Name: Mahmoud Mohamed Hussein Moustafa
ID: 67

Matrix Multiplication Using Threads

Code Organization

The code is divided into 4 files. The following points describe briefly what each file is responsible for.
1- main.c : The file that contains the main method, starts the execution of the multiplication methods, calls the reading and writing methods.
2- multiply.c : Contains the 2 methods of multiplying using threads.
3- model.c : Contains the structure of the matrix which is

- Number of rows
- Number of columns
- Data matrix

4- file_manager.c : It is responsible for handling the read and write operations.

Main Functions

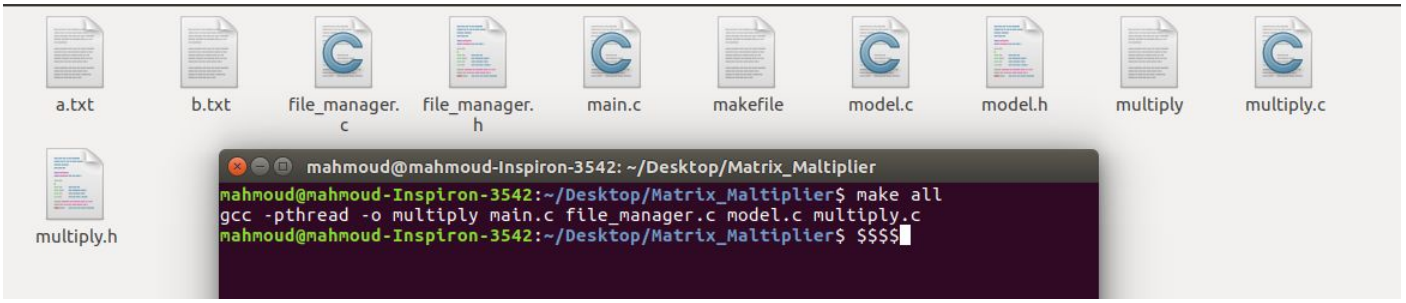
- ***main*** → It starts the execution of the program and calls the solving methods to get the answer matrix then it calls the write_matrix method to write the answer to a file.
  It also calculates the execution time.
- ***multiply_method_1*** → It is the function that contains the logic of the first method of multiplying. It uses the function "solve_row" which is called when a new thread is created to get the answer for a specific row.
- ***multiply_method_2*** → It is the function that contains the logic of the second method of multiplying. It uses the function "solve_cell" which is called when a new thread is created to get the answer for a specific cell.
- ***read_matrix*** → It reads a matrix from a file.
- ***write_matrix*** → It writes a matrix to a file.
- ***write_str*** → It writes a char* to a file.

Compile

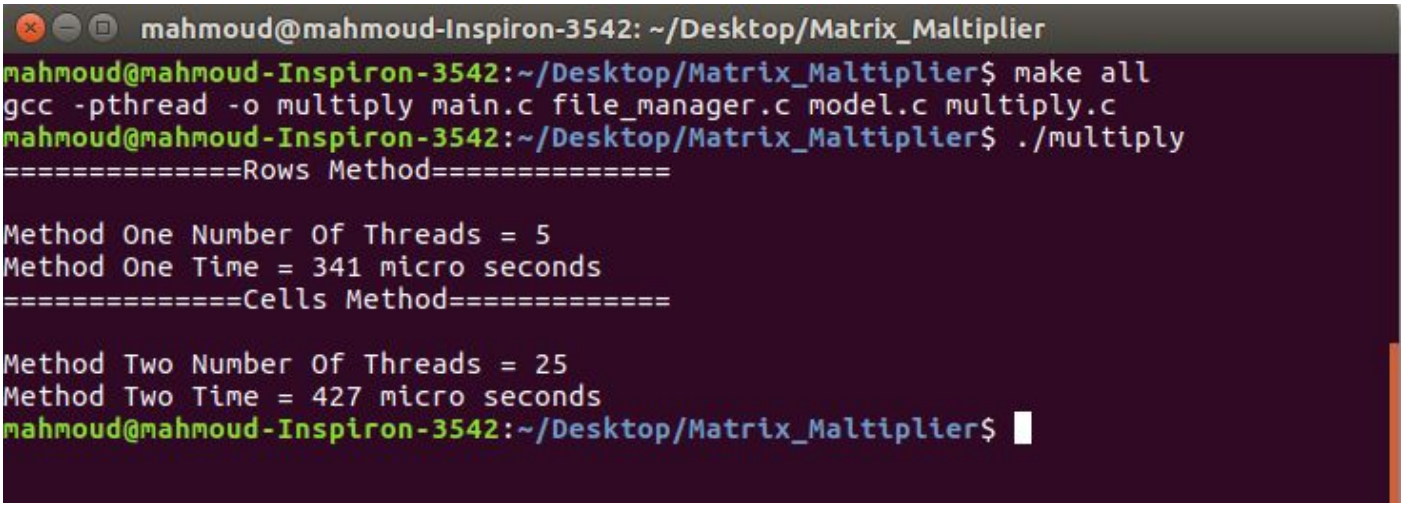
- Open the terminal in the project path and enter “make all”
- An executable file with the name multiply will be generated
- In the same path enter “./multiply” to use the default input files ‘a.txt’ & ‘b.txt’ and the default output file ‘c.out’.
- In the same path enter “./multiply <input_matrix_file_A> <input_matrix_file_B> <output_matrix_file_C> to use custom input and output files.
- To undo the changes in the same path enter “make clean”

Sample runs

- For Compiling



- For running with default settings



- For running with custom input

```
mahmoud@mahmoud-Inspiron-3542:~/Desktop/Matrix_Maltiplier$ ./multiply $HOME/Desk
top/a.txt $HOME/Desktop/b.txt $HOME/Desktop/c.out
==============Rows Method==============

Method One Number Of Threads = 5
Method One Time = 572 micro seconds
==============Cells Method=============

Method Two Number Of Threads = 25
Method Two Time = 763 micro seconds
mahmoud@mahmoud-Inspiron-3542:~/Desktop/Matrix_Maltiplier$
```

- Output file

```
==============Rows Method==============

215     230     245     260     275
490     530     570     610     650
765     830     895     960     1025
1040    1130    1220    1310    1400
1315    1430    1545    1660    1775
==============Cells Method=============

215     230     245     260     275
490     530     570     610     650
765     830     895     960     1025
1040    1130    1220    1310    1400
1315    1430    1545    1660    1775
```

Comparison

After running the two algorithms multiple time, The second algorithm seems to be faster because it uses more number of threads.