COL380

Introduction to
Parallel & Distributed Programming

Agenda

- Understanding memory consistency

Memory Model

@ atom
Address space

Memory Model

- Register size?
- Limit on the number of readers and writers?
- Asynchronous reads and writes
  - ➡ Global knowledge of ‘time?’

Memory Model

- Register size?
- Limit on the number of readers and writers?
- Asynchronous reads and writes
  - ➡ Global knowledge of ‘time?’

✳ What is the value read? (Correctness)

Memory Model

- Register size?
- Limit on the number of readers and writers?
- Asynchronous reads and writes
  - ➡ Global knowledge of ‘time?’

Register

✳ What is the value read? (Correctness)

“Most recent Write”

Memory Model

- Register size?
- Limit on the number of readers and writers?
- Asynchronous reads and writes
    - ➡ Global knowledge of 'time?'

✴ What is the value read? (Correctness)

- Increasing power ⇒ reducing performance

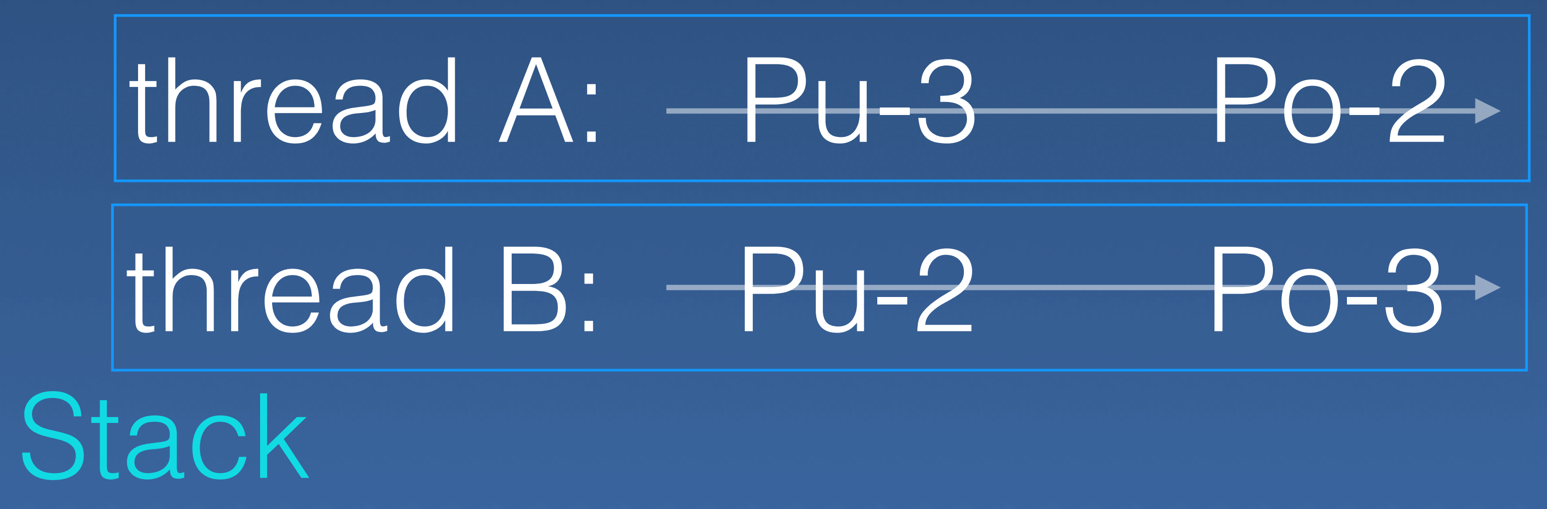
Memory Model

- Register size?
- Limit on the number of readers and writers?
- Asynchronous reads and writes
  - ➡ Global knowledge of ‘time?’

✳ What is the value read? (Correctness)

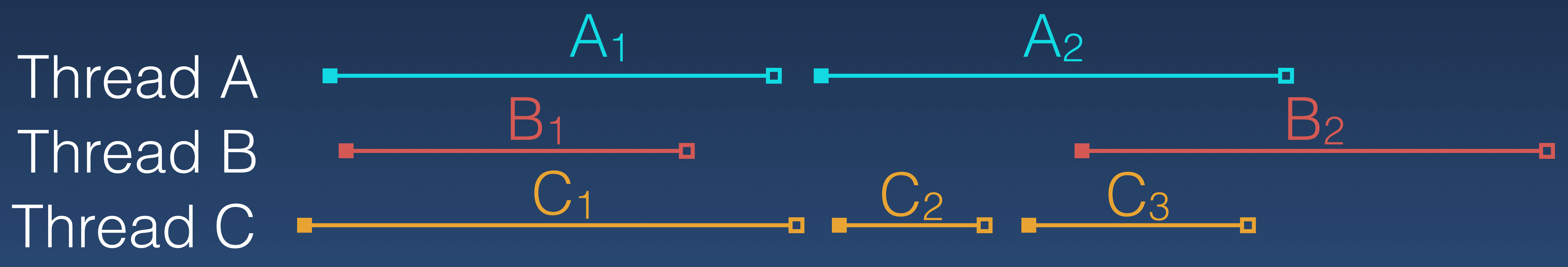
- Increasing power ⇒ reducing performance

Same ideas apply to higher level data structures



Linearizable

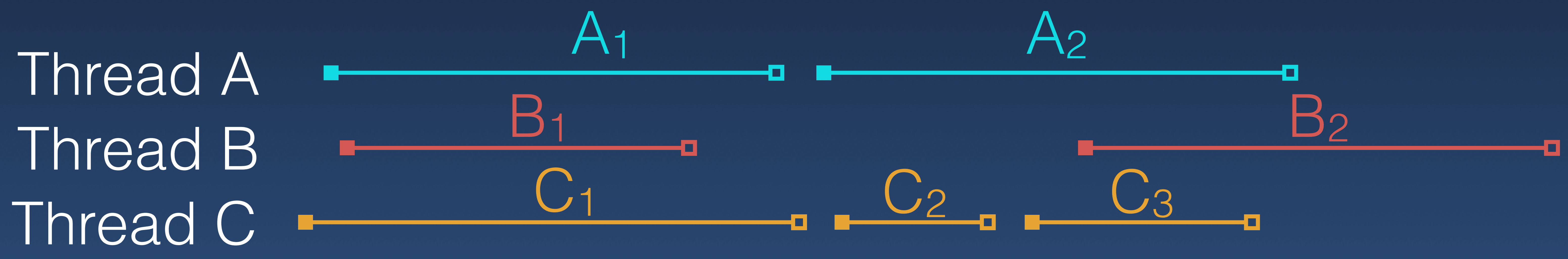
- Operations appear to take effect at some instant
  - ➡ between their start and end
  - ➡ If there is no overlap, ordering is well defined



Linearizable

- Operations appear to take effect at some instant
  - ➡ between their start and end
  - ➡ If there is no overlap, ordering is well defined

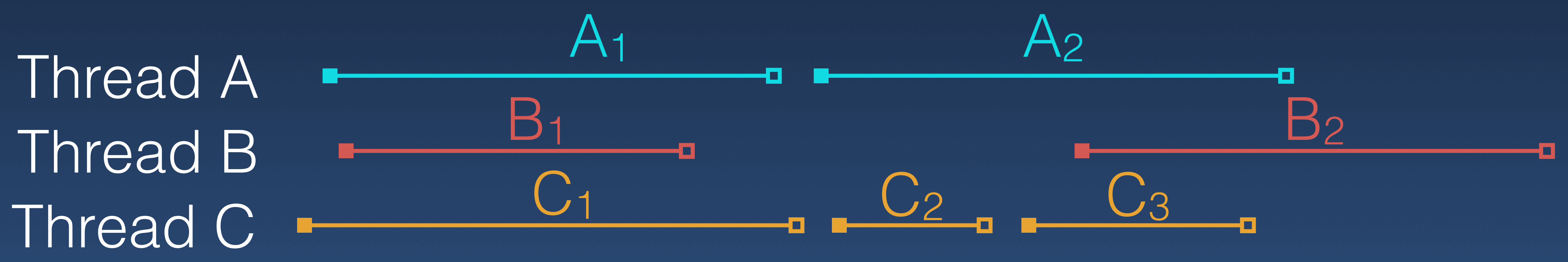
Strict Consistency:
Operations are instantaneous



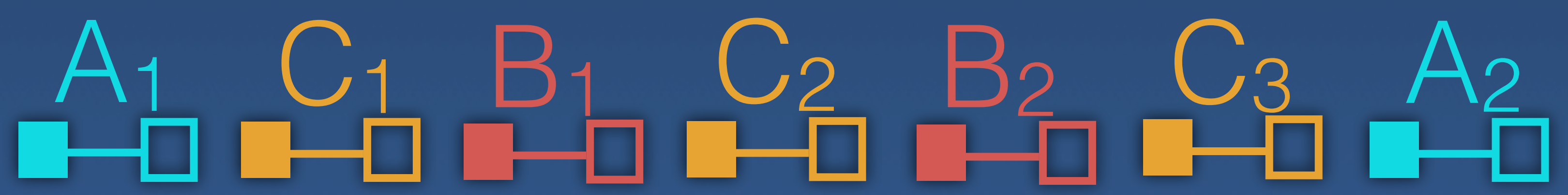
Linearizable

- Operations appear to take effect at some instant
  - ➡ between their start and end
  - ➡ If there is no overlap, ordering is well defined

Strict Consistency:
Operations are instantaneous



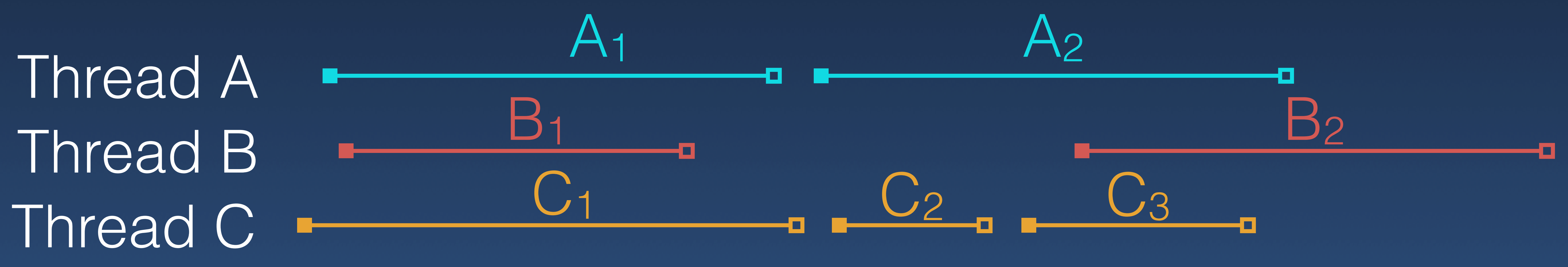
Equivalent sequential history S exists:



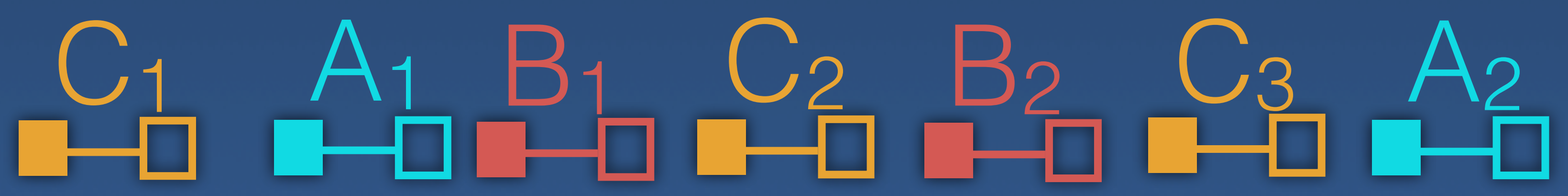
Linearizable

- Operations appear to take effect at some instant
  - ➡ between their start and end
  - ➡ If there is no overlap, ordering is well defined

Strict Consistency:
Operations are instantaneous



Equivalent sequential history S exists:

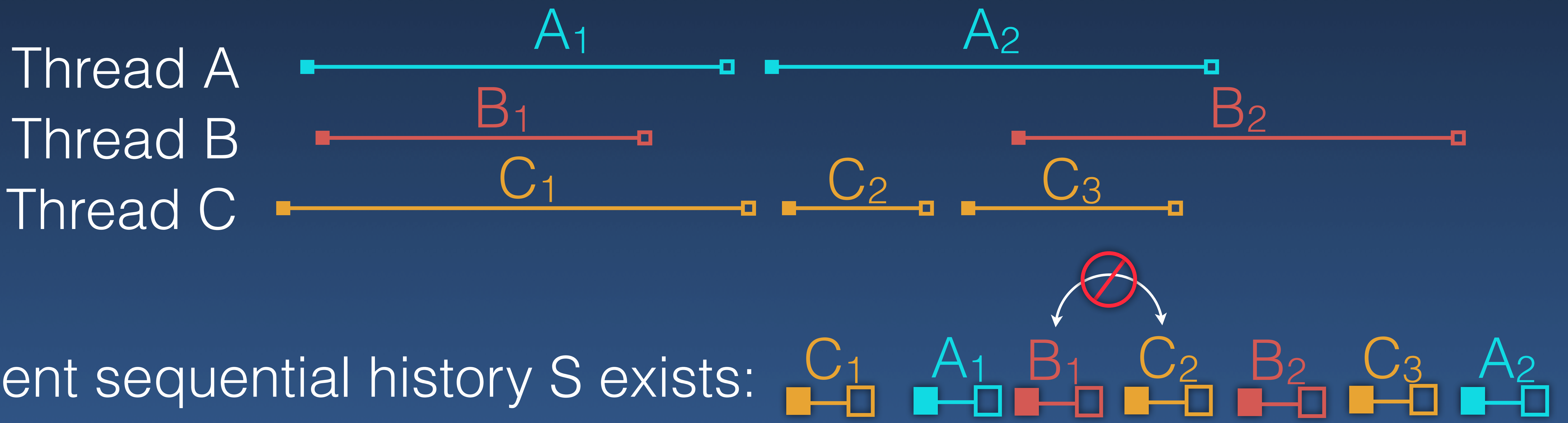


Subodh Kumar

Linearizable

- Operations appear to take effect at some instant
  - ➡ between their start and end
  - ➡ If there is no overlap, ordering is well defined

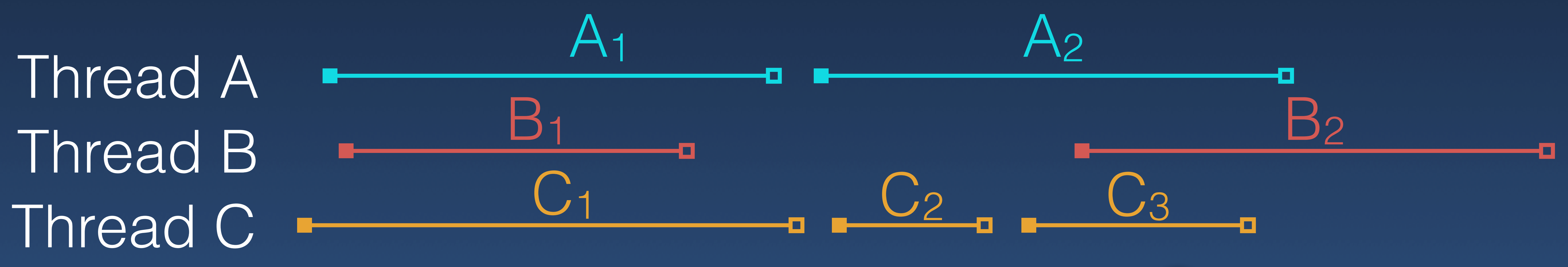
Strict Consistency:
Operations are instantaneous


Thread A
Thread B
Thread C
A_1
A_2
B_1
B_2
C_1
C_2
C_3
Equivalent sequential history S exists:
C_1 A_1 B_1 C_2 B_2 C_3 A_2

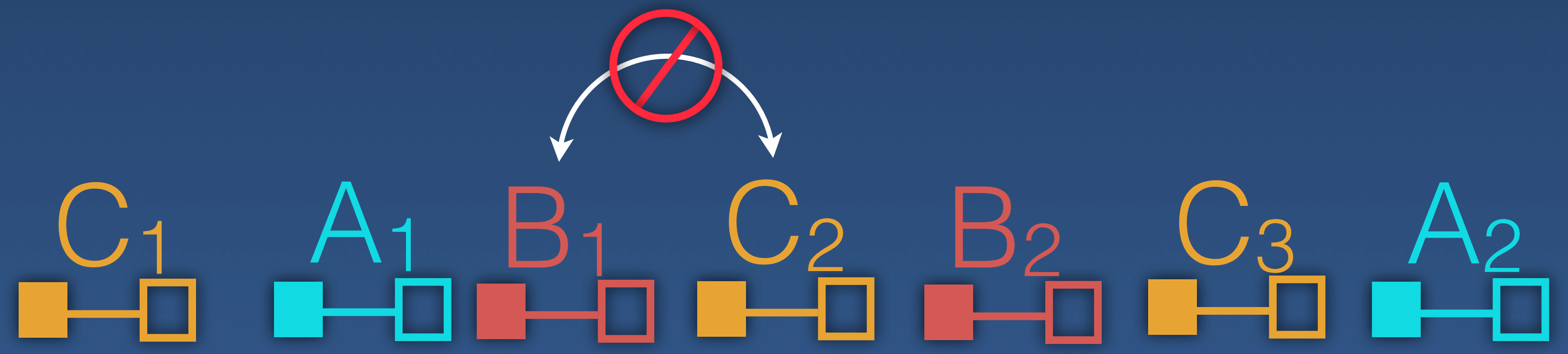
Linearizable

- Operations appear to take effect at some instant
  - ➡ between their start and end
  - ➡ If there is no overlap, ordering is well defined

Strict Consistency:
Operations are instantaneous



Equivalent sequential history S exists:

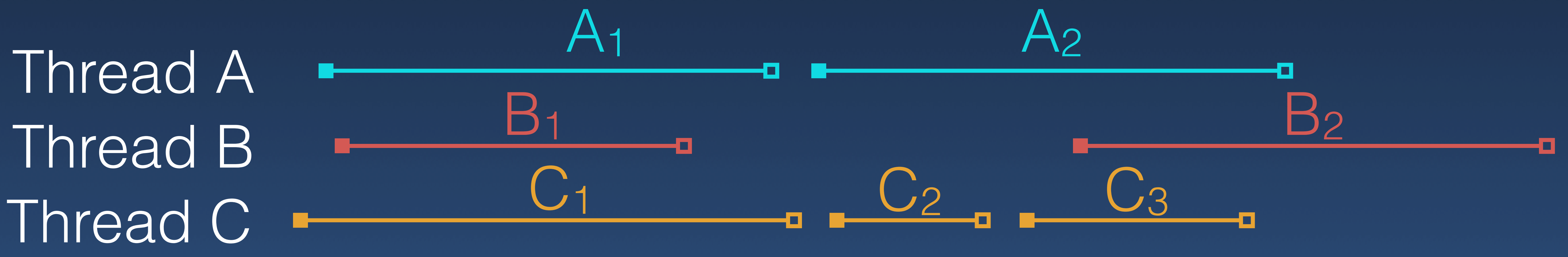


1. Got the result S would get
2. No thread's history is violated in S
3. Non-overlapping operations retain order in S

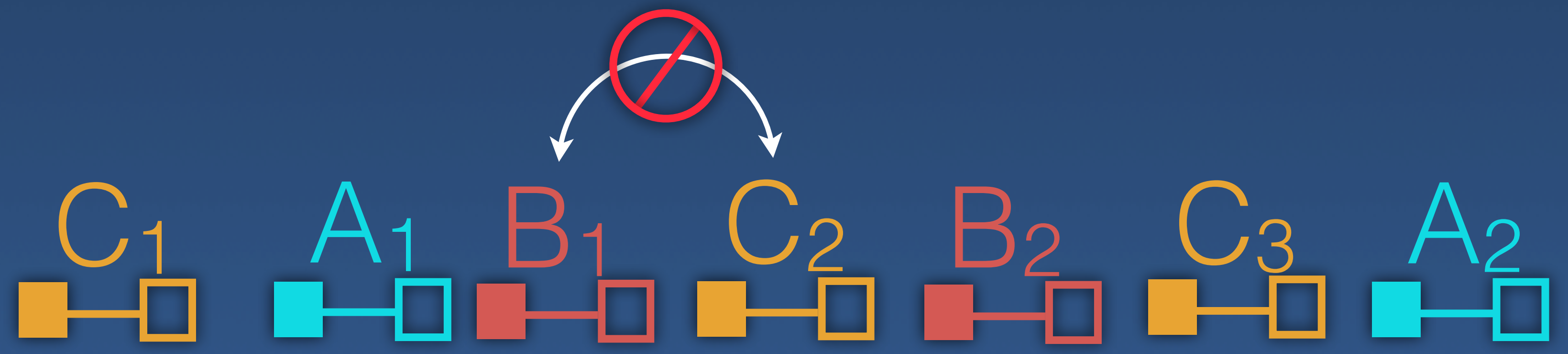
Linearizable

- Operations appear to take effect at some instant
  - ➡ between their start and end
  - ➡ If there is no overlap, ordering is well defined

Strict Consistency:
Operations are instantaneous



Equivalent sequential history S exists:



1. Got the result S would get
2. No thread's history is violated in S
3. Non-overlapping operations retain order in S

Linearizability is Composable

Types of Registers

- Linearizable registers
  - ➡ SRSW 1-bit safe register
  - ➡ MRMW n-bit atomic linearizable register
- Sequentially consistent registers
- Causally consistent registers
- FIFO consistent registers
- Weakly consistent registers

Sequential Consistency

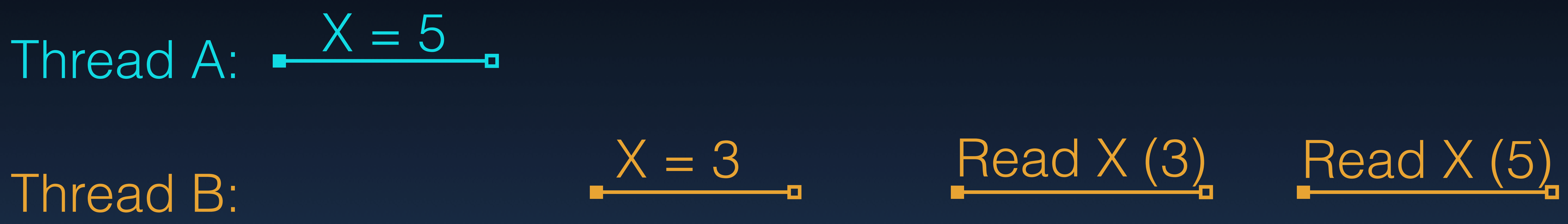
- "A multiprocessor is sequentially consistent if the result of any execution is the same as if the operations of all the processors were executed in some sequential order, and the operations of each individual processor appear in this sequence in the order specified by its program." [Lamport, 1979]

Sequential Consistency

- “A multiprocessor is sequentially consistent if the result of any execution is the same as if the operations of all the processors were executed in some sequential order, and the operations of each individual processor appear in this sequence in the order specified by its program.” [Lamport, 1979]

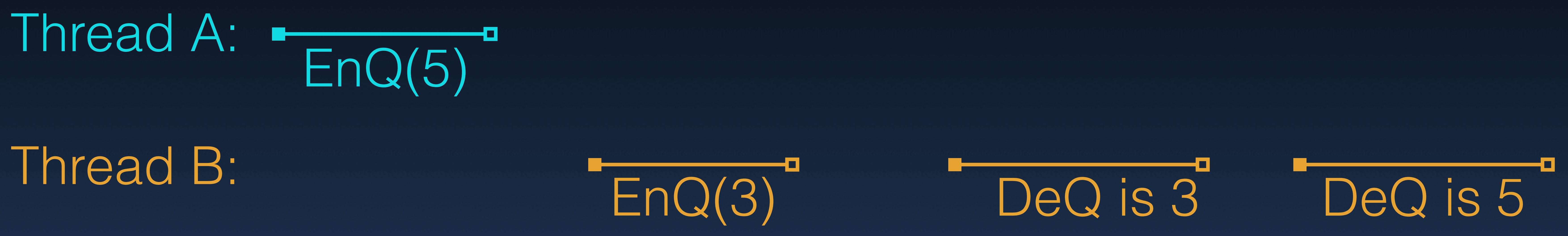
Weaker than Linearizability

Sequentially Consistent



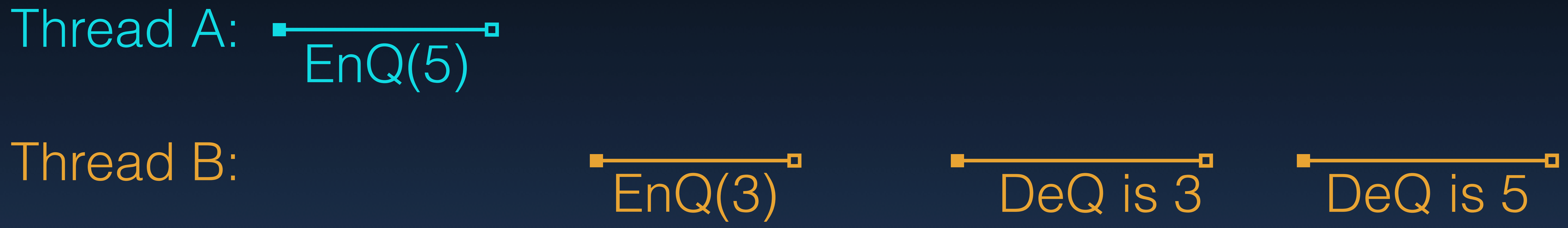
- No global notion of time
  - ➡ Only consistent Order

Sequentially Consistent

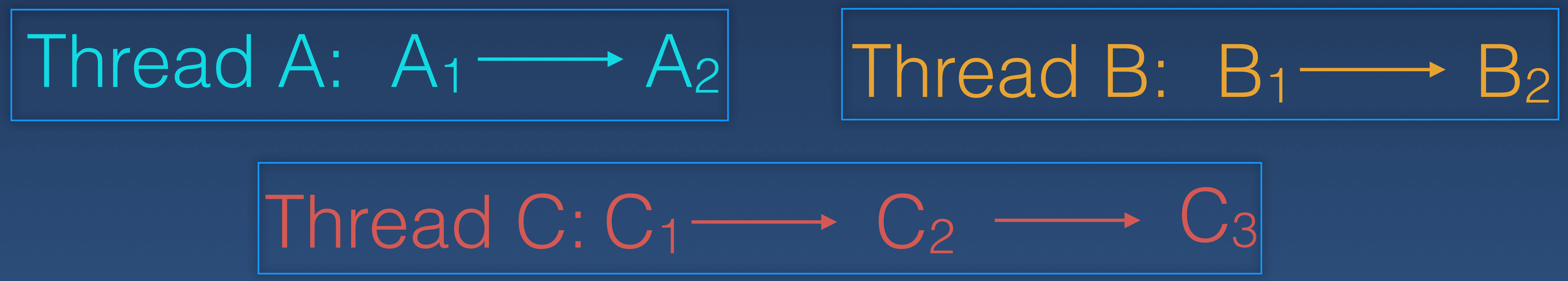


- No global notion of time
  - ➡ Only consistent Order

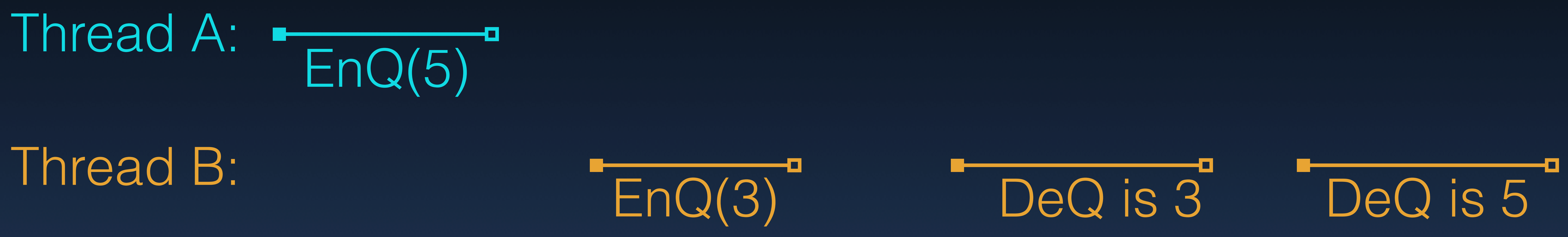
Sequentially Consistent



- No global notion of time
  - ➡ Only consistent Order



Sequentially Consistent



- No global notion of time
  - ➡ Only consistent Order

Thread A: $A_1 \longrightarrow A_2$

Thread B: $B_1 \longrightarrow B_2$

Thread C: $C_1 \longrightarrow C_2 \longrightarrow C_3$

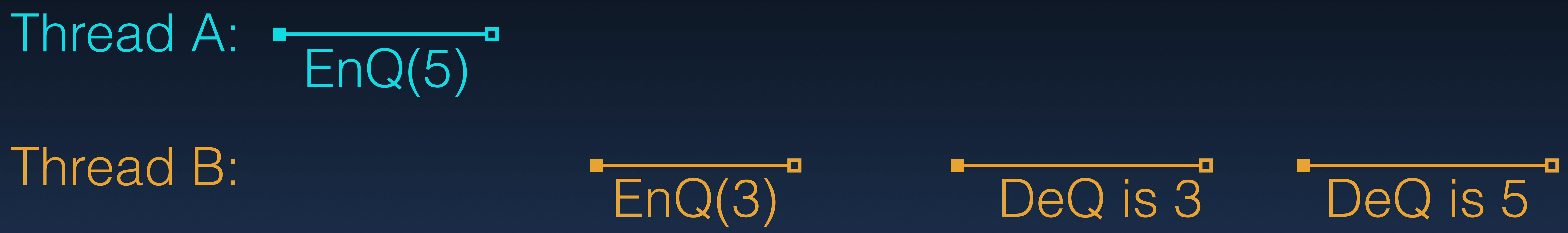
Sequential History

$A_1 \to A_2 \quad C_1 \to C_2 \to C_3 \quad B_1 \to B_2$

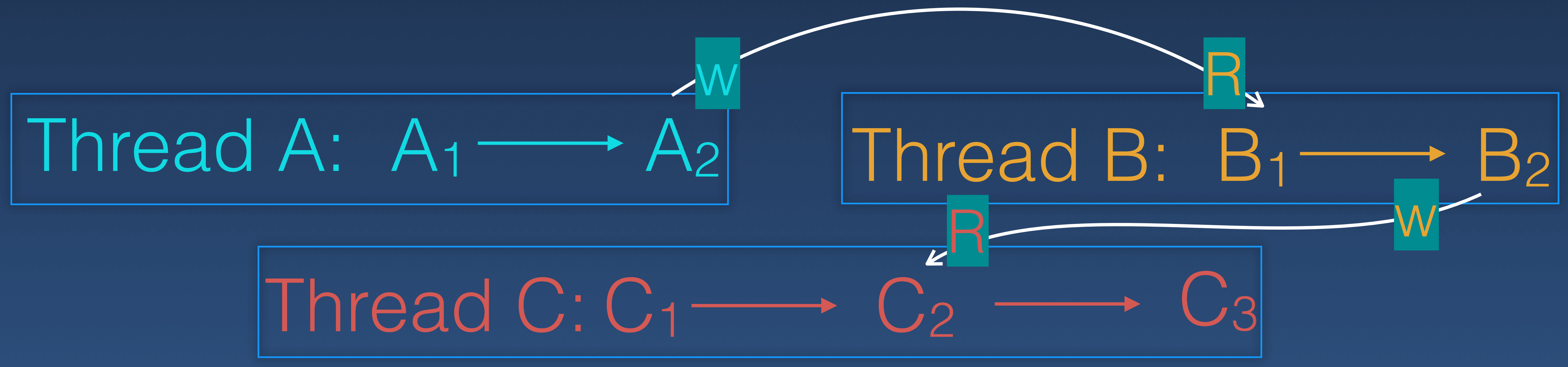
$$\begin{tikzcd}
A_1 \arrow[rr, bend left] & C_1 \arrow[rrrr, bend right] & A_2 & B_1 \arrow[r] & B_2 & C_2 \arrow[r] & C_3
\end{tikzcd}$$

Subodh Kumar

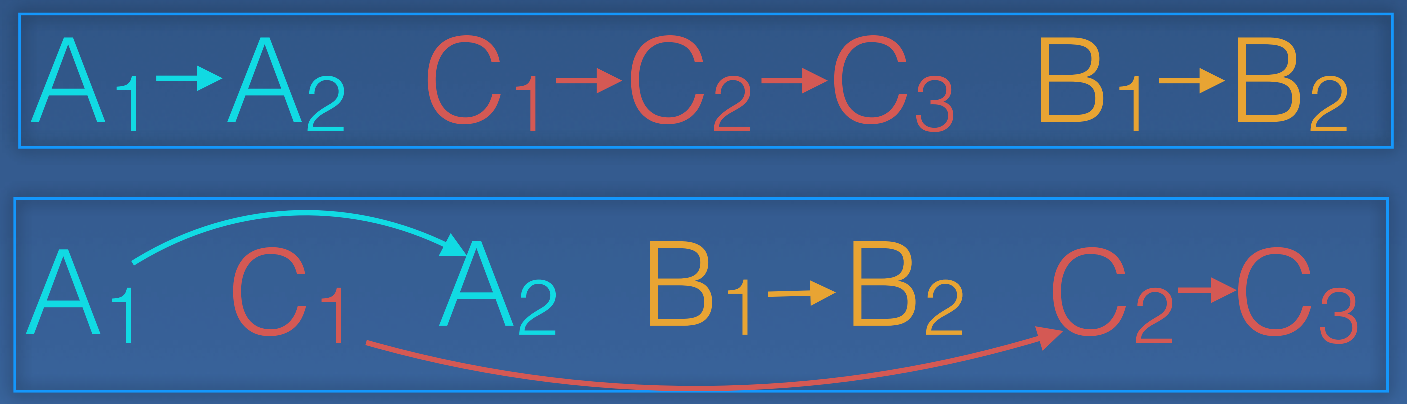
Sequentially Consistent



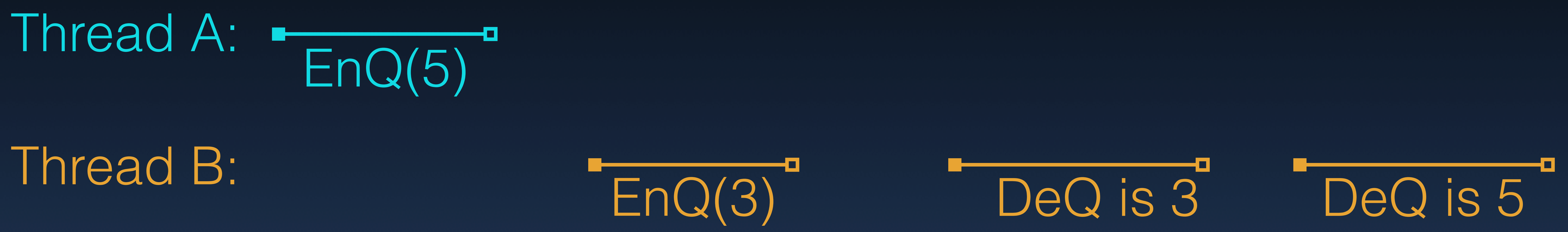
- No global notion of time
  - ➡ Only consistent Order



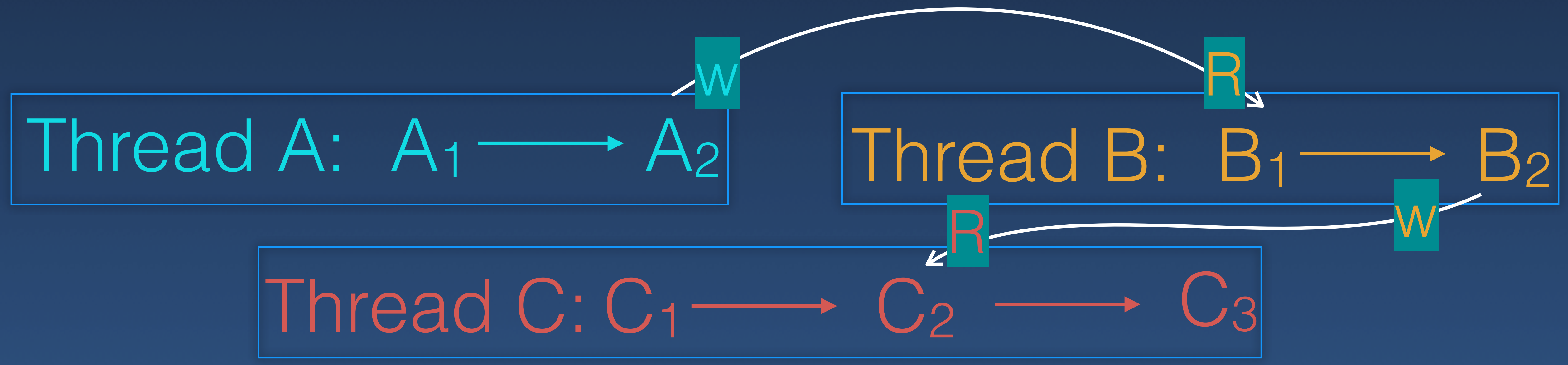
Sequential History



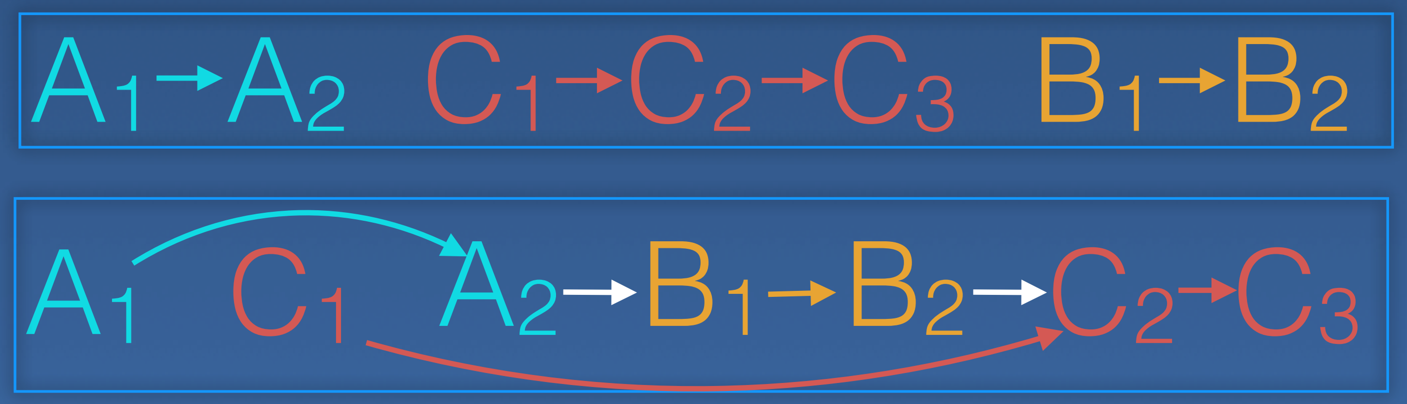
Sequentially Consistent



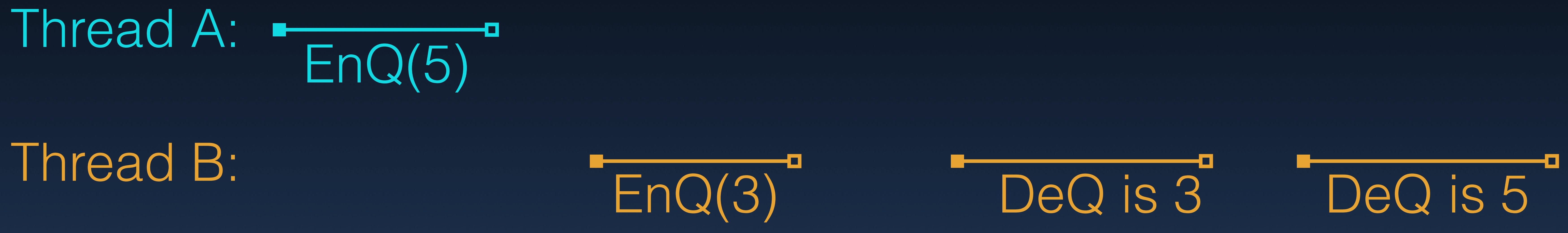
- No global notion of time
  - ➡ Only consistent Order



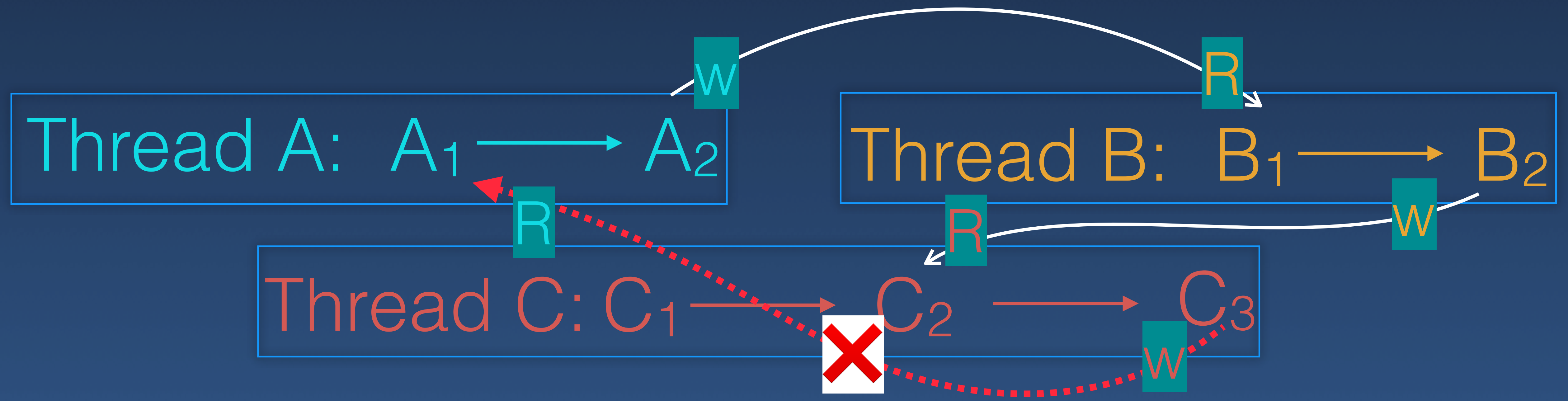
Sequential History



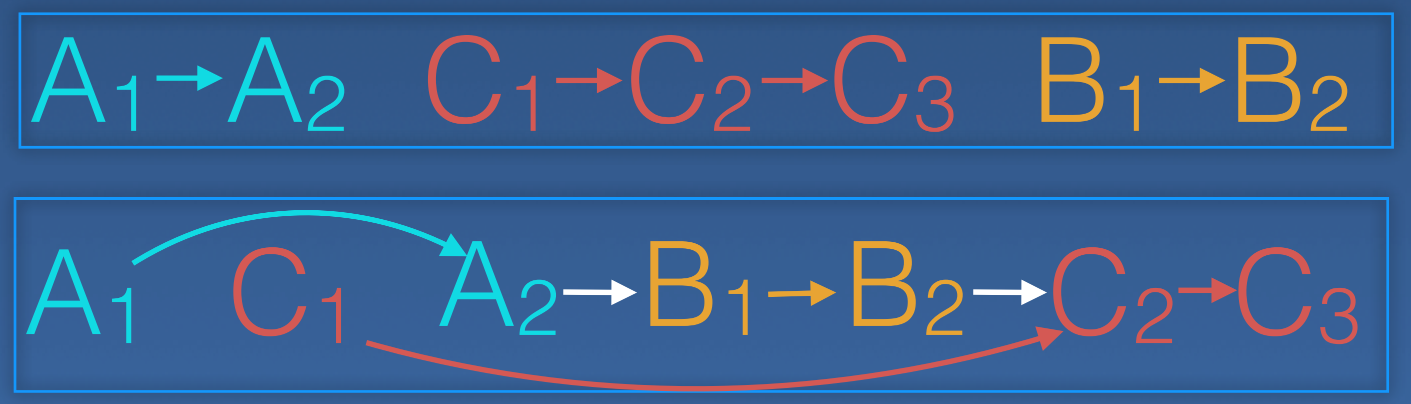
Sequentially Consistent

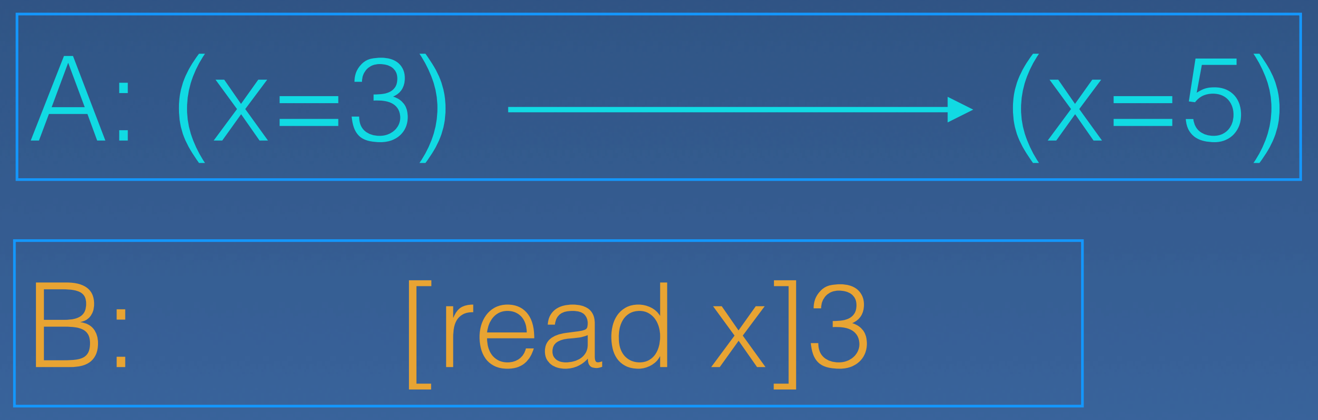


- No global notion of time
  - ➡ Only consistent Order

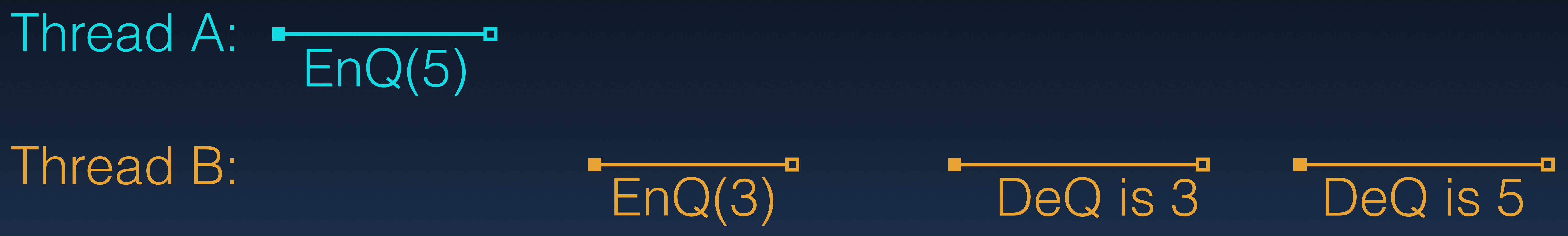


Sequential History

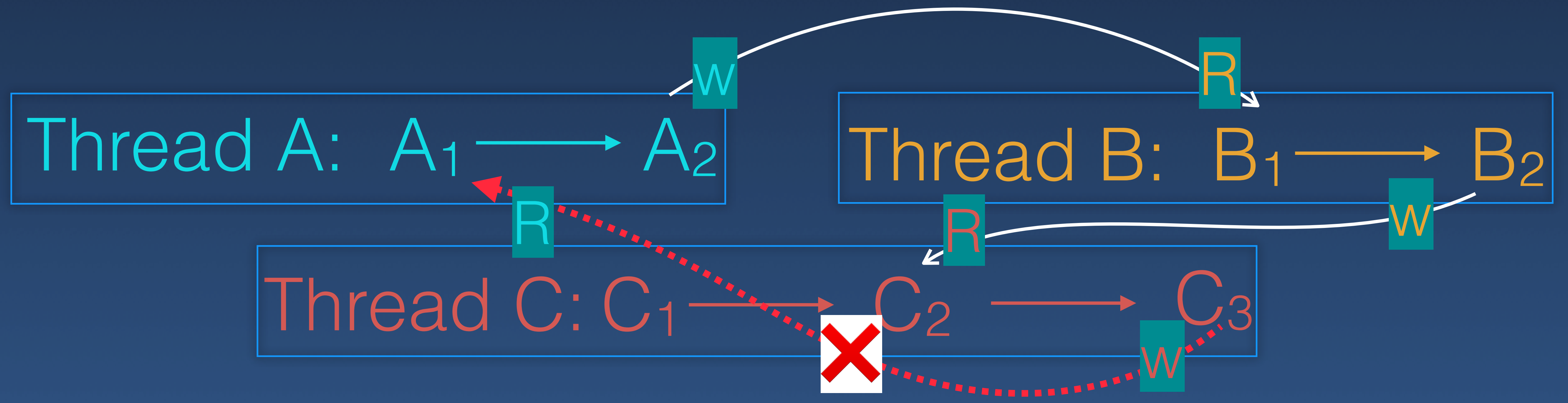




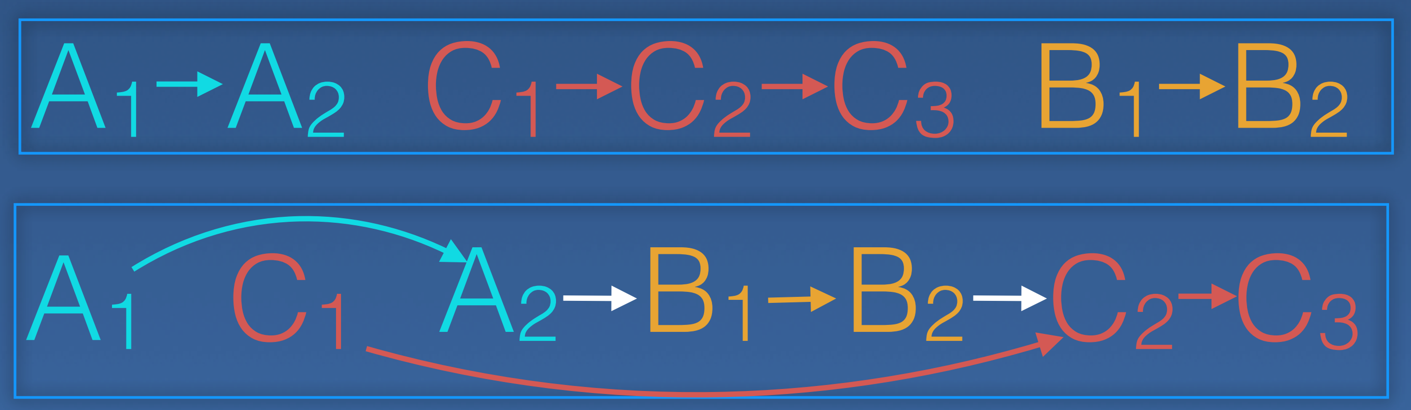
Sequentially Consistent

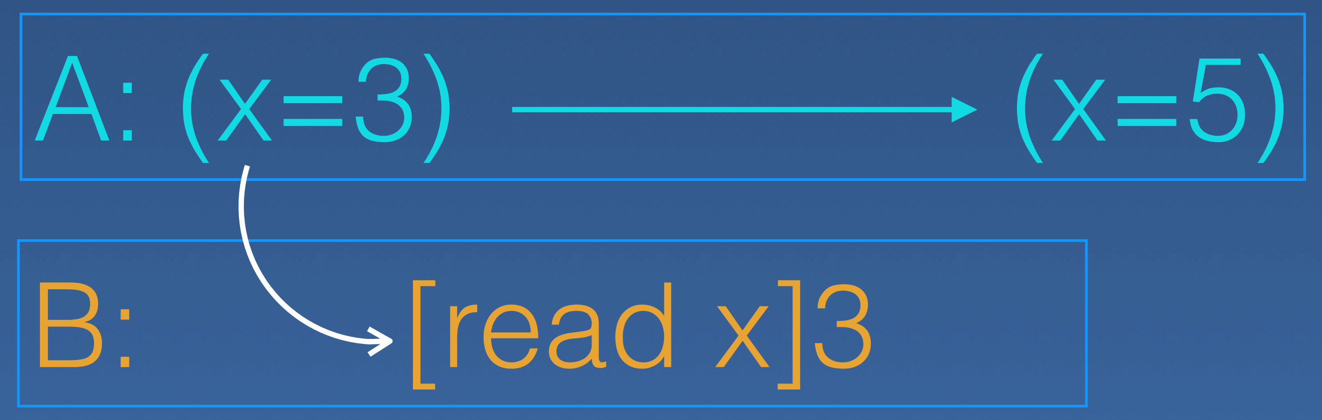


- No global notion of time
  - ➡ Only consistent Order



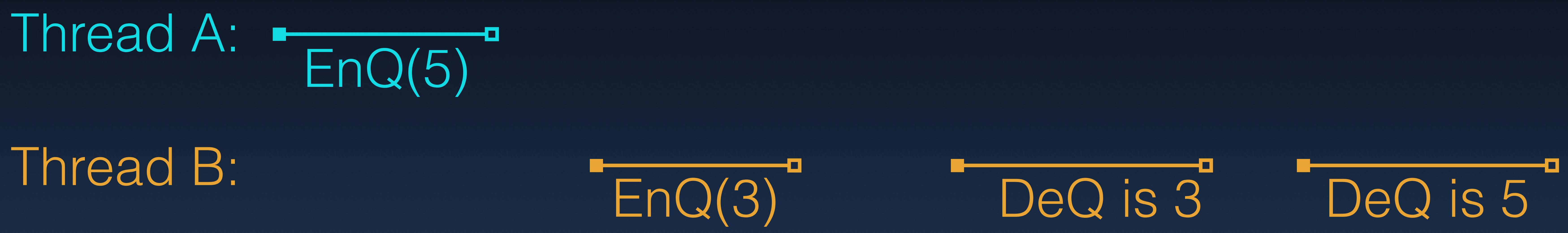
Sequential History



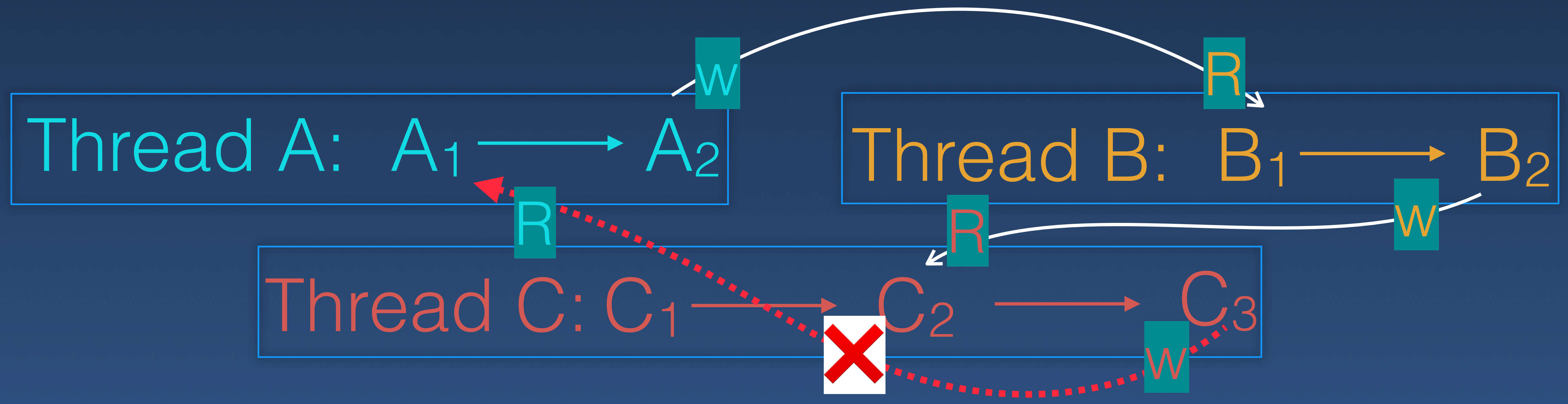


Subodh Kumar

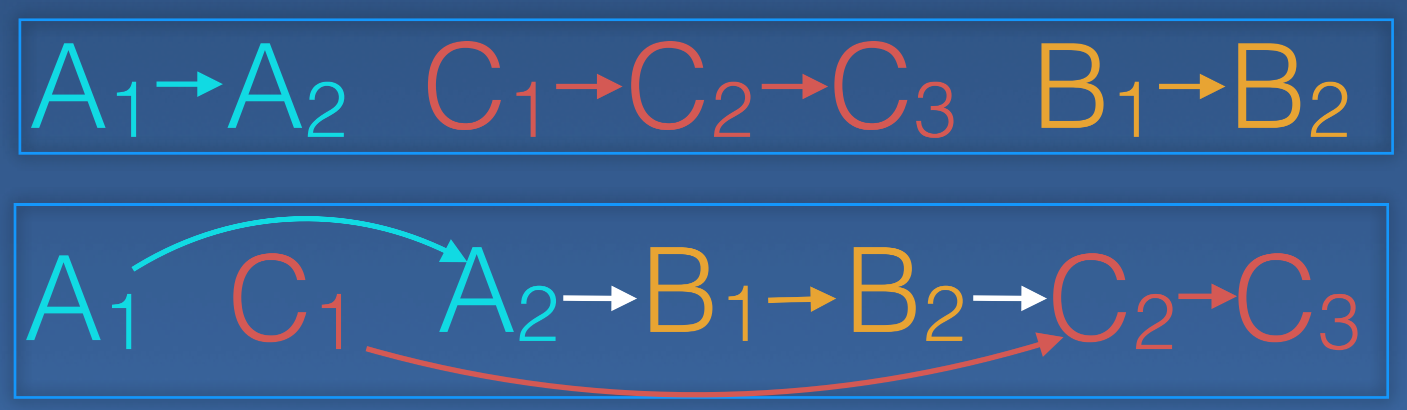
Sequentially Consistent

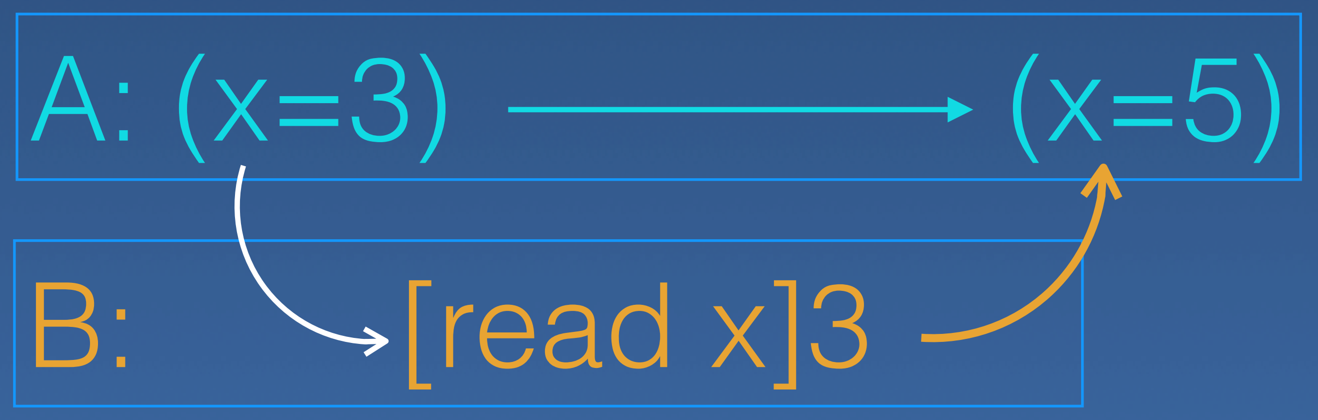


- No global notion of time
  - ➡ Only consistent Order



Sequential History





Memory Consistency Semantics

- Threads always see values written by some thread
  - ➡ No garbage Update is atomic
- The value seen is constrained by thread-order
  - ➡ for every thread

Memory Consistency Semantics

- Threads always see values written by some thread
  - ➡ No garbage Update is atomic
- The value seen is constrained by thread-order
  - ➡ for every thread

```
initially:    ready=0, data=0

  thread A           thread B

data = 1;             while(!ready);
ready = 1;            pvt = data;
```

Need: If B sees the new value of ready (1), B must also see the new value of data (1)

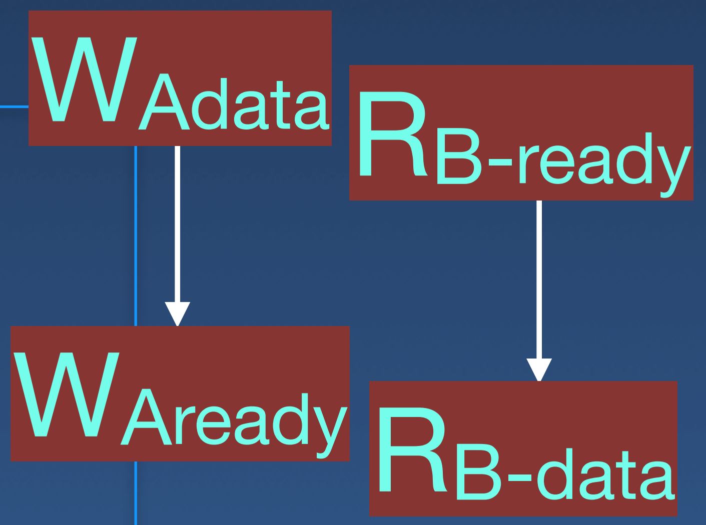
Memory Consistency Semantics

- Threads always see values written by some thread
  - ➡ No garbage Update is atomic
- The value seen is constrained by thread-order
  - ➡ for every thread

```
initially:    ready=0, data=0

  thread A          thread B

data = 1;            while(!ready);
ready = 1;           pvt = data;
```


$$\begin{tikzcd} W_{Adata} \arrow[d] & R_{B\text{-}ready} \arrow[d] \\ W_{Aready} & R_{B\text{-}data} \end{tikzcd}$$

Need: If B sees the new value of ready (1), B must also see the new value of data (1)

Memory Consistency Semantics

- Threads always see values written by some thread
  - ➡ No garbage Update is atomic
- The value seen is constrained by thread-order
  - ➡ for every thread

```
initially:    ready=0, data=0

  thread A          thread B

data = 1;            while(!ready);
ready = 1;           pvt = data;
```

$$\begin{tikzcd}
W_{Adata} \arrow[d] & \\
W_{Aready} & \\
 & R_{B\text{-}ready} \arrow[d] \\
 & R_{B\text{-}data}
\end{tikzcd}$$

Need: If B sees the new value of ready (1),
B must also see the new value of data (1)

Memory Consistency Semantics

- Threads always see values written by some thread
  - ➡ No garbage Update is atomic
- The value seen is constrained by thread-order
  - ➡ for every thread

```
initially:    ready=0, data=0

  thread A          thread B

data = 1;             while(!ready);
ready = 1;            pvt = data;
```

$$\begin{tikzcd} W_{Adata} \arrow[d] & \\ W_{Aready} & \\ & R_{B\text{-}ready} \arrow[d] \\ & R_{B\text{-}data} \end{tikzcd}$$

If B sees ready =	then B may see data =
0	1
0	0
1	1

Need: If B sees the new value of ready (1), B must also see the new value of data (1)

Review

- Linearizability
- Sequential Consistency