Operating Systems - II: CS3523

Lab Exam

Report

Sagar Jain - CS17BTECH11034

May 4, 2019

Contents

Program Design

I have extended the mutual exclusion problem to the class mutual exclusion problem in the following way:

1. I have used additional semaphore locks to make sure that only one class of threads can be in the critical section at any one point of time.

2. I have used a special bool array to make sure that not more than one threads belonging to the same class are waiting for the class semaphore. The other threads belonging to the same class wait for the class to change to their respetive class while the thread waiting on the class semaohore will set the class to its class once it is into the critical section.

3. I keep track of the number of threads of the current class that are in the critical sectin, when this number hits 0, I signal to the class semaphore.

Conversion to Normal Mutual Exclusion

The following changes were made to convert the algorithm.

1. Change number of classes to number of threads.

2. Change bool array length to number of threads.

3. This essentially maps every thread to a class and thus every class(thread) is run mutually exclusively.

Program Output

The program outputs a file. The log file contains discrete events of the threads entering and exiting the critical section

Example Output:

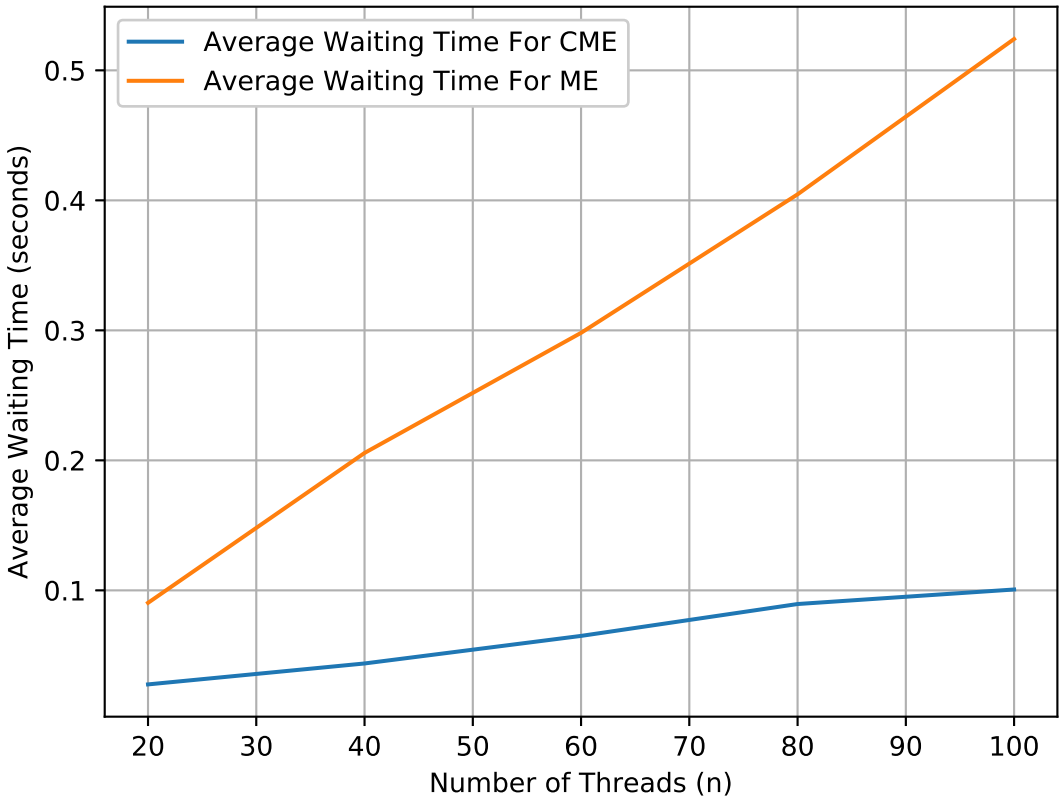
Log File:

0 th CS entry at 17:47:41 by thread 4 for session 4.
0 th CS exit at 17:47:41 by thread 4 for session 4.
1 th CS request at 17:47:42 by thread 4 for session 4.
1 th CS entry at 17:47:42 by thread 4 for session 4.
0 th CS entry at 17:47:41 by thread 0 for session 4.

0 th CS request at 17:47:41 by thread 9 for session 3.
0 th CS request at 17:47:41 by thread 8 for session 5.
1 th CS exit at 17:47:42 by thread 4 for session 4.
2 th CS request at 17:47:42 by thread 4 for session 2.
0 th CS request at 17:47:42 by thread 15 for session 1.
0 th CS exit at 17:47:42 by thread 0 for session 4.
0 th CS entry at 17:47:42 by thread 2 for session 2.

Results & Graphs

Average Waiting Times For Class Mutual Exclusion and Simple Mutual Exclusion with Increasing Number of Threads



Explaination of Results

1. The time taken by the simple mutual exclusion is constanty more than the time taken by the Class Mutual Exclusion.

2. This is explained by the fact that at any moment of time the number of threads executing in the class mutual exclusion is ≥ 1 while in simple mutual exclusion the number of threads is $= 1$.

3. Clearly the throughput for CME is higher than Simple Mutual Exclusion.

4. The rate of increase of Average time is also higher for the Simple Mutual Exclusion since along with number of threads increasing the number of threads fighting for every semaphore is also increasing.