CS 6378 Advanced Operating Systems
Project 2: Due date 1st Nov, 2017

This project must be executed by a team of 2. Your team should demonstrate the project to the TA. Sharing your work with other teams is strictly prohibited.

The goal of the project is to study the performance of the two token based distributed mutual exclusion algorithms by Suzuki-Kasami and Raymond.

Distributed Computation

You need to create a distributed system first as you did in Project 1. (Reuse the code from Project 1.) After initialization the steps executed by each process are as follows.

1. Wait for a period of time uniformly distributed over [t1, t2] msecs. Ensure that you seed each process with different values for the random function to be truly random.
2. Request to enter the critical section using the appropriate algorithm. Print that the process has requested to enter the CS with the process id and time.
3. Enter the critical section. Print that the process has entered the CS with the process id and time. Wait for t3 msecs and then exit the CS.
4. Repeat the steps 1 to 3 several times randomly chosen over the interval [20, 40]. Again seed appropriately.
5. Send a *completed* message to the coordinator along with the data that were collected.
6. If you are a coordinator wait for *completed* messages from all the processes. Then send a *terminate* message to all the processes. Evaluate the performance of the algorithm and print the appropriate output, and then terminate.
7. All processes terminate on receipt of *terminate* message

This part needs to executed for both the algorithms separately. The values for t1, t2, and t3 must be specified in the config file.

Performance Study

You need to evaluate the performance of both algorithms for values of n = 5, 10, 15, 20, and 25 where n is the number of processes. The parameters that you need evaluate are average number messages used to enter CS, average synchronization delay, and the average waiting time to enter CS.

Write a short report on the performance study. Your report should contain graph depicting the performance of the above metrics for various values of n. The report should also contain a short description of how the performance data are collected and measured.

Grading Policy

Implementation: 30%. Source code should be structured well with adequate comments clearly showing the different parts and functionalities implemented.

Correctness: 30%. CS should not be violated.

Performance study: 40%

The source code, config files, and the performance report should be submitted through elearning. The TA will compile the code when you demonstrate. Add a note at the top of the source code in case you are using any special flags for compilers/linkers.