

Threads

lab2

Ahmed Nawar -4092| Operating Systems | 04/11/2018

# Overview

Threads lab is divided into two sub-programs:

* Merge Sort using threads
* Matrix Multiplication using threads

# Features

* This lab is to experiment developing programs using threads and comparing the execution time.

# Run

* Matrix multiplication: Run the cbp file, which will take input 2 matrices and their dimensions from input.txt then output the result in output.txt along with the cpu running time for two cases 1- The computation of each element of the output matrix happens in a thread. 2- The computation of each row of the output matrix happens in a thread.

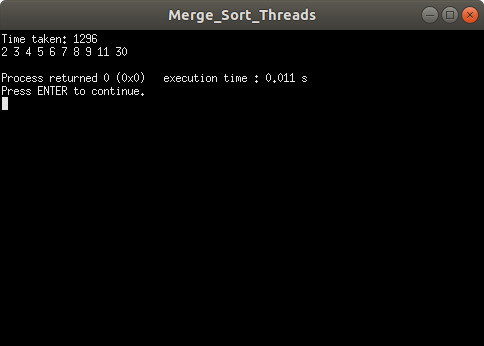
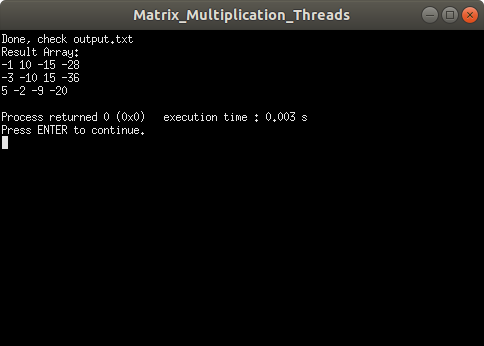
# Flow of Program for Matrix Multiplication Using Threads

1. Read two input matrices and their dimensions from input.txt file
2. Compute the output when each element is computed in a thread(Case 1)
3. Compute the output when each row is computed in a thread(Case 2)
4. The computation is done in 3 loops (O(n^3))
5. In case 1 we create a thread in the second loop using pthread\_create() and let the thread call a function containing the 3rd loop
6. I make a pool of threads according to the number of elements of the output array
7. I make an array of arguments because each thread has its own set of variables and we don’t want them mixed up, therefore a struct of arguments was needed to be passed as method argument containing the indices of the loop to continue as normal.
8. In Case 2, we create the thread at the first loop
9. I also have a pool of threads according to the number of rows of the output
10. The called method of the thread should not change any variables because we cant assume the order of execution of threads.
11. At the end we wait for al threads to finish using pthread\_join()
12. Then I write the output to output.txt and compute the time taken for each algorithm to run and compare them.

# Flow of Program for Merge Sort Using Threads

1. Read array from file and its’s length
2. Recursively we divide our array into 2 until we have N sorted arrays of length 1
3. Then we Merge them into 1 array
4. With each division we create two threads, each one will take care of half the array
5. We wait for them to join the main thread and merge them together in O(n)
6. Compute the time of the execution in nano seconds

Sample Runs



# Support

* If you encounter any problem or a bug, please let me know at [ahmednawar2007@gmail.com](mailto:ahmednawar2007@gmail.com)