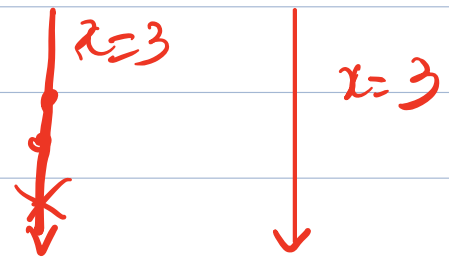
Agenda

- Reasons to do replication
  - Strong consistency (informally)
  - Primary backup replication
  - Chain replication
  - Latency & Throughput
  - Exam Review

Reasons to do replication

- Mitigating data loss
  i.e fault tolerance



- Data locality
  - Having data close to the clients that need it.

- dividing up the work

Downsides of doing replication:

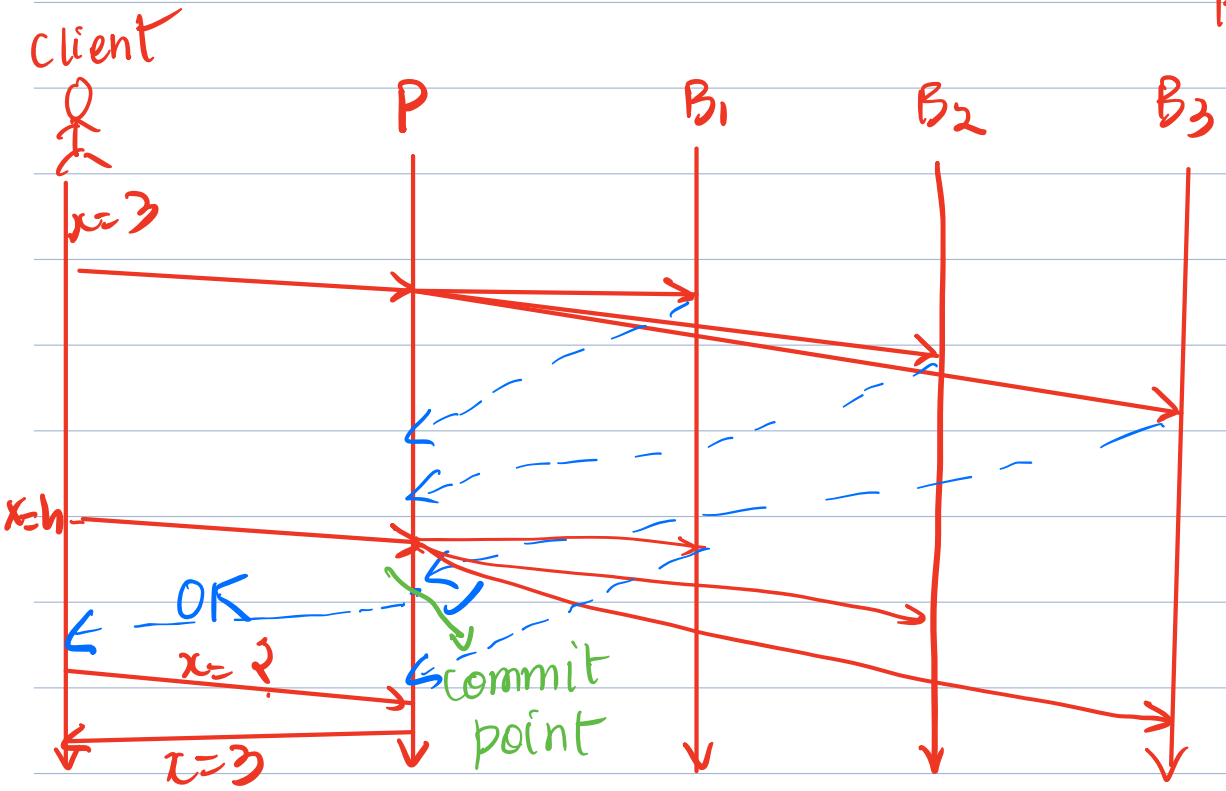
- Expensive
- Have to keep replicas consistent (often!)

Strong consistency (informally)

Definition:

Replicated storage system is strongly consistent if clients cannot tell that the data is replicated

Primary backup replication (strongly consistent protocol)



✓ Fault tolerance: Doing very well

x Data locality: } Not so good.

x Dividing up the work: }

Above scenario:

- Client only interacts with primary
- Suppose system is K-V store. Client wants

to write $x=3$

- Send request to P. P will broadcast request to all backups.
- Backups will write value $x=3$ and send ack (blue line)
- Once P receives ack from ALL backups, that point is called commit point.
- Now, P can write $x=3$ to its own DB and send ACK to client

For reading:

- client requests value of $x=?$ from primary
- Primary responds with value $x=3$ to client request.

Why can't we read from backups?

- Inconsistency!

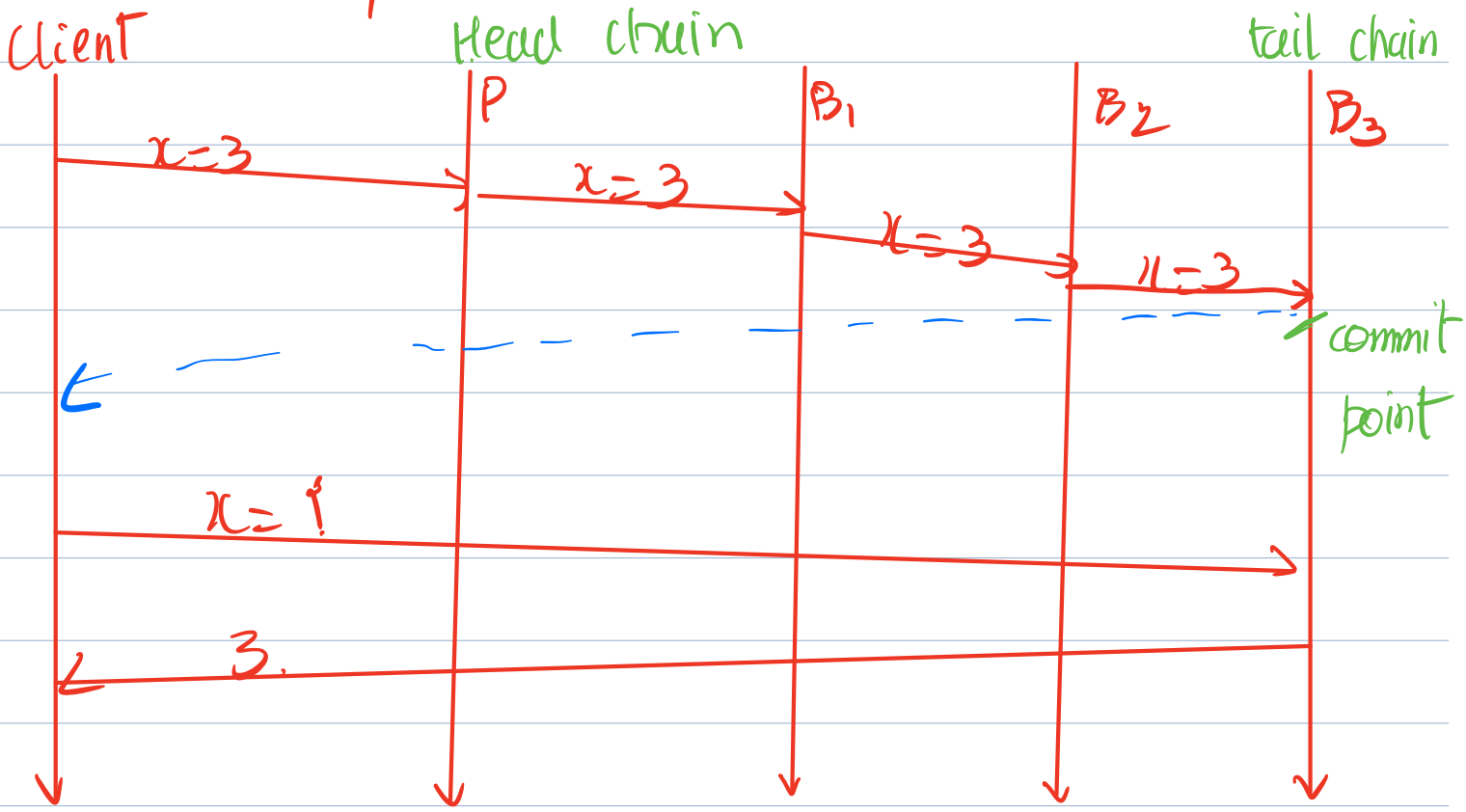
- Suppose after $x=3$, we send request for $x=4$
- B_1 receives broadcast from P, writes data & sends ack.

- But P has not received ACKs from all other backups, so it has not reached commit point!
- Inconsistent data.

How to improve on this protocol?

Chain Replication

- Have one backup serve reads. Make it ACK to primary when write is complete.



✓ Fault tolerance: Good

😐 Data locality: Not great

🙂 Dividing up the work: Little better

Paper Name: Chain replication for supporting High Throughput & Availability
Van Renesse & Schneider (2004)

Throughput

- Number of actions per unit time

Depending on workload (mix of writes/reads)

CR can give you better throughput than PB.
CR > PB.

If workload is only reads?
CR = PB

If all writes?
CR = PB

50-50 split btw read & writes is not ideal for CR since writes are more expensive

For CR, 15% writes, 85% reads — More or less optimal.

Downsides of chain replication?

Latency: time between start & end of one action.

- For read latency, CR = PB
- For writes: In PB, parallel processing since broadcast sent from primary at the same time
  In CR, msg sends one by one in the chain.
  adding more servers increases latency!

∴ CR has worse write latency than PB depending on chain length.

Midterm review!