Notes 11: View Serializability

Arthur Keller

Motivating example

Schedule Q

\[
\begin{array}{c|c|c}
T_1 & T_2 & T_3 \\
Read(A) & Write(A) & Write(A) \\
Write(A) & & \\
\end{array}
\]

Same as

\[Q = \text{r}_1(A) \ w_2(A) \ w_1(A) \ w_3(A)\]

P(Q):

\[
\begin{array}{c}
T_1 \\ \downarrow \\ T_2 \\ \downarrow \\ T_3 \\
\end{array}
\]

\[
\text{Not conflict serializable!}
\]

But now compare Q to Ss, a serial schedule:

\[
\begin{array}{c|c|c}
Q & T_1 & T_2 & T_3 \\
\text{r}_1(A) & \text{w}_2(A) & \text{w}_1(A) & \text{w}_3(A) \\
\text{Read(A)} & \text{Write(A)} & \text{Write(A)} & \\
\text{Write(A)} & & & \\
\end{array}
\]

\[
\begin{array}{c|c|c}
Ss & T_1 & T_2 & T_3 \\
\text{r}_1(A) & \text{w}_2(A) & \text{w}_1(A) & \text{w}_3(A) \\
\text{Read(A)} & \text{Write(A)} & \text{Write(A)} & \\
\text{Write(A)} & & & \\
\end{array}
\]

- T_1 reads same thing in Q, Ss
- T_2, T_3 read something (nothing?)
- After Q or Ss, DB is left in same state

So what is wrong with Q?
Definition Schedules S₁, S₂ are View Equivalent if:

1. If in S₁: wₗ(A) ⇒ rᵢ(A) then in S₂: wₛ(A) ⇒ rᵢ(A)
2. If in S₁: rᵢ(A) reads initial DB value, then in S₂: rᵢ(A) also reads initial DB value
3. If in S₁: Ti does last write on A, then in S₂: Ti also does last write on A

⇒ means "reads value produced"

Definition
Schedule S₁ is View Serializable if it is view equivalent to some serial schedule

Lemma
Conflict Serializable ⇒ View Serializable

Proof:
Swapping non-conflicting actions does not change what transactions read nor final DB state

Venn Diagram

Note: All view serializable schedules that are not conflict serializable, involve useless write

S = w₂(A) ... w₃(A)...... no reads
How do we test for view-serializability?

P(S) not good enough...
(see schedule Q)

One problem: some swaps involving conflicting actions are OK… e.g.:

\[ S = \ldots w_2(A) \ldots r_1(A) \ldots w_3(A) \ldots w_4(A) \]

This action can move if this write exists.

Another problem: useless writes

S = \ldots W_2(A) \ldots W_1(A) \ldots

No A reads

To check if S is View Serializable

(1) Add final transaction \( T_f \) that reads all DB
(eliminates condition 3 of V-S definition)

E.g.: \[ S = \ldots W_1(A) \ldots W_2(A) \ldots r_f(A) \]

Last A write?

(2) Add initial transaction \( T_b \)
that writes all DB
(eliminates condition 2 of V-S definition)

E.g.: \[ S = w_b(A) \ldots r_1(A) \ldots w_2(A) \ldots \]

(3) Create labeled precedence graph of S:
(3a) If \( w_i(A) \Rightarrow r_j(A) \) in S, add \( T_i \leadsto T_j \)
(3b) For each \( w_i(A) \Rightarrow r_j(A) \) do consider each \( w_k(A) \): \([T_k \neq T_b]\)
- If \( T_i \neq T_b \land T_j \neq T_f \) then insert \( T_k \rightarrow T_i \) some new \( p \)
- If \( T_i = T_b \land T_j \neq T_f \) then insert \( T_j \rightarrow T_k \)
- If \( T_i \neq T_b \land T_j = T_f \) then insert \( T_k \rightarrow T_i \)

(4) Check if \( LP(S) \) is “acyclic” (if so, \( S \) is V-S)
- For each pair of “p” arcs (\( p \neq 0 \)), choose one

Example: check if \( Q \) is V-S:
\[
Q = r_1(A) w_2(A) w_1(A) w_3(A) \\
Q' = w_b(A) r_1(A) w_2(A) w_1(A) w_3(A) r_f(A)
\]

Another example:
\[
Z = w_b(A) r_1(A) w_2(A) r_3(A) w_1(A) w_3(A) r_f(A)
\]

\( S_s = w_b(A) r_1(A) w_1(A) w_2(A) r_3(A) w_3(A) r_f(A) \)

\( \overline{T_1 \ T_2 \ T_3} \)

- Checking view serializability is expensive
- Still, V-S useful in some cases...
**Example on useless transactions:**

\[ S = w_1(A) r_2(A) w_2(B) r_1(B) w_3(A) w_3(B) \]

\[ S' = T_b w_1(A) \rightarrow r_2(A) w_2(B) r_1(B) w_3(A) w_3(B) \rightarrow T_f \]

- If we only care about final state \} remove \( T_1, T_2 \); i.e., remove useless transactions
- If we care what \( T_1, T_2 \) read (view equivalence), then do **not** remove useless transactions

**Implications:**

If no blind writes, view-serial \( \iff \) conf-serial

\( P(S) \) acyclic \( \iff \) all transactions read the same as in a serial schedule

**Proof(?):** say \( S_1 \) is view-serial and no blind writes. \( S_1 \) V-equiv to \( S_s \), serial schedule.

1. Goal: Show that \( T_1 \rightarrow T_2 \) in \( P(S_1) \) \( \Rightarrow \) \( T_1 <_{ss} T_2 \)
2. Assume \( T_1 \rightarrow T_2 \)
   - if \( S_1 = \ldots w_1(A) \ldots r_1(A) \ldots, \) clearly \( T_1 <_{ss} T_2 \)
   - if \( S_1 = \ldots w_1(A) \ldots r_1(A) w_3(A) \ldots r_2(A) \ldots \)
     also \( T_1 <_{ss} T_2 \)
   - if \( S_1 = \ldots r_1(A) r_3(A) \ldots w_3(A) \ldots w_3(A) \ldots r_2(A) \ldots \) not possible: \( T_1, T_3 \) not serializable
   - Other cases similar...