CS F372 Operating Systems
First Semester 2022-23
Assignment 2
Total Marks: 60
Deadline: 29th November 23:59 hours

- All programs are to be written in C
- You may use the same groups as in Assignment 1 or change your groups. If you are changing your groups, the individual late days for assignment 1 will carry over for you. That is, if you are moving from a group which has used 2 late days, to a group which has not used any late days, 2 late days will still apply to you individually. Groups should comprise 5-6 students and may be formed across the L1 and L2 sections.
- All members of the group must contribute to the assignment. A demo-cum-viva session will be scheduled during which it will be ascertained that all members have contributed. If any member does not contribute, the member will be awarded 0 in the assignment.
- All submissions will be passed through a code and text similarity checker. If the codes of two or more groups match then all the group members will be summarily awarded 0 in the assignment. This is irrespective of if only one member of a group is the offender. Secure the code that your group is writing. If another group is unethically able to access your code then you have not secured it enough and you will be penalised. **There is no partial penalty for dishonesty. Lifting code from the Internet also constitutes cheating.**
- Honest but incorrect submissions will be awarded partial credit through the demo-cum-viva session
- Discussion is encouraged between groups, copying is strictly prohibited. What is the difference? Check the last part of this page: https://www.cse.iitd.ac.in/~mausam/courses/col772/spring2019/. Use Piazza for discussions.
- Mention names of all people you have discussed with and Internet sources you have referenced in your submission

Problem Statements:

a) This assignment requires you to multiply arbitrarily large matrices using parallelism provided by the Linux process and threads libraries

b) Write a C program P1 which takes three filenames (in1.txt, in2.txt and out.txt) as command line arguments. The first two files (in1.txt and in2.txt) contain two matrices of arbitrary size but satisfying the criteria for matrix multiplication. The sizes will be passed in the command line. P1 spawns n threads which each read row(s) and column(s) each from in1.txt and in2.txt.

   i) Different threads should read different parts of the file. Vary the number of threads from 1… to arbitrarily large

   ii) Record the time that it takes to read the entire file into memory with different number of threads (1, 2, … n). The timing should be at the granularity of nanoseconds.

iii) Plot time against the number of threads for different input sizes. Analyse the findings from the plots.

c) Write a C program (P2) which uses IPC mechanisms to receive the rows and columns read by P1. P2 spawns multiple threads to compute the cells in the product matrix. The program stores the product matrix in the file out.txt
   i) Vary the number of threads from 1… to arbitrarily large.
   ii) Record the time it takes to compute the product with different number of threads. The timing should be at the granularity of nanoseconds.
   iii) Plot the time against the number of threads for different input sizes. Analyse the findings from the plots.

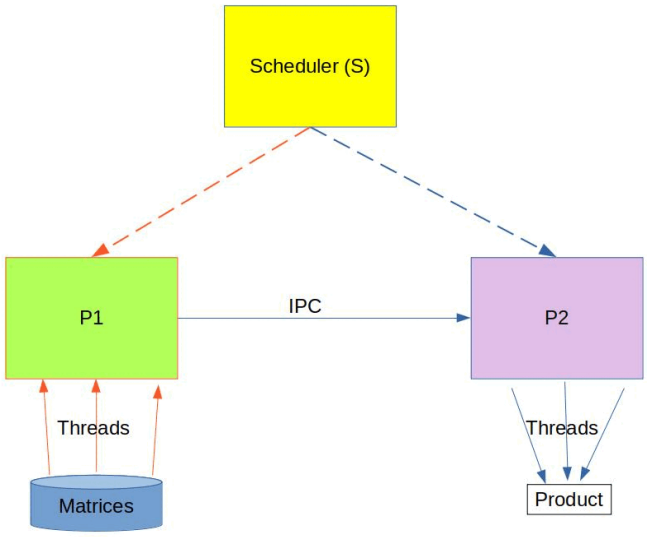
d) Write a scheduler program S. S spawns 2 children processes which exec to become the processes P1 and P2 in part (b) and part (c) above. S uses some mechanism to simulate a uniprocessor scheduler. That is, it suspends P1 and lets P2 execute, and vice versa. Simulate the following scheduling algorithms in S:
   i) Round Robin with time quantum 2 ms
   ii) Round Robin with time quantum 1 ms
   ~~iii) Completely Fair Scheduler where the vruntime is updated every 1 ms~~
   iv) Plot the total turnaround time vs workload size and waiting time vs workload size for the different scheduling algorithms. How do the two algorithms compare for the same workload size? Analyse your findings.
   v) What is the switching overhead in the different cases?

e) You might need to take care of race conditions which might arise at different parts of the assignment

f) Using high level maths libraries is not allowed. It will be ensured in the test cases that each element of the product matrix fits into a long long int

g) Sample input, output and a python script to generate and multiply matrices is here.

h) How to run: `./groupX_assignment2.out i j k in1.txt in2.txt out.txt`
   i) Where `in1.txt` contains a `ixj` matrix, `in2.txt` contains a `jxk` matrix and the `ixk` product matrix is stored in `out.txt`



What to submit:
A single zip file for each group containing:

1. A text file with the details of the group members, including the names and BITS email addresses
2. C program(s) for the three parts above
3. CSV files with the timing data
4. A PDF file containing the analysis as required above. Credit will be given for plots and analysis which clearly elucidate what is happening. You are free to innovate on the plots and the analysis to earn credit. You should use gnuplot or matplotlib tool to generate the plots automatically within the programs. Credit will be given for optimising the code and explanation for each design decision taken
5. Submit on CMS. It is sufficient for one group member to submit.

Evaluation:
Evaluation will be done through a demo-cum viva session where your code will be checked for correctness using different test cases. It will be checked that you have satisfied the different requirements of the assignment. Your report will be checked and questions will be asked regarding the code and analysis. General questions related to concepts in the assignment might also be asked. All group members must know all aspects of the assignment. Marks will be awarded for good optimisations in using threads and working around race conditions.

File Naming Convention: `groupX_assignment2.c` [Where X is the group no.]

Late Day rules: Each student gets 3 free late days in the semester. The 3 late days can be used by a student across the two assignments in any manner that she/he thinks fit (e.g. 1 day for assignment 1 and 2 for assignment 2). Beyond the 3 free days a 10% penalty per day will

apply (partial days will be rounded up). The late days of an individual student will carry forward even if she/he changes the group for the next assignment.