CS386D Problem Set #6

Consider the following transactions:

```
T1: r1[a]  r1[b]  w1[f]  c1
T2: w2[d]  w2[b]  c2
T3: w3[b]  w3[a]  c3
```

and suppose the DBMS scheduler receives these operations in the following order:

```
r1[a]  w2[d]  r1[b]  w2[b]  w3[b]  c2  w3[a]  w1[f]  c1  c3
```

[1] what is the history graph for this execution?

For each of the following boolean-valued questions, justify your answer:

[2] is this a serial execution?

[3] is this execution equivalent to a serial execution? (If so, what is the schedule?)

[4] is this a strict execution?

[5] if T2 aborts instead of commits, would cascading aborts be needed to affect recovery?

[6] if T1 aborts instead of commits, would cascading aborts be needed to affect recovery?
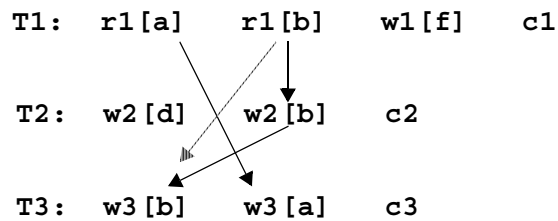
[7] is this a recoverable execution?

Solutions

Consider the following transactions:

```
T1: r1[a] r1[b] w1[f] c1
T2: w2[d] w2[b] c2
T3: w3[b] w3[a] c3
```

and suppose the DBMS scheduler receives these operations in the following order:

```
r1[a] w2[d] r1[b] w2[b] w3[b] c2 w3[a] w1[f] c1 c3
```

[1] what is the history graph for this execution?

```
T1:   r1[a]     r1[b]     w1[f]     c1


T2:   w2[d]     w2[b]      c2


T3:   w3[b]     w3[a]      c3
```

For each of the following boolean-valued questions, justify your answer:

[2] is this a serial execution?

This is not a serial execution. Operations of T1, T2, and T3 are interleaved.

[3] is this execution equivalent to a serial execution? (If so, what is the schedule?)

This execution is equivalent to a serial schedule. There are no cycles in the serialization graph:

```
        _____
       /        \
T1 ──────▶T2 ──────▶ T3
```
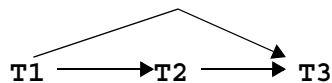
The execution order is T1, T2, T3.

[4] is this a strict execution?

No, this is not a strict execution. A strict execution does not permit transactions to write over uncommitted values. The operation `w3[b]` *overwrites the uncommitted write* `w2[b]`.

[5] if T2 aborts instead of commits, would cascading aborts be needed to affect recovery?

No. Cascading aborts would not be needed. Cascading aborts arise when a transaction reads an uncommitted data item. T1 and T3 do not read any data item that was written by T2.

[6] if T1 aborts instead of commits, would cascading aborts be needed to affect recovery?

No. Cascading aborts would not be needed. Neither T2 or T3 perform reads.

[7] is this a recoverable execution?

This is a recoverable execution. No transaction reads an uncommitted data item. (A recoverable execution means that transactions that read uncommitted data items cannot commit until the transactions that wrote those data items have committed).