Timestamps for Serializability

Alternative to Locks for Concurrency Control
Timestamp Serializability Requirement

- Lock Version: Schedule is equivalent to some serial Schedule
- Timestamp Version: Schedule is equivalent to exactly one serial Schedule
- That schedule: TIMESTAMP ORDERED
TIMESTAMP ORDER

• Timestamp ordered Serial Schedule: transactions execute in the same order as they began in the database.

• Goal: Schedule that is serializable to Timestamp Ordered Serial Schedule.
Punish Out of Order Transactions

• Proper order for conflicting pairs:
  Older gets to data item first
  Younger gets to it next

• Out of order discovered when older turns up late.

• Older punished with death.
Example TO Schedules
Example TO Schedules

Timestamps in parentheses

T1(10)  T2(20)  T3(30)

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E
Example TO Schedules

Timestamps in parentheses

T1(10)   T2(20)   T3(30)

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E

Serial in the Timestamp Order
Example TO Schedules

Timestamps in parentheses

T1(10)  T2(20)  T3(30)

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Serial in the Timestamp Order

Read C
Write C
Write E
Example TO Schedules

Timestamps in parentheses
T1(10) T2(20) T3(30)

Serial in the Timestamp Order

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E
Example TO Schedules

Timestamps in parentheses

$T_1(10)$  $T_2(20)$  $T_3(30)$

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E

Serial in the Timestamp Order

$T_1(10)$  $T_2(20)$  $T_3(30)$

Read C
Read A
Read B
Write B
Read D

Write C
Write B
Write E

Write A
Example TO Schedules

Timestamps in parentheses

T1(10)  T2(20)  T3(30)

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E

Serial in the Timestamp Order

T1(10)  T2(20)  T3(30)

Read C
Read A
Read B
Write B
Read D

Write C
Write E
Write A
Example TO Schedules

T1(10)  T2(20)  T3(30)
Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E

Serial in the Timestamp Order

Timestamps in parentheses

T1(10)  T2(20)  T3(30)
Read C
Read B
Write B
Read D

Read A
Read B
Write B
Read D

Write C
Write E
Write A
Example TO Schedules

Timestamps in parentheses
T1(10)  T2(20)  T3(30)

Read A
Read B
Write A

Read C
Read B
Write B
Read C

Read C
Write C
Write E

Serial in the Timestamp Order

Equivalent: each pair in the same order
Another Example

Timestamps in parentheses
T1(10)  T2(20)  T3(30)

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Serial in the Timestamp Order
Read C
Write C
Write E
Another Example

Timestamps in parentheses
T1(10)  T2(20)  T3(30)

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E

Serial in the Timestamp Order

T1(10)

Read A
Read B
Write A
Another Example

Timestamps in parentheses
T1(10)  T2(20)  T3(30)
Read A  Read B  Write A
Read C  Read B  Read D
Write B  Write C  Write E

Serial in the Timestamp Order

T1(10)  T2(20)
Read C  Read A
Read B  Read B
Write B  Write D
Write A
Another Example

Timestamps in parentheses
T1(10)  T2(20)  T3(30)
Read A
Read B
Write A

Serial in the
Timestamp Order

Read C
Read B
Write B
Read D

Read C
Write C

Read A
Read B
Write A

Read C
Write C
Write E

Read C
Write E
Another Example

Timestamps in parentheses
T1(10)  T2(20)  T3(30)
Read A
Read B
Write A
Read C
Read B
Write B
Read D
Read C
Write C
Write E

Serial in the Timestamp Order

Obviously Serializable

...
Another Example

Timestamps in parentheses
T1(10)  T2(20)  T3(30)
Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E

Serial in the Timestamp Order

T1(10)  T2(20)  T3(30)
Read C
Write C
Write E

Obviously Serializable ...

Read C
Read B
Write B
Read D

Read C
Write C
Write E

Read A
Read B
Write A
Another Example

Timestamps in parentheses
T1(10)  T2(20)  T3(30)

Read A
Read B
Write A

Read C
Read B
Write B
Read D

Read C
Write C
Write E

Serial in the Timestamp Order

Obviously Serializable

...
Another Example

Timestamps in parentheses
T1(10)  T2(20)  T3(30)
Read A
Read B
Write A
Read C
Read B
Write B
Read D
Read C
Write C
Write E
Serial in the Timestamp Order

Obviously Serializable
...

T1(10)  T2(20)  T3(30)
Read C
Write C
Write E
Read C
Write B
Write E
Read D
Read A
Read B
Write A
But NOT equivalent -- so invalid under this protocol
Protocol Method

- Identify Violations of TS order
- Kill Violator
- Restart Violator with new TS
- Mechanism: Each Data Item is Timestamped
  - INHERITED from TS of youngest transaction to access item
  - NOT FROM TIME OF ACCESS
  - item gets Read TS and Write TS
Proper Order: Write-Read

• Only concerned with conflicting Operations
• Proper: Older Writes value then Younger Reads it
• Violation: Younger Reads value, then Older Writes it
• Older should have written first
• Older dies when it tries to write
TS Order: Older Writes, Younger Reads

A
ReadTS
WriteTS

A
ReadTS
WriteTS
TS Order: Older Writes, Younger Reads

T4(40)    T5(50)
Write A    Read A

A
ReadTS
WriteTS

A
ReadTS
WriteTS
TS Order: Older Writes, Younger Reads

T4(40)  T5(50)
Write A  Read A
A
ReadTS
WriteTS  40

A
ReadTS
WriteTS
TS Order: Older Writes, Younger Reads

T4(40)  T5(50)

Write A → Read A

A
ReadTS
WriteTS 40

OK to Read:
T5’s Timestamp
is > A’s WriteTS
TS Order: Older Writes, Younger Reads

T4(40)  T5(50)

Write A

A
ReadTS  50
WriteTS  40

OK to Read:
T5’s Timestamp
is > A’s WriteTS

A
ReadTS
WriteTS
TS Order: Older Writes, Younger Reads

T4(40)   T5(50)
Write A    Read A

A
ReadTS    50
WriteTS    40

OK to Read: T5’s Timestamp
is > A’s WriteTS

T4(40)   T5(50)
Write A    Read A

A
ReadTS
WriteTS
TS Order: Older Writes, Younger Reads

T4(40)  T5(50)

Write A  Read A

A
ReadTS  50
WriteTS  40

OK to Read:
T5’s Timestamp
is > A’s WriteTS
TS Order: Older Writes, Younger Reads

T4(40)  T5(50)
Write A  Read A
A
ReadTS  50
WriteTS 40

OK to Read: T5’s Timestamp is > A’s WriteTS

T4(40)  T5(50)
Write A  Read A
A
ReadTS  50
WriteTS

Violation: T4’s Timestamp is < A’s ReadTS.
TS Order: Older Writes, Younger Reads

T4(40)  T5(50)
Write A  Read A
A  
ReadTS  50
WriteTS  40

OK to Read: T5’s Timestamp is > A’s WriteTS

T4(40)  T5(50)
Write A  Read A
A  
ReadTS  50
WriteTS

Violation: T4’s Timestamp is < A’s ReadTS. T4 should have been there already. T4 dies.
TSO: Older Reads before Younger Writes

A
ReadTS
WriteTS

A
ReadTS
WriteTS
TSO: Older Reads before Younger Writes

T4(40)  T5(50)

Read A  Write A

A
ReadTS
WriteTS

A
ReadTS
WriteTS
TSO: Older Reads before Younger Writes

T4(40) T5(50)

Read A → Write A

A
ReadTS 40
WriteTS

A
ReadTS
WriteTS
TSO: Older Reads before Younger Writes

T4(40)  T5(50)

Read A  Write A

A
ReadTS  40
WriteTS

OK to Write:
T5’s Timestamp is > A’s ReadTS
TSO: Older Reads before Younger Writes

T4(40)  T5(50)

Read A → Write A

OK to Write: T5’s Timestamp is > A’s ReadTS

A
ReadTS  40
WriteTS  50

A
ReadTS
WriteTS
TSO: Older Reads before Younger Writes

T4(40)  T5(50)

Read A  Write A

A
ReadTS   40
WriteTS  50

OK to Write: T5’s Timestamp is > A’s ReadTS

T4(40)  T5(50)

Read A  Write A

A
ReadTS
WriteTS
TSO: Older Reads before Younger Writes

T4(40)  T5(50)

Read A  →  Write A

A
ReadTS   40
WriteTS  50

OK to Write:
T5’s Timestamp is > A’s ReadTS

T4(40)  T5(50)

Read A  →  Write A

A
ReadTS
WriteTS  50
**TSO: Older Reads before Younger Writes**

**Left:**
- **T4(40) T5(50)**
- **Read A**
- **Write A**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>ReadTS</td>
<td>40</td>
</tr>
<tr>
<td>WriteTS</td>
<td>50</td>
</tr>
</tbody>
</table>

**OK to Write:**
- T5’s Timestamp is > A’s ReadTS

**Right:**
- **T4(40) T5(50)**
- **Read A**
- **Write A**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
</tr>
<tr>
<td>ReadTS</td>
<td></td>
</tr>
<tr>
<td>WriteTS</td>
<td>50</td>
</tr>
</tbody>
</table>

**Violation:**
- T4’s Timestamp is < A’s WriteTS.
TSO: Older Reads before Younger Writes

T4(40)  T5(50)

Read A  Write A

A
ReadTS  40
WriteTS 50

OK to Write: T5’s Timestamp is > A’s ReadTS

T4(40)  T5(50)

Read A  Write A

A
ReadTS
WriteTS 50

Violation: T4’s Timestamp is < A’s WriteTS. T4 should have been there already. T4 dies.
TSO: Older Writes before Younger Writes

A
ReadTS
WriteTS

A
ReadTS
WriteTS
TSO: Older Writes before Younger Writes

T4(40) T5(50)
Write A

A
ReadTS
WriteTS

A
ReadTS
WriteTS
TSO: Older Writes before Younger Writes

T4(40)  T5(50)

Write A  Write A

<table>
<thead>
<tr>
<th>A</th>
<th>ReadTS</th>
<th>WriteTS 40</th>
</tr>
</thead>
</table>

| A   | ReadTS | WriteTS |
TSO: Older Writes before Younger Writes

T4(40) T5(50)
Write A

A
ReadTS
WriteTS 40

OK to Write: T5’s Timestamp is > A’s WriteTS

A
ReadTS
WriteTS
TSO: Older Writes before Younger Writes

T4(40)  T5(50)

Write A

A
ReadTS
WriteTS  50

OK to Write: T5’s Timestamp is > A’s WriteTS

A
ReadTS
WriteTS
TSO: Older Writes before Younger Writes

T4(40)  T5(50)

Write A  Write A

A
ReadTS
WriteTS  50

OK to Write: T5’s Timestamp is > A’s WriteTS
TSO: Older Writes before Younger Writes

T4(40)  T5(50)

Write A  Write A

A
ReadTS
WriteTS  50

OK to Write: T5’s Timestamp is > A’s WriteTS

T4(40)  T5(50)

Write A  Write A

A
ReadTS
WriteTS  50
TSO: Older Writes before Younger Writes

T4(40)  T5(50)

Write A  Write A

A
ReadTS  50
WriteTS

OK to Write: T5’s Timestamp is > A’s WriteTS

T4(40)  T5(50)

Write A  Write A

A
ReadTS
WriteTS  50

Violation: T4’s Timestamp is < A’s WriteTS.
TSO: Older Writes before Younger Writes

T4(40)  T5(50)

Write A  Write A

A
ReadTS  50
WriteTS

OK to Write: T5's Timestamp is > A's WriteTS

T4(40)  T5(50)

Write A

A
ReadTS
WriteTS  50

Violation: T4's Timestamp is < A's WriteTS. T4 should have been there already. T4 dies.
Summary

• **Reading:** you cannot read a younger transaction’s value
  – Transaction’s **TS** must be \( \geq \) data item’s **write TS**

• **Writing:** you cannot write what a younger transaction has read
  – Transaction’s **TS** must be \( \geq \) data item’s **read TS**

• **Writing:** you cannot overwrite a younger transaction’s value
  – Transaction’s **TS** must be \( \geq \) data item’s **write TS**
Notes

• TSO: killed transaction restarted with new Timestamp
  – younger transactions favored by rules
• TSO protocol does not use locks so NO Deadlock
• TSO protocol guarantees serializable schedules
• TSO drawback: transactions killed unnecessarily