Tutorial 11

CSC343
Fall 2019

Oluwaseun Cardoso (OLUWASEUN.CARDOSO@MAIL.UTORONTO.CA)

Saihiel Bakshi (SAIHIEL.BAKSHI@MAIL.UTORONTO.CA)

UNIVERSITY OF TORONTO MISSISSAUGA

UNIVERSITY OF TORONTO MISSISSAUGA

Transactions

- A sequence of many actions which are considered to be one unit of work.
  - Example:

    T1: R(A) R(B) W(B) W(A) Commit

- R(A): Read database object A
- W(A): Writing (to) an object A

- Commit: Committing transaction
- Abort: Aborting transaction

Schedules

- A list of actions from a set of transactions in a specific order

Example:

- **T1:** R(A) R(B) W(B) W(A) Commit
- **T2:** R(B) W(A) Commit
- **S:** $R_1(A)$ $R_1(B)$ $R_2(B)$ $W_2(A)$ $W_1(B)$ $W_1(A)$ $Commit_1$ $Commit_2$

S	**T1**	R(A)	R(B)			W(B)	W(A)	Commit	
	T2			R(B)	W(A)				Commit

UNIVERSITY OF TORONTO MISSISSAUGA

Conflict Operations

Two operations in a schedule are said to be conflict if they satisfy all three of the following conditions:

1. they belong to different transactions;
2. they access the same item A; and
3. at least one of the operations is a write(A).

Example in Sa: R1(A), R2(A), W1(A), W2(A), A1, C2

- R1(A),W2(A) conflict, so do R2(A),W1(A),
- R1(A), W1(A) do not conflict because they belong to the same transaction,
- R1(A),R2(A) do not conflict because they are both read operations

Write Read Conflict

S	**T1**	R(A)	W(A)				R(B)	W(B)	Abort
	T2			R(A)	W(A)	Commit			
	Action	X = A	A = X + 200	Y = A	A = Y * 1.05				

1. they belong to different transactions? ✔
2. they access the same item? ✔ (A)
3. at least one of the operations is a write? ✔

This is a **W**rite**R**ead conflict. (dirty read)

The problem is we are reading data which is not committed (and later aborted).

Read Write Conflict

S	**T1**	R(A)				W(A)	Commit
	T2		R(A)	W(A)	Commit		
	Action	X = 5	Y = 5	A = Y - 1		A = X -1	

1. they belong to different transactions? ✔
2. they access the same item? ✔ (A)
3. at least one of the operations is a write? ✔

This is a $\mathbf{R}_{ead}\mathbf{W}_{rite}$ conflict.

The problem is that after T1 has read A, T2 runs and modifies A and then T1 continues to

Write Write Conflict

S	**T1**	W(A)				W(B)	Commit
	T2		W(A)	W(B)	Commit		
	Action	A = 1000	A = 2000	B = 2000		B = 1000	

1. they belong to different transactions? ✔
2. they access the same item? ✔ (A)
3. at least one of the operations is a write? ✔

This is a $\mathbf{W}_{\text{rite}}\mathbf{W}_{\text{write}}$ conflict.

T2 runs during T1 running and overwrites the data that T1 wrote into A. The objective of T1 is to set both A and B to 1000, and the objective of T2 is to set both A and B to 2000. At the end of this schedule we have an inconsistent state.

Serializable

- A schedule is **serializable** if the results of executing that schedule is identical to executing the transactions in the schedule in some serial order.

S	**T1**	R(A)	W(A)			R(B)	W(B)		
	T2			R(A)	W(A)			R(B)	W(B)

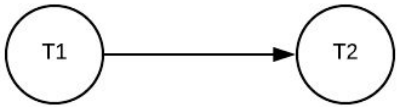
- S is serializable because it is equivalent to running, **T1; T2;**
- **T1**'s read and write of B (shaded in grey) is not affected by **T2** in S (because R(A) W(A) do not affect B).

Conflict Serializable

- A schedule is **conflict serializable** if it is conflict equivalent to some serial schedule.
- (two schedules are conflict equivalent if they involve the same actions of the same transactions, and the order of conflicts is the same)

S	**T1**	R(A)	W(A)			R(B)	W(B)		
	T2			R(A)	W(A)			R(B)	W(B)

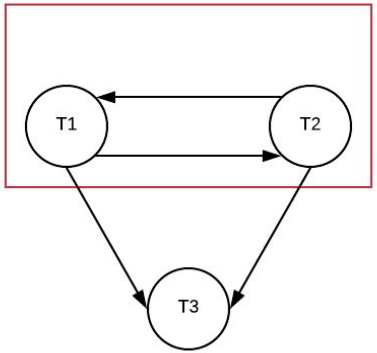
- S **is** conflict serializable because the precedence/serializability graph is acyclic.



Conflict Serializable

S	**T1**	R(A)			W(A)	Commit		
	T2		W(A)	Commit				
	T3						W(A)	Commit

- S is **not** conflict serializable because the precedence/serializability graph is not Acyclic (cycle between T1 and T2).



UNIVERSITY OF TORONTO MISSISSAUGA

View Serializable

- A schedule is **view serializable** if it is view equivalent to some serial schedule.

- For 2 schedules to be view equivalent, they must abide by the following:
  a. If T_i reads the initial value of an object, O in S, it must also read the initial value of O, in S'
  b. If T_i reads a value of O written by T_j in S, it must also read the value of O written by in T_j S'.
  c. For each data object, the transaction, T_x that performs the final write on the object in S, must also perform the final write on the object in S'.

- If a schedule is conflict serializable, it is also view serializable. This is not true, the other way around.

View Serializable

S	**T1**	R(A)			W(A)	Commit		
	T2		W(A)	Commit				
	T3						W(A)	Commit

- Let's propose S' and then test it. S' = T1; T2; T3;

  a. T1 reads the initial value of A in S. In our S', it also reads the initial value of A.
  b. No transaction reads the value of A after it is written by another transaction.
  c. For the object A, T3 performs the final write on the object in S. In our S', T3 also performs the final write.

- Therefore, S' is view serializable, even though it was not conflict serializable.

Recoverable

- In a recoverable schedule, a transaction commits only after all transactions it reads from commit.

T1	R(A)	W(A)			Commit	
T2			R(A)	W(A)		Commit

T1	R(A)	W(A)			Abort	
T2			R(A)	W(A)		Abort

Since T2 reads A from T1, it cannot commit until T1 commits. If T1 aborts, T2 has to abort as well.

Unrecoverable

T1	R(A)	W(A)				Commit
T2			R(A)	W(A)	Commit	

T2 read A from T1 but commited before T1 commited.

T1	R(A)	W(A)			Abort	
T2			R(A)	W(A)		Commit

T2 read A from T1 but T1 never commited.

Avoiding Cascading Aborts (ACA)

- In regular recoverable schedules if T2 reads A from T1, but T1 is later aborted, T2 has to be aborted as well. ACA avoids that.
- A transaction can only read data from committed transactions.

T1	R(A)	W(A)	Commit		
T2				R(A)	W(A)

.

- ACA => Recoverable (Recoverable ≠> ACA). Serial schedules are ACA and recoverable.
- This schedule is recoverable, but non ACA:

T1	R(A)	W(A)			Commit	
T2			R(A)	W(A)		Commit

UNIVERSITY OF TORONTO MISSISSAUGA

Task:

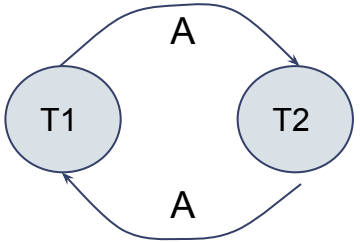
Consider the two schedules below; what can you say about them? (ie: what properties to they maintain?)

- **S1:** $R_1(A)\ R_2(A)\ W_1(A)\ W_2(A)$
- **S2:** $R_1(A)\ R_2(B)\ W_3(A)\ R_2(A)\ R_1(B)$

Task Solution:

- **S:** $R_1(A)\ R_2(A)\ W_1(A)\ W_2(A)$
  - Serial? **No**
  - Serializable? **No**
  - Conflict Serializable? **No**
  - View Serializable? **No**
  - Recoverable? **Yes**
  - ACA? **Yes**

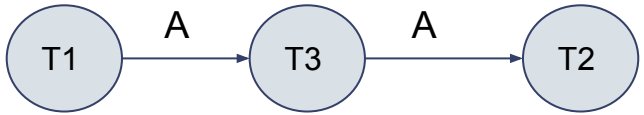
S	**T1**	R(A)		W(A)	
	T2		R(A)		W(A)



Cycle!

- **S:** $R_1(A)\ R_2(B)\ W_3(A)\ R_2(A)\ R_1(B)$
  - Serial? **No**
  - Serializable? **Yes (T1;T3;T2)**
  - Conflict Serializable? **Yes**
  - View Serializable? **Yes**
  - Recoverable? **Not sure**
  - ACA? **No**

S	**T1**	R(A)				R(B)
	T2		R(B)		R(A)	
	T3			W(A)		



No cycle.

UNIVERSITY OF TORONTO MISSISSAUGA

Any Questions?

- Do you have any questions?
  1. Check piazza
  2. Post the question on piazza
     (unless it's a personal question then email one of the TAs)
- Thank you for a great semester! :)