CS 6/73201 # Homework #2 **Advanced OS**

Due in class at 5pm on Monday 31 October 2005
typed answers preferred

1. **How does the meaning of the "reply" message differ between Lamport's Algorithm and Ricart and Agrawala's Algorithm for mutual exclusion in a distributed environment?**

   In Lamport's Algorith, the "reply" message is simply an acknowledgement that the process has received the request message — it does not give permission to enter the critical section but neither does it deny permission. In Ricard and Agrawala's Algorthm, the "reply" message gives that process's permission to enter the critical section.

2. **In what ways is Suzuki and Kasami's Broadcast Algorithm for mutual exclusion better and worse than the algorithms mentioned in question 1 above?**

   Suzuki and Kasami is <u>better</u> because: It allows a process to enter the critical section repeatedly if no one else is requesting the token. It requires less messages. Processes not holding the token do not have to send any messages (though they do have to receive messages and update their RN accordingly).

   Suzuki and Kasami is <u>worse</u> because: It doesn't necessarily implement "happened before" unless the queue is sorted using logical clocks. Its messages are larger (includes LN and Q). Detecting the loss of the token is difficult (a new type of failure).

3. **The Chandy, Misra, and Haas Edge-Chasing Algorithm for deadlock detection in a distributed environment does not report false deadlock. Why not?**

   Because the probe follows the path from blocked process waiting on a resource to another blocked process waiting on a resource and so on until it reaches the probe's sender — following a circular hold-and-wait, in an environment that does not allow preemption and demands mutual exclusion, meaning all four deadlock conditions are satisfied. If this truly is a deadlock, the deadlock will not "break" behind the probe as it continues on, meaning this is a real deadlock and not a false deadlock.

4. **One of the methods of deadlock prevention is to allow preemption. What is the major problem with this method?**

   The major problem is that preemption permits starvation, meaning a process may never receive its requested resources and thus may never make progress. Another problem is that preemption only works for resources whose state can be saved and restore, which limits the applicability of the technique in general.

5. **One concurrency control mechanism for deadlock prevention is lock timeouts. What are the problems with this method?**

   It is overly conservative — it breaks vulnerable locks and preempts the resource it wants, forcing the vulnerable transaction to abort even if that transaction it is making progress and is not deadlocked. This also opens the possibility of starvation and cascaded rollbacks.