GTU

**DEPARTMENT OF COMPUTER ENGINEERING**

**CSE 344 – Spring 2022**

**HOMEWORK 5  
REPORT**

SÜLEYMAN GÖLBOL 1801042656

# REQUIREMENTS

***NONFUNCTIONAL REQUIREMENTS***

1. Portability → The application should be portable. All computers that have Linux Distro with POSIX and GCC compiler can run the program.

2. Maintainability → In case of an error occurrence, the system uses perror in order to give feedback on terminal.

3. Performance → The system should initially be able to process as many entries as possible. Each request must be processed with different terminals. The system’s performance should be fast enough to show user the feedback.

***FUNCTIONAL REQUIREMENTS***

In order to compile the program, user have to use “make” command that uses gcc. If make or gcc is not installed user can install it via *“sudo apt-get install build-essential”* command.   
Make command runs 1 command.

**gcc -Wall \*.c -lm -o hw5**  
To run, we need to use command line arguments with parameters i, j, o,n and m.

**./hw5 -i files/file1.txt -j files/file2.txt -o output.csv -n 4 -m 2**

If input file exists and we have permissions to read the input file and write to the output file, the executable will run successfully.

To run, also size of each input file should be less than (2n)2.  Because row and column size of matrix will be 2n. Also (2n % 4) has to be divisible with m . Because there are m threads, and each thread is responsible for 2n / m columns of matrix.

So also, m should be less than 2n. When these rules are obtained, it should be fine.

1. **PROBLEM SOLUTION APPROACH AND REQUIREMENT ACHIEVEMENT**

Firstly, the first big problem for to get synchronization between part1 and part2. So, I implemented a barrier. To implement it I used both mutex and conditional variable which both of them are available on pthread library.

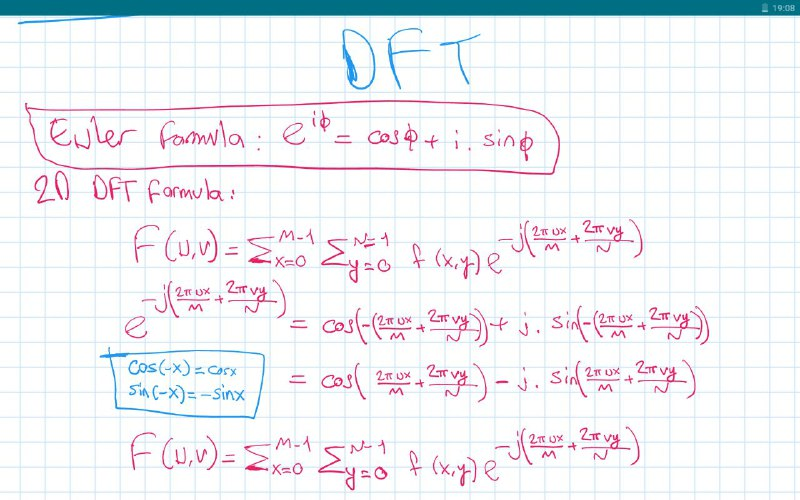
Then other problem of mine was to send more than 1 variable to thread. So I’ve created a struct which contains thread id index, size, number of columns to calculate.

Another problem that I have was freeing allocated memory. But I got double free error and I fixed it.

The most difficult problem that I’ve had was to calculate discrete fourier transform because the resources on internet are not helpful enough (for a beginner). Until last week, I didn’t know about that topic.

Finally I solved it using complex numbers and so on.

This is the formula that I’ve used.



Last problem I encountered was printing timestamp with formatted output with same print function. For this I created a function called tprintf and to merge timestampt with formatted string I used snprintf. Then with variadic function I used variadic list and vprintf to print formatted text.

**3 ) TEST CASES AND RESULTS**

# 3.a) ./hw5 -i files/file1.txt -j files/file2.txt -o output -n 4 -m 5

# Because of m is not divisible by 2n, it gives error.

# 

# 3.b) ./hw5 -i files/file1.txt -j files/file2.txt -o output -n 4 -m 17

# Because of m is bigger than 2n, it gives error.

# 

# Then it goes like this when reading and consuming.

# 3.c) ./hw5 -i files/file1.txt -j files/dfdffhfj.txt -o output -n 4 -m 17

# 

# If user tries to enter a non-existing file, it gives error.

# 3.d) ./hw5 -i files/file1.txt -j files/file2.txt -o output -n 8 -m 2

# 

# If user tries to put n as big number and if file size is less than (2n)2 , it gives error.

# 3.e) ./hw5 -i files/file1.txt -j files/file2.txt -o output -n 4 -m 2

# Text Description automatically generated with medium confidence

# *RESULTS WITH DIFFERENT INPUTS*

# Number of cores in my pc is 4 cores.

# A picture containing diagram Description automatically generated

# *SAME SMALL FILES DIFFERENT NUMBER OF THREADS*

# Text Description automatically generated with medium confidence

# Text Description automatically generated with medium confidence

# I made thread size 2 times bigger, and I made n = 3 instead of 4. Now it’s much faster.

# In the first one, because of thread size is 2, each thread was calculating 2^4/2 = 8 columns.

# In the second one, because of thread size is 4 and n is 3, each thread is calculating 2^3/4 = 2 columns.

# *ANOTHER EXAMPLE*

# Text Description automatically generated

# Text Description automatically generated with medium confidence

# This time number of threads are same. But n is different so In the first one it calculates 2^3/2 = 4 columns per each thread. But in the second it’s 2^4/2 = 8 columns per each thread. So it’s much slower in the second.

# *VALGRIND MEMORY RESULTS*

# The output from valgrind about heap and leaks is like below:

# Text Description automatically generated

# *CHECKING FOR ZOMBIE PROCESSES ETC.*