

Consistency Models



COS 418: Distributed Systems

Lecture 15

Wyatt Lloyd

Consistency Models

- Contract between a distributed system and the applications that run on it
- A consistency model is a set of **guarantees** made by the distributed system

Linearizability

- All replicas execute operations in **some** total order
- That total order preserves the **real-time ordering** between operations
 - If operation A **completes** before operation B **begins**, then A is ordered before B in real-time
 - If neither A nor B completes before the other begins, then there is no real-time order
 - (But there must be *some* total order)

Real-Time Ordering Examples

$P_A \vdash w(x=1) \dashv$

$P_B \quad \vdash w(x=2) \dashv$

$P_C \quad \quad \vdash w(x=3) \dashv$

$P_D \quad \quad \vdash w(x=4) \dashv \quad \vdash w(x=5) \dashv$

$P_E \quad \quad \quad \vdash w(x=6) \dashv$

Linearizable?

$P_A \vdash w(x=1) \dashv$

$P_B \quad \vdash w(x=2) \dashv$

$P_C \quad \quad \quad \vdash w(x=3) \dashv$

$P_D \quad \quad \quad \vdash w(x=4) \dashv \quad \vdash w(x=5) \dashv$

$P_E \quad \quad \quad \quad \quad \quad \vdash w(x=6) \dashv$

$P_F \vdash r(x)=1 \dashv \dashv r(x)=2 \dashv \dashv r(x)=3 \dashv \dashv r(x)=6 \dashv \dashv r(x)=5 \dashv \quad \checkmark$

Linearizable?

$P_A \vdash w(x=1) \dashv$

$P_B \quad \vdash w(x=2) \dashv$

$P_C \quad \quad \quad \vdash w(x=3) \dashv$

$P_D \quad \quad \quad \vdash w(x=4) \dashv \quad \vdash w(x=5) \dashv$

$P_E \quad \quad \quad \quad \quad \vdash w(x=6) \dashv$

$P_G \vdash r(x)=1 \dashv \dashv r(x)=2 \dashv \dashv r(x)=5 \dashv \dashv r(x)=6 \dashv \dashv r(x)=5 \dashv \quad X$

Linearizable?

$P_A \vdash w(x=1) \dashv$

$P_B \quad \vdash w(x=2) \dashv$

$P_C \quad \quad \quad \vdash w(x=3) \dashv$

$P_D \quad \quad \quad \vdash w(x=4) \dashv \quad \vdash w(x=5) \dashv$

$P_E \quad \quad \quad \quad \quad \quad \vdash w(x=6) \dashv$

$P_H \vdash r(x)=1 \dashv \dashv r(x)=4 \dashv \dashv r(x)=2 \dashv \dashv r(x)=3 \dashv \dashv r(x)=6 \dashv \quad \checkmark$

Linearizable?

$P_A \vdash w(x=1) \dashv$

$P_B \quad \vdash w(x=2) \dashv$

$P_C \quad \quad \quad \vdash w(x=3) \dashv$

$P_D \quad \quad \quad \vdash w(x=4) \dashv \quad \vdash w(x=5) \dashv$

$P_E \quad \quad \quad \quad \quad \quad \vdash w(x=6) \dashv$

$P_I \vdash r(x)=1 \dashv \dashv r(x)=4 \dashv \dashv r(x)=5 \dashv \dashv r(x)=6 \dashv \dashv r(x)=3 \dashv \quad \mathbf{X}$

Linearizability ==

“Appears to be a Single Machine”

- Single machine processes requests one by one in the order it receives them
 - Will receive requests ordered by real-time in that order
 - Will receive all requests in some order
- Atomic Multicast, Viewstamped Replication, Paxos, and RAFT provide Linearizability
- Single machine processing incoming requests one at a time also provide Linearizability 😊

Linearizability is Ideal?

- Hides the complexity of the underlying distributed system from applications!
 - Easier to write applications
 - Easier to write correct applications
- But, performance trade-offs

Stronger vs Weaker Consistency

- **Stronger consistency models**
 - + Easier to write applications
 - More guarantees for the system to ensure
Results in performance tradeoffs
- **Weaker consistency models**
 - Harder to write applications
 - + Fewer guarantees for the system to ensure

Strictly Stronger Consistency

- A consistency model A is strictly stronger than B if it allows a strict subset of the behaviors of B
 - Guarantees are strictly stronger

Sequential Consistency

- All replicas execute operations in **some** total order
- That total order preserves the **process ordering** between operations
 - If process P issues operation A before operation B, then A is order before B by the process order
 - If operations A and B and done by different processes then there is no process order between them
 - (But there must be *some* total order)

Sequential Consistency \approx “Appears to be a Single Machine”

- Single machine processes requests one by one in the order it receives them
 - Will receive requests ordered by process order in that order
 - Will receive all requests in some order

Linearizability is strictly stronger than Sequential Consistency

- Linearizability: \exists total order + real-time ordering
- Sequential: \exists total order + process ordering
 - Process ordering \subseteq Real-time ordering

Consistency Hierarchy

Linearizability

e.g., RAFT



Sequential Consistency



Causal+ Consistency

e.g., Bayou



Eventual Consistency

e.g., Dynamo

Causal+ Consistency

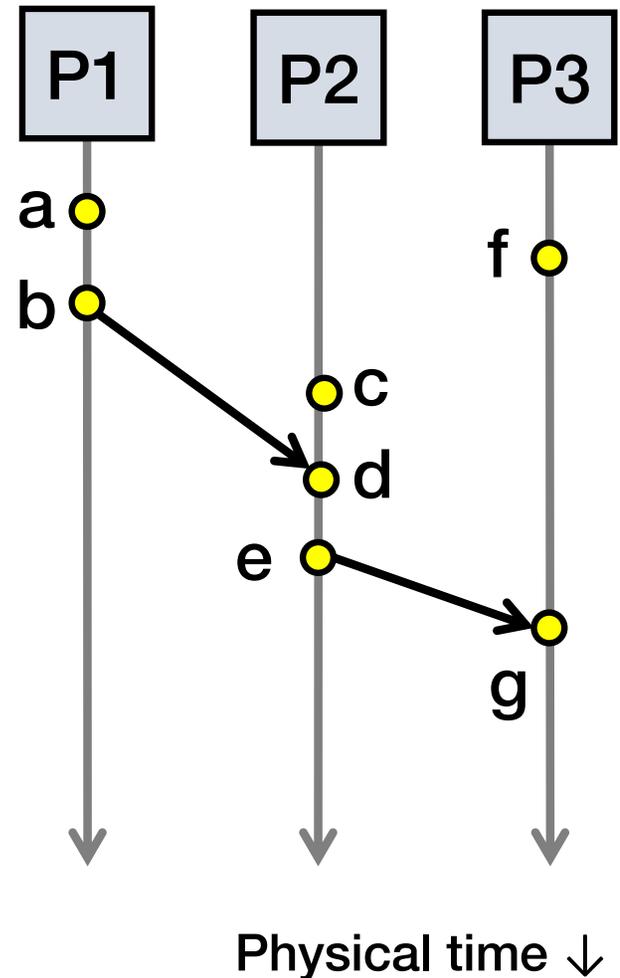
- Partially orders all operations, does not totally order them
 - Does not look like a single machine
- Guarantees
 - For each process, \exists an order of all writes + that process's reads
 - Order respects the happens-before (\rightarrow) ordering of operations
 - + replicas converge to the same state
 - Skip details, makes it stronger than eventual consistency

Causal Consistency

1. Writes that are **potentially** causally related must be seen by all processes in same order.
 2. Concurrent writes may be seen in a different order on different processes.
- Concurrent: Ops not causally related

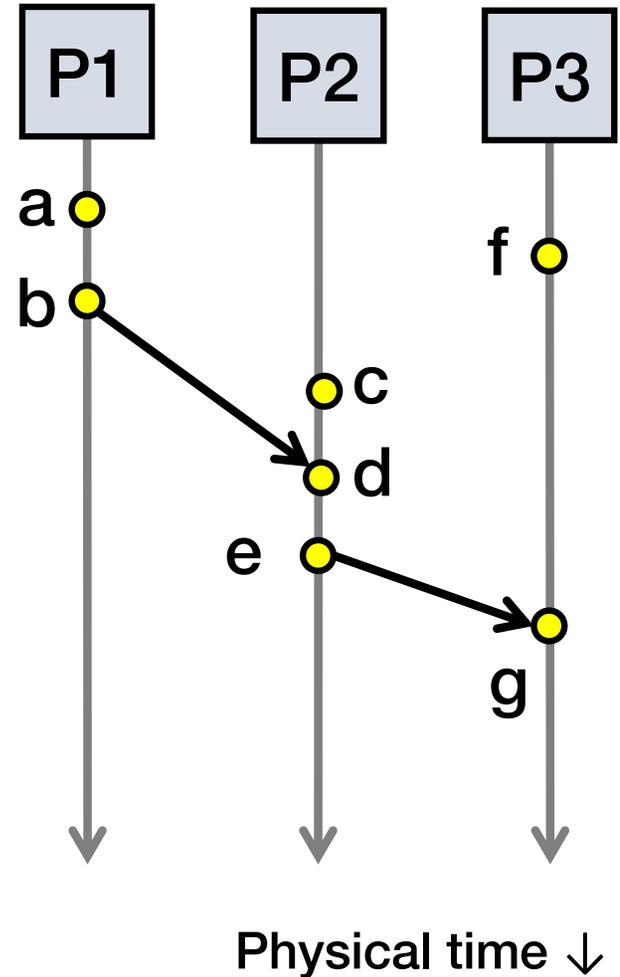
Causal Consistency

1. Writes that are **potentially** causally related must be seen by all processes in same order.
 2. Concurrent writes may be seen in a different order on different processes.
- Concurrent: Ops not causally related



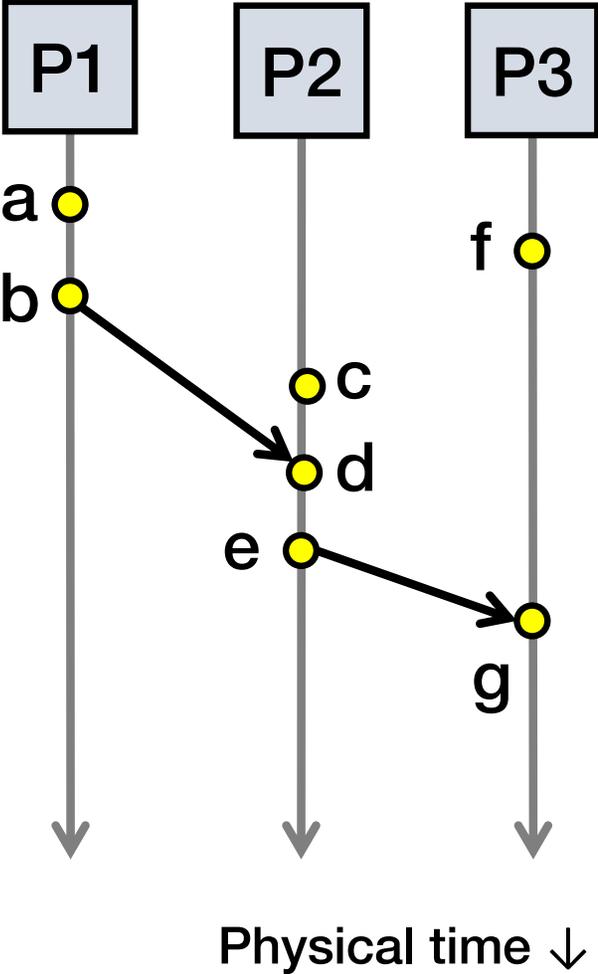
Causal Consistency

Operations	Concurrent?
a, b	
b, f	
c, f	
e, f	
e, g	
a, c	
a, e	



Causal Consistency

Operations	Concurrent?
a, b	N
b, f	Y
c, f	Y
e, f	Y
e, g	N
a, c	Y
a, e	N



Causal+ But Not Sequential

$P_A \vdash w(x=1) \dashv \vdash \vdash r(y)=0 \dashv \vdash$

$P_B \vdash w(y=1) \dashv \vdash \vdash r(x)=0 \dashv \vdash$

✓ Casual+

✗ Sequential

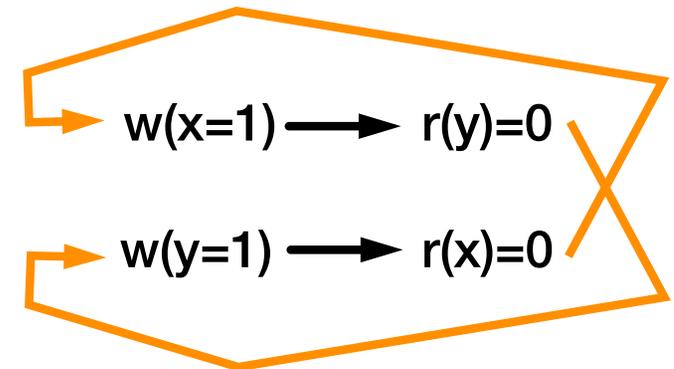
Happens Before Order
 $w(x=1) \longrightarrow r(y)=0$
 $w(y=1) \longrightarrow r(x)=0$

Process Ordering
 $w(x=1) \longrightarrow r(y)=0$
 $w(y=1) \longrightarrow r(x)=0$

P_A Order: $w(x=1), r(y=0), w(y=1)$

P_B Order: $w(y=1), r(x=0), w(x=1)$

No Total Order



Eventual But Not Causal+

$P_A \vdash w(x=1) \dashv \vdash \vdash w(y)=1 \dashv \vdash$

P_B

$\vdash r(y)=1 \dashv \vdash \vdash r(x)=0 \dashv \vdash$

✓ Eventual

As long as P_B eventually would see $r(x)=1$ this is fine

✗ Causal+

Happens Before Ordering

$w(x=1) \longrightarrow w(y)=1$
 $r(y)=1 \longrightarrow r(x)=0$

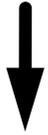
No Order for P_B

$w(x=1) \longrightarrow w(y)=1$
 $r(y)=1 \longrightarrow r(x)=0$

Consistency Hierarchy

Linearizability

e.g., RAFT



Sequential Consistency



Causal+ Consistency

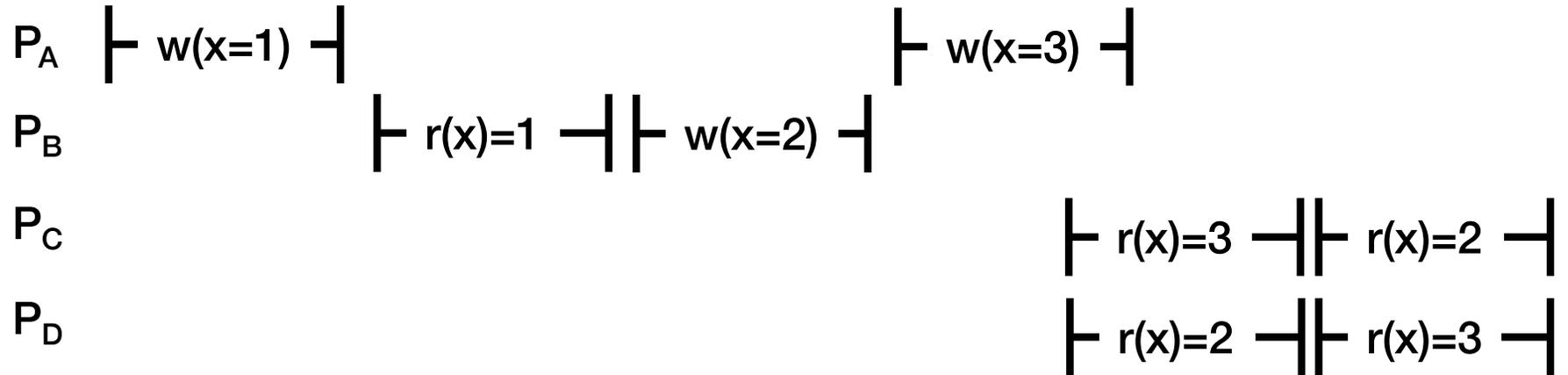
e.g., Bayou



Eventual Consistency

e.g., Dynamo

Causal Consistency: Quiz



- Valid under causal consistency
- **Why?** $x=3$ and $x=2$ are concurrent
 - So all processes don't (need to) see them in same order
- P_C and P_D read the values '1' and '2' in order as potentially causally related. No 'causality' for '3'.

Causal Consistency

$P_A \vdash w(x=1) \dashv$

$P_B \vdash r(x)=1 \dashv \dashv \vdash w(x=2) \dashv$

$P_C \vdash r(x)=2 \dashv \dashv \vdash r(x)=1 \dashv$

$P_D \vdash r(x)=1 \dashv \dashv \vdash r(x)=2 \dashv$

X $x=2$ happens after $x=1$

Causal Consistency

$P_A \vdash w(x=a) \dashv$

$P_B \vdash w(x=b) \dashv$

$P_C \vdash r(x)=b \dashv \dashv \vdash r(x)=a \dashv$

$P_D \vdash r(x)=a \dashv \dashv \vdash r(x)=b \dashv$

✓ P_B doesn't read value of 1 before writing 2

