The University of Melbourne
School of Computing and Information Systems
COMP90020 Distributed Algorithms

Tutorial Week 12: Distributed Transactions.

Notes

Exercises

51. A three-phase commit protocol has the following parts:

- *Phase 1:* Is the same as for two-phase commit.
- *Phase 2:* The coordinator collects the votes and makes a decision. If it is No, it aborts and informs participants that voted Yes; if the decision is Yes, it sends a preCommitrequest to all the participants. Participants that voted Yes wait for a preCommitor doAbortrequest. They acknowledge preCommitrequests and carry out doAbortrequests.
- *Phase 3:* The coordinator collects the acknowledgements. When all are received, it commits and sends doCommitrequests to the participants. Participants wait for a doCommitrequest. When it arrives, they commit.

Explain how this protocol avoids delay to participants during their 'uncertain' period due to the failure of the coordinator or other participants. Assume that communication does not fail.

52. Give an example of the interleavings of two trnasactions that is serially equivalent at each server but is not serially equivalent globally.

53. Extend the definition of two-phase locking to apply to distributed transactions. Explain how this is ensured by distributed transactions using strict two-phase locking locally.

54. In the edge chasing algorithm, every transaction involved in a deadlock cycle can cause deadlock detection to be initiated. The effect of several transactions in a cycle initiating deadlock detection is that detection may happen at several different servers in the cycle, with the result that more than one transaction in the cycle is aborted. Explain a possible solution to this problem.

55. Consider a binary, perfectly balanced tree of processes of height n where all leaf nodes have the same distance from the root; so there is a total number of $m = 2^n - 1$ nodes in the tree. Assume that the root is the coordinator of the commit protocol. Determine the number of messages and forced log writes for the presumed nothing (2PC) protocol in the following situations, assuming that there is a heirarchic communication organization:

(a) all processes have performed updates and the transaction commits.

(b) all processes have performed updates and the transaction aborts (root decides to abort after receiving 2 YES).