U O W

Introduction to Transaction Processing

(Database transactions, principles and correctness)

CSIT882: Data Management Systems

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

Outline

- Database transactions
- Principles of transaction processing
- Correctness

An interesting experiment

```
$sqlplus jrg@csci
```

```
SQL> SELECT COUNT(*)
  2  FROM SKILL;

  COUNT(*)
----------
        19
```

```
$sqlplus jrg@csci
```

```
SQL> SELECT COUNT(*)
  2  FROM SKILL;

  COUNT(*)
----------
        19
```

An interesting experiment

```
SQL> INSERT INTO SKILL
  2  VALUES('singing');

1 row created.
```

```
SQL> SELECT COUNT(*)
  2  FROM SKILL;

  COUNT(*)
----------
        20
```

```
SQL> SELECT COUNT(*)
  2  FROM SKILL;

  COUNT(*)
----------
        19
```

An interesting experiment

```
SQL> COMMIT;

Commit complete.
```

```
SQL> SELECT COUNT(*)
  2  FROM SKILL;

  COUNT(*)
----------
        20
```

```
SQL> SELECT COUNT(*)
  2  FROM SKILL;

  COUNT(*)
----------
        20
```

Transaction ? What is it ?

A partially ordered set of *read*, *write* operations on the database items is called as a transaction

A transaction might be a whole program, or a part of a program, or several statements, or a single statement.

Outline

- Database transactions
- Principles of transaction processing
- Correctness

Principles of transaction processing

Database users interact with a database system by processing the programs

Processing of a program is equivalent to processing of a partially ordered set of *read*, *write* operations on data

Database is visible to transactions as a collection of data items

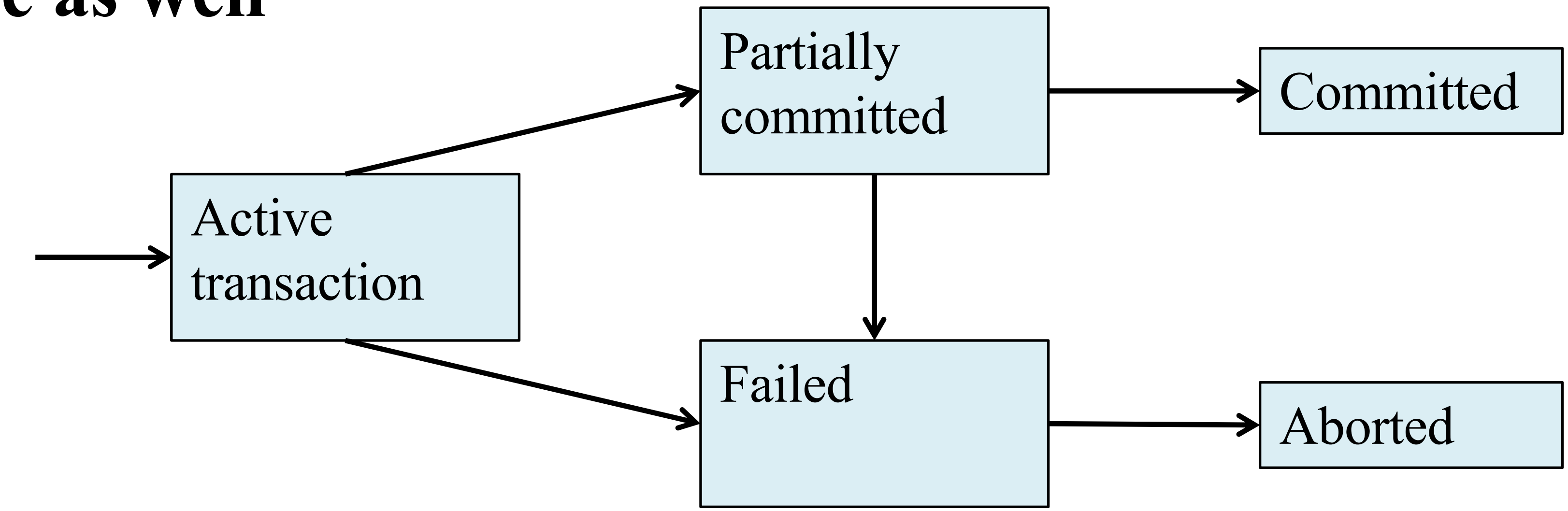
Concurrently running transactions interleave their operations

Transactions have no impact on processing of their operations and transaction do not communicate with each other

Principles of transaction processing

Each transaction terminates by either *commit* or *abort (rollback)* operation

Each transaction arrives at a consistent database state and must leave a database in a consistent state as well



Properties of transactions

Basic properties for all transactions are called ACID.

Atomicity: A transaction unit is indivisible. It must be processed entirely or not at all.

Consistency: A transaction must transform the database from one consistent state to another consistent state.

Isolation: Each transaction must be processed independently.

Durability: The data affected by a committed transaction must be permanently recorded in the database.

Outline

- Database transactions
- Principles of transaction processing
- Correctness

So, where is a problem ?

T_1	T_2	x = $100
v=read(x)		x = $100
	w=read(x)	x = $100
write(x,v-10)		x = $90
	write(x,w+20)	x = $120
	commit	x = $120
commit		x = $120

Correctness condition

What makes concurrent execution of database transaction correct/incorrect ?

How do we define a correct concurrent execution of database transactions ?

Schedule

A schedule is a processing of concurrent transactions that preserves the order of the operations in each transaction.

T_1	T_2	T_3
`read(x)`		
`write(x)`		
`commit`		
	`read(y)`	
		`read(z)`
	`write(y)`	
	`commit`	
		`write(z)`
		`commit`

Serial schedule

A serial schedule consists of a set of concurrent transactions, that processed their operation consecutively

T_1	T_2	T_3
`read(x)`		
`write(x)`		
`commit`		
		`read(z)`
		`write(z)`
		`commit`
	`read(y)`	
	`write(y)`	
	`commit`	

Nonserial schedule

A nonserial schedule consists of a set of concurrent transactions that are interleaved.

T_1	T_2	T_3
read(x)		
write(x)		
commit		
	read(y)	
		read(z)
	write(y)	
	commit	
		write(z)
		commit

Serializability

A serial schedule never leaves the database in an inconsistent state even if different results may be generated.

In a nonserial schedule if concurrent transactions produce the same results as in a certain serial schedule, and each transaction reads the same data items then the nonserial schedule is called view serializable.

Serializability

Problem:

Testing if a schedule is view serializable is NP-complete

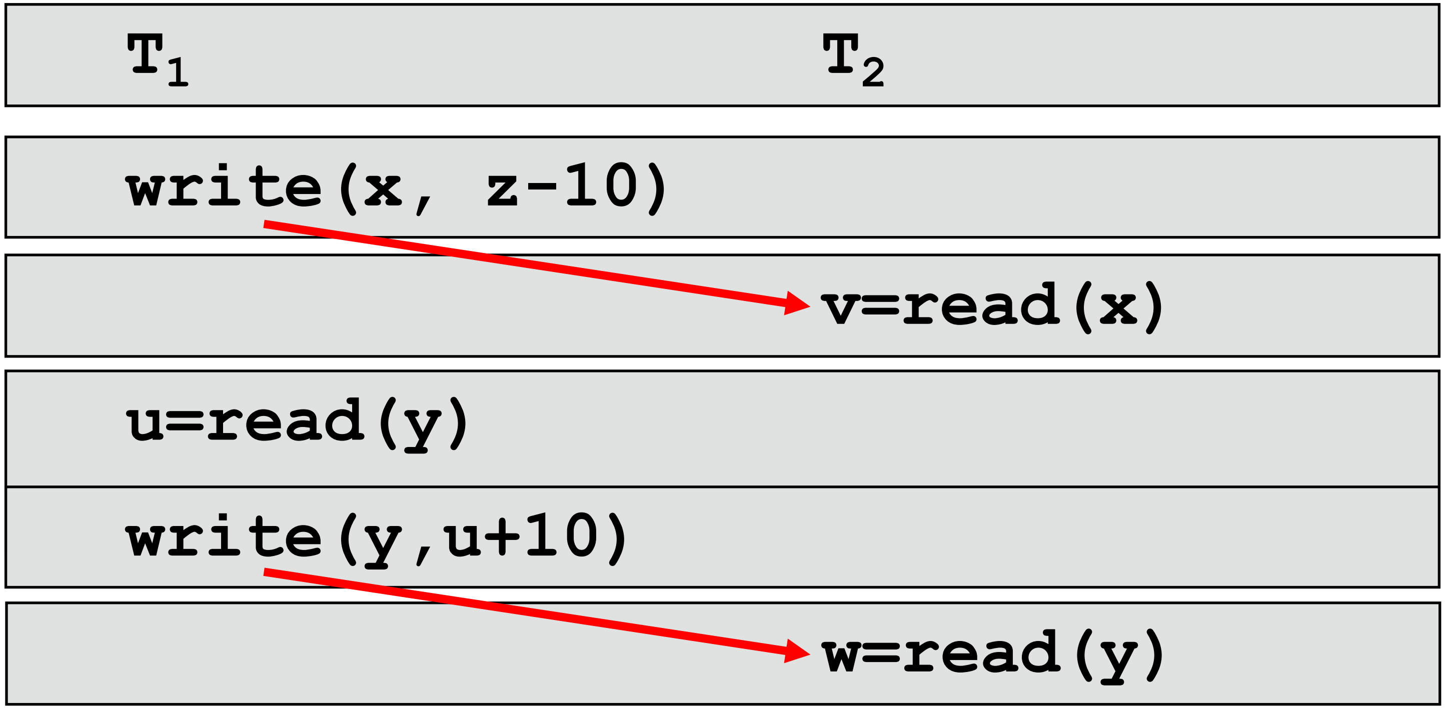
It means, that time needed to decide whether a given schedule is view serializable (correct) grows exponentially with the grows of the total number of transactions involved in a schedule

Conflicting operations

We say operations read and write performed by two transactions conflict…

… if operations act on the same data item and one of these operations is write operation.

X	read	write
read	NO	YES
write	YES	YES

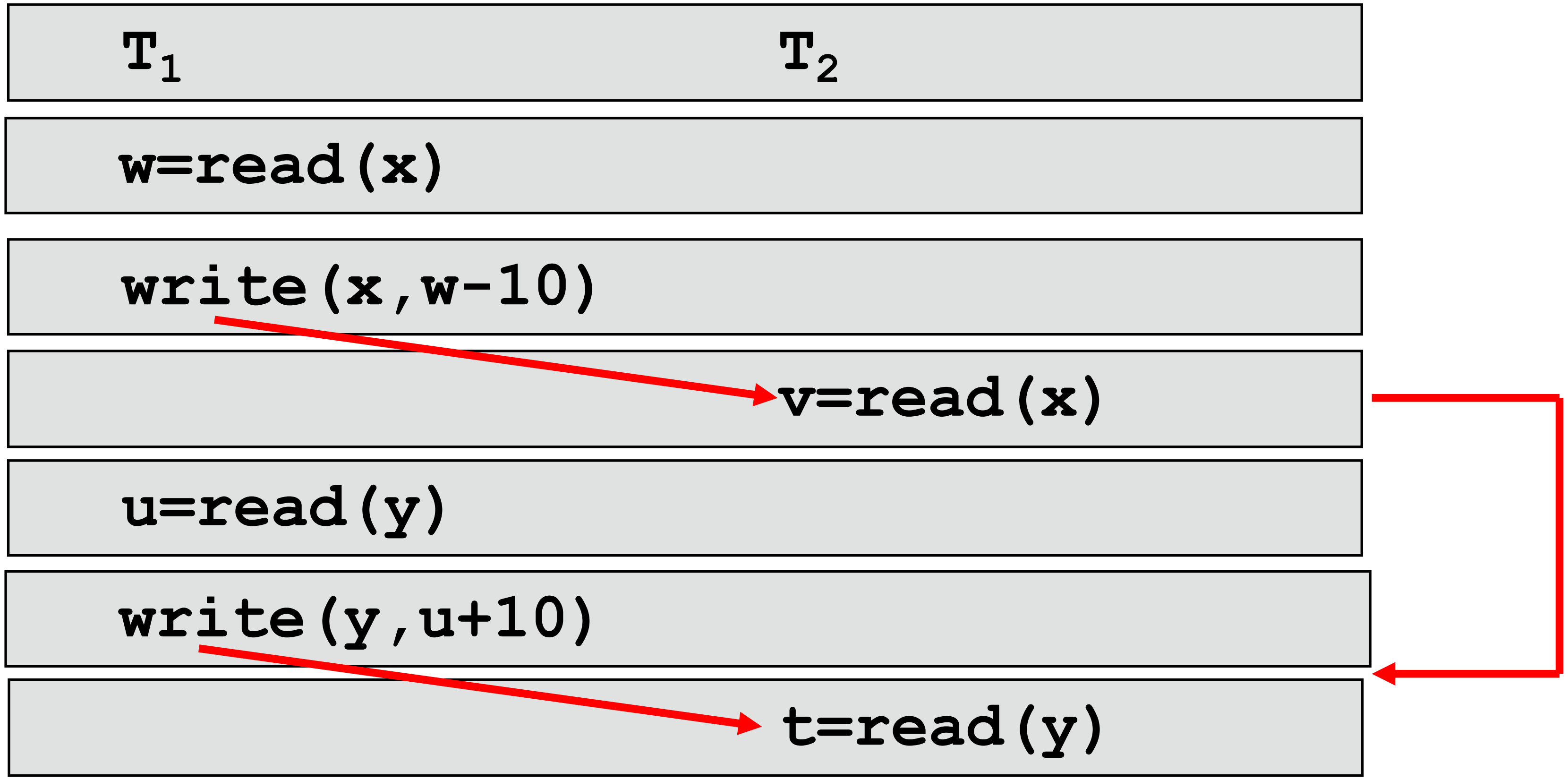


T_1	T_2
`write(x, z-10)`	
	`v=read(x)`
`u=read(y)`	
`write(y,u+10)`	
	`w=read(y)`

Conflict serializability condition

Nonserial schedule of database transactions is conflict serializable if there exists a possible serial schedule of the same set of transactions such that in both schedules the order of conflicting operations is the same

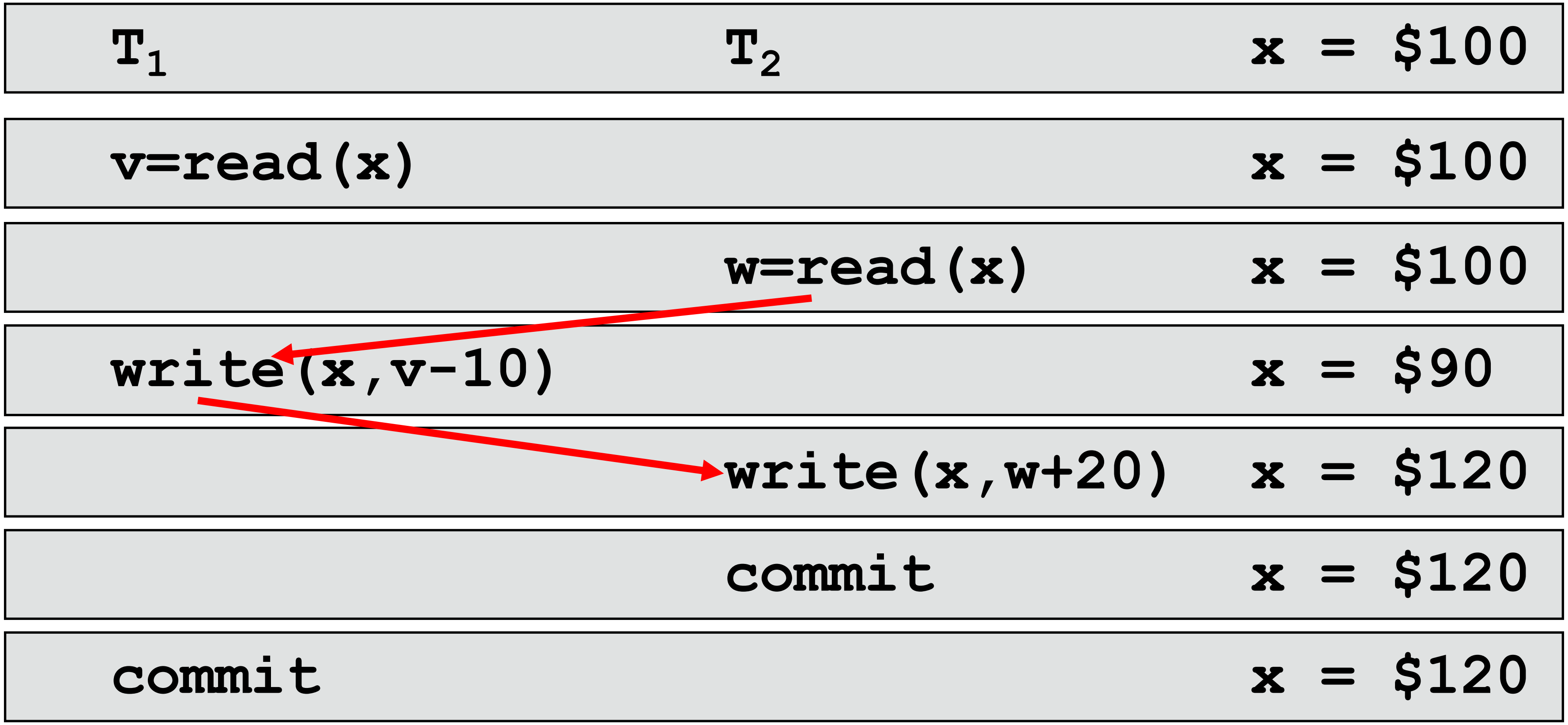
Conflict serializable execution



T_1	T_2
`w=read(x)`	
`write(x,w-10)`	
	`v=read(x)`
`u=read(y)`	
`write(y,u+10)`	
	`t=read(y)`

Order of conflicting operations: T_1 before T_2

Conflict nonserializable execution



T_1	T_2	x = \$100
v=read(x)		x = \$100
	w=read(x)	x = \$100
write(x,v-10)		x = \$90
	write(x,w+20)	x = \$120
	commit	x = \$120
commit		x = \$120

Order of conflicting operations:

T_1 before T_2 and T_2 before T_1 impossible to serialize

References

Elmasri R., Navathe S., Fundamentals of Database Systems, 6th edition, chapter 21 Introduction to Transaction Processing Concepts and Theory, pp. 747-779

Elmasri R., Navathe S., Fundamentals of Database Systems, 6th edition, chapters 22.1, 22.3 Concurrency Control Techniques, pp. 780-794