Locking

CS157B

Chris Pollett

May 2, 2005.

Outline

- Lock Types
- Lock Scheduler Architecture

Lock Types

- Last day we talked about shared and exclusive locks.
- Recall if T_i has performed $xl_i(X)$ then no $xl_j(X)$ or $sl_j(X)$ can occur in the schedule without an intervening $u_i(X)$. Further, if $sl_i(X)$ appears in the schedule, then there can be no $xl_j(X)$ where j and i are different without an intervening $u_i(X)$.
- The two-phase condition for such locks is that no $sl_i(X)$ or $xl_i(X)$ for a transaction T_i can be preceded by an action $u_i(X)$.

Compatibility Matrix

- The information about what is legal to do with shared and exclusive locks can be listed in a matrix:

		Lock Requested	
		S	X
Lock held in mode	S	Y	N
	X	N	N

Upgrading Locks

- If a transaction T already has a shared lock on X, it can request an exclusive lock.
- If no one else has a lock on X, then the request will be granted.
- $u_i(X)$ releases all locks held on X. That is, if one has both a shared and an exclusive lock on X.
- For example, the following schedule is possible: $sl\_{}_1(A)$, $r_1(A)$, $sl_2(A)$, $r_2(A)$, $sl_2(B)$, $r_2(B)$, $sl_1(B)$, $r_1(B)$, $xl_1(B)$--denied, $u_2(A)$, $u_2(B)$, $xl_1(B)$, $w_1(B)$, $u_1(A)$, $u_1(B)$.
- The ability to upgrade lock can cause deadlocks. For example, $sl_1(A)$, $sl_2(A)$, $xl_1(A)$ --denied, $xl_2(A)$ -- denied .

Update Locks

- One way to avoid deadlocks is to introduce a new type of lock - the *update lock*.
- We now no longer let a transaction get an exclusive lock if they have a shared lock. One can upgrade to an update lock, however.
- An update lock can be used to read and can be upgraded to an exclusive lock, but only one transaction at a time is allowed to hold an update lock.

Update Lock Compatibility Matrix

- Here’s what the matrix looks like:

		Lock Requested		
		S	X	U
Lock held in mode	S	Y	N	Y
	X	N	N	N
	U	N	N	N

Increment Locks

- Many transactions operate on the database only by incrementing or decrementing a value.
- For example, the number of seats on an airplane after one ticket purchased goes down by one.
- Another example, is money transfers of a fixed dollar amount (say $40).
- These kinds of operations commute with each other: INC(X,40), INC(X,2) = INC(X,2), INC(X,40).
- Can introduce a new kind of operation: inc(X,a) -- increment X by a -- and a new lock called an increment lock il(X).

Compatibility Matrix for Increment Locks

- Here’s the associate matrix which guarantees serializable database schedules

		Lock Requested		
		S	X	I
Lock held in mode	S	Y	N	N
	X	N	N	N
	I	N	N	Y

Lock Scheduler Architecture

We next discuss the design of a simple lock scheduler. The design principles we will follow are:

1. Transactions themselves do not request locks. It is the job of the scheduler to insert lock actions into the stream of reads, writes, etc.
2. Transactions do not release locks. Rather, the scheduler releases locks when the transaction manager tells it that the transaction will commit or abort.

A Two Part Scheduler

- We will split our scheduler into two parts:
  1. The first part inserts locks into the stream of operations from the transaction. Its main job is to determine the lock type.
  2. The second part then checks the modified stream. If the operation is a DB operation, then it executes it. If the operation was a lock then it checks if the lock can be granted. If yes, the lock table is modified. If no, then a request is noted in the lock table, and the transaction must wait on all further actions until the lock request is granted.
- When a transaction commits or aborts all of its locks are released.
- It is also the job of the second part of the scheduler to figure out which of the waiting transactions should get a given lock.