Outline For Today

Serializability:

1. Execution Histories
2. Conflict Equivalence
3. Checking For Serializability

System’s Perspective (and more next lecture)
Outline For Today

Serializability:

1. Execution Histories
2. Conflict Equivalence
3. Checking For Serializability
Concurrency is achieved by interleaving operations across trxs.

Q: Does an interleaving correspond to a serializable execution?

Execution history model and conflict equivalences is a formal method to answer this question.
Representing Single Transactions

- Database is a set of *data items* (often will denote as x, y, z…)
- Trx $T_i$ is a *total order* of read/write & commit/abort operations on items
  - $r_i(x)$ indicates $T_i$ reads item x
  - $w_i(x)$ indicates $T_i$ writes item x
  - c indicates commit (a could indicate abort but won’t be needed)
    - Suppose: $T_i$ does the following in this *chronological order*:
      - Read(x), Read(y), $x \leftarrow x + y$, Write(x), commit
      - $T_i = \{r_i(x) < r_i(y) < w_i(x) < c_i\}$ or simply as:
        - $T_i = \{r_i(x), r_i(y), w_i(x), c_i\}$ or $r_i(x), r_i(y), w_i(x), c_i$
    - Note: total order is simply the chronological order of operations of $T_i$
An execution history over a set of transactions $T_1 \ldots T_n$ is an interleaving of the operations of $T_1 \ldots T_n$ in which the operation total order imposed by each transaction is preserved.

Example: $T_1 = \{ w_1[x], w_1[y], c_1 \}, \ T_2 = \{ r_2[x], r_2[y], c_2 \}$

- $H_a = w_1[x]r_2[x]w_1[y]r_2[y]c_1c_2$
- $H_b = w_1[x]w_1[y]c_1r_2[x]r_2[y]c_2$
- $H_c = w_1[x]r_2[x]r_2[y]w_1[y]c_1c_2$
- $H_d = r_2[x]r_2[y]c_2w_1[x]w_1[y]c_1$
Examples for Valid Execution History

\[ T_1 = \{ w_1[x], w_1[y], c_1 \}, \quad T_2 = \{ r_2[x], r_2[y], c_2 \} \]
Examples for Invalid Execution History

- $T_1 = \{ w_1[x], w_1[y], c_1 \}$, $T_2 = \{ r_2[x], r_2[y], c_2 \}$

Incorrect orders:
Serial Execution Histories

- $T_1 = \{ w_1[x], w_1[y], c_1 \}$, $T_2 = \{ r_2[x], r_2[y], c_2 \}$

- Serial history: no interleaving of ops from different transactions

<table>
<thead>
<tr>
<th>$T_1$</th>
<th>$T_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1(x)$</td>
<td>$r_2(x)$</td>
</tr>
<tr>
<td>$w_1(y)$</td>
<td>$r_2(y)$</td>
</tr>
<tr>
<td>$c_1$</td>
<td>$c_2$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$T_1$</th>
<th>$T_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1(x)$</td>
<td>$w_1(x)$</td>
</tr>
<tr>
<td>$w_1(y)$</td>
<td>$w_1(y)$</td>
</tr>
<tr>
<td>$c_1$</td>
<td>$c_1$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$T_1$</th>
<th>$T_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_2(x)$</td>
<td>$r_2(x)$</td>
</tr>
<tr>
<td>$r_2(y)$</td>
<td>$r_2(y)$</td>
</tr>
<tr>
<td>$c_2$</td>
<td>$c_2$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$T_1$</th>
<th>$T_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1(x)$</td>
<td>$w_1(x)$</td>
</tr>
<tr>
<td>$w_1(y)$</td>
<td>$w_1(y)$</td>
</tr>
<tr>
<td>$c_1$</td>
<td>$c_1$</td>
</tr>
</tbody>
</table>
Equivalent Histories

- \( H_a \) is “equivalent” to \( H_b \) if values read by each \( T_i \) in \( H_a \) and \( H_b \) is same, and the order or write operations on each item is the same.

<table>
<thead>
<tr>
<th>( T_1 )</th>
<th>( T_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write 4</td>
<td>Write 4</td>
</tr>
<tr>
<td>Write 5</td>
<td>Write 5</td>
</tr>
<tr>
<td>Read 4</td>
<td>Read 4</td>
</tr>
<tr>
<td>Read 5</td>
<td>Read 5</td>
</tr>
</tbody>
</table>

\( H_a \)

\( H_b \)

\( T_2 \) sees all the updates by \( T_1 \)
- \( T_2 \) reads \( x \) written by \( T_1 \)
- \( T_2 \) reads \( y \) written by \( T_1 \)

\( x=3, y=1 \) (before \( T_1 \) and \( T_2 \))
Example Non-equivalent Histories

➢ $H_a$ is “equivalent” to $H_b$ if values read by each $T_i$ in $H_a$ and $H_b$ is same, and the order or write operations on each item is the same.

$H_b$:

- Write 4
- Write 5
- Read 5
- Read 4

$H_c$:

- Write 4
- Write 5
- Read 1
- Read 4

$x=3, y=1$ (before $T_1$ and $T_2$)
Outline For Today

Serializability:

1. Execution Histories
2. Conflict Equivalence
3. Checking For Serializability
Conflict Equivalence

➢ Dfn: Two operations conflict if:

1. they belong to different transactions,
2. they operate on the same object, and at least one of the operations is a write

2 types of conflicts: (1) Read-Write and (2) Write-Write

➢ Dfn: Two histories are (conflict) equivalent if

1. they are over the same set of transactions, and
2. the ordering of each pair of conflicting operations is the same in each history
Example 1

➢ Consider:

\[ H_a = w_1[x]r_2[x]w_1[y]r_2[y]c_1c_2 \]

\[ H_b = w_1[x]w_1[y]r_2[x]r_2[y]c_1c_2 \]

➢ Step 1: check if they are over the same set of transactions

\[ T_1 = \{w_1[x], w_1[y]\}, \quad T_2 = \{r_2[x], r_2[y]\} \]

➢ Step 2: check if all the conflicting pairs have the same order

<table>
<thead>
<tr>
<th>Conflicting pairs</th>
<th>( H_a )</th>
<th>( H_b )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( w_1[x], r_2[x] )</td>
<td>&lt;</td>
<td>&lt;</td>
</tr>
<tr>
<td>( w_1[y], r_2[y] )</td>
<td>&lt;</td>
<td>&lt;</td>
</tr>
</tbody>
</table>
Motivation & Intuition For Conflict Equivalence

➢ If two histories $H_a$ and $H_b$ are conflict equivalent then, we can make $H_a$ exactly the same as $H_b$ by iteratively swapping two consecutive non-conflicting operations in $H_a$ and/or $H_b$.

➢ $H_a = w_1[x]r_2[x]w_1[y]r_2[y]c_1c_2 \Rightarrow H'_a = w_1[x]w_1[y]r_2[x]r_2[y]c_1c_2$

➢ $H_b = w_1[x]w_1[y]r_2[x]r_2[y]c_1c_2$

➢ Proof Sketch: Move all ops on item $x_1$ to the beginning by swapping in both $H_a$ and $H_b$ ($x_1$ ops “move past” ops on other items).

➢ End with the order imposed by the conflicts on $x_1$

➢ If $H_a$ & $H_b$ are conflict eq. this prefix ops on $x_1$ will be the same order.

➢ Then repeat for $x_2$, $x_3$, etc. and we will arrive at the same histories.

➢ Therefore: Every read by each trx has the same value in $H_a$ & $H_b$

➢ Therefore: $H_a$ & $H_b$ lead to the same output database state.
Consider

- $H_A: r_1[x]r_3[x]w_4[y]r_2[u]w_4[z]r_1[y]r_3[u]r_2[z]w_2[z]r_3[z]r_1[z]w_3[y]$
- $H_B: r_1[x]w_4[y]r_3[x]r_2[u]r_1[y]r_3[u]r_2[z]w_2[z]w_4[z]r_1[z]r_3[z]w_3[y]$

Step 1: check if they are over the same set of transactions

Step 2: check if all the conflicting pairs have the same order
Example 2

Consider

- $H_A: r_1[x]r_3[x]w_4[y]r_2[u]w_4[z]r_1[y]r_3[u]r_2[z]w_2[z]r_3[z]r_1[z]w_3[y]$
- $H_B: r_1[x]w_4[y]r_3[x]r_2[u]r_1[y]r_3[u]r_2[z]w_2[z]w_4[z]r_1[z]r_3[z]w_3[y]$

Step 1: check if they are over the same set of transactions

\{r_1[x] \ r_1[y] \ r_1[z] \}, \ \{r_2[u] \ r_2[z]w_2[z]\}, \ \{r_3[x] \ r_3[u] \ r_3[z]w_3[y]\}, \ \{w_4[y] \ w_4[z]\}$

Step 2: check if all the conflicting pairs have the same order
Identify All Conflicting Pairs

- $H_A: r_1[x]r_3[x]w_4[y]r_2[u]w_4[z]r_1[y]r_3[u]r_2[z]w_2[z]r_3[z]r_1[z]w_3[y]$

- Conflicting pairs:
  - Related to x: no conflicting pairs, as all are reads
  - Related to y: $w_4[y], r_1[y], w_3[y]$
    - $w_4[y] < r_1[y]$
    - $w_4[y] < w_3[y]$
    - $r_1[y] < w_3[y]$
  - Related to z: $w_4[z], r_2[z], w_2[z], r_3[z], r_1[z]$
    - $w_4[z] < r_2[z]$
    - $w_4[z] < w_2[z]$
    - $w_4[z] < r_3[z]$
    - $w_4[z] < r_1[z]$
    - $r_2[z], w_2[z]$ are not, as they are from the same transactions
    - $w_2[z] < r_3[z]$
    - $w_2[z] < r_1[z]$
Consider

- $H_A: r_1[x] r_3[x] w_4[y] r_2[u] w_4[z] r_1[y] r_3[u] r_2[z] w_2[z] r_3[z] r_1[z] w_3[y]$
- $H_B: r_1[x] w_4[y] r_3[x] r_2[u] r_1[y] r_3[u] r_2[z] w_2[z] w_4[z] r_1[z] r_3[z] w_3[y]$

Step 1: check if they are over the same set of transactions
- $\{r_1[x] r_1[y] r_1[z]\}$, $\{r_2[u] r_2[z] w_2[z]\}$, $\{r_3[x] r_3[u] r_3[z] w_3[y]\}$, $\{w_4[y] w_4[z]\}$

Step 2: check if all the conflicting pairs have the same order

<table>
<thead>
<tr>
<th>Conflicting pairs</th>
<th>$H_A$</th>
<th>$H_B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_4[y], r_1[y]$</td>
<td>&lt;</td>
<td>&lt;</td>
</tr>
<tr>
<td>$w_4[y], w_3[y]$</td>
<td>&lt;</td>
<td>&lt;</td>
</tr>
<tr>
<td>...</td>
<td>&lt;</td>
<td>&lt;</td>
</tr>
<tr>
<td>$w_4[z], w_2[z]$</td>
<td>&lt;</td>
<td>&gt;</td>
</tr>
<tr>
<td>...</td>
<td>&lt;</td>
<td>&lt;</td>
</tr>
</tbody>
</table>
Outline For Today

Serializability:

1. Execution Histories
2. Conflict Equivalence
3. Checking For Serializability
Serializability

A history $H$ is said to be (conflict) **serializable** if there exists some serial history $H'$ that is conflict equivalent to $H$.

In Example 1 we showed: $H_a = H_b$
Does $H_c$ have an equivalent serial execution?

$H_c = w_1[x] r_2[x] r_2[y] w_1[y] c_1 c_2$

Only 2 serial execution to check:

- $H_b$: $T_1$ followed by $T_2$: $w_1[x] w_1[y] c_1 r_2[x] r_2[y] c_2$
  - $r_2[y]$ reads different value as in $H_c$

- $H_d$: $T_2$ followed by $T_1$: $r_2[x] r_2[y] c_2 w_1[x] w_1[y] c_1$
  - $r_2[x]$ reads different value as in $H_c$

<table>
<thead>
<tr>
<th>Conflicting pairs</th>
<th>$H_b$</th>
<th>$H_c$</th>
<th>$H_d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1[x], r_2[x]$</td>
<td>$&lt;$</td>
<td>$&lt;$</td>
<td>$&gt;$</td>
</tr>
<tr>
<td>$w_1[y], r_2[y]$</td>
<td>$&lt;$</td>
<td>$&gt;$</td>
<td>$&gt;$</td>
</tr>
</tbody>
</table>

Do we need to check all the serial executions?
More practical test for serializability

Serialization graph $SG_H(V, E)$ for history $H$:

- $V = \{T | T$ is a committed transaction in $H\}$
- $E = \{T_i \rightarrow T_j$ if $o_i \in T_i$ and $o_j \in T_j$ conflict and $o_i < o_j\}$

Theorem: A history is serializable iff its serialization graph is acyclic.
Example 1

$H_a = w_1[x]r_2[x]w_1[y]r_2[y] c_1 c_2$

$w_1[x]$ and $r_2[x]$ conflict, and $w_1[x] < r_2[x]$

$w_1[y]$ and $r_2[y]$ conflict, and $w_1[y] < r_2[y]$

Serialization graph: no cycles $\rightarrow$ serializable
Example 2

Example: \( H_c = w_1[x]r_2[x]r_2[y]w_1[y]c_1c_2 \)

\( w_1[x] \) and \( r_2[x] \) conflict, and \( w_1[x] < r_2[x] \);
\( w_1[y] \) and \( r_2[y] \) conflict, and \( r_2[y] < w_1[y] \)

Not serializable
Example 3

\[ r_1[x]r_3[x]w_4[y]r_2[u]w_4[z]r_1[y]r_3[u]r_2[z]w_2[z]r_3[z]r_1[z]w_3[y] \]

Conflicting pairs:

- Related to x: no conflicting pairs, as all are reads
- Related to y: w4[y], r1[y], w3[y]
  - \( w_4[y] < r_1[y] \) \( T4 \rightarrow T1 \)
  - \( w_4[y] < w_3[y] \) \( T4 \rightarrow T3 \)
  - \( r_1[y] < w_3[y] \) \( T1 \rightarrow T3 \)
- Related to z: w4[z], r2[z], w2[z], r3[z], r1[z]
  - \( w_4[z] < r_2[z] \) \( T4 \rightarrow T2 \)
  - \( w_4[z] < w_2[z] \) \( T4 \rightarrow T2 \)
  - \( w_4[z] < r_3[z] \) \( T4 \rightarrow T3 \)
  - \( w_4[z] < r_1[z] \) \( T4 \rightarrow T1 \)
  - \( r_2[z], w_2[z] \) are not, as they are from the same transactions
  - \( w_2[z] < r_3[z] \) \( T2 \rightarrow T3 \)
  - \( w_2[z] < r_1[z] \) \( T2 \rightarrow T1 \)
• $r_1[x]r_3[x]w_4[y]r_2[u]w_4[z]r_1[y]r_3[u]r_2[z]w_2[z]r_3[z]r_1[z]w_3[y]$

• No cycles in this serialization graph
  • Topological sort: T4 -> T2 -> T1->T3

• The history above is (conflict) equivalent to
  $w_4[y]w_4[z]r_2[u]r_2[z]w_2[z]r_1[x]r_1[y]r_1[z]r_3[x]r_3[u]r_3[z]w_3[y]$
  • Note: we ignore the commits at the end for simplicity
Next Lecture: 2 Phase-Locking Protocol That Guarantees Conflict Serializability
(And Logging-based Recovery)