CS 499 – Distributed System  
Wednesday, October 21, 2009

TODO:

* Look over the answers for the quiz below, they me show up again on the next exam.

Class Notes:

* Quiz 5
  + What is two-phase locking”?
  + See 1: what additional requirement does “strict two-phase locking” add?
  + What are the protocol rules in optimistic concurrency control that (automatically) guarantees strict execution? Remember that you need to look at reads-reads and writes-writes of past transactions versus the current transaction.
  + // this may show up again it the exam.

Lecture Notes:

* Mutual Exclusion (ME) :
  + // “liveness” will be used, not proper still but still could be used
  + Conditions: (continued from last lecture)
    - ME 3 (fairness): processes are completed in the order they arrive
  + Algorithm:
    - Multicast & Logical Clocks (creators: Ricard & Agravola)
      * + *On initialization*

*state* := RELEASED;

* + - * + *To enter the section*

*state* := WANTED;

Multicast *request* to all processes;

T := request’s timestamp;

*Wait until* (number of replies received = (N -1));

*state* := HELD:

* + - * + *On receipt of a request <Ti, pi> at pj( i ≠ j )*

*if* (*state* = HELD or (*state* = WANTED *and* (*T, pj*) < (*T, pi*)))

*then*

queue *request* from *pi* without replying;

*else*

reply immediately to *pi*;

end if

* + - * + *To exit the critical section*

*state* := RELEASD

reply to any queued requests;

* + - * What does this algorithm provide us?
      * Because of the “if” statement we are ensuring fairness because we are comparing timestamps.
        + No two processes should have the same timestamp.
    - Maekawa’s Algorithm:
      * Just mentioned in passing
      * will not be on the next exam
  + Performance of Algorithms:
    - Central Server Algorithm
      * Enter: 2 (round trip)
      * Exit: 1
      * Sync Delay: 2 (round trip)
        + time for process to releasing the Critical Section and for another to pick it back up
    - Ring Based
      * Enter: 0…N
      * Exit: 1
      * Sync Delay: 1…N
        + time for process to releasing the Critical Section and for another to pick it back up