Distributed Algorithms

Assignment 10

Group 11

Xugang Zhou	352032
Fangzhou Yang	352040
Yuwen Chen	352038

January 28, 2014

1 Transactions

- a). Transaction is atomic execution of a set of instructions, which ensures consistency even in the face of failures. For example in a banking system, when a customer transfers money from a saving account to a checking account, the transaction must consist of three separate operations: 1.decrement the saving account; 2.Increment the checking account; 3. Record the transaction in the transaction journal. Now if an interrupting error happens between the first and second operation, then the database must roll back the entire transaction so that the balance of all accounts is correct. But if the three operations are separated and independent, then the transferred money will be missing.
- b). ACID Properties:
  - Atomicity: All tasks of a transaction are performed or none if them are. There are no partial transactions.
  - Consistency: The transaction takes the database from one consistent state to another consistent state.
  - Isolation: The effect of a transaction is not visible to other transactions until the transaction is committed.
  - Durability: Changes made by committed transactions are permanent. After a transaction completes, the database ensures through its recovery mechanisms that changes from the transaction are not lost.
- c). The ACID Properties are endangered by interfering concurrent transactions and by faulty environments. For example, if two clients are visiting and modifying the same data in the database system, or an fatal error happens during the execution of a transaction.

2 Properties of Schedules

- a). A serial schedule is one in which no transaction starts until a running transaction has ended. A serializable schedule is equivalent to a serial schedule in its outcome, but the order in which the actions of transactions are executed is not the same, a transaction can start before other running transactions end.
- b). In Two Phase Locking, a thread must hod all locks until it needs no further lock, it ensures serializability but not recoverability, avoiding cascading aborts and also strict schedule. Because a thread can still read, write or commit the uncommitted data, which are already unlocked but no yet committed in other threads.
- c). In Strict Two Phase Locking, all write locks are hold until threads end. Thus, the data that other threads read and write are always committed.

- d). Keeping write locks until the end of a transaction guarantees other threads cannot read and write uncommitted data. However for read locks, it won't affect the consistency when other threads re-read and write data before it is committed.

3 Lock Escalation

The basic idea of Lock Escalation is that, threads starts locking items of fine granularity, if a thread acquires too many locks, the granularity of locks is increased, which results in a conversion from multiple fine-grained locks to fewer coarse-grained ones. To achieve consistency of multiple different locks, when performing the lock escalation, the scope of each lock needs to be identified. Locks being consolidated should not have overlap fields.

4 2PC

In the implementation the test scenario will be set up by a single configuration file. The program execute periodically 3 test cases in turn of:

- successful completion - The Coordinator receives one commit command and after receiving all votes the commit will be performed.
- unsuccessful completion - The Coordinator receives one commit command but the transaction is aborted due to an abort vote from a certain node.
- abort command - The Coordinator receives an abort command and the transaction is aborted.