

Database Administration

José Orlando Pereira

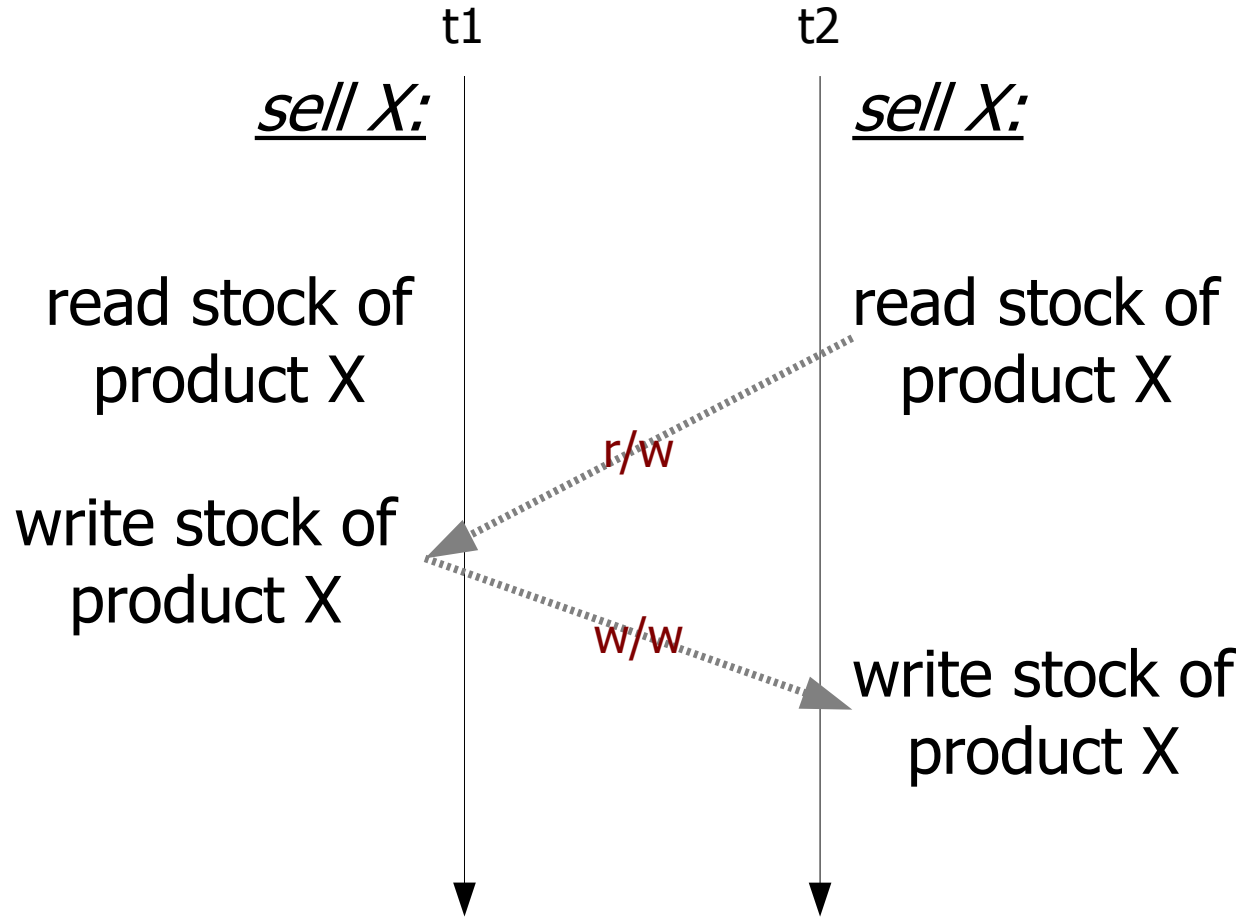
Departamento de Informática
Universidade do Minho



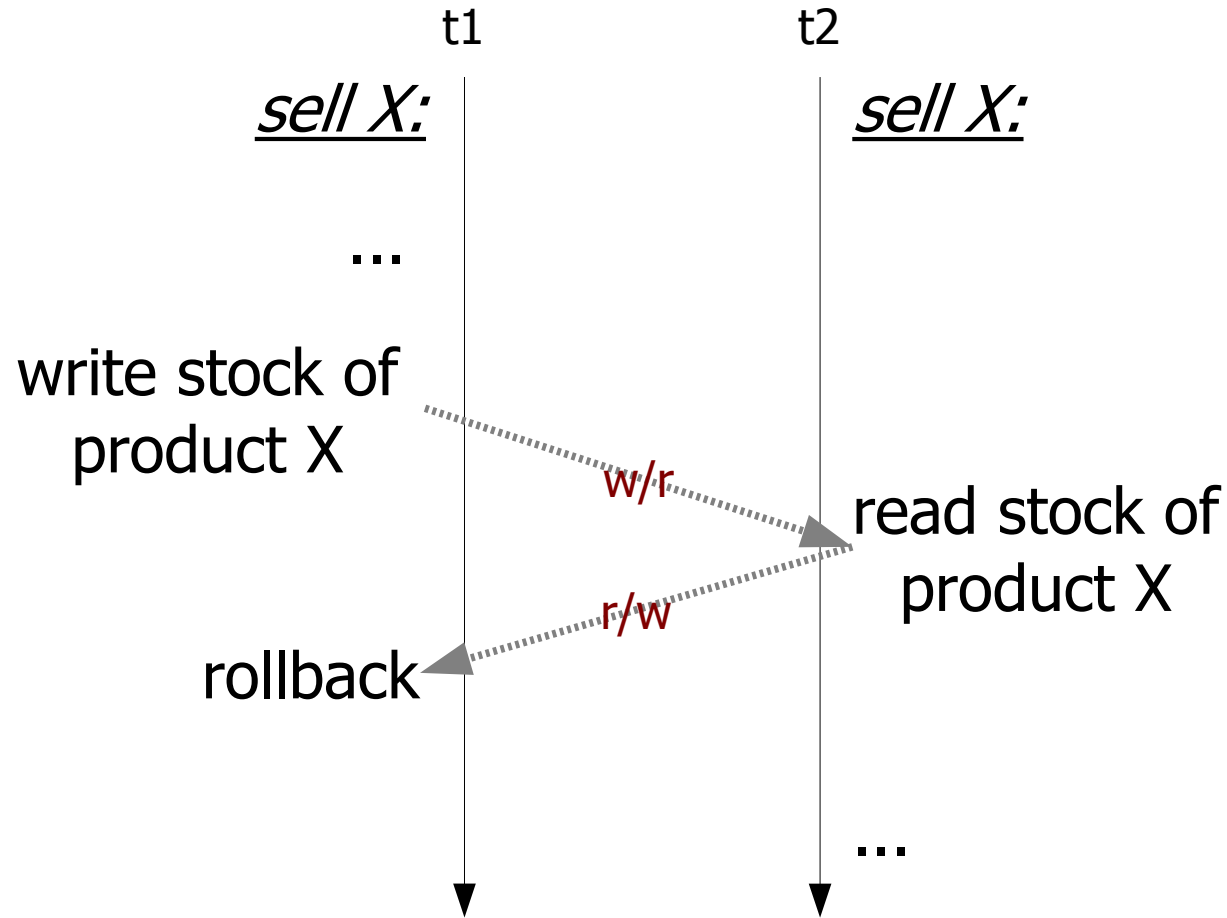
Motivation

- First, implement each operation assuming that no two operations run concurrently
- Then, assume that any operations can run concurrently
- What can go wrong?

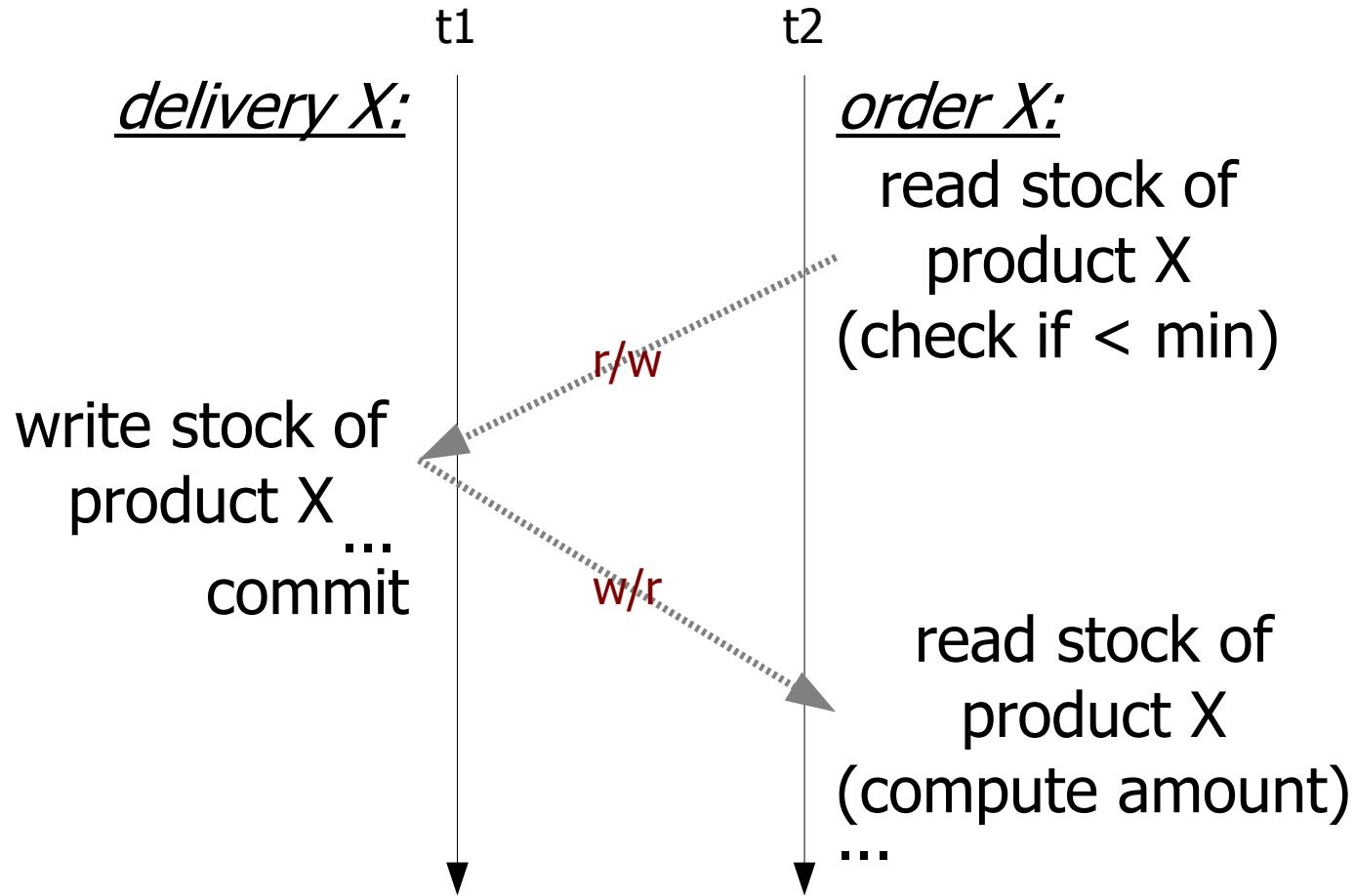
Lost update (RWW)



Dirty read (WRW)



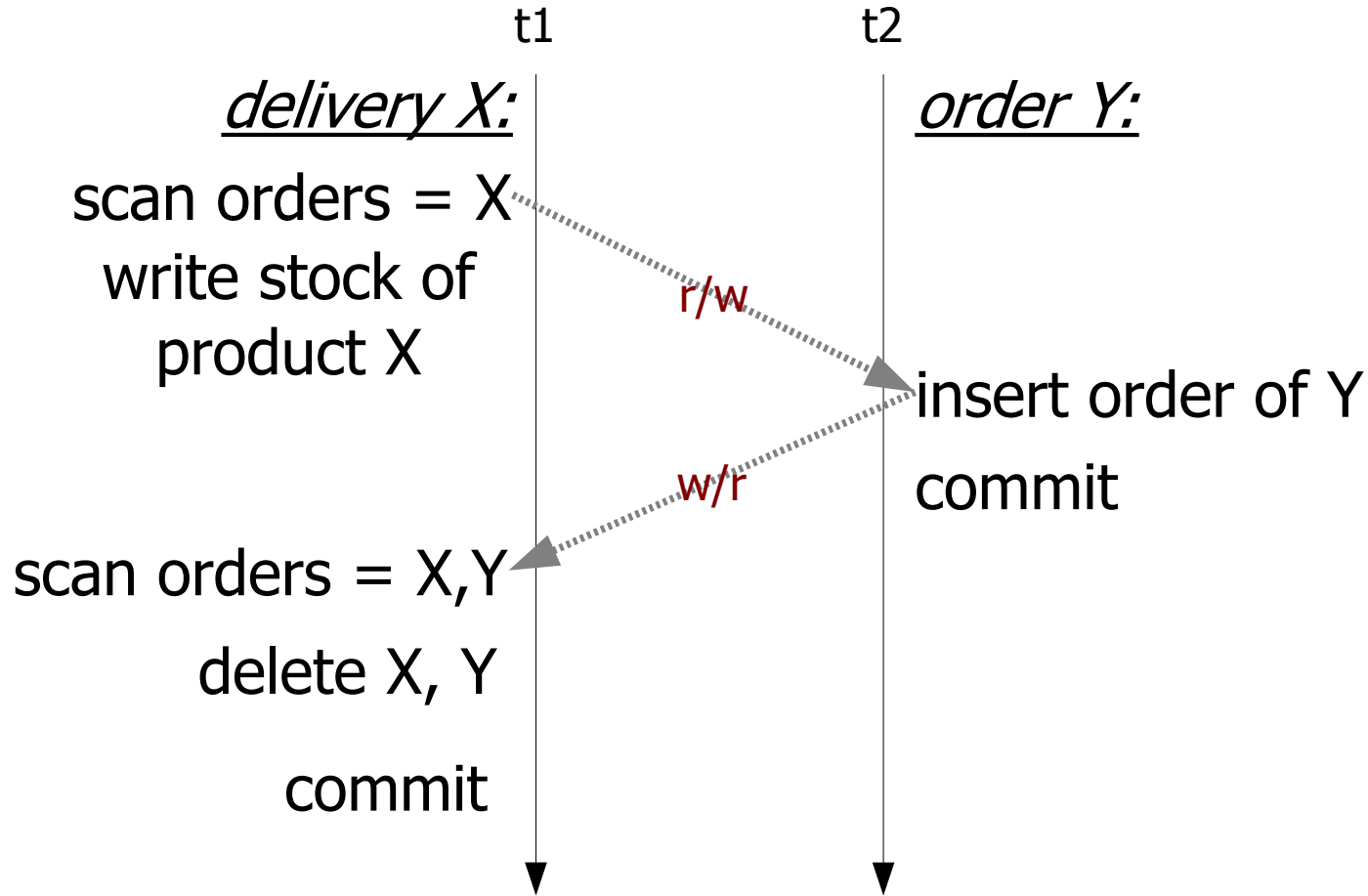
Non-repeatable read (RWR)



Other anomalies?

- Read after read is not a problem
 - Thus no RRW, WRR...
- Why no WWR?
 - Assume no “blind writes”
 - Actually a **RWW**R (lost update)

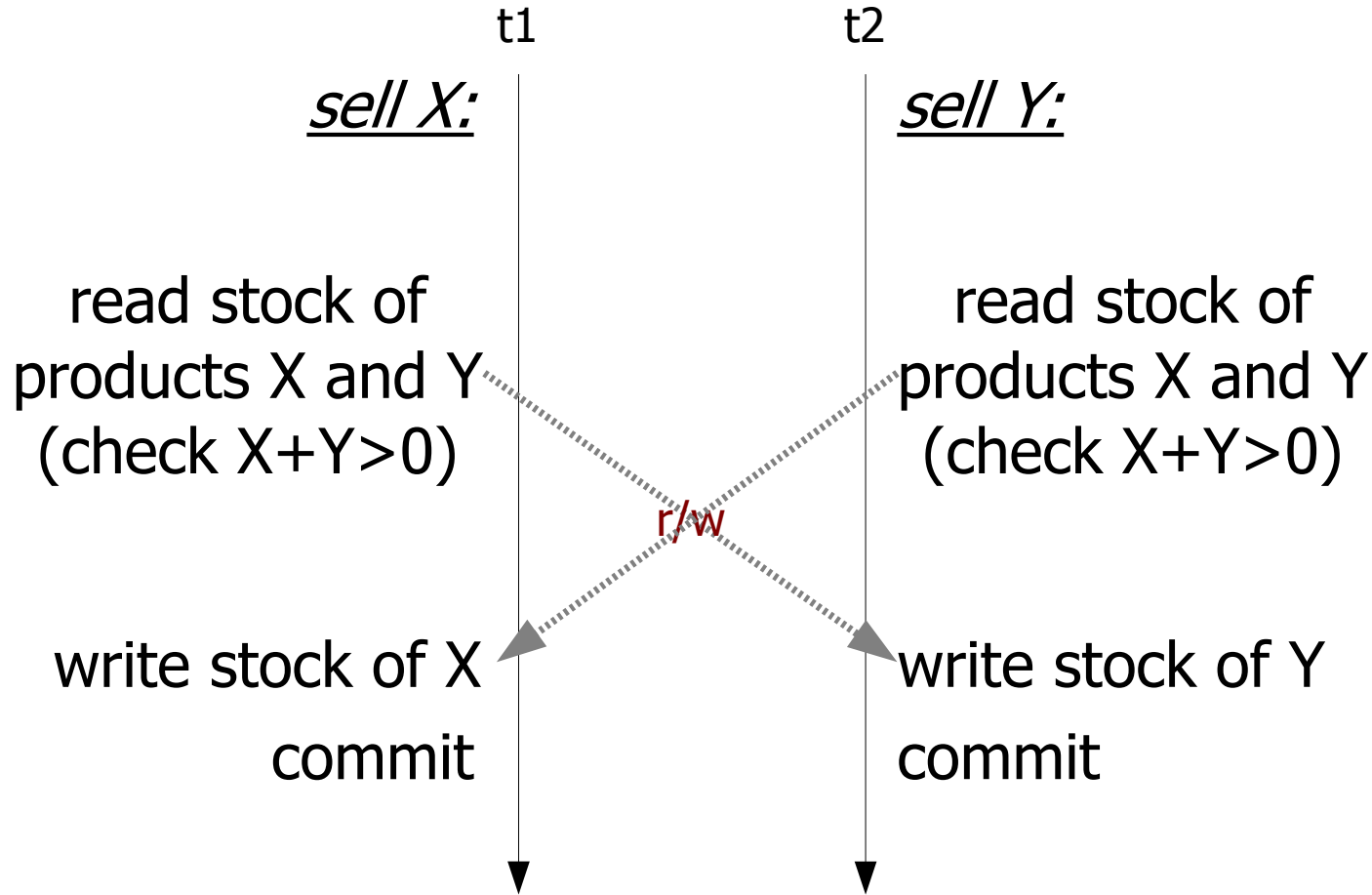
Phantoms



Phantoms

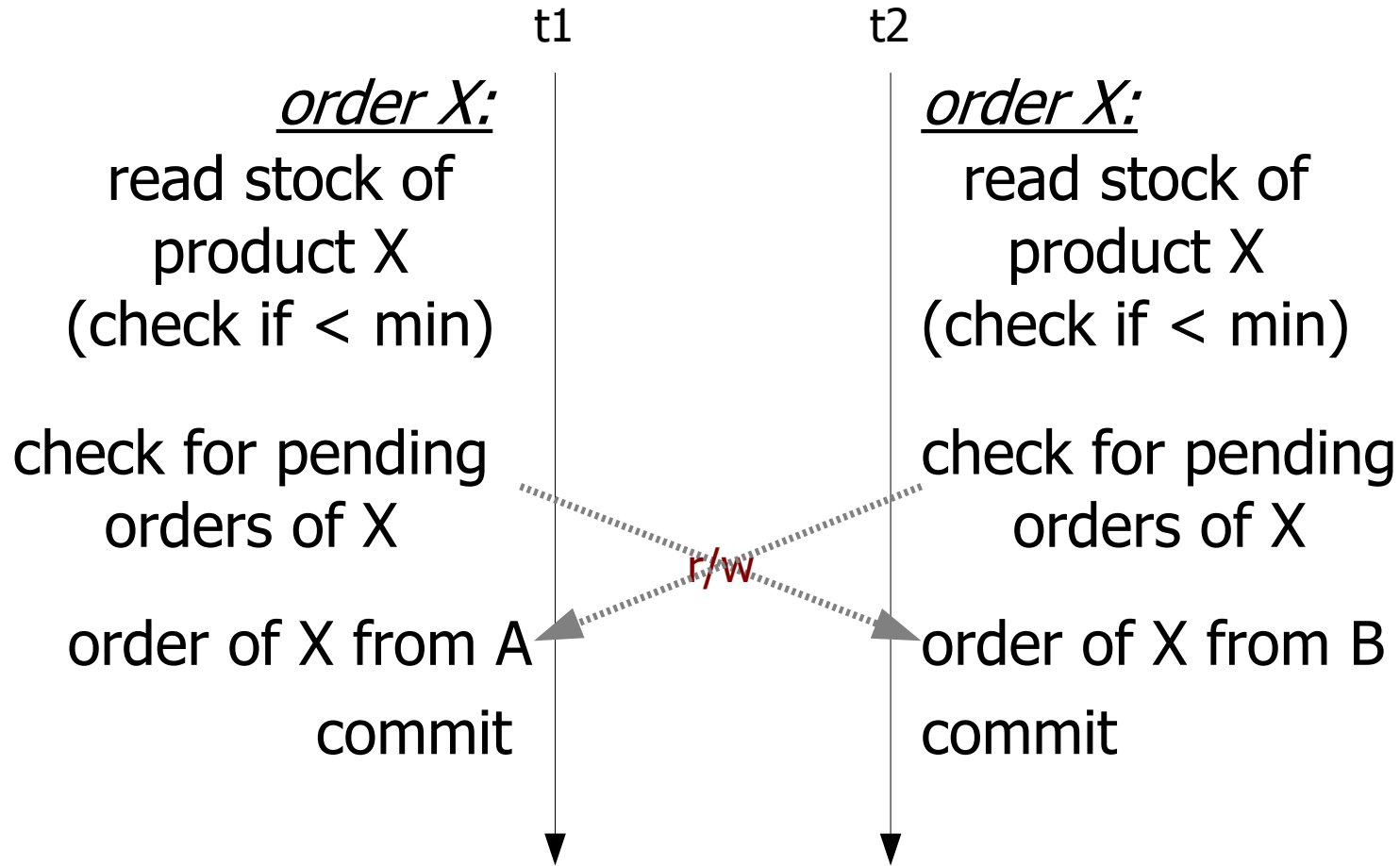
- It is actually a non-repeatable read (RWR) on the collection
- Why no dirty read (WRW) for collections?
 - Solved by having no dirty reads on the item
- Why no RWW (lost update) for collections?
 - Means allocating the same physical space for two records!
 - Very dangerous: corruption, etc...

Write skew (aka “short fork”)



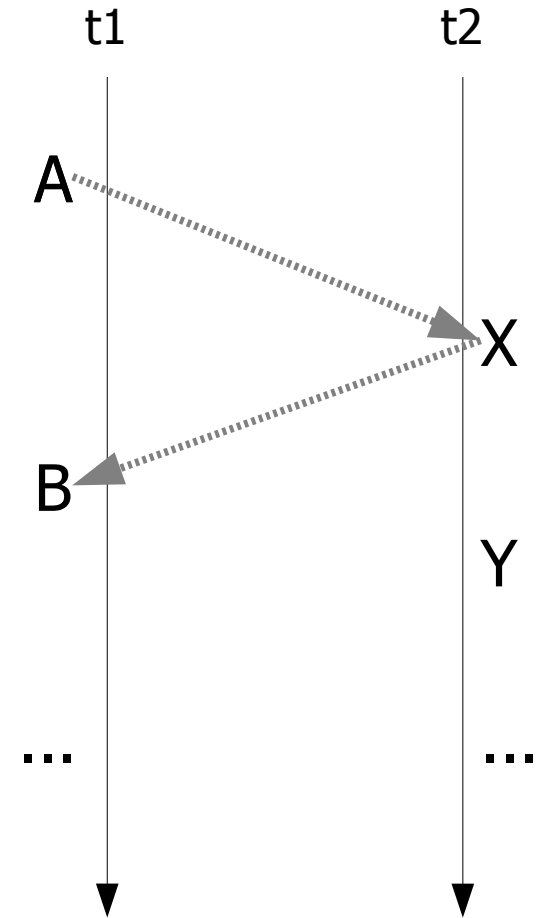
(assume X is backup for Y and vice-versa)

Write skew on collections



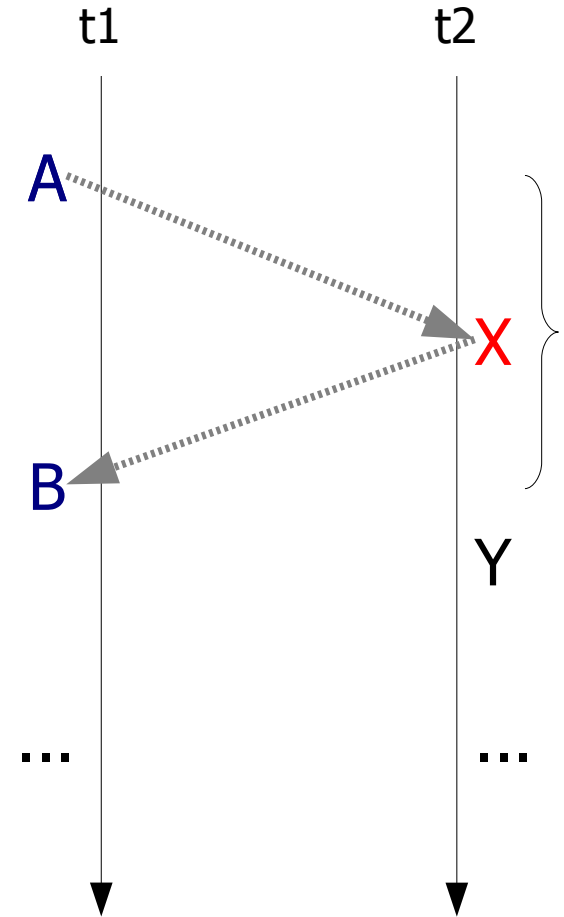
General problem

- No serial execution is conceivable:
 - Some t1 must be ordered after t2
 - But t2 must be ordered after t1
- The user cannot be fooled into thinking that transactions execute serially (i.e. serialized)



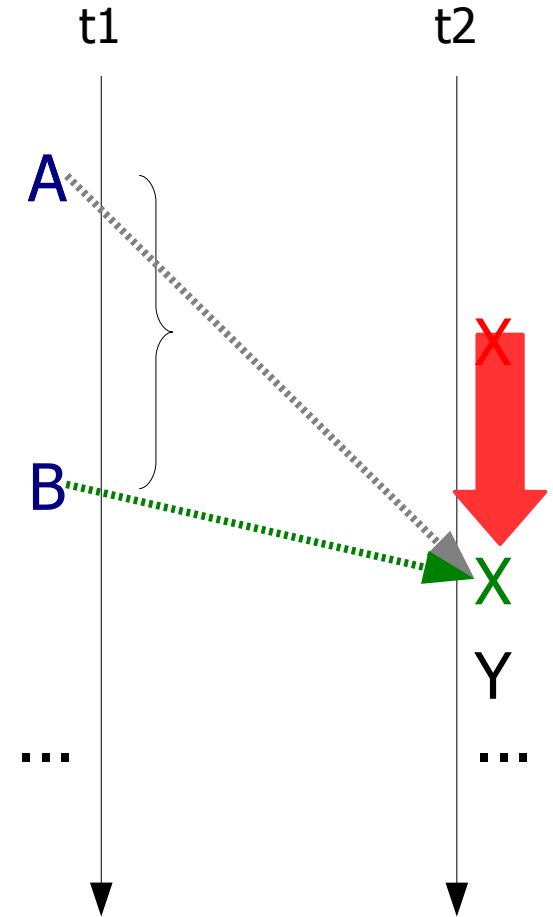
General problem

- In detail, some operation X should not be happening between A and B...



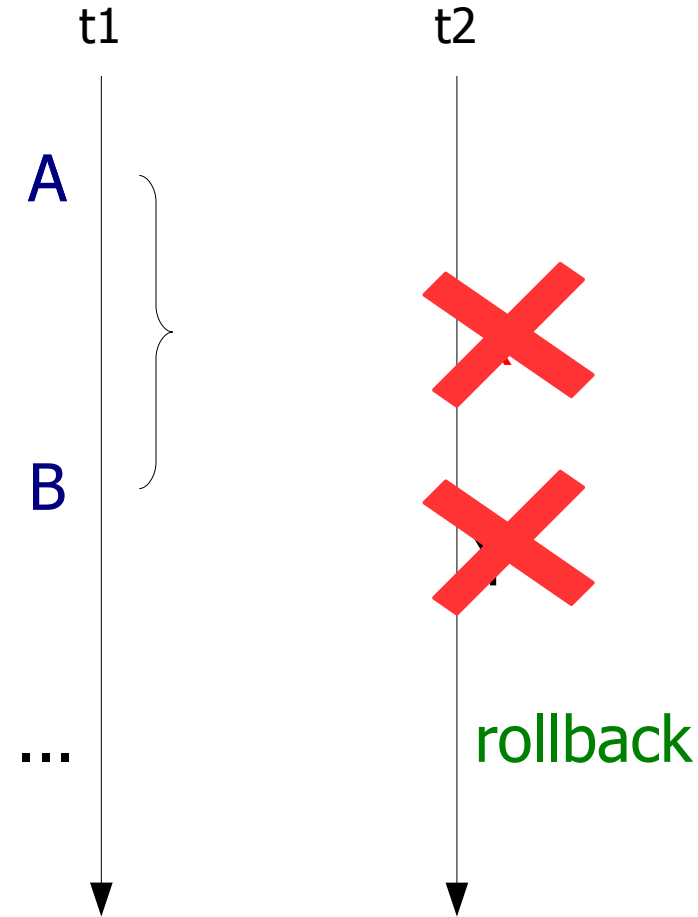
General approach

- Delay X (and all its consequences) until B
 - t1 precedes t2



General approach

- Remove X and related operations
 - t1 executes alone



General approach

- Anticipate X (i.e. execute X before the application has requested it!)
 - t2 precedes t1
- (Can you propose a mechanism to do this!?)

