CSE 344 SYSTEMS PROGRAMMING

HOMEWORK 5 REPORT

MURAT BEYAZ

151044004

Problem Definition: The objective of the project is doing some mathematical operations on matrices with POSIX threads. The program takes 5 command line arguments.

FilePath1 : Path of first input file.
FilePath2 : Path of second input file.
Output : Path of output file.
N : Power of 2 to read from file and create a matrix.
M : Number of threads.

Input files are ASCII files with arbitrary length. The process will read the input files and write them into matrices. The process will read the files 2^n by 2^n characters. If the file doesn't contain enough characters then the process will acknowledge the user and exits.

The process will create m threads and let them to do the necessary calculations.

The threads have 2 different operations. Threads will share the matrices to calculate. Meaning if matrix has 2^n columns each thread will calculate the $2^n/m$ columns. First threads will calculate C=AxB operation. Threads will work according to synchronization barrier paradigm.

The threads needs the whole C matrix to calculate 2D DFT operation so that threads will wait the others to continue calculating DFT. This is synchronization barrier problem. The solution of this problem is using a condition variable and a mutex together. Solution code section is shown below.

```
pthread_mutex_lock(&lock);
++arrived;
clock_t end = clock();
double time_sp = 0.0;
time_sp = (double)(end - begen) / CLOCKS_PER_SEC;
time(&ltime);
timeinfo = localtime ( &ltime );
printf("%s Thread %d has reached the rendezvous point in %f seconds.\n", asctime(timeinfo),thread_index, time_sp);
if(arrived < thread_count){
    pthread_cond_wait(&cond1, &lock);
    time(&ltime);
    timeinfo = localtime ( &ltime );
    printf("%s Thread %d is advancing to the second part.\n", asctime(timeinfo),thread_index);
}
else{
    for(i=0;i<thread_count-1;i++)
        pthread_cond_signal(&cond1);
    time(&ltime);
    timeinfo = localtime ( &ltime );
    printf("%s Thread %d is advancing to the second part.\n", asctime(timeinfo),thread_index);
}
pthread_mutex_unlock(&lock);
```

As you can see after every thread comes to the rendezvous point checks the arrived variable if it is equal to total thread number signals the other threads and wakes them up. Until then other threads are waiting with pthread_cond_wait function.

After all threads woke up they start to calculate 2D DFT on C matrix. The algorithm to calculate 2D DFT is shown below.

```
int k,l,m,n;
for(k=0;k<dimension;k++){ // k
    for(l=start_index;l<=end_index;l++){ // l
        reel = 0.0;
        imaginer = 0.0;
        for(m=0;m<dimension;m++){ // m
            for(n=0;n<dimension;n++){ // n
                reel = reel + matrixC[n][m]*cos(2 * 3.141592 * (((double)(k*n)/(double)dimension) + ((double)(m*l)/(double)dimension)));
                imaginer = imaginer + matrixC[n][m]*sin(-2 * 3.141592 * (((double)(k*n)/(double)dimension) + ((double)(m*l)/(double)dimension)));
                if(sigint_Flag == 1){
                    free_all();
                    printf("Thread %d is received sigint exiting.\n", thread_index);
                    return NULL;
                }
            }
        }
        matrixReel[k][l] = reel;
        matrixImag[k][l] = imaginer;
    }
}
```

There is a 4 layered for loop. First two is for recording results to the assigned indexes of the thread. Inner 2 loop is for traversing on matrix C and calculate the 2D DFT. While calculating the DFT the formula below is used.

$$X_{lm} = \sum_{j=0}^{N-1} \sum_{k=0}^{M-1} x_{jk} e^{-2\pi i \left(\frac{jl}{N} + \frac{km}{M} \right)}$$

Calculation of the reel and imaginary parts are calculated with cos and sin functions. These results recorded in 2 different matrices matrixReel and matrixImag. These result will be printed into given output file.

The function to write results into csv file is shown below.

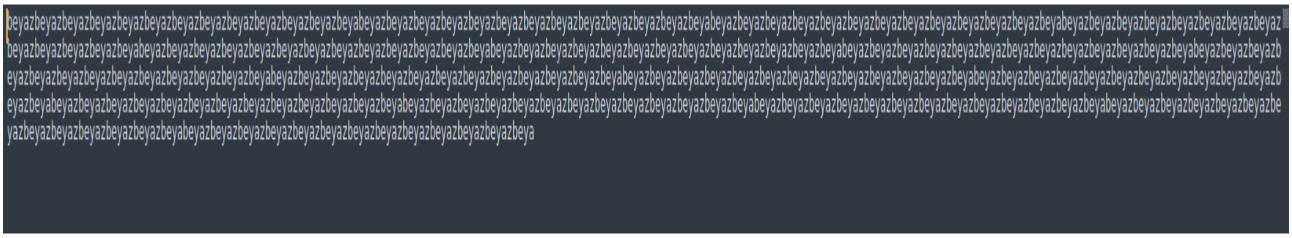
```
void write_array_to_file(int dimension){
    int i,j;
    int fd = open(outputfile, O_CREAT|O_WRONLY|O_APPEND, 0777);
    char line[1024];
    for(i=0;i<dimension;i++){
        for(j=0;j<dimension;j++){
            sprintf(line, "%.3f+%.3fi", matrixReel[i][j], matrixImag[i][j]);
            write(fd, line, strlen(line));
            if(j != dimension-1){
                write(fd,",",strlen(","));
            }
        }
        write(fd,"\n",strlen("\n"));
    }
    close(fd);
}
```

Meanwhile the process waits the threads with pthread_join, writes result to output file and exits.

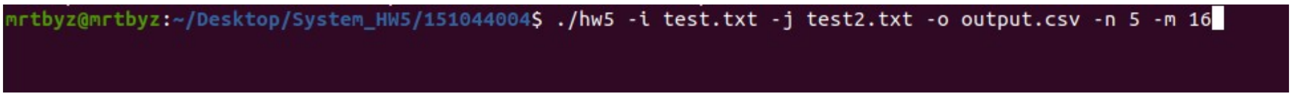
TEST

Input file 1: (1024 characters)

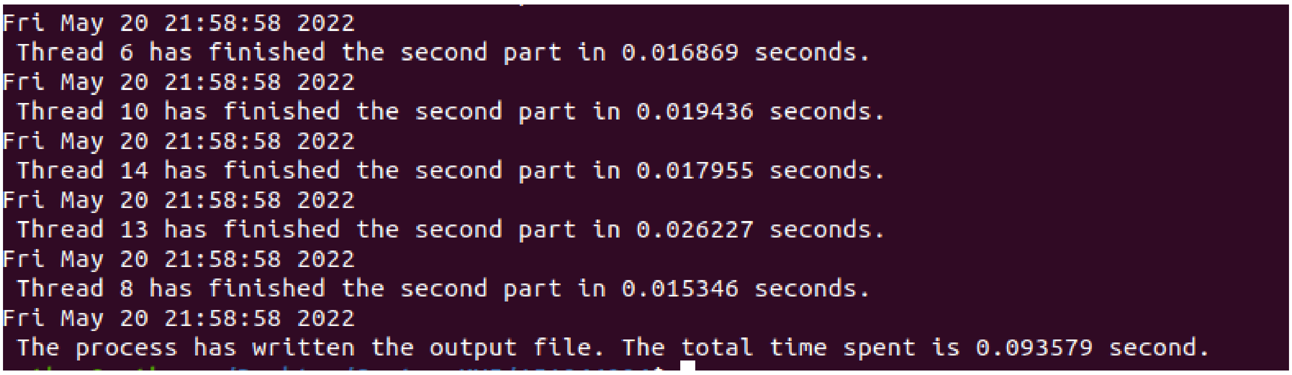
Input file 2: (1024 characters)



Run Command 1:



Result 1:



Run Command 2:

```
mrtbyz@mrtbyz:~/Desktop/System_HW5/151044004$ ./hw5 -i test.txt -j test2.txt -o output.csv -n 5 -m 8
```

Result 2:

```
Fri May 20 21:59:53 2022
 Thread 3 has finished the second part in 0.052261 seconds.
Fri May 20 21:59:53 2022
 Thread 1 has finished the second part in 0.085299 seconds.
Fri May 20 21:59:53 2022
 Thread 4 has finished the second part in 0.087612 seconds.
Fri May 20 21:59:53 2022
 Thread 7 has finished the second part in 0.035291 seconds.
Fri May 20 21:59:53 2022
 Thread 5 has finished the second part in 0.025583 seconds.
Fri May 20 21:59:53 2022
 The process has written the output file. The total time spent is 0.099607 second.
```

Run Command 3:

```
mrtbyz@mrtbyz:~/Desktop/System_HW5/151044004$ ./hw5 -i test.txt -j test2.txt -o output.csv -n 5 -m 4
```

Result 3:

```
Fri May 20 22:01:13 2022
 Thread 2 is advancing to the second part.
Fri May 20 22:01:13 2022
 Thread 1 has finished the second part in 0.091012 seconds.
Fri May 20 22:01:13 2022
 Thread 3 has finished the second part in 0.095720 seconds.
Fri May 20 22:01:13 2022
 Thread 2 has finished the second part in 0.095706 seconds.
Fri May 20 22:01:13 2022
 Thread 0 has finished the second part in 0.097908 seconds.
Fri May 20 22:01:13 2022
 The process has written the output file. The total time spent is 0.100625 second.
```

Run Command 4:

```
mrtbyz@mrtbyz:~/Desktop/System_HW5/151044004$ ./hw5 -i test.txt -j test2.txt -o output.csv -n 5 -m 2
```

Result 4:

```
Fri May 20 22:02:10 2022
 Two matrices of size 32x32 have been read. The number of threads is 2
Fri May 20 22:02:10 2022
 Thread 0 has reached the rendezvous point in 0.000299 seconds.
Fri May 20 22:02:10 2022
 Thread 1 has reached the rendezvous point in 0.000390 seconds.
Fri May 20 22:02:10 2022
 Thread 1 is advancing to the second part.
Fri May 20 22:02:10 2022
 Thread 0 is advancing to the second part.
Fri May 20 22:02:10 2022
 Thread 0 has finished the second part in 0.086605 seconds.
Fri May 20 22:02:10 2022
 Thread 1 has finished the second part in 0.088762 seconds.
Fri May 20 22:02:10 2022
 The process has written the output file. The total time spent is 0.090129 second.
```

Thread Count	2	4	8	16
Total Time	0.09	0.10	0.99	0.93

As you can see from the table total time to calculate result is consuming less time.

Test 2:

Argument tests are shown below: