbitrary number of client processes.

Problem Solution

Program X: This program takes 2 command line arguments from the user first is server fifo file path and the second is the path of the test file. Test file is a csv file contains a square matrix. Each line of the file is the one row of the matrix.

After taking these arguments client X creates a request object. Request struct is shown below:

```c
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <errno.h>


#define SERVER_FIFO "/tmp/seqnum_sv"
#define CLIENT_FIFO_TEMPLATE "/tmp/seqnum_cl.%ld"
#define CLIENT_FIFO_NAME_LEN (sizeof(CLIENT_FIFO_TEMPLATE)+20)

struct request{
    pid_t pid;
    char reqtxt[500];
    int seqLen;
};


struct response{
    char yes_or_no[5];
};
```

After all the operations on server Y or server Z result turns back to client X via client fifo. Client fifo name is defined as seqnum_cl.<PID> to avoid race of the clients. Also response struct is shown in the image above. This is the struct returns from the worker process.

While the client X waits for the response I used the code shown below so I didn't create a busy waiting condition.

```
time_t begin = time(NULL);
clientFd = open(clientFifo, O_RDONLY);
if(clientFd == -1)
  fprintf(stderr,"open %s", clientFifo);
if(read(clientFd, &resp, sizeof(struct response)) != sizeof(struct response))
  fprintf(stderr,"Can't read response from server");
```

After the response comes client prints the result to the console and exits.

Program Y: Program Y takes 5 arguments from the command line.

-s : server fifo path
-o : log file path
-p : pool size of server y
-r : pool size of server z
-t : sleep time for worker processes

Server Y after getting the arguments creates two semaphores to create synchronization between them. Server Y forks and creates server Z.

After the fork both server Y and server Z become daemon processes with become_daemon function shown below:

```
int becomeDaemon(int flags){
    int fd;
    switch(fork()){
        case -1:return -1;
        case 0: break;
        default:_exit(EXIT_SUCCESS);
    }

    if(setsid() == -1)
        return -1;

    switch(fork()){
        case -1:return -1;
        case 0: break;
        default:_exit(EXIT_SUCCESS);
    }


    if (!(flags & BD_NO_UMASK0))
        umask(0);


    if (!(flags & BD_NO_REOPEN_STD_FDS)) {
        close(STDIN_FILENO);

        fd = open("/dev/null", O_RDWR);

        if (fd != STDIN_FILENO)
            return -1;
        if (dup2(STDIN_FILENO, STDOUT_FILENO) != STDOUT_FILENO)
            return -1;
        if (dup2(STDIN_FILENO, STDERR_FILENO) != STDERR_FILENO)
            return -1;
    }

    return 0;
}
```

This function is the same with the one we saw during our class. After this function call processes becomes daemon and contact of the process with the terminal is broken.

After becoming daemon both server Z and server Y created their pools of worker process.

Server Y creates a 2d array for creating pipes to all worker processes and while forking created semaphores to each child process with the name of “sem_<%d>. This semaphore is created for both making synchronization and understanding if the child process is available or not. As mentioned before if all the worker processes are busy then we will redirect the request to server Z. To understand the availability of child processes I used sem_getvalue() function. If there is available worker process sending the request to it via dedicated pipe else redirecting to server z via pipe between Z and Y. Also posted semaphore of z to wake up and direct request to worker process.

Server Z shares requests to worker processes with a shared memory. I created a queue on shared memory. Struct of queue and struct of nodes of the queue shown below:

```
struct QNode {
    char request[500];
    size_t next;
};

struct Queue {
    QNode nodes[REQ_SIZE];
    size_t used;
    size_t empty;
    size_t head;
};
```

After Z receives a request puts it in a node object and adds to queue. After adding to shared memory calls a worker process’s semaphore and continues. If there is not available worker process just adds to queue and iterates until a worker process becomes available. Worker processes removes nodes from the queue after reading them.

Worker Processes: Worker processes first reads the test file’s first row and finds the number of integers then allocates space for nxn matrice. Then reads the file into matrix and calls findInvertibility function to find either the matrix is invertible or not. FindInvertibility function uses findcofact function to find cofactor of sub matrices.

After the calculations sleeps t seconds and returns to client via client fifo.

When all clients are done Y, Z and worker processes keeps waiting for new requests. Then I sent kill -2 <PID_of_Y> command and Server Y killed all of it’s child process and send signal to server Z and server Z killed it’s child processes.

TEST

1) Server Y ran with command shown below:

```
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$ ./serverY -s /home/mrtbyz/Desktop/System_Midterm/serfifo
 -o lokum.log -p 4 -r 4 -t 2
```

Client X called 3 times as shown below:

```
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$ ./client -s /home/mrtbyz/Desktop/System_Midterm/serfifo
-o test.csv & ./client -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv & ./client -s /home
/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv
```

Log file result shown below:

serverY.c | lokum.log

```
Fri Apr 15 16:02:15 2022
 Server Y (/home/mrtbyz/Desktop/System_Midterm/serfifo lokum.log p=4 t=2) started (27380)
Fri Apr 15 16:02:15 2022
 Instantiated server Z
Fri Apr 15 16:02:17 2022
 Worker PID#27370 is handling client PID#27403 matrix size 3x3
Fri Apr 15 16:02:17 2022
 Worker PID#27370 is responding to client PID#27403: the matrix is NOT invertible
Fri Apr 15 16:02:17 2022
 Worker PID#27371 is handling client PID#27405 matrix size 3x3
Fri Apr 15 16:02:17 2022
 Worker PID#27371 is responding to client PID#27405: the matrix is NOT invertible
Fri Apr 15 16:02:19 2022
 Worker PID#27370 is handling client PID#27404 matrix size 3x3
Fri Apr 15 16:02:19 2022
 Worker PID#27370 is responding to client PID#27404: the matrix is NOT invertible
Fri Apr 15 16:02:34 2022
 SIGINT received exiting server Y. Total requests: 3, 0 invertible, 3 not.
```

Client prompt output:

```
Fri Apr 15 16:02:17 2022
 Client PID#27403 (test.csv) is submitting a 3x3 matrix
Fri Apr 15 16:02:17 2022
 Client PID#27404 (test.csv) is submitting a 3x3 matrix
Fri Apr 15 16:02:17 2022
 Client PID#27405 (test.csv) is submitting a 3x3 matrix
Fri Apr 15 16:02:19 2022
 CLient PID#27403: the matrix is NOT invertible, total time 2 seconds, goodbye.
Fri Apr 15 16:02:19 2022
 CLient PID#27405: the matrix is NOT invertible, total time 2 seconds, goodbye.
[21]   Done                    ./client -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv
[22]-  Done                    ./client -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$ Fri Apr 15 16:02:21 2022
 CLient PID#27404: the matrix is NOT invertible, total time 4 seconds, goodbye.
```

After killing server Y:

```
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$ ps -aux | grep Y
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mrtbyz     26020  2.0  1.3 710508 109748 ?       Sl   15:41   0:25 /snap/sublime-text/112/opt/sublime
_text/sublime_text --detached /home/mrtbyz/Desktop/System_Midterm/TEST/serverY.c
mrtbyz     27370  0.0  0.0   2732   112 pts/0    S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27371  0.0  0.0   2732   112 pts/0    S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27372  0.0  0.0   2752  1268 ?        S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27373  0.0  0.0   2732   112 pts/0    S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27374  0.0  0.0   2736   112 pts/0    S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27375  0.0  0.0   2736   112 ?        S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27377  0.0  0.0   2740   112 ?        S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27378  0.0  0.0   2744   112 ?        S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27379  0.0  0.0   2748   112 ?        S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27380  0.0  0.0   2740   112 ?        S    16:02   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     27417  0.0  0.0  11776   720 pts/3    S+   16:02   0:00 grep --color=auto Y
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$ sudo kill -2 27380
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$ ps -aux | grep Y
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mrtbyz     26020  2.0  1.2 710508 102184 ?       Sl   15:41   0:25 /snap/sublime-text/112/opt/sublime
_text/sublime_text --detached /home/mrtbyz/Desktop/System_Midterm/TEST/serverY.c
mrtbyz     27446  0.0  0.0  11776   720 pts/3    S+   16:02   0:00 grep --color=auto Y
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$
```

As you have seen only killing server Y killed other processes also.

2) Server Y ran with command shown below:

```
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$ ./serverY -s /home/mrtbyz/Desktop/System_Midterm/serfifo
 -o lokum.log -p 4 -r 4 -t 2
```

Client X called 20 times as shown below:

```
mrtbyz@mrtbyz:~/Desktop/System_Midterm/TEST$ ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o te
st.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv & ./X -s /home/mrtbyz/Desktop
/System_Midterm/serfifo -o test.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv
& ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test2.csv & ./X -s /home/mrtbyz/Desktop/Syste
m_Midterm/serfifo -o test2.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv & ./X
 -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv & ./X -s /home/mrtbyz/Desktop/System_Midt
erm/serfifo -o test.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv & ./X -s /ho
me/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/ser
fifo -o test.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv & ./X -s /home/mrtb
yz/Desktop/System_Midterm/serfifo -o test.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o
 test2.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test2.csv & ./X -s /home/mrtbyz/De
sktop/System_Midterm/serfifo -o test.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test
.csv & ./X -s /home/mrtbyz/Desktop/System_Midterm/serfifo -o test.csv & ./X -s /home/mrtbyz/Desktop/S
ystem_Midterm/serfifo -o test.csv
```

Log file result shown below:

```
Fri Apr 15 17:16:19 2022
 Server Y (/home/mrtbyz/Desktop/System_Midterm/serfifo lokum.log p=4 t=2) started (28607)
Fri Apr 15 17:16:19 2022
 Instantiated server Z
Fri Apr 15 17:19:39 2022
 Worker PID#28597 is handling client PID#28692 matrix size 3x3
Fri Apr 15 17:19:39 2022
 Worker PID#28597 is responding to client PID#28692: the matrix is NOT invertible
Fri Apr 15 17:19:39 2022
 Worker PID#28600 is handling client PID#28690 matrix size 3x3
Fri Apr 15 17:19:39 2022
 Worker PID#28600 is responding to client PID#28690: the matrix is NOT invertible
Fri Apr 15 17:19:39 2022
 Worker PID#28601 is handling client PID#28697 matrix size 3x3
Fri Apr 15 17:19:39 2022
 Worker PID#28601 is responding to client PID#28697: the matrix is NOT invertible
Fri Apr 15 17:19:39 2022
 Worker PID#28603 is handling client PID#28695 matrix size 3x3
Fri Apr 15 17:19:39 2022
 Worker PID#28603 is responding to client PID#28695: the matrix is NOT invertible
Fri Apr 15 17:19:39 2022
 Forwarding request of client PID#28705 to serverZ, matrix size 3x3
Fri Apr 15 17:19:39 2022
 Z: Worker PID#28602 is handling client PID#28705 matrix size 3x3
Fri Apr 15 17:19:39 2022
 Z: Worker PID#28602 is responding to client PID#28705: the matrix is NOT invertible
Fri Apr 15 17:19:39 2022
 Forwarding request of client PID#28696 to serverZ, matrix size 3x3
Fri Apr 15 17:19:39 2022
 Z: Worker PID#28604 is handling client PID#28696 matrix size 3x3
Fri Apr 15 17:19:39 2022
 Z: Worker PID#28604 is responding to client PID#28696: the matrix is NOT invertible
Fri Apr 15 17:19:39 2022
 Forwarding request of client PID#28709 to serverZ, matrix size 4x4
Fri Apr 15 17:19:39 2022
 Z: Worker PID#28606 is handling client PID#28709 matrix size 4x4
Fri Apr 15 17:19:39 2022
 Z: Worker PID#28606 is responding to client PID#28709: the matrix is invertible
Fri Apr 15 17:19:39 2022
 Forwarding request of client PID#28707 to serverZ, matrix size 3x3
Fri Apr 15 17:19:39 2022
 Z: Worker PID#28608 is handling client PID#28707 matrix size 3x3
Fri Apr 15 17:19:39 2022
 Z: Worker PID#28608 is responding to client PID#28707: the matrix is NOT invertible
Fri Apr 15 17:19:41 2022
 Worker PID#28597 is handling client PID#28691 matrix size 3x3
Fri Apr 15 17:19:41 2022
 Worker PID#28597 is responding to client PID#28691: the matrix is NOT invertible
Fri Apr 15 17:19:41 2022
 Worker PID#28600 is handling client PID#28693 matrix size 3x3
Fri Apr 15 17:19:41 2022
 Worker PID#28600 is responding to client PID#28693: the matrix is NOT invertible
Fri Apr 15 17:19:41 2022
 Worker PID#28601 is handling client PID#28698 matrix size 3x3
Fri Apr 15 17:19:41 2022
 Worker PID#28601 is responding to client PID#28698: the matrix is NOT invertible
Fri Apr 15 17:19:41 2022
 Worker PID#28603 is handling client PID#28694 matrix size 3x3
```

```
Fri Apr 15 17:19:41 2022
 Worker PID#28603 is responding to client PID#28694: the matrix is NOT invertible
Fri Apr 15 17:19:41 2022
 Forwarding request of client PID#28700 to serverZ, matrix size 3x3
Fri Apr 15 17:19:41 2022
 Z: Worker PID#28602 is handling client PID#28700 matrix size 3x3
Fri Apr 15 17:19:41 2022
 Z: Worker PID#28602 is responding to client PID#28700: the matrix is NOT invertible
Fri Apr 15 17:19:41 2022
 Forwarding request of client PID#28706 to serverZ, matrix size 3x3
Fri Apr 15 17:19:41 2022
 Z: Worker PID#28604 is handling client PID#28706 matrix size 3x3
Fri Apr 15 17:19:41 2022
 Z: Worker PID#28604 is responding to client PID#28706: the matrix is NOT invertible
Fri Apr 15 17:19:41 2022
 Forwarding request of client PID#28702 to serverZ, matrix size 3x3
Fri Apr 15 17:19:41 2022
 Z: Worker PID#28606 is handling client PID#28702 matrix size 3x3
Fri Apr 15 17:19:41 2022
 Z: Worker PID#28606 is responding to client PID#28702: the matrix is NOT invertible
Fri Apr 15 17:19:41 2022
 Forwarding request of client PID#28699 to serverZ, matrix size 3x3
Fri Apr 15 17:19:41 2022
 Z: Worker PID#28608 is handling client PID#28699 matrix size 3x3
Fri Apr 15 17:19:41 2022
 Z: Worker PID#28608 is responding to client PID#28699: the matrix is NOT invertible
Fri Apr 15 17:19:43 2022
 Worker PID#28597 is handling client PID#28708 matrix size 3x3
Fri Apr 15 17:19:43 2022
 Worker PID#28597 is responding to client PID#28708: the matrix is NOT invertible
Fri Apr 15 17:19:43 2022
 Worker PID#28600 is handling client PID#28701 matrix size 3x3
Fri Apr 15 17:19:43 2022
 Worker PID#28600 is responding to client PID#28701: the matrix is NOT invertible
Fri Apr 15 17:19:43 2022
 Forwarding request of client PID#28704 to serverZ, matrix size 3x3
Fri Apr 15 17:19:43 2022
 Z: Worker PID#28602 is handling client PID#28704 matrix size 3x3
Fri Apr 15 17:19:43 2022
 Z: Worker PID#28602 is responding to client PID#28704: the matrix is NOT invertible
Fri Apr 15 17:19:43 2022
 Forwarding request of client PID#28703 to serverZ, matrix size 3x3
Fri Apr 15 17:19:43 2022
 Z: Worker PID#28604 is handling client PID#28703 matrix size 3x3
Fri Apr 15 17:19:43 2022
 Z: Worker PID#28604 is responding to client PID#28703: the matrix is NOT invertible
Fri Apr 15 17:20:16 2022
 SIGINT received exiting server Y. Total requests: 10, 0 invertible, 10 not.
```

As you can see from the log file server Z's processes are handling the requests but evendough I used the same code I couldn't write the Z's informative message to the log file only result of Y appears at the end.
After killing server Y:

```
mrtbyz@mrtbyz:~/Desktop/System_Midterm/151044004$ ps -aux | grep Y
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mrtbyz     28597  0.0  0.0   2736   112 pts/0    S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28599  0.8  0.0   2840  1376 ?        S    17:16   0:01 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28600  0.0  0.0   2740   112 pts/0    S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28601  0.0  0.0   2740   112 pts/0    S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28602  0.0  0.0   2824   112 ?        S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28603  0.0  0.0   2744   112 pts/0    S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28604  0.0  0.0   2828   112 ?        S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28606  0.0  0.0   2832   112 ?        S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28607  0.0  0.0   2740   112 ?        S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28608  0.0  0.0   2836   112 ?        S    17:16   0:00 ./serverY -s /home/mrtbyz/Desktop/
System_Midterm/serfifo -o lokum.log -p 4 -r 4 -t 2
mrtbyz     28758  0.0  0.0  11776   716 pts/3    S+   17:20   0:00 grep --color=auto Y
mrtbyz@mrtbyz:~/Desktop/System_Midterm/151044004$ sudo kill -2 28607
[sudo] password for mrtbyz:
mrtbyz@mrtbyz:~/Desktop/System_Midterm/151044004$ ps -aux | grep Y
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
mrtbyz     28794  0.0  0.0  11776   720 pts/3    S+   17:20   0:00 grep --color=auto Y
mrtbyz@mrtbyz:~/Desktop/System_Midterm/151044004$
```