Operating Systems Lab (CS 470):

Lab 3: Given a sequence of numbers $x_{0,}$ x_1, …$x_n{}^2 \in \{0,1\}$ in a binary file, where x_i represents square matrix element in a linear fashion (concatenate the lines of the matrix for example), write in C/C++ a simulation process as follows: a) start M threads, b) each thread generates a column and a row randomly, c) considering the element location in the matrix given by the generated column and row change the value of the matrix element to 0 if its 4/8 neighbors are predominantly 0, and 1 otherwise.

Overview

Reading and modifying (increment, decrement, assign) a shared variable between different threads needs extra attention due to inconsistency which might occur if the processes run in parallel. In order to avoid this, different mechanism are implemented to solve the critical section problem.

Instructions

The numbers in the binary file should be generated randomly. After each update the values of the matrix should be written in the binary file. No storage in the memory is allowed for the matrix. The simulations process ends when all elements in the matrix turn 0 or 1. To lock the file consider the $\text{fcntl}()$ function. To protect the critical section consider $\text{pthread\_mutex\_lock}()$ and $\text{pthread\_mutex\_unlock}()$ functions, respectively. The stopping criteria should run in a separate process.

Notes

- The number elements (n) and the number of threads (M) should be read from the consol.
- Each step in the simulation process should be traceable. Printing should be provided.

Rubric

Task	**Points**
Error handling	1
Simulation running in M threads	8
Print the different steps	1