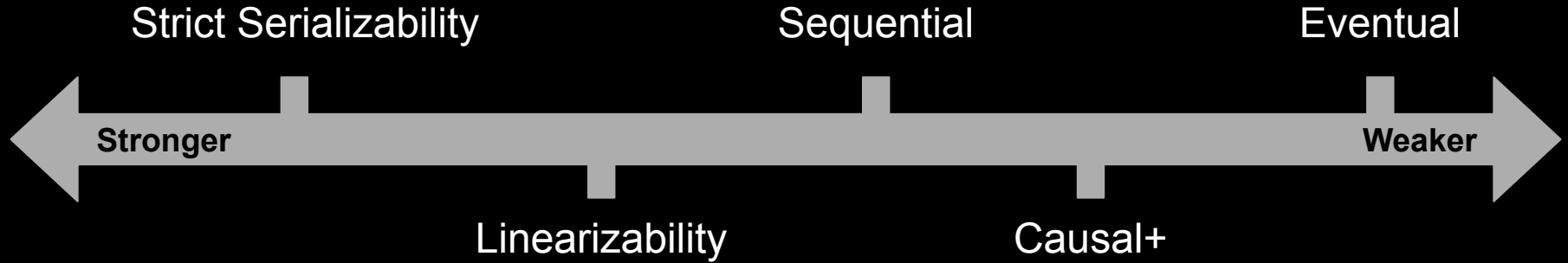


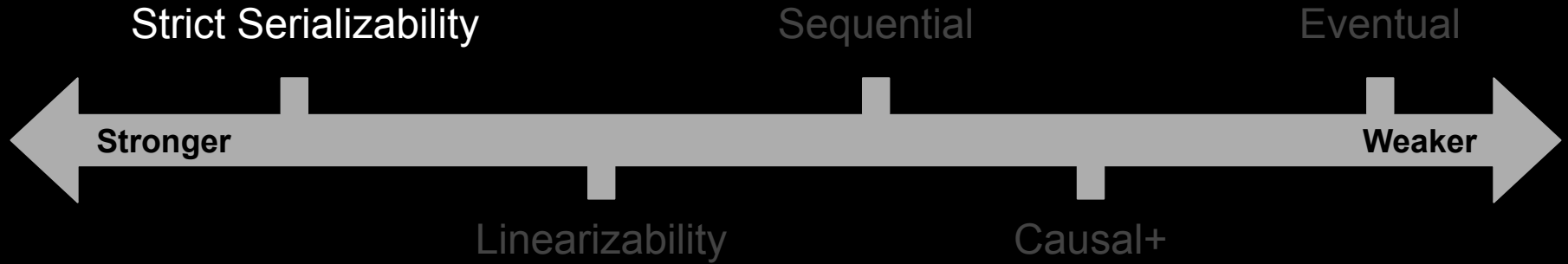
# Consistency

Nov 10, 2022

# Consistency Models



# Consistency Models



# Strict Serializability

- **Transactions**: operations that span multiple objects (e.g., keys in KV store) *atomically* commit (or abort).
- **Total order**: There exists some legal total ordering of transactions.
  - Legal (intuitively defined for strict serializability): in the total ordering, read operations “see” the latest write operation.
- Preserves **real-time commit order**: if *txn A* commits before *txn B* begins, then *txn A* occurs before *txn B* in the total order.
  - Write ops in a committed txn are visible to all future txns’ read ops.
  - Intuition: once a read “sees” a txn and commits, all future reads must also “see” that txn.

**Pros**: applications can easily reason about correctness of transactions.

**Cons**: strict serializability imposes high read and write latencies on system.

# Strict Serializability Example

**Strictly Serializable?** **Yes**

P1: {W(x)b, W(y)b}

P2: {W(x)a}

P3: {R(x)a} {R(x)b}

P4: {R(x)b} {R(y)b}

**Strictly Serializable?** **No**

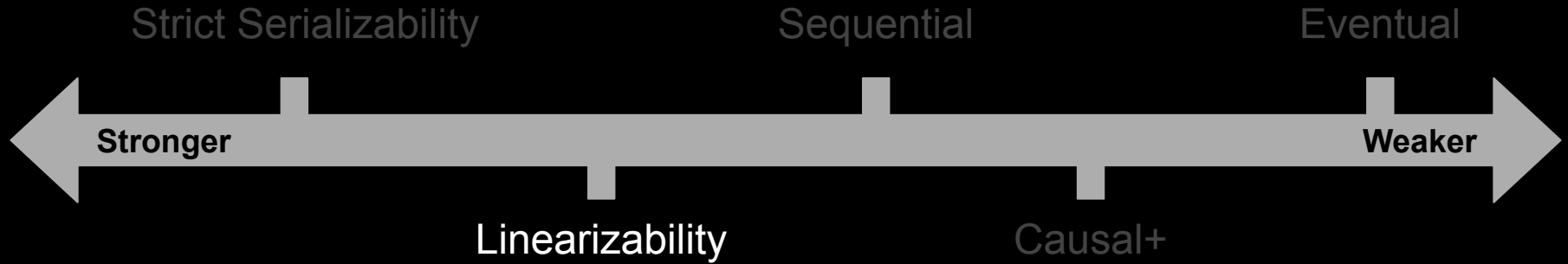
P1: {W(x)b, W(y)b}

P2: {W(x)a}

P3: {R(y)b} {R(x)a}

P4: {R(x)b} {R(y)b}

# Consistency Models



# Linearizability

- **Total order**: There exists some legal total order of **operations (not txns)**.
- Difference from *strict serializability*?
  - Single-object operations! No transactions!
- **Preserves real-time ordering**: if an operation  $A$  completes before operation  $B$  begins, then op  $A$  occurs before op  $B$  in the total order.
  - A completed write op is visible to all future read ops.
  - Intuition: once a read “sees” a new write, all future reads must also “see” that write.

**Pros:** Easy to reason about correctness

**Cons:** High read and write latencies

# Linearizability Example

**Linearizable?**

**No**

P1: W(x)a

P2: W(x)b

P3: R(x)b R(x)a

P4: R(x)b R(x)a

**Linearizable?**

**Yes**

P1: W(x)a

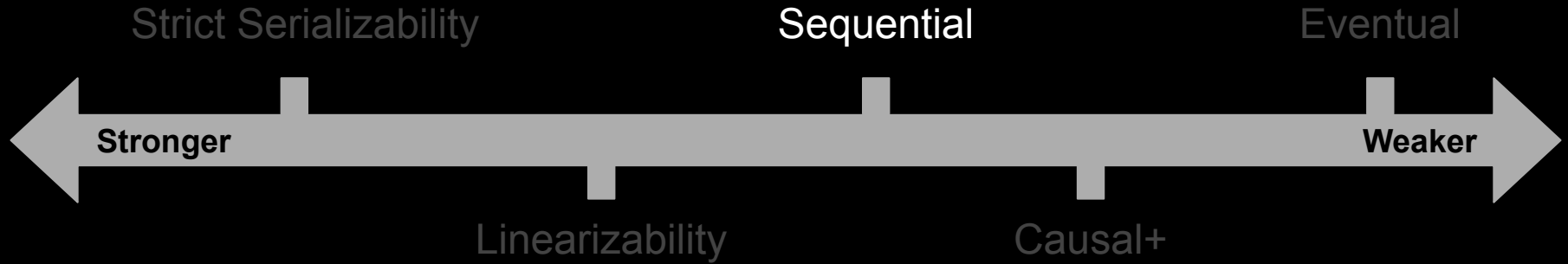
P2: W(x)b

P3: R(x)a R(x)b

P4: R(x)a R(x)b



# Consistency Models



# Sequential Consistency

- **Total order**: there exists some legal total order of operations.
- **Preserves process ordering**: total order respects order of each process's operations.
- Difference from *linearizability*?
  - Order of ops across processes not determined by real-time

**Pros:** Can allow more orderings than linearizability → better performance.

**Cons:** Many possible sequential executions → increased application complexity

# Sequential Consistency Example

**Sequentially Consistent?** **Yes**

P1: W(x)a

P2: W(x)b

P3: R(x)b R(x)a

P4: R(x)b R(x)a

**Sequentially Consistent?** **No**

P1: W(x)a

P2: W(x)b

P3: R(x)b R(x)a

P4: R(x)a R(x)b

# Consistency Models



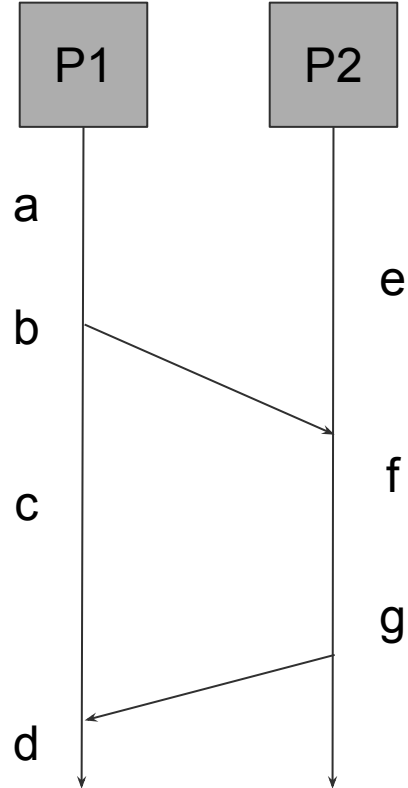
# Causal+ Consistency

- **Partial order**: order causally related ops the same way across all processes
- **+**: replicas' total order eventually converge.
- Difference from *sequential consistency*?
  - Only causally related ops need to be ordered: **no guaranteed total order.**
  - Concurrent ops may be ordered differently across different processes.

**Pros:** preserves causality while improving efficiency.

**Cons:** harder to reason about concurrency.

Ops	Concurrent
a,b	<b>No</b>
a,e	<b>Yes</b>
a,g	<b>No</b>
c,e	<b>Yes</b>
c,d	<b>No</b>
d,g	<b>No</b>
d,f	<b>No</b>
e,g	<b>No</b>
a,d	<b>No</b>



# Causal+ Consistency Example

**Causally+ Consistent? Yes**

P1: W(x)a  
P2: W(x)b  
P3: R(x)b R(x)a  
P4: R(x)a

**Causally+ Consistent? No**

P1: W(x)a  
P2: R(x)a W(x)b  
P3: R(x)b R(x)a  
P4: R(x)a

# Consistency Models





# Eventual Consistency

- **Eventual convergence**: If no more writes, all replicas *eventually* agree.
- Difference from *causal consistency*?
  - Does not preserve causal relationships
  - Is the “+” in causal+.
- Frequently used with application conflict resolution, anti-entropy

**Pros:** highly available; think Bayou.

**Cons:** no safety guarantees, need conflict resolution

# In a nutshell...

**Strict Serializability**: total order + real time guarantees over *transactions*

**Linearizability**: total order + real time guarantees over *operations*

**Sequential consistency**: total order + process order

**Causal+ consistency**: causally ordered + replicas eventually converge

**Eventual consistency**: tventually, everyone should agree on state

# Exercise 1:

P1: {W(x) 1, W(y) 2}                      {R(y) 4}

P2:                      {W(x) 1, R(y) 4}

P3:                      {W(x) 0, W(y) 4}

P4:                      {R(x) 0}                      {R(x) 1}

## Consistency Model:

Strictly Serializable **Yes**

Linearizable **Yes**

Sequential **Yes**

Causal+ **Yes**

Eventual **Yes**

# Exercise 2:

P1: W(x) 1 R(y) 4  
P2: R(x) 1 R(y) 4  
P3: R(x) 1 W(y) 4  
P4: R(x) 1 R(y) 4

## Consistency Model:

Linearizable **Yes**

Sequential **Yes**

Causal+ **Yes**

Eventual **Yes**

# Exercise 3:

P1: W(x) 3

W(y) 7

P2: W(x) 1

P3:

R(x) 1

R(x) 3

R(y) 7

P4:

R(x) 1

R(x) 3

R(y) 7

P5:

R(x) 1

R(x) 3

R(y) 7

## Consistency Model:

Linearizable **No**

Sequential **Yes**

Causal+ **Yes**

Eventual **Yes**

# Exercise 4:

P1: W(x) 3

W(y) 7

P2: W(x) 1

P3: R(x) 1 R(x) 3

R(y) 7

P4: R(x) 3 R(x) 1

R(y) 7

P5: R(x) 1 R(x) 3

R(y) 7

## Consistency Model:

Linearizable **No**

Sequential **No**

Causal+ **Yes**

Eventual **Yes**

# Exercise 5:

P1: W(x) 1

P2: W(x) 3

P3: W(x) 7

P4: R(x) 3 R(x) 7 R(x) 1

P5: R(x) 3 R(x) 1 R(x) 7

## Consistency Model:

Linearizable **No**

Sequential **No**

Causal+ **Yes**

Eventual **Yes**

# Exercise 6:

P1: W(x) 1

P2: W(x) 3

P3: R(x) 3 W(x) 7

P4: R(x) 3 R(x) 7 R(x) 1

P5: R(x) 3 R(x) 1 R(x) 7

## Consistency Model:

Linearizable **No**

Sequential **No**

Causal+ **Yes**

Eventual **Yes**



# Exercise 7:

P1: W(x) 1

P2: R(x) 1 W(x) 3

P3: R(x) 3 W(x) 7

P4: R(x) 3 R(x) 7 R(x) 1

P5: R(x) 3 R(x) 1 R(x) 7

## Consistency Model:

Linearizable **No**

Sequential **No**

Causal+ **No**

Eventual **Yes**