Transaction Processing and Concurrency Control

Objectives

- Transaction Processing Concepts +
- Introduction to Transactions +
- Introduction to Concurrency Control +

- Transaction Processing Concepts ...

- Transaction processing systems are systems with large DBs and hundreds of concurrent users that are executing DB transactions.

- Examples of such systems are:
  - airline reservation systems
  - banking systems
  - KFUPM student information systems, etc.

- These systems require high availability and fast response time for hundreds of concurrent users.

- In this lecture we will introduce the major concepts of transaction processing and the concept of concurrency control

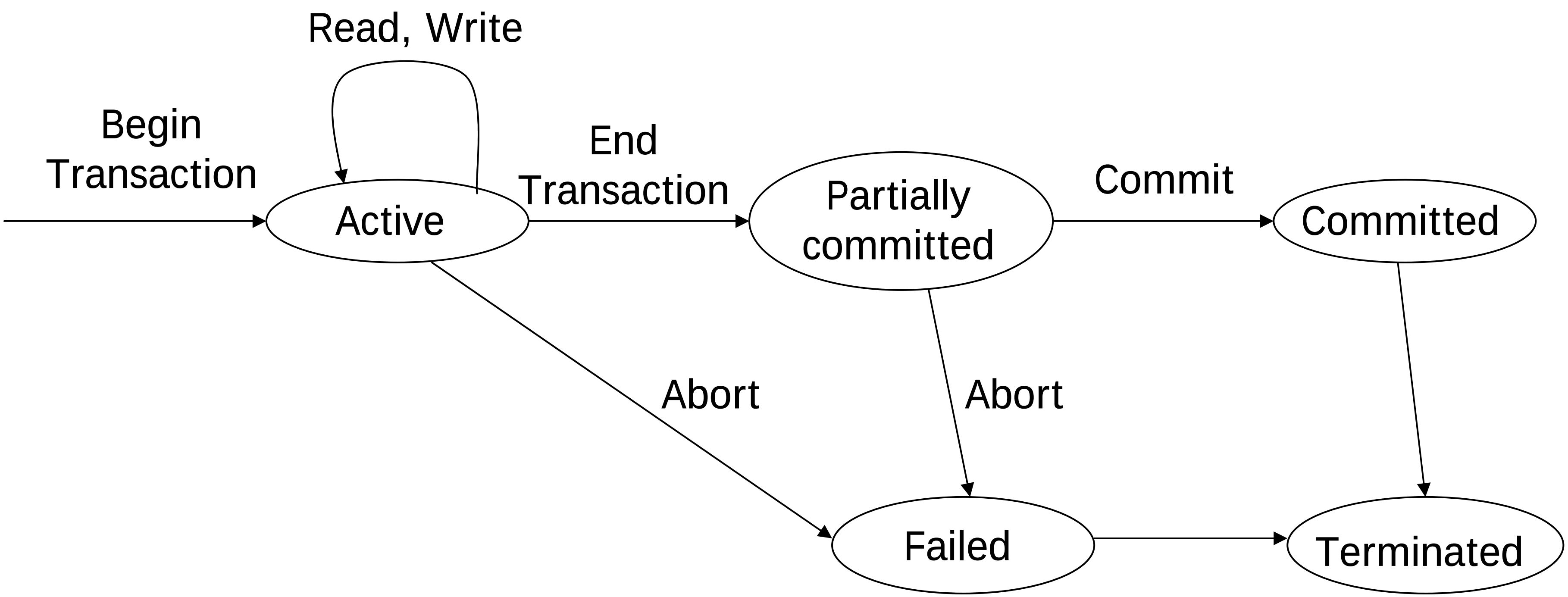
... - Transaction Processing Concepts ...

- A **transaction** is an action or series of actions, carried out by users or applications, which accesses or changes the contents of a DB.
- A transaction is the logical unit of a DB processing.
- An application program is a series of transactions with non-DB processing in between.
- A transaction transforms a DB from one consistent state to another

... - Transaction Processing Concepts ...

- A transaction can have one of two outcomes:
  - **Success** – transaction commits and DB reaches a new consistent state.
  - **Failure** – transaction aborts, and DB must be restored (rolledback) to the consistent state before its started the failed transaction.
  - Committed transaction can not be aborted.
  - Aborted transactions that are rolled back can be restarted later.

... - Transaction Processing Concepts ...



... - Transaction Processing Concepts ...

- Basic properties of a transaction are (**ACID**)
  - **Atomicity**: All or nothing property.
  - **Consistency preservation**: Must transform DB from one consistency state to another.
  - **Isolation**: Partial effects of incomplete transactions should not be visible to other transactions.
  - **Durability**: Effects of a committed transaction are permanent and must not be lost because of other failures.

- Introduction to Concurrency Control ...

- Most real-time DB systems are **multi-user system**; i.e. many users can access and use the system concurrently.

- For example, an airline reservation system is used by hundreds of travel agents and reservation clerks concurrently. Database systems in banks, insurance agencies, stock exchange, supermarkets, and the like are operated on by many users who submit transactions concurrently to the system.

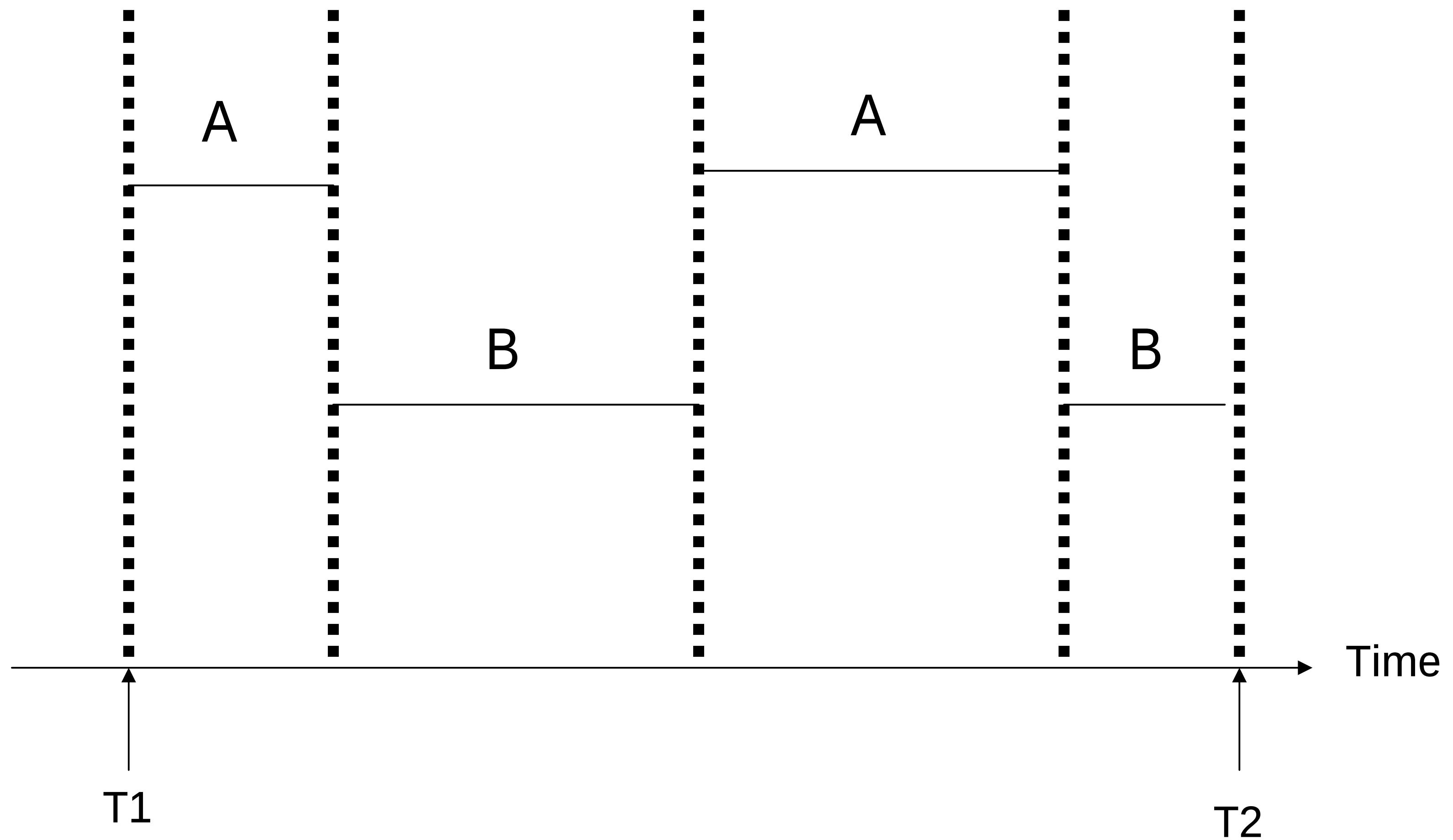
- Multiple users can access DB and use computer systems simultaneously because of the concepts of multiprogramming, which allows the computer to execute multiple programs (processes) at the same time.

... - Introduction to Concurrency Control ...

- If only a single CPU exists, it can actually execute at most one process at a time. However, multiprogramming operating systems execute some commands from one process, then suspend that process and execute some commands from the next process, and so on.

- A process is resumed at the point where it was suspended whenever it gets its turn to use CPU again. Hence, **concurrent execution** of processes is actually **interleaved**.

-- Example: Interleaved Processing



... - Introduction to Concurrency Control ...

- **Concurrency control** is a the process of managing simultaneous operations on the DB without having them interfere with one another.
- The objective of concurrency control is to prevent interference when two or more users are accessing DB simultaneously and at least one is updating data.
- Although two transactions may be correct in themselves, interleaving of operations may produce an incorrect result.

... - Introduction to Concurrency Control

- Several problems can occur when concurrent transactions execute in an uncontrolled manner. Some of the potential problems are caused by uncontrolled concurrency and illustrate the need for concurrency control are:
  - The lost update problem +
  - The temporary update problem +
  - The incorrect analysis problem +

-- The Lost Update Problem

- This problem occurs when two transactions that access the same DB items have their operations interleaved in a way that makes the value of some DB items incorrect.
- For the following two transactions assume the initial value of X is 80, N = 5, and M = 4.

Time

T1	T2
Read_item(X); X := X - N;	
	Read_item(X); X:= X+M;
Write_item(X); Read_item(Y);	
	Write_item(X);
Y:= Y + N; Write_item(Y);	

Item X has an
Incorrect value
Because its update
By T1 is lost

-- The Lost Update Problem

- This problem occurs when two transactions that access the same DB items have their operations interleaved in a way that makes the value of some DB items incorrect.
- For the following two transactions assume the initial value of bal is 1000, M = 100, and N = 100.

Time

T1	T2
Read X; X := X - M;	
	Read X; X := X+N;
Write X;	
	Write N;

Item X has an
Incorrect value
Because its update
By T1 is lost

-- Temporary Update Problem

- This problem occurs when one transaction updates a DB item and then the transaction fails for some reason. The updated item is accessed by another transaction before it is changed back to its original value.

Time

T1	T2
Read X; **X:=X-N;** **Write X;**	
	Read X; **X:=X+M;** **Write X;**
Read Y;	

T1 fails and must change the value Of X back to its old value; Meanwhile T2 has read The “temporary” incorrect Value of X

-- The Incorrect Analysis Problem

- If one transaction is calculating an aggregate summary function on a number of records while other transactions are updating some of these records, the aggregate function may calculate some values before they are updated and others after they are updated.

T1	T3
	Sum := 0; **Read A ;** **Sum := sum+A;**
Read X; **X:=X-N;** **Write X;**	
	Read X **Sum :=sum+X;** **Read Y;** **Sum := sum+Y;**
Read Y **Y:=Y+N;** **Write Y ;**	

T3 reads X after N is Subtracted and reads Y before N is added; The result is a wrong Summary (off by N)