Replication and Consistency Models

Data Replication

Replicate data at many nodes

- **Performance**: local reads
- **Reliability**: no data-loss unless data is lost in all replicas
- **Availability**: data available unless all replicas fail or become unreachable
- **Scalability**: balance load across nodes for reads

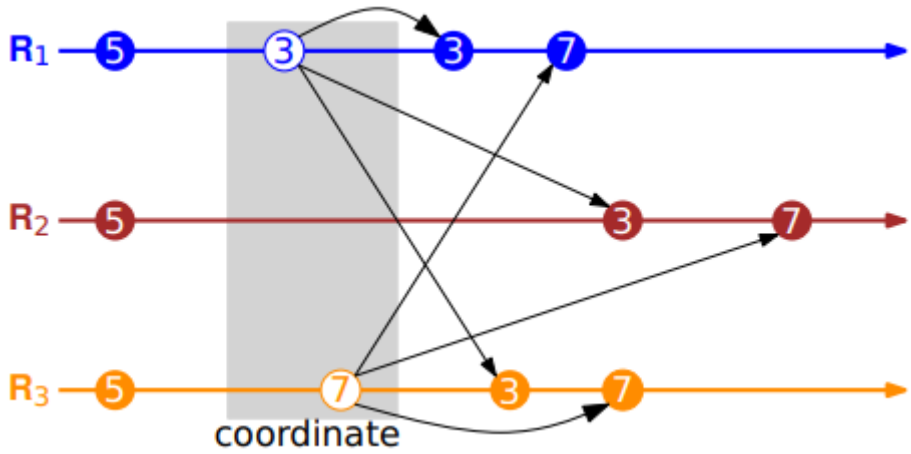
Upon on update, push data to all replicas
Challenge: Ensure data consistency

- Updating at different erplicas may lead to different results e.g. inconsistent data

Strong Consistency

All replicas execute updates in the same order

- same initial state leads to same result



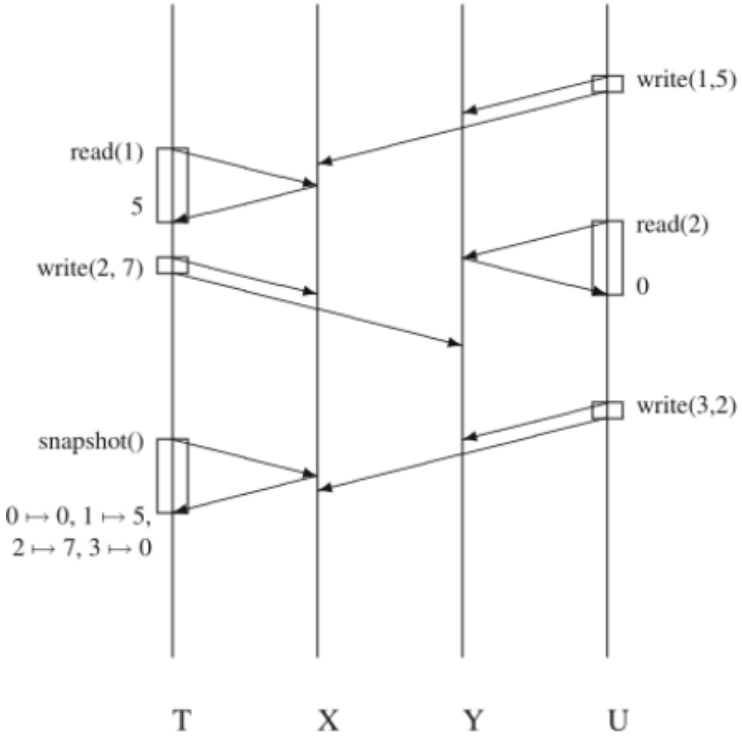
Models

Sequential Consistency

An execution is sequential consistent iff it is identical to a sequential execution of all the operations in that execution such that all operations executed by any thread, appear in the order in which they were executed by the corresponding thread.

- Model provided by a multi-threaded system on a uniprocessor
- Protocol
    - **Read**: reads from one replica

    - **Write**: writes to all replicas in same order; have no reply → return after sending the write request messages to all replicas
    - **Snapshot**: reads from one replica
- Is not Composable
    - the same algorithm to replicate each of the sub-arrays, and thus ensure sequential consistency on each array
    - The combined execution may not be sequential consistent



1. *write*(1, 5)
2. *read*(1)
3. *read*(2)
4. *write*(2, 7)
5. *snapshot*()
6. *write*(3, 2)

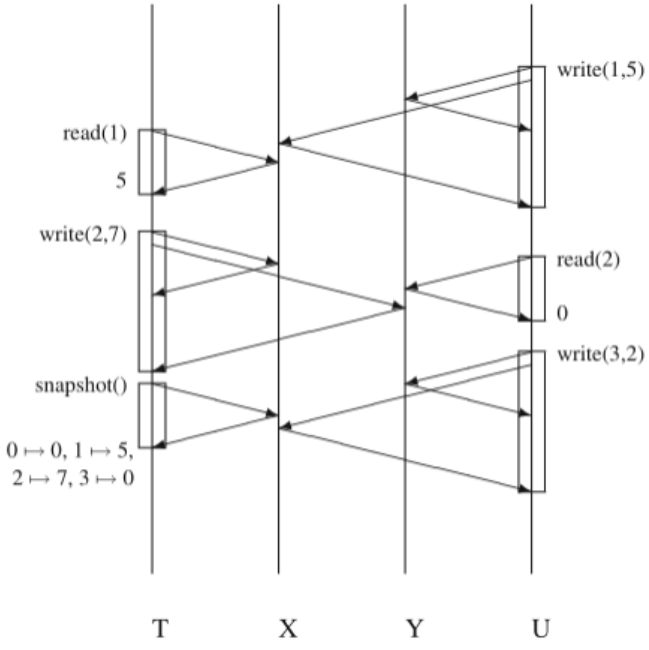
What other ordering would be possible?

Linearizability

An execution is linerizable iff it is sequential consistent and if op1 occurs before op2, according to one omnisciente observer, then op1 appears before op2.

- Guaranteeing linearizability usually requires more synchronization
- Protocol
  - **Read**: reads from one replica
  - **Write**: writes to all replicas in same order; **Wait for ack from all replicas before returning**

- **Snapshot**: reads from one replica



1. *write*(1, 5)
2. *read*(1)
3. *read*(2)
4. *write*(2, 7)
5. *snapshot*()
6. *write*(3, 2)

Is the other sequentially consistent order also linearizable?

src: Fekete and Ramamritham 09

One-copy Serializability (Transaction-based Systems)

The execution of a set of transactions is one-copy serializable iff its outcome is similar to the execution of those transactions in a single copy

- it's essentially the sequential consistency model, when the operations executed by all processor are transactions
  - isolation: ensures that the outcome of the concurrent eecution of a set of transactions is equal to some sequential execution of those transactions
- **Serializability**: databases (preserve complex application-specific invariants)
- **Sequential Consistency**: multiprocesing (programmers are expected to reason about concurrency)