CS 6378-001: Project I

Instructor: Ravi Prakash

Assigned on: February 9, 2017
Due date: February 23, 2017

You are required to implement the Ricart-Agrawala algorithm for distributed mutual exclusion, with the optimization proposed by Roucairol and Carvalho.

This is an individual project and you are expected to demonstrate its operation to the instructor and/or the TA.

The program must run on UTD machines (dc01, dc02, ..., dc45).

1 Requirements

1. There are ten nodes in the system, numbered from zero to nine.
2. There are reliable socket connections (TCP) between each pair of nodes. The messages pertaining to the mutual exclusion algorithm are sent over these connections.
3. The execution of your code is split over two phases as described below.
4. Phase 1: Each node goes through the following sequence of operations until each node has successfully exited the critical section 20 times:
   (a) Waits for a period of time that is uniformly distributed in the range [[illegible]] time units before trying to enter the critical section.
   (b) On entry into the critical section the node do the following:
      i. Print “entering” followed by node number and current physical time on a new line,
      ii. Let 3 units of time elapse, and then exit the critical section.
5. Phase 2: Following Phase 1, the even numbered nodes continue to issue requests for entry into the critical section at the same rate as before. The odd numbered nodes, after exiting the critical section, wait for a period of time that is uniformly distributed in the range [45, 50] time units before trying to enter the critical section. As in Phase 1, the critical section execution lasts 3 time units and on entering the critical section the node prints “entering,” node number and current physical time on a new line.
6. Once a node has successfully existed the critical section 40 times (20 each in Phases 1 and 2), it does not make any more attempt to enter the critical section, and sends a *completion notification* to node zero.
7. Node zero brings the entire distributed computation to an end once its has received *completion notification* from all the nodes.

2 Data Collection

For your implementation of the mutual exclusion algorithm report the following (either show it on the screen or write it to a file):

1. The total number of messages sent by each node from the beginning until it sends the *completion notification*.

2. The total number of messages received by each node from the beginning until it sends the *completion notification*.

3. For each node, report the following for each of its attempts to enter the critical section:

    (a) The number of messages exchanged.

    (b) The elapsed time between making a request and being able to enter the critical section (latency).

4. If, during the execution of your code, there is any noteworthy change in the number of messages exchanged per critical section execution or latency at any stage provide an explanation for the change.

3 Point Distribution

Implementation (50%): Source code of your well structured and well documented program. You may write your code in C or C++ or Java.

Correctness (50%): Output that your program produces and the performance data and its analysis.

4 Submission Information

Submission should be done through eLearning.