## 2.7. Examples:

$H_1 = W_1(X)\ R_2(X)\ W_2(Y)\ R_1(Y)\ C_2\ C_1$

$H_2 = R_1(X)\ W_2(X)\ R_3(X)\ W_1(X)\ W_3(X)\ C_3\ C_2\ C_1$

$H_3 = W_1(X)\ W_1(Y)\ R_2(U)\ W_2(X)\ R_2(Y)\ W_2(Y)\ C_2\ W_1(Z)\ C_1$

$H_4 = R_1(X)\ W_1(Z)\ R_2(Z)\ W_2(X)\ R_2(X)\ W_1(X)\ C_1\ C_2$

$H_5 = R_1(X)\ W_2(X)\ R_3(X)\ W_1(X)\ W_3(X)\ C_2\ C_1\ C_3$

$H_6 = W_1(X)\ W_1(Y)\ R_2(U)\ W_2(X)\ R_2(Y)\ W_2(Y)\ W_1(Z)\ C_1\ C_2$

$H_7 = R_1(X)\ W_1(X)\ W_2(Y)\ W_1(Y)\ C_1\ W_2(X)\ R_2(Y)\ C_2$

$H_8 = R_1(X)\ W_2(X)\ C_2\ R_3(X)\ W_1(X)\ W_3(X)\ C_1\ C_3$

$H_{10} = W_1(X)\ W_1(Y)\ R_2(V)\ W_1(Z)\ W_1(U)\ C_1\ W_2(X)\ R_2(Y)\ W_2(Y)$

$H_9 = W_1(X)\ W_2(Y)\ R_2(U)\ W_2(X)\ W_1(Z)\ C_1\ R_2(Y)\ W_2(Y)\ C_2$

$H_{11} = R_1(X)\ W_2(X)\ C_2\ R_3(X)\ W_1(X)\ C_1\ W_3(X)\ C_3$

$H_{12} = W_1(X)\ W_1(Y)\ R_2(U)\ W_1(Z)\ C_1\ W_2(X)\ R_2(Y)\ W_2(Y)\ C_2$

2.12

Consider the following serializable History $H'$:

$$
H' = \begin{tikzcd}
w_3(x) \arrow[r] & w_3(y) \arrow[r] & c_3 \\
& r_2(y) \arrow[u] \arrow[r] & c_2 \\
r_1(x) \arrow[uu] \arrow[r] & c_1 &
\end{tikzcd}
\quad , \quad
H' = \begin{tikzcd}
r_1(x) \arrow[r] & c_1 \arrow[r] & w_3(x) \arrow[r] & c_3 \\
r_2(y) \arrow[r] & c_2 \arrow[r] & w_3(y) \arrow[ur] &
\end{tikzcd}
$$

Serialization Graph for $H'$ is acyclic implying $H$ is serializable.

In every possible history $H''$ equivalent to $H'$: $T_1$ precedes $T_3$ and $T_2$ precedes $T_3$.

We can construct $H_1$ : $r_1(x)\ c_1\ r_2(y)\ c_2\ w_3(x)\ w_3(y)\ c_3$

- Clearly : $H_1$ is serializable:
  - ↳ $T_1$ & $T_2$ are not interleaved in $H_1$
  - ↳ $T_1$ precedes $T_2$ in $H_1$
  - ↳ $\exists\ H_2$ : $r_2(y)\ c_2\ r_1(x)\ c_1\ w_3(x)\ w_3(y)\ c_3$ which is a serial history equivalent to $H_1$ ($H_2 \equiv H' \equiv H_1$) and in $H_2$, $T_2$ precedes $T_1$.

Therefore; $H_1$ is our required serializable history.

2.10.

If each transaction $T_i$ can read / write a data item more than once, we need to store the access index. We define a superscript for $r$ & $w$ in order to index reads or writes on the same data item.

$o_i^j[x]$ : Transaction $i$'s - $j$'th access for data item '$x$'.

$o$ can be '$r$' or '$w$' - read or write.

Defining Transaction: A transaction $T_i$ is a partial order relation with ordering relation $<_i$ where:

1. $T_i \subseteq \{ r_i^j[x], w_i^j[x] \mid x \text{ is a data item}, j \in \mathbb{N} \}$
$\cup \{a_i, c_i\}$

2. $a_i \in T_i$ iff $c_i \notin T_i$

3. if $t$ is $c_i$ or $a_i$ (whichever in in $T_i$), for any other operation $p \in T_i$, $p <_i t$

4. if $r_i^j[x], w_i^k[x] \in T_i$, then either $r_i^j[x] <_i w_i^k[x]$ or $w_i^k[x] <_i^j r_i[x]$ : $j, k \in \mathbb{N}$.

5. $r_i^j[x] \in T_i \Rightarrow r_i^{j-1}[x] \in T_i$ & $w_i^j[x] \in T_i \Rightarrow w_i^{j-1}[x] \in T_i$ $\forall j \geq 2$

6. $r_i^{j-1}[x] <_i r_i^j[x]$ iff $r_i^j[x] \in T_i$ and
$w_i^{j-1}[x] <_i w_i^j[x]$ iff $w_i^j[x] \in T_i$ $\forall j \geq 2$

- We ensure that if data item $x$ has been accessed (read or write) $N$ number of times in transaction $T_i$: these accesses are ordered in $j$ s.t $o_i^1[x] <_i o_i^2[x] <_i \ldots <_i o_i^N[x]$ - & the ordering is done WLG.

- The condition ensures that $j \in \mathbb{N}$, if $o_i^j[x]$ exists, then $o_i^{j-1}[x]$ also exists & $o_i^{j-1}[x] <_i o_i^j[x]$

- The above conditions ensure a well defined ordering & is both necessary & sufficient.

: Now, we can define: History, Complete History, Equivalence of Histories & serialization graph.

- The definitions for History & Complete History need not change. It only needs to consider the modified definition for $T_i$. if $T = \{T_1, T_2, \dots, T_n\}$ be the set of transactions; each $T_i$ must satisfy our new definition of transaction. The remaining conditions defining the complete history is unchanged, & History is still simply a prefix of a complete history.

We need to update the definition of conflicting & include the clause: two operations accessing the same data item causes a conflict iff atleast one of them is a write, operations can belong to different transactions or even the same transaction.

- This is economical & clarifies point 3 of definition of complete History. Sticking to previous convention, conflicts are considered at the level of history.
- We only need this update in how conflicts are defined : i.e $r_i^j[x]$ & $w_k^l[x]$ are conflicting operations.

• The definitions of equivalence of histories & serialization graph will also be unchanged — with the redefinition of conflicts. Note that the existing definition does not consider conflicts within the transactions & only defines edges for $T_i$ to $T_j$ $(i \neq j)$ — this ensures no loops (self-loops) even in the new setting. Thus, we don't need to change anything and our serialization graph will not have self loops.

→ Serializability Theory: A history H is serializable iff SG(H) is acyclic.

if) Suppose H is a history over $T = \{T_1, T_2, \dots, T_n\}$ : $T_i$ being a transaction according to our definition. WLG, assume $T_1, T_2, \dots, T_m$ $(m \leq n)$ are transactions in T committed in H. $\therefore T_1, T_2, \dots, T_m$ are nodes in SG(H). SG(H) is acyclic → can be topologically sorted. Let $i_1, i_2, \dots, i_m$ be a permutation of $1, 2, \dots, m$ s.t $T_{i_1}, T_{i_2}, \dots, T_{i_m}$ is a topological sort of SG(H). Let $H_S$ be the serial History $T_{i_1}, T_{i_2}, \dots, T_{i_m}$. We claim : $C(H) \equiv H_S$. Let $p_i \in T_i$ & $q_j \in T_j$ : $T_i, T_j$ are committed in H. Suppose : $p_i, q_j$ conflict and $p_i <_H q_j$. By defn of SG(H), $T_i \to T_j$ is an edge in SG(H). So, in any topological sort of SG(H), $T_i$ precedes $T_j$. $\therefore p_i <_{H_S} q_j$. $\therefore C(H) \equiv H_S$. Also $H_S$ is serial by construction $\rightarrow$ H is SR.

only if > Let H be SR and $H_S$ is a serial history s.t $H_S \equiv C(H)$. Consider edge $T_i \rightarrow T_j$ in SG(H). $\exists$ 2 conflicting operations $p_i, q_j$ of $T_i, T_j$ respectively s.t $p_i <_H q_j$. As $C(H) \equiv H_S$, $p_i <_H q_j$. As $H_S$ is serial & $p_i \in T_i$ precedes $q_j \in T_j$; it follows that $T_i$ precedes $T_j$ in $H_S$. $\therefore T_i \rightarrow T_j$ is in $SG(H) \Rightarrow T_i$ precedes $T_j$ in $H_S$. If possible let there be a cycle in SG(H). WLG, let $T_1, T_2, \ldots, T_k \rightarrow T_1$ be the cycle. The edges imply: $T_1$ precedes $T_2$, $T_2$ precedes $T_3, \ldots$, $T_k$ precedes $T_1$. Or $T_1$ precedes $T_1$ in $H_S$ which is impossible. $\therefore \nexists$ any cycle in SG(H).

Note: We updated the definition of conflicts. The idea was simple: have a well defined order within each $T_i$, do not consider self loops in the SG(H). Thus, in $H_S$ & SG(H) - it appears as if within each $T_i$, the reads and writes on each data item is atomic & equivalent to a single read-write or read only or write only.