COSC 6352, Advanced Operating Systems

Spring 2019, Programing Assignment #4

Date Assigned: Wednesday, April 17, 2019

Due Date: Monday, April 29, 2019 at 11:59 p.m. (use blackboard)

DESCRIPTION

Write a C or C++ program named suzuki.c or suzuki.cpp (executable name should be suzuki) to implement Suzuki and Kasami's Broadcast Algorithm using MPI. The screenshot from jumpshot should be saved as suzuki.pdf.

Each process records its state of being outside the critical section (RELEASED), requesting entry (REQUESTED) or being in the critical section (GRANTED). A process can randomly request entry to critical section after waiting for some time, if it has not already requested one or is not currently in the critical section.

In order to get access to the critical section a process must send (N - 1) requests and wait for (N-1) reply if it is not already holding the token. At the start of the simulation randomly assign the token to any process.

When a process is requesting entry to the critical section, it defers processing requests from other processes until its own requests have been sent to all (N-1) processes.

On every process maintain a data structure which contains the largest sequence number for each thread received (RN vector). The token comprises of a data structure (LN vector) which contains the sequence number of the latest executed request from a thread i and another data structure for maintaining a queue (Q) of requesting threads.

INPUT TO THE PROGRAM

The input to the program is the number of processes, the simulation time and the maximum amount of time a process waits before requesting access to critical section. The number of processes should be specified as an argument to your program.

PROGRAM OUTPUT

```
mpirun –np 4 aosproj3 30 5
```

- The number of processes requested is 4 (maximum 8 processes).

- The simulation time is 30 seconds (minimum 10 seconds), after which no process can request access to critical section.
- The maximum wait time for a process is 5 seconds (Therefore a process can randomly wait between 0 to 5 seconds before requesting access to critical section again).

```
Process with rank 1 and sequence number 1 is requesting critical
section
Broadcast message (1:1)
Rank 2 received critical section request from rank 1
Rank 3 received critical section request from rank 1
Rank 0 received critical section request from rank 1
Rank 3 is sending the token to rank 1
Process with rank 2 and sequence number 1 is requesting critical
section
Broadcast message (2:1)
Rank 1 has received the token from Rank 3 and entering into
critical section
Rank 3 received critical section request from rank 2
Rank 0 received critical section request from rank 2
Rank 1 has exited critical section
Rank 1 received critical section request from rank 2
Rank 1 is sending the token to rank 2
Rank 2 has received the token from Rank 1 and entering into
critical section
Rank 2 has exited critical section
.
.
.
.

.so on until the simulation time ends
```