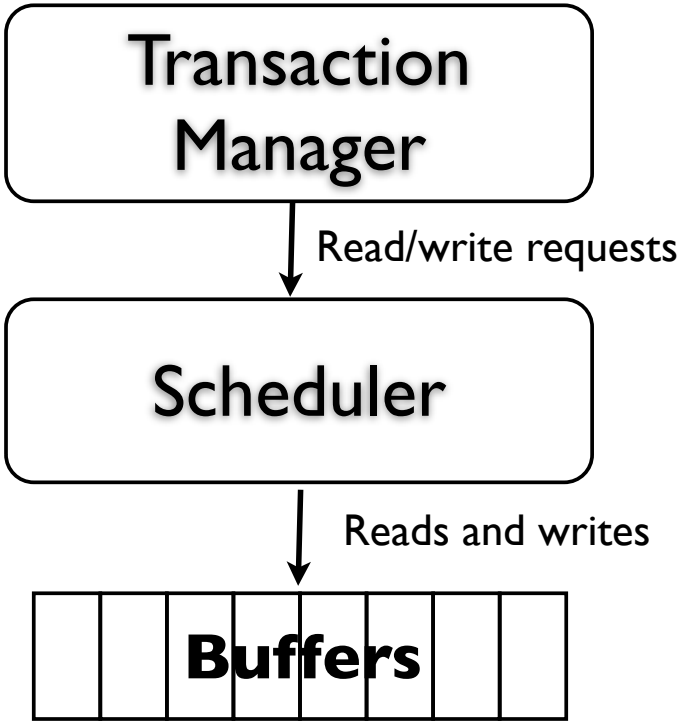
Section 5: Concurrency Control

Thursday, April 30 2009

Concurrency Control



- What is the purpose of the scheduler?

to ensure serializability.

Optimistic vs Pessimistic

- What is the difference?
- When is it preferable to have optimistic concurrency control?
  - Poor when there are many conflicts (rollbacks)
  - Great when there are few conflicts
- When is it preferable to have pessimistic concurrency control?
  - Great when there are many conflicts
  - Poor when there are few conflicts

Pessimistic Concurrency Control: Locks

- Won't cover in section since it was covered in class!

Optimistic Concurrency Control

- Timestamps
- Validation (will not be covered in this class)

Concurrency Control: Timestamps

- Key idea: The timestamp order defines the serialization order.
- Scheduler maintains:
  - **TS(T)** for all transactions T
  - **RT(X)**, **WT(X)**, and **C(X)** for all data elements X

Scheduler receives request from transaction T ...

- grant request
- rollback T
- delay T

Scheduler receives request from transaction T ...

1. If read request $r_T(X)$:

2. If write request $w_T(X)$:

3. Commit request:

4. Abort request:

See textbook - section 18.8

Exercises

1. st1; st2; st3; r1(A); r2(B);r2(C); r3(B); com2; w3(B);w3(C)
2. st1; st2; r1(A), r2(B);w2(A); com2; w1(B)
3. st1; st3; st2; r1(A); r2(B); r3(B);w3(A);w2(B); com3; w1(A)
4. st1; r1(A); w1(A); st2; r2(C); w2(B); r2(A); w1 (B)

Exercise 1:

st1; st2; st3; r1(A); r2(B);r2(C); r3(B); com2; w3(B);w3(C)

TS(T1) = 1
TS(T2) = 2
TS(T3) = 3

T1	T2	T3		A	B	C	Comments
r1(A)				RT=1			
	r2(B)				RT=2		
	r2(C)					RT=2	
		r3(B)			RT=3		
	commit						
		w3(B)			WT=3 c= 0		
		w3(C)				WT=3 c=0	**GRANT**

Exercise 2:

st1; st2; r1(A), r2(B);w2(A); com2; w1(B)

TS(T1) = 1
TS(T2) = 2

T1	T2		A	B	Comments
r1(A)			RT=1		
	r2(B)			RT=2	
	w2(A)		WT=2 C=0		
	commit		C=1		
w1(B)					**ROLLBACK.** TS(T1) < RT(B) so T1 is writing too late!

Exercise 3:

st1; st3; st2; r1(A); r2(B); r3(B);w3(A);w2(B); com3; w1(A)

TS(T1) = 1
TS(T2) = 3
TS(T3) = 2

T1	T2	T3		A	B	Comments
r1(A)				RT=1		
	r2(B)				RT=3	
		r3(B)				RT doesn't change because TS(T3) < RT(B).
		w3(A)		WT=2 C=0		
	w2(B)				WT=3 C=0	
		commit		C=1		
w1(A)						**IGNORE,** because TS(T1) < WT(A) and C (A) = 1. This is the Thomas Write Rule.

Exercise 4:

st1; r1(A); w1(A); st2; r2(C); w2(B); r2(A); w1(B)

TS(T1) = 1
TS(T2) = 2

T1	T2		A	B	C	Comments
r1(A)			RT=1			
w1(A)			WT=1 C=0			
	r2(C)				RT=2	
	w2(B)			WT=2 C=0		
	r2(A)		RT=2			
w1(B)						**DELAY.** TS(T1) < WT(B) but C(B) = 0. So T1 waits until T2 commits or aborts.

Multiversion Timestamps

- Keep multiple version of each data element along with the write timestamp.
- Will reduce number of aborts due to read-too-late problem.

Didn't get this far in section.

Exercises

On whiteboard.