CSE 344 – SYSTEM PROGRAMMING

HW5

Threads synchronization

151044072 – ALPER YAŞAR

Purpose of this Project?

The idea is to use POSIX threads to parallelize a couple of simple mathematical tasks.

Firstly taking and checking arguments. If argument number is less than 11 giving error.

The process will start by reading the two files into memory. It will read sequentially 2^nx2^n characters from each file. 'n' value must be bigger than 2. There is a m threads. 'm' value must be bigger than 1.

After the arguments check then creating sigaction for SIGINT signal. If the user push to CTRL+C then program quit. Closing all files and freeing all datas.

Then opening given 2 files for read only and opening the output file. For output file there is extra special conditions. If file is not exist creating and truncate it.

Allocation

After that allocating the matrixes. Allocating as 2^nx2^n matrix. When allocating matrixes reading files as given 2^nx2^n. If files less than given condition then giving error and exiting.

Then initializing mutex and condition variables.

After the initialization creating threads.

Thread

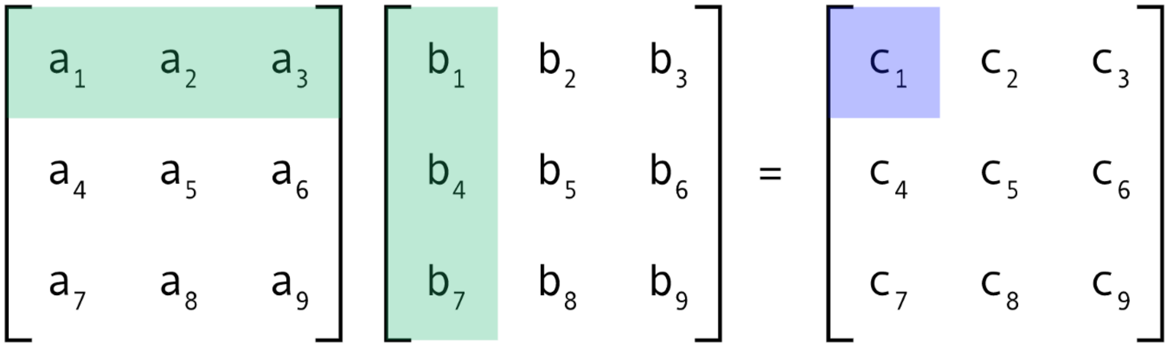
Firstly assigning time for print. Then calculating row number and calculating start and finish column for thread.

*start = ((int)(r/m))*index;*

That is giving the start column and same way finding the finish column.

Multiplication of matrices

First loop starting from “*start column*” to “*end column*”. There is 2 nested loop in this loop. They are started from 0 to 2^n .



The image is the example of matrix multiplication. The loop doing it.

```
for (int i = start; i < finish; ++i)  {
    for (int k = 0; k < r; ++k)   {
      temp =0;
      for (int j = 0; j < r; ++j)   {
        temp += matrix1[k][j]*matrix2[j][i];
      }
      resMat[k][i] = temp;
    }
  }
```

Each thread printed calculation time after calculate.

Then waiting for all threads calculation with mutex. If all threads calculation are done then unlock mutex and all of them calculating the second part of task.

The second task and calculate the 2D Discrete Fourier Transform. I took the formula from https://youtu.be/Iz6C1ny-F2Q

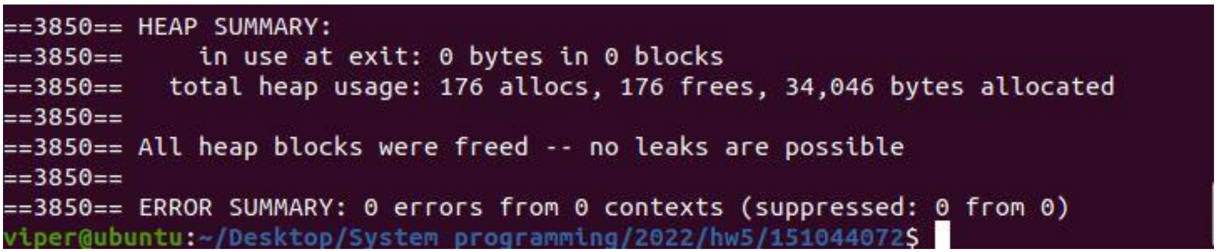
```
  for (int x = start; x < finish; ++x)   {
    for (int y = 0; y < r; ++y)   {
      float ak=0;
      float bk=0;
      for (int z = 0; z < r; ++z)   {
        for (int t = 0; t < r; ++t)    {
          float x1=-2.0*PI*x*z/(float)r;
          float y1=-2.0*PI*y*t/(float)r;
          ak+=resMat[z][t]*cos(x1+y1);
          bk+=resMat[z][t]*1.0*sin(x1+y1);
        }
      }
      resRE[x][y]=ak;
      resIM[x][y]=bk;
    }
  }
```

Then printing the calculation time of each thread and exiting. The main process collecting each thread with pthread_join.

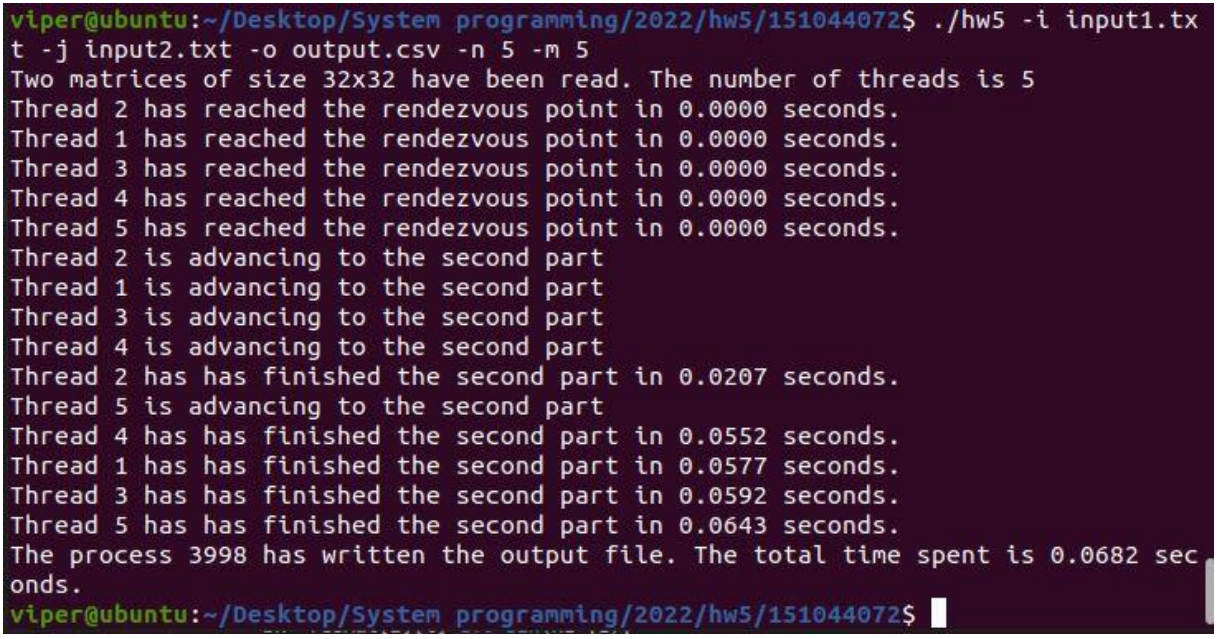
Writing the results to output file, printing the all calculation time and then exiting from system.

When exiting closing all files freeing all datas.

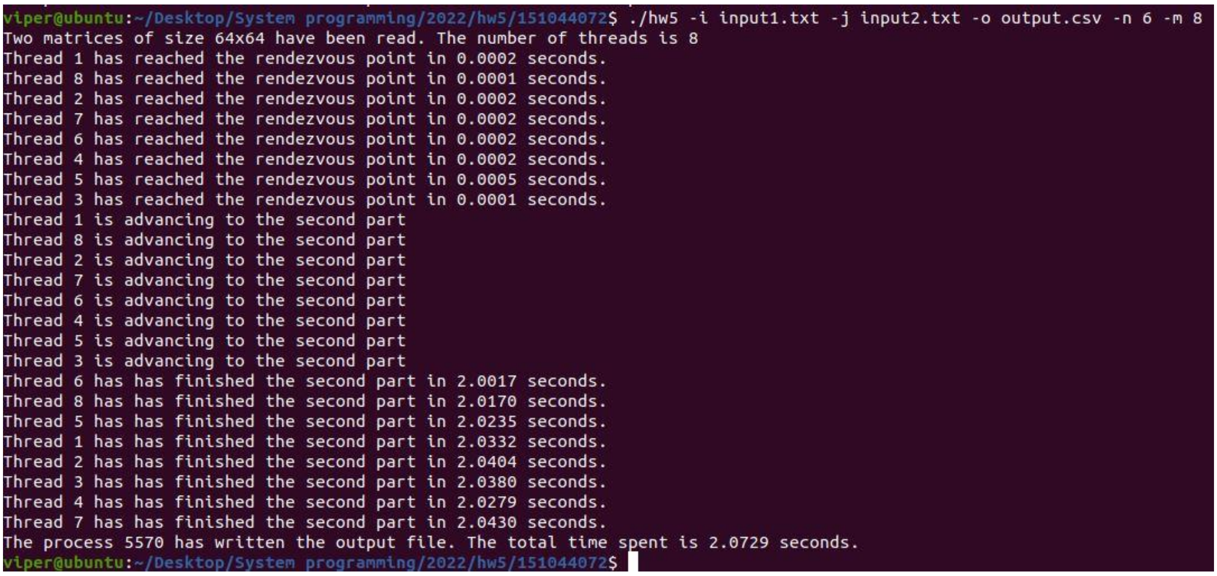
Results



```
==3850== HEAP SUMMARY:
==3850==     in use at exit: 0 bytes in 0 blocks
==3850==   total heap usage: 176 allocs, 176 frees, 34,046 bytes allocated
==3850==
==3850== All heap blocks were freed -- no leaks are possible
==3850==
==3850== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
viper@ubuntu:~/Desktop/System programming/2022/hw5/151044072$
```

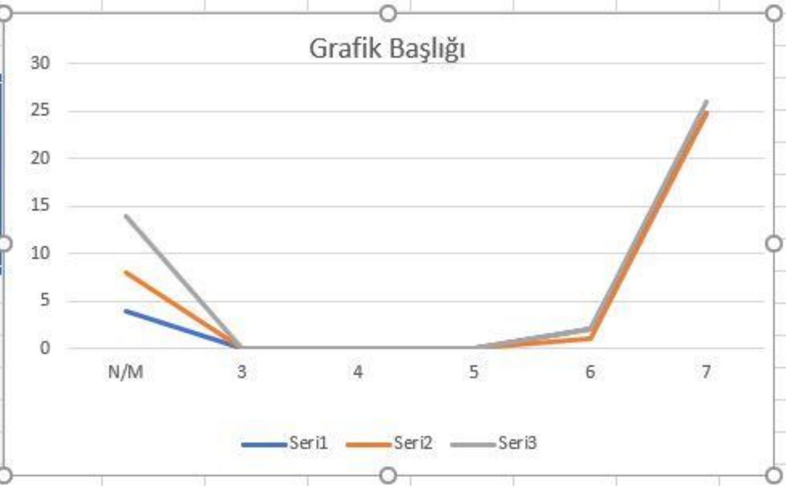


```
viper@ubuntu:~/Desktop/System programming/2022/hw5/151044072$ ./hw5 -i input1.tx
t -j input2.txt -o output.csv -n 5 -m 5
Two matrices of size 32x32 have been read. The number of threads is 5
Thread 2 has reached the rendezvous point in 0.0000 seconds.
Thread 1 has reached the rendezvous point in 0.0000 seconds.
Thread 3 has reached the rendezvous point in 0.0000 seconds.
Thread 4 has reached the rendezvous point in 0.0000 seconds.
Thread 5 has reached the rendezvous point in 0.0000 seconds.
Thread 2 is advancing to the second part
Thread 1 is advancing to the second part
Thread 3 is advancing to the second part
Thread 4 is advancing to the second part
Thread 2 has has finished the second part in 0.0207 seconds.
Thread 5 is advancing to the second part
Thread 4 has has finished the second part in 0.0552 seconds.
Thread 1 has has finished the second part in 0.0577 seconds.
Thread 3 has has finished the second part in 0.0592 seconds.
Thread 5 has has finished the second part in 0.0643 seconds.
The process 3998 has written the output file. The total time spent is 0.0682 sec
onds.
viper@ubuntu:~/Desktop/System programming/2022/hw5/151044072$
```



```
viper@ubuntu:~/Desktop/System programming/2022/hw5/151044072$ ./hw5 -i input1.txt -j input2.txt -o output.csv -n 6 -m 8
Two matrices of size 64x64 have been read. The number of threads is 8
Thread 1 has reached the rendezvous point in 0.0002 seconds.
Thread 8 has reached the rendezvous point in 0.0001 seconds.
Thread 2 has reached the rendezvous point in 0.0002 seconds.
Thread 7 has reached the rendezvous point in 0.0002 seconds.
Thread 6 has reached the rendezvous point in 0.0002 seconds.
Thread 4 has reached the rendezvous point in 0.0002 seconds.
Thread 5 has reached the rendezvous point in 0.0005 seconds.
Thread 3 has reached the rendezvous point in 0.0001 seconds.
Thread 1 is advancing to the second part
Thread 8 is advancing to the second part
Thread 2 is advancing to the second part
Thread 7 is advancing to the second part
Thread 6 is advancing to the second part
Thread 4 is advancing to the second part
Thread 5 is advancing to the second part
Thread 3 is advancing to the second part
Thread 6 has has finished the second part in 2.0017 seconds.
Thread 8 has has finished the second part in 2.0170 seconds.
Thread 5 has has finished the second part in 2.0235 seconds.
Thread 1 has has finished the second part in 2.0332 seconds.
Thread 2 has has finished the second part in 2.0404 seconds.
Thread 3 has has finished the second part in 2.0380 seconds.
Thread 4 has has finished the second part in 2.0279 seconds.
Thread 7 has has finished the second part in 2.0430 seconds.
The process 5570 has written the output file. The total time spent is 2.0729 seconds.
viper@ubuntu:~/Desktop/System programming/2022/hw5/151044072$
```

N/M	4	8	14
3	0,0022	0,0018	0,0027
4	0,0071	0,0069	0,0083
5	0,0682	0,0754	0,0789
6	2,1224	1,0876	2,0698
7	24,7688	24,8037	26,0111


Grafik Başlığı
N/M
Seri1
Seri2
Seri3