YEDITEPE UNIVERSITY

CSE331
OPERATING SYSTEMS DESIGN
FALL 2024

ASSIGNMENT II

Last Submission Date:
19 December 2024, 23:59

POSIX SEMAPHORES AND SHARED SEMAPHORES

In this assignment, you will compare two programming concepts: a multi-process program with shared semaphores and a multi-threaded program with POSIX semaphores. So you will write a multi-process and a multi-threaded program. For each program, you will write a code that does unsharp filtering on the images. The images will be given as raw image matrices in a ".dat" input file and with different sizes to you. You can use the given Python code to visualize the ".dat" images.

IMPLEMENTATION

a) Multi processes program with Shared Semaphores (30 pts):

Before the filtering operation, the parent process will do the following:

1) The parent process creates the Matrix of the Laplacian Filter as the following 3x3 matrix: ***L***=(0,-1,0;1,4,1;0,-1,0).

2) The parent process will read the first line of the file and get the size ***N*** of the image.

3) The parent process will create 2 two-dimensional ***NxN*** arrays in a shared memory area. These matrices will be named as ***A*** and ***B***.

4) The parent process will read the remaining lines of the file and fill ***A***. The value of the element is between 0 and 255.

For the filtering, the parent process will create 2 groups (***G1, G2***) of ***N*** child processes which run concurrently, **to be created at the same time** (i.e, the creation of processes should not be dependent on the completion of the operation of other created processes) and do the following operations below. The parent process will start to wait for the completion of the operation of all child processes.

i) A child process in the ***G1*** will do the convolution operation using the ***L*** matrix to a row in ***A*** associated with it. The first created child in ***G1*** does its convolution operation to the first row, the second child in ***G1*** does its convolution operation to the second row, and so on. The child processes in ***G1*** will write the results to ***B***.

ii) A child process in the ***G2*** will do ***B=A-B*** matrix subtraction. A child process in ***G2*** will do this calculation to a row associated with it. The first created child in ***G2*** does this to the first row, the second child in ***G2*** does this to the second row, and so on. To obtain the right result of the filtering, the ***i***th child process of ***G2*** should wait to complete the calculation performed by the ***i***th child processes of ***G1***. So, you should synchronize the calculation operation using POSIX semaphore primitives and you should design the right synchronization mechanism for overall operation.

After the termination of all child processes, the parent process will print the content of ***B*** to the screen.

b) Multi-threaded program with Posix Semaphores (30 pts):

Before the filtering operation, the main thread will do the following:

1) The main thread creates the Matrix of the Laplacian Filter as the following 3x3 matrix: ***L***=(0,-1,0;1,4,1;0,-1,0).

2) The main thread will read the first line of the file and get the size ***N*** of the image.

3) The main thread will create 2 two-dimensional ***NxN*** arrays in a shared memory area. This matrices will be named as ***A*** and ***B***.

4) The main thread will read the remaining lines of the file and fill ***A***. The value of the element is between 0 and 255.

For the filtering, the main thread will create 2 groups (***G1, G2***) of ***N*** threads which run concurrently, **to be created at the same time** (i.e, the creation of threads should not be dependent on the completion of the operation of other created threads) and do the following operations below. The main thread will start to wait for the completion of the operation of all other threads.

i) A thread in the ***G1*** will do the convolution operation using the ***L*** matrix to a row in ***A*** associated with it. The first created thread in ***G1*** does its convolution operation to the first row, the second thread in ***G1*** does its convolution operation to the second row, and so on. The threads in ***G1*** will write the results to ***B***.

ii) A thread in the ***G2*** will do ***B=A-B*** matrix subtraction. A thread in ***G2*** will do this calculation to a row associated with it. The first created thread in ***G2*** does this to the first row, the second thread in ***G2*** does this to the second row, and so on. To obtain the right result of the filtering, the ***i***th thread of ***G2*** should wait to complete the calculation performed by the ***i***th thread of ***G1***. So, you should synchronize the calculation operation using POSIX semaphore primitives and you should design the right synchronization mechanism for overall operation.

After the termination of all threads, the main thread will print the content of ***B*** to the screen.

REPORT (40 points)

You are required to write a report that expresses the design, implementation, and execution flow of your programs including your understanding of semaphores and shared semaphores. By obtaining execution times from each program you have written, compare multi-process with shared semaphores and multi-threaded with semaphores concepts. Express the total execution time variations for each program with different input image files. **In addition, compare those programming concepts with another choice of your metric which you consider crucial for the expression of the difference.** Explain your results clearly and support them with graphics in your report. (40 pts)

SUBMISSION RULES

- **Writing clean, readable code and using comments are recommended.**
- **Include the student name and ID at the top of the code, as comments are required.**
- **Filenames of the C files must be hw2a.c and hw2b.c.**
- **For report files, only PDF format is accepted. The filename of the report file must be Report2.pdf**
- **This homework will be sent to YULEARN.**
- **The submission must be zipped only as a zip file, no other zip formats will be evaluated.**
- **The name of the zip file should consist of student ID, assignment type and number, excluded from space and special characters.**
- **Your C files will be checked for plagiarism using the MOSS system.**
- **Do not send your files via email. The files sent by email will not be taken into account!**
- **Any copied work (including ChatGPT etc. generated text or code!!!) will be evaluated as 0 points and the assignment will not be accepted!!!**
- You are welcome to ask questions and come up with new ideas about the homework, but reading the instructions explained here carefully, and studying from the course book are highly recommended to have a general understanding before asking questions. More creative ideas can get higher points.

The due date for this assignment is strictly 19 December at 23:59 and submit your own work to stay away from consequences. Good luck...