Deadlock Control

Prevention and Detection
## Deadlock

<table>
<thead>
<tr>
<th>Transaction A</th>
<th>Transaction B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read X</td>
<td>Read Y</td>
</tr>
<tr>
<td>Read Y</td>
<td>Read X</td>
</tr>
<tr>
<td>Write X</td>
<td>Write Y</td>
</tr>
</tbody>
</table>
Deadlock

Transaction A
- Read X
- Writelock X
- Read Y
- Write X

Transaction B
- Read Y
- Read X
- Write Y
<table>
<thead>
<tr>
<th>Transaction A</th>
<th>Transaction B</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Write</strong> X</td>
<td><strong>Write</strong>lock Y</td>
</tr>
<tr>
<td>Read X</td>
<td>Read Y</td>
</tr>
<tr>
<td>Read Y</td>
<td>Read X</td>
</tr>
<tr>
<td>Write X</td>
<td>Write Y</td>
</tr>
</tbody>
</table>
Deadlock

Transaction A
Read X
Read Y
Write X

Transaction B
Write Y
Read Y
Read X
Write Y
### Deadlock

<table>
<thead>
<tr>
<th>Transaction A</th>
<th>Transaction B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read X</td>
<td>Write lock Y</td>
</tr>
<tr>
<td>Read Y</td>
<td>Read Y</td>
</tr>
<tr>
<td>Write X</td>
<td>Read X</td>
</tr>
</tbody>
</table>

Transaction A requests write lock on X, then read lock on Y, and finally write lock on X. Transaction B requests write lock on Y, then read lock on Y, and finally read lock on X. Transaction A cannot get lock on X due to transaction B's write lock on Y, and sleep.
Deadlock

Transaction A

Read X

Writelock X

Read Y

Readlock Y

Write X

Cannot get lock due to B’s
Writelock -- A sleeps

Transaction B

Read Y

Writelock Y

Write Y

Readlock X

Read X
Deadlock

Transaction A
Read X
Read Y
Write X

Transaction B
Read Y
Read X

Internal
Cannot get lock due to B's
Readlock Y
Writelock Y
B sleeps

Internal
Cannot get lock due to A's
Readlock X
Writelock -- A sleeps

Readlock Y
Writelock X
A sleeps
Deadlock

Transaction A
Read X
Read Y
Write X

Transaction B
Read Y
Read X
Write Y

Writelock X
Readlock Y
Writelock Y
Readlock X

Cannot get lock due to B’s
Writelock -- A sleeps

Cannot get lock due to A’s
Writelock -- B sleeps

Each sleeps, waiting for the other
Deadlock

Transaction A
- Read X
- Read Y
- Write X

Cannot get lock due to B's Writelock
A sleeps

Transaction B
- Read Y
- Readlock X
- Read X
- Write Y

Cannot get lock due to A's Writelock
B sleeps

Each sleeps, waiting for the other
Deadlock

Transaction A
- Read X
- Write X
- Read Y

Transaction B
- Read Y
- Write Y
- Readlock X
- Writelock Y

Each sleeps, waiting for the other

- Cannot get lock due to B's Writelock
- A sleeps

- Cannot get lock due to A's Writelock
- B sleeps
Deadlock Prevention Strategy

• Change some rule about locks
• Normally: Can’t get lock ⇒ Sleep on Queue
• New rule: Can’t get lock ⇒ Some transaction may die
• Strategy: find something (X) that always happens when there is going to be a deadlock:
  \[
  \text{If deadlock about to happen Then X is true}
  \]
• Turn it around to make the rule:
  \[
  \text{If X is true Then deadlock is about to happen}
  \text{And someone must die to prevent it.} 
  \]
Prevention Solutions

- Conservative 2PL
- Wait-Die Protocol
- Wound-Wait Protocol
- No Waiting
- Cautious Waiting
Conservative 2PL

• Saw this one earlier
• Get all needed locks at beginning
• If you cannot get them, die and retry
Wait-Die Protocol

• If you cannot get a lock, sometimes die instead of waiting

• Rule: Compare age of transaction who holds lock and transaction who wants it
  – Older Transaction will wait
  – Younger Transaction must die
## Wait-Die Example

<table>
<thead>
<tr>
<th>OLDER</th>
<th>YOUNGER</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Transaction A</strong></td>
<td><strong>Transaction B</strong></td>
</tr>
<tr>
<td>Read X</td>
<td>Read Y</td>
</tr>
<tr>
<td>Read Y</td>
<td>Read X</td>
</tr>
<tr>
<td>Write X</td>
<td>Write Y</td>
</tr>
</tbody>
</table>
Wait-Die Example

OLDER
Transaction A
Read X
Write lock X
Read Y
Write X

.transaction B
Read Y
Read X
Write Y

YOUNGER
Wait-Die Example

**OLDER**

Transaction A
- Read X
- **Writelock X**
- Read Y
- Write X

Transaction B
- **Writelock Y**
- Read Y
- Read X
- Write Y

**YOUNGER**
Wait-Die Example

Transaction A
- Write lock X
- Read Y
- Read X

Transaction B
- Write lock Y
- Read Y
- Read X

Write lock X

Write lock Y
Wait-Die Example

**OLDER**

Transaction A
- Read X
- Writelock X
- Read Y
- Write X

**YOUNGER**

Transaction B
- Read Y
- Writelock Y
- Read X
- Write Y

---

Cannot get lock due to B's Writelock -- A is older: waits
Wait-Die Example

**OLDER**

Transaction A
- Read X
- Read Y
- Write X

Transaction B
- Read Y
- Readlock X
- Writelock Y
- Writelock X

**YOUNGER**

Cannot get lock due to B’s Writelock -- A is older: waits
Wait-Die Example

Transaction A
Read X
Read Y
Write X

Transaction B
Write Y
Read Y
Read X
Write Y

Cannot get lock due to B's Writelock -- A is older: waits

Cannot get lock due to A's Writelock -- B is younger: dies
Wait-Die Example

Transaction A
- Read X
- Write X
- Read Y
- Readlock Y
- Writelock X

Cannot get lock due to B's Writelock -- A is older: waits

Transaction B
- Read Y
- Writelock Y
- Read X
- Readlock X
- Write Y

Cannot get lock due to A's Writelock -- B is younger: dies

No Deadlock
### Unnecessary Deaths

<table>
<thead>
<tr>
<th>Younger</th>
<th>Older</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transaction A</td>
<td>Transaction B</td>
</tr>
<tr>
<td>Read X</td>
<td>Read Y</td>
</tr>
<tr>
<td>Read Y</td>
<td>Read W</td>
</tr>
<tr>
<td>Write X</td>
<td>Write Y</td>
</tr>
</tbody>
</table>
Unnecessary Deaths

Transaction A

YOUNGER

Read X

Write X

Transaction B

OLDER

Read Y

Write Y

Writelock X
Unnecessary Deaths

Transaction A
Read X
Write X

Transaction B
Read Y
Write Y

OLDER

YOUNGER

Writelock X
Writelock Y

Read Y
Read W
Unnecessary Deaths

Transaction A
- Read X
- Writelock X
- Read Y
- Write X

Transaction B
- Writelock Y
- Readlock Y
- Read W
- Write Y
Unnecessary Deaths

Transaction A
- Read X
- Writelock X
- Read Y
- Write X

Transaction B
- Writelock Y
- Read Y
- Read W
- Write Y

Cannot get lock due to B's Writelock -- A is younger: dies

OLDER

YOUNGER
Unnecessary Deaths

**OLDER**
Transaction B
Writelock Y
Read Y
Readlock W
Read W
Write Y

**YOUNGER**
Transaction A
Writelock X
Read X
Read Y
Readlock Y
Write X

*Cannot get lock due to B’s Writelock -- A is younger: dies*
Unnecessary Deaths

Transaction A
- Read X
- Read Y
- Write X
- Writelock X
- Readlock Y

Transaction B
- Read Y
- Readlock W
- Read W
- Writelock Y
- Unlock W

Cannot get lock due to B's Writelock -- A is younger: dies
Unnecessary Deaths

Transaction A
- Read X
- Read Y
- Write X

Transaction B
- Read Y
- Readlock W
- Readlock Y
- Writelock Y

Cannot get lock due to B's Writelock -- A is younger: dies

OLDER

YOUNGER
Unnecessary Deaths

Transaction A

Read X

Write X

Read Y

Readlock Y

Cannot get lock due to B's Writelock -- A is younger: dies

Transaction B

Read Y

Writelock Y

Readlock W

Write Y

Unlock Y

Unlock W

No Deadlock would have occurred if A waited
Wound-Wait Protocol

• If you cannot get a lock, sometimes kill instead of waiting
• Rules:
  – Older Transaction kills younger
  – Younger Transaction will wait
• Death comes to younger lock holder like bolt from the blue
  – Not when younger does anything
  – When older tries for lock
<table>
<thead>
<tr>
<th>Wound-Wait Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>YOUNGER</strong></td>
</tr>
<tr>
<td>Transaction A</td>
</tr>
<tr>
<td>Read X</td>
</tr>
<tr>
<td>Read Y</td>
</tr>
<tr>
<td>Write X</td>
</tr>
<tr>
<td><strong>OLDER</strong></td>
</tr>
<tr>
<td>Transaction B</td>
</tr>
<tr>
<td>Read Y</td>
</tr>
<tr>
<td>Read X</td>
</tr>
<tr>
<td>Write Y</td>
</tr>
</tbody>
</table>
# Wound-Wait Example

<table>
<thead>
<tr>
<th>Younger</th>
<th>Older</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Transaction A</strong></td>
<td><strong>Transaction B</strong></td>
</tr>
<tr>
<td>Write lock X</td>
<td></td>
</tr>
<tr>
<td>Read X</td>
<td>Read Y</td>
</tr>
<tr>
<td>Read Y</td>
<td>Read X</td>
</tr>
<tr>
<td>Write X</td>
<td>Write Y</td>
</tr>
</tbody>
</table>
Wound-Wait Example

**YOUNGER**

Transaction A
- Writelock X
- Read X
- Read Y
- Write X

**OLDER**

Transaction B
- Writelock Y
- Read Y
- Read X
- Write Y
Wound-Wait Example

**YOUNGER**

Transaction A
- Writelock X
- Read X
- Readlock Y
- Read Y
- Write X

**OLDER**

Transaction B
- Writelock Y
- Read Y
- Read X
- Write Y
Wound-Wait Example

Transaction A
- Writelock X
- Read X
- Read Y
- Write X

Transaction B
- Writelock Y
- Read Y
- Read X
- Write Y

Cannot get lock due to B's Writelock -- A is younger: waits

OLDER

YOUNGER
Wound-Wait Example

Transaction A
- Writelock X
- Read X
- Read Y
- Write X

Transaction B
- Writelock Y
- Read Y
- Readlock X
- Read X
- Write Y

Cannot get lock due to B’s Writelock -- A is younger: waits
Wound-Wait Example

**Transaction A**
- Writelock X
- Read Y
- Write X

**Transaction B**
- Writelock Y
- Read Y
- Readlock X

**B YOUNGER**
- Cannot get lock due to B's Writelock -- A is younger: waits

**A OLDER**
- Cannot get lock due to A's Writelock -- B is older: kills A
**Wound-Wait Example**

**YOUNGER**

Transaction A
- Writelock X
- Read X
- Read Y
- Write X
- **Cannot get lock due to B's Writelock -- A is younger: waits**

**OLDER**

Transaction B
- Writelock Y
- Read Y
- **Readlock X**
- **Readlock Y**
- **Cannot get lock due to A's Writelock -- B is older: kills A**

**No Deadlock**
No Waiting

• If you cannot get a lock, assume deadlock
  – Die
  – Wait
  – Restart

• Never a Queue for a lock
Cautious Waiting

• Not so rash as ‘No Waiting’
• If you cannot get a lock, check lock holder
  – If lock holder is already waiting, die and restart
  – If lock holder is active, wait
## Cautious Waiting Example

<table>
<thead>
<tr>
<th>Transaction A</th>
<th>Transaction B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read X</td>
<td>Read Y</td>
</tr>
<tr>
<td>Read Y</td>
<td>Read X</td>
</tr>
<tr>
<td>Write X</td>
<td>Write Y</td>
</tr>
</tbody>
</table>
## Cautious Waiting Example

<table>
<thead>
<tr>
<th>Transaction A</th>
<th>Transaction B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read X</td>
<td>Read Y</td>
</tr>
<tr>
<td>Write X</td>
<td>Read X</td>
</tr>
<tr>
<td>Read Y</td>
<td>Read Y</td>
</tr>
<tr>
<td><strong>Writelock X</strong></td>
<td>Write Y</td>
</tr>
</tbody>
</table>
Cautious Waiting Example

Transaction A
Read X
Writelock X
Read Y
Write X

Transaction B
Writelock Y
Read Y
Read X
Write Y
### Cautious Waiting Example

<table>
<thead>
<tr>
<th>Transaction A</th>
<th>Transaction B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read X</td>
<td>Read Y</td>
</tr>
<tr>
<td><strong>Writelock X</strong></td>
<td><strong>Writelock Y</strong></td>
</tr>
<tr>
<td>Readlock Y</td>
<td>Read X</td>
</tr>
<tr>
<td>Read Y</td>
<td>Write Y</td>
</tr>
<tr>
<td>Write X</td>
<td></td>
</tr>
</tbody>
</table>
Cautious Waiting Example

Transaction A
- Read X
- Writelock X
- Readlock Y
- Read Y
- Write X

Transaction B
- Writelock Y
- Read Y
- Read X
- Write Y

Cannot get lock due to B's Writelock -- B is active: A waits
Cautious Waiting Example

Transaction A

Read X
Write X
Read Y
Readlock Y
Cannot get lock due to B's Writelock -- B is active: A waits

Transaction B

Writelock X
Read Y
Readlock X
Read X
Write Y
Cautious Waiting Example

Transaction A
Read X
Read Y
Write X

Transaction B
Read X
Read Y
Write Y

Writelock X
Readlock Y
Cannot get lock due to B's Writelock -- B is active: A waits

Writelock Y
Readlock X
Cannot get lock due to A's Writelock -- A is sleeping: B dies
Cautious Waiting Example

Transaction A
- Read X
- Read Y
- Write X

Transaction B
- Writelock Y
- Read Y
- Readlock X

**Cannot get lock due to B’s Writelock -- B is active: A waits**

**Cannot get lock due to A’s Writelock -- A is sleeping: B dies**

No Deadlock
Deadlock Detection

• Use locks with no deadlock prevention
• Every so often, check for deadlock
  – use Wait-for Graph
• If found, kill some transaction to break deadlock
No deadlock

No deadlock

Cycle found: deadlock

Wait-for Graph
Cost-Benefit of Detection

• No unnecessary slaughter
• Transactions sit in deadlock for a while instead of restarting
• Time taken to calculate wait-for graph every so often