Assignment 3 - Part 1: All about synchronization of threads

Fall 2024

Due: Wednesday, Oct 9, 11:59 pm

Coding Task 1: Choose a synchronization technique IF NECESSARY (mutex, condition variables, or barrier) to correct the problem (if any) of the multithreaded array sum program from Assignment 2 part 2. Use your previous submission as a starter code.

Hint: The partialSums array is global variable and the threads are overlapping when modifying the array which leads to inconsistent and incorrect results.

Note: Make sure to modify the chunk size calculation to handle cases where there is uneven division of work among the threads (ex. when NUM_THREADS = 7)

Coding Task 2: Choose a synchronization technique IF NECESSARY (mutex, condition variables, or barrier) to correct the problem (if any) of the multithreaded matrix multiplication program from Assignment 2 part 2. Use your previous submission as a starter code.

Hint: The matrices are global variables, all the threads see the same resultant matrix C.

Note: Make sure to modify the chunk size calculation to handle cases where there is uneven division of work among the threads (ex. when NUM_THREADS = 7)

Task 3: Explain the reasoning behind your chosen synchronization technique (if you used one) and how it fixed the problem of shared memory. Also explain your reasoning if you decided not to use a synchronization technique.

Deliverables:

- A zipped folder named A3_part1 submitted to D2L dropbox that contains:
    - Your source code files for both coding tasks 1 and 2.

    - A pdf file for Task 3 and a screenshot showing your codes fixed the issue and work as expected.
    - A README.txt file
- Available on Github inside the folder where you added me as a collaborator.

Total Points (100)

- Code runs and works as expected – Task 1: 40 points
- Code runs and works as expected – Task 2: 40 points
- Clear and detailed explanation – Task 3: 10 points
- Proper commenting and screenshot of code output: 5 points
- Available on GitHub: 5 points