Lamport clocks

Dave Eckhardt
de0u@andrew.cmu.edu

Synchronization

- No class Friday
  - Spring Carnival (“Mobot” races @ noon)

Outline ***(not)***

- Chapter 15 ("Distributed System Structures")
  - Zooming past distributed systems
    - Process migration!
  - Network protocol stacks
    - "The Internet in one easy lesson"
  - You can read it yourselves...
    - ***...and you probably should.***

Outline

- Lamport clocks
  - Covered in 17.1, 17.2 (different focus from today)
  - Time, Clocks, and the Ordering of Events in a Distributed System
    - CACM 21:7 (1978)

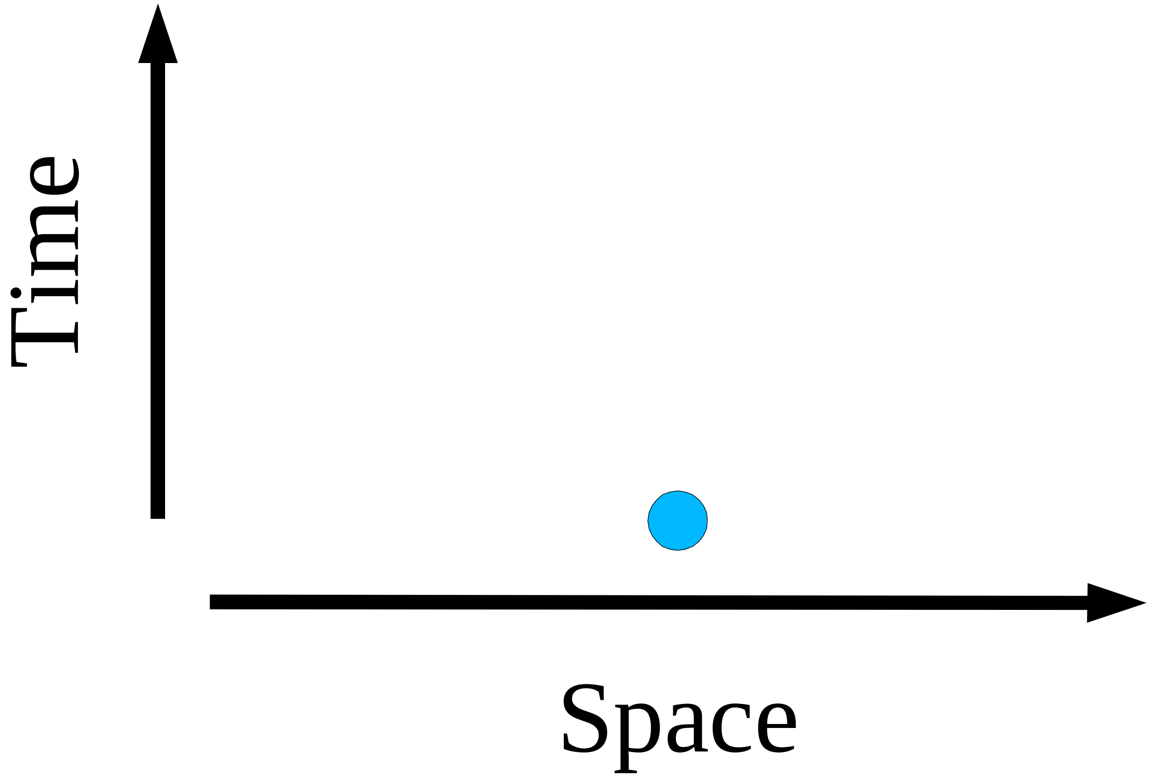
Overview

- Light cones
- Meeting for beer
- “Happened before” partial order
- Logical clocks
- Advanced techniques

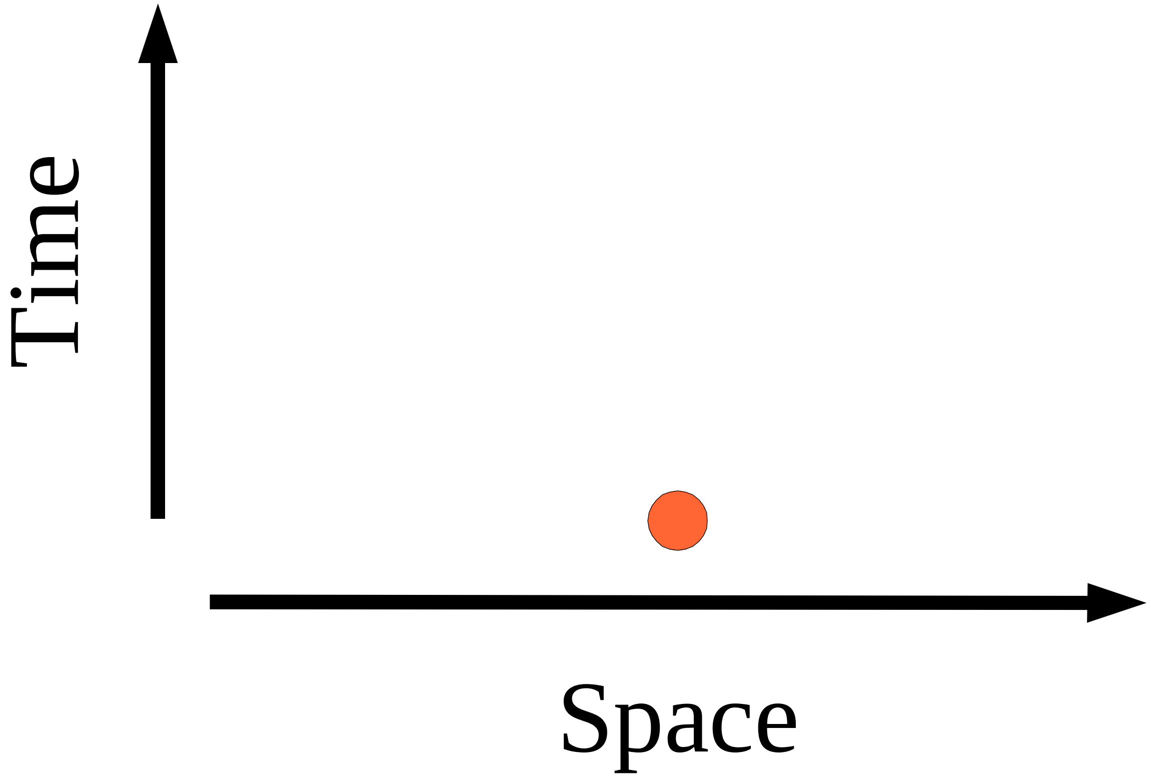
Light cones

- Concept
  - Effects propagate at or below speed of light
    - Objects, light/radio/X-rays, gravity
  - ***Knowledge*** of events limited the same way
  - Event propagation modeled by expanding sphere
    - Four-dimensional “cone”

Light cones



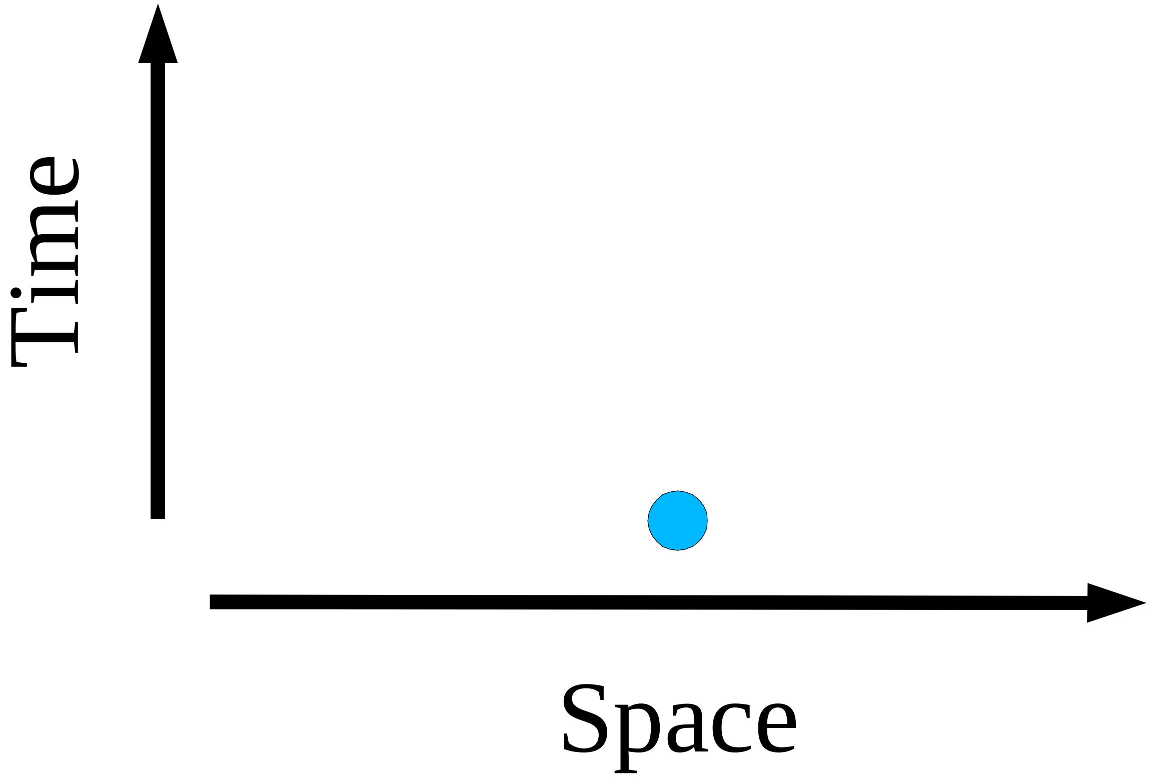
Light cones



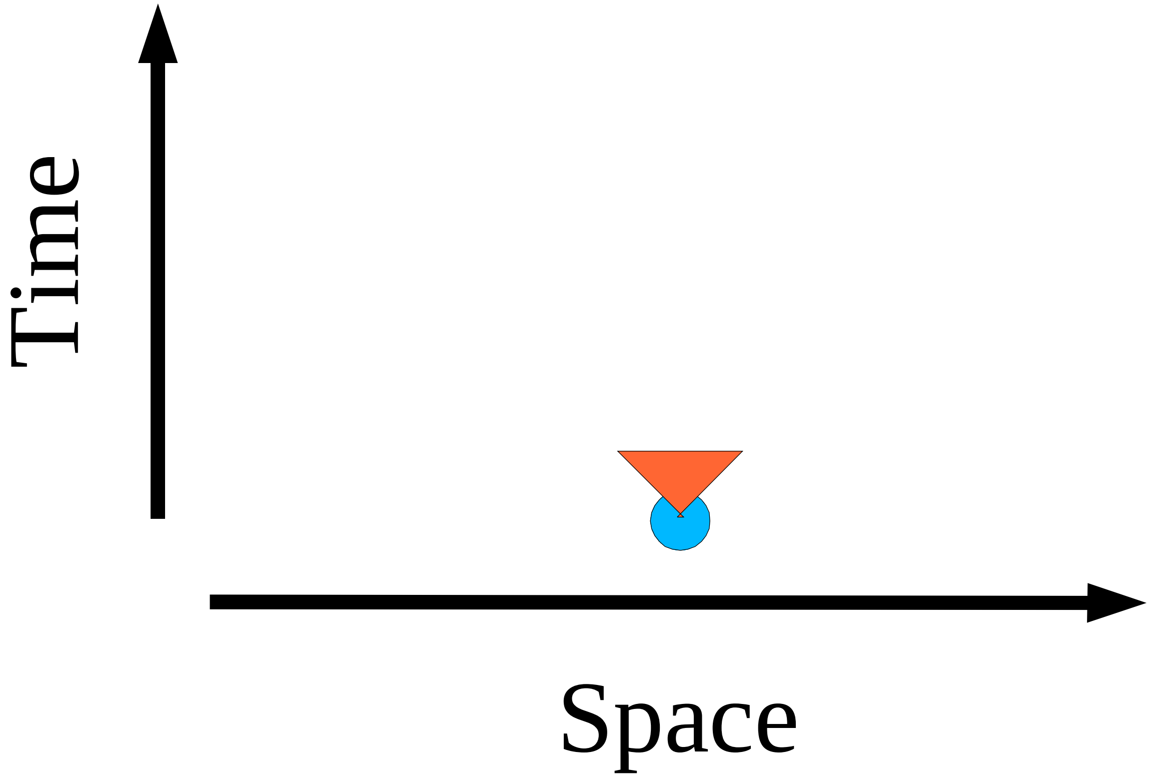
Time

Space

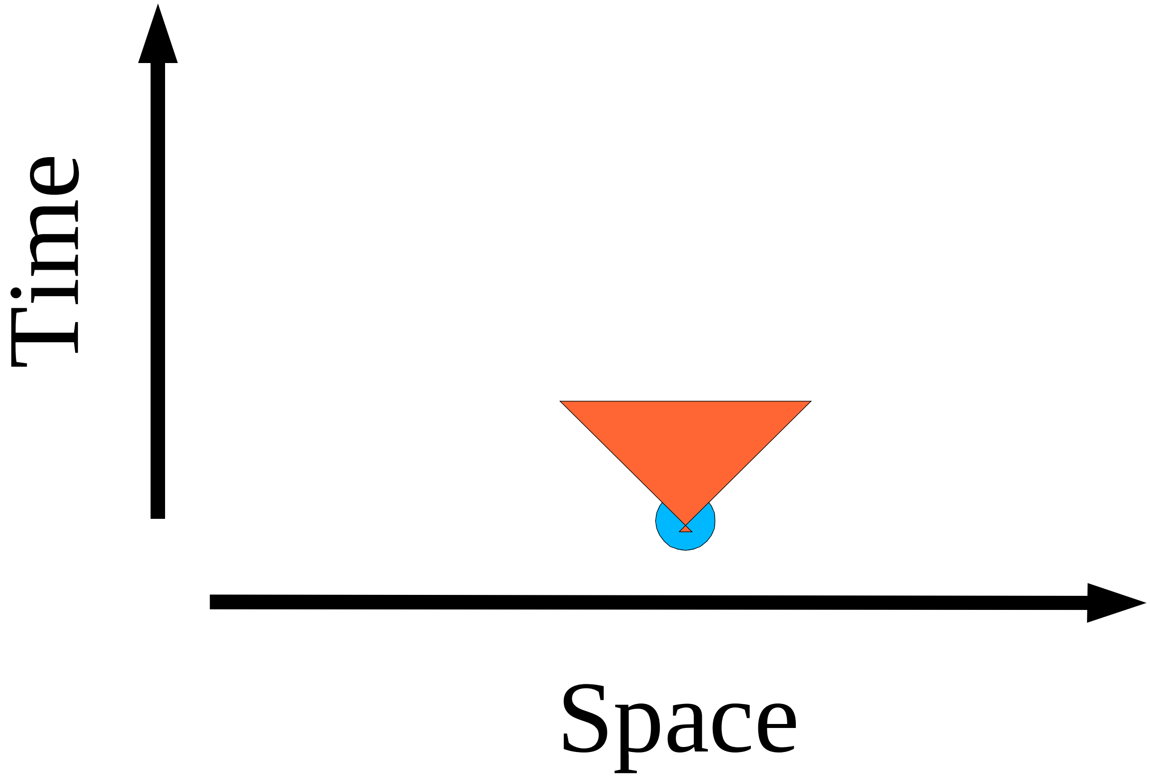
Light cones



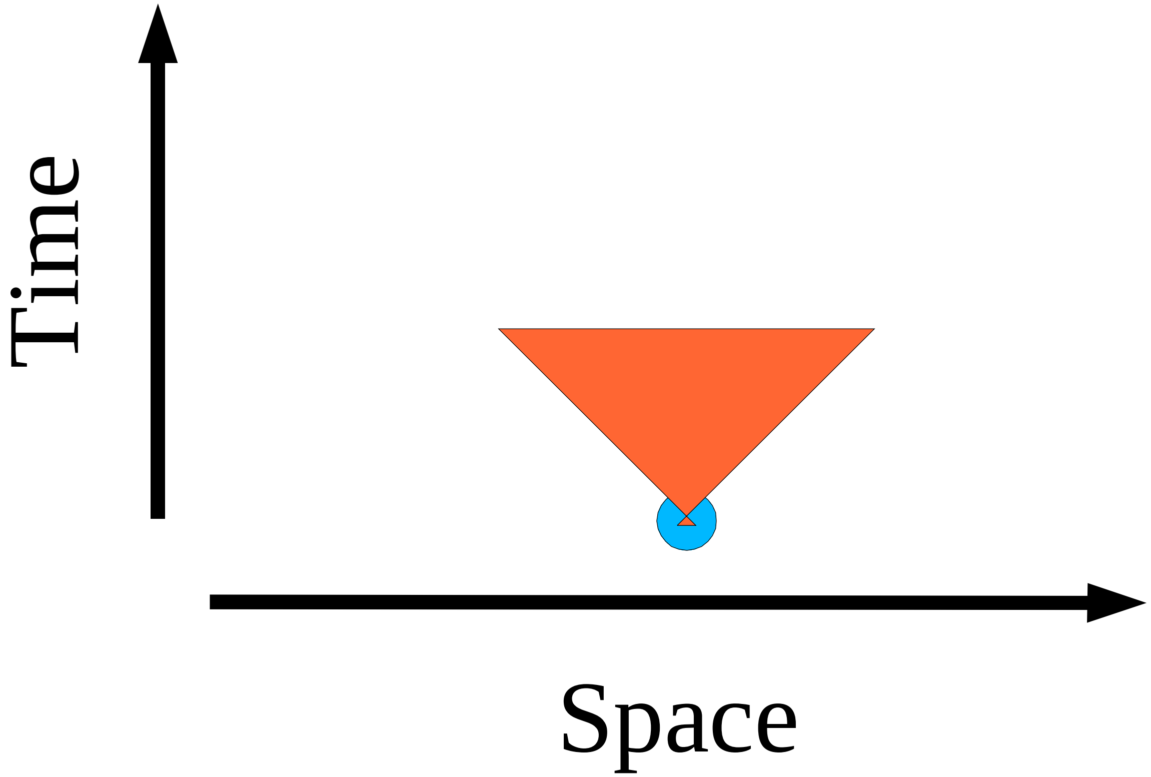
Light cones



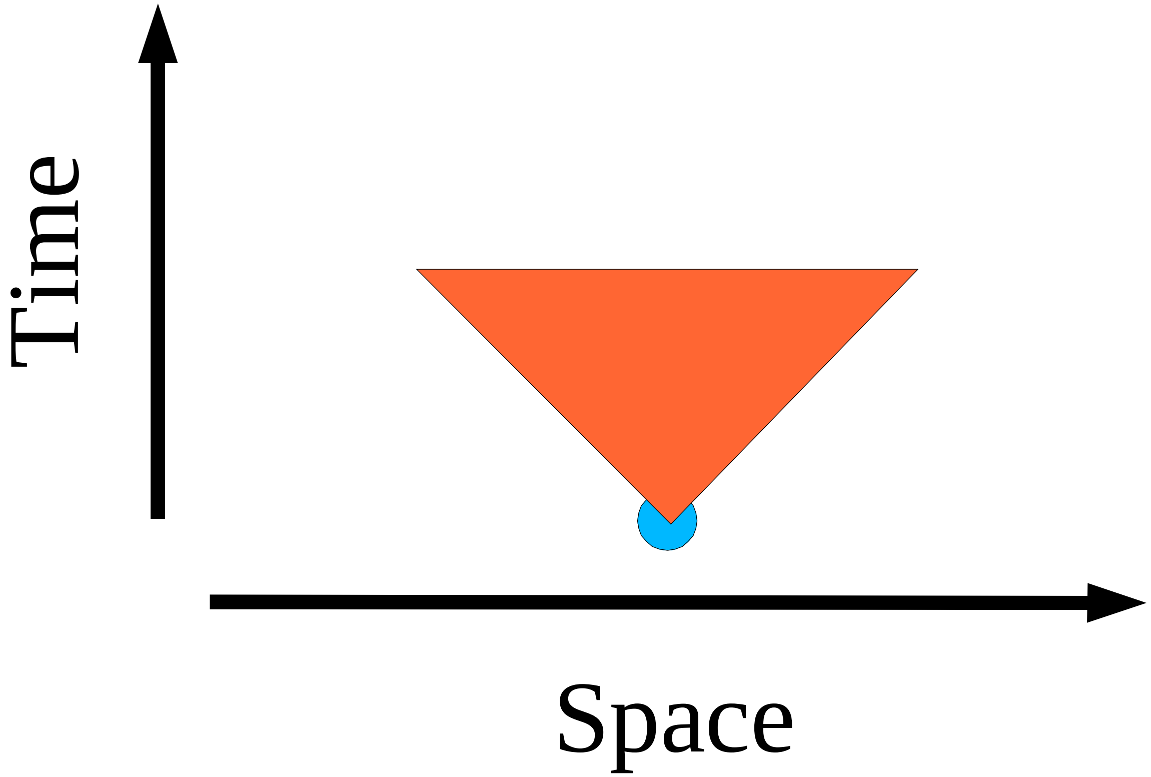
Light cones



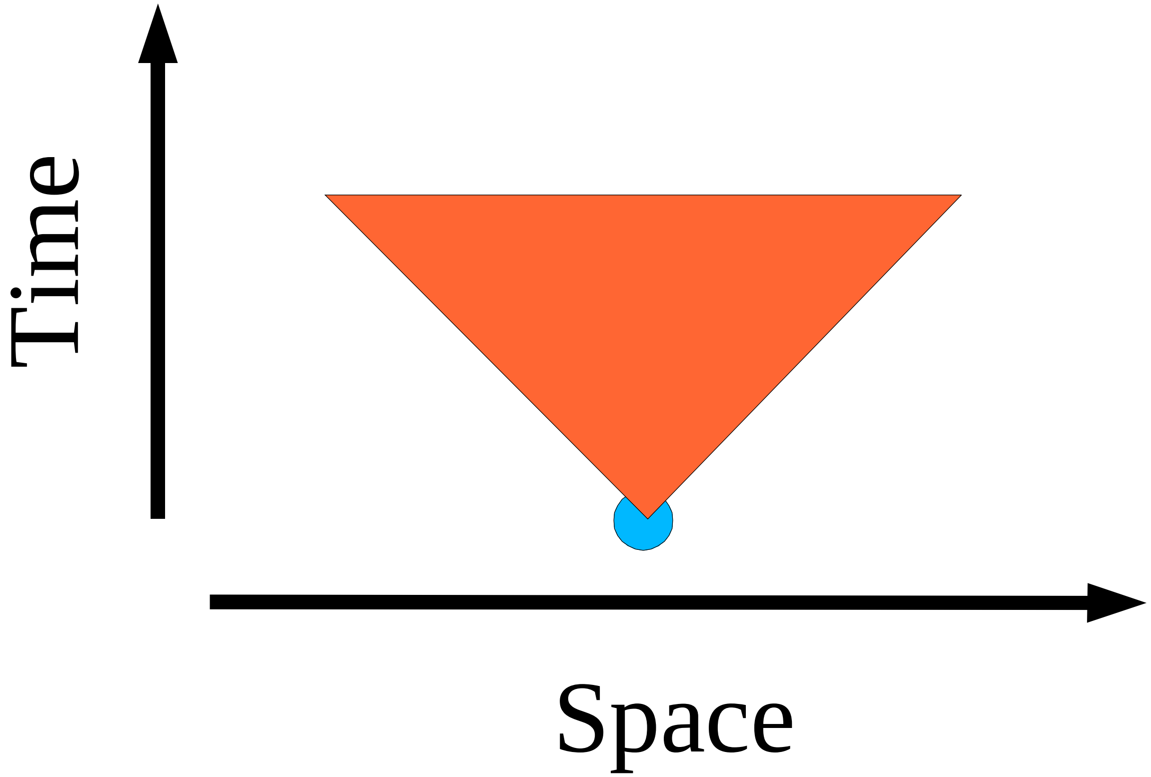
Light cones



Light cones



Light cones



Light cones

- Future light cone
  - The part of spacetime influenced by an event
- Past light cone
  - The part of spacetime that could have influenced an event

Meeting for Beer

- P1 transmits “Panther Hollow Inn” to blackboard

Meeting for Beer

- P1 transmits “Panther Hollow Inn” to blackboard
- P1 transmits to P2
  - Hey, P2, let's go have a beer.
  - I have transmitted the bar's name to the blackboard.
  - See you there!

Meeting for Beer

- P1 transmits “Panther Hollow Inn” to blackboard
- P1 transmits to P2
  - Hey, P2, let's go have a beer.
  - I have transmitted the bar's name to the blackboard.
  - See you there!
- P2 receives P1's message

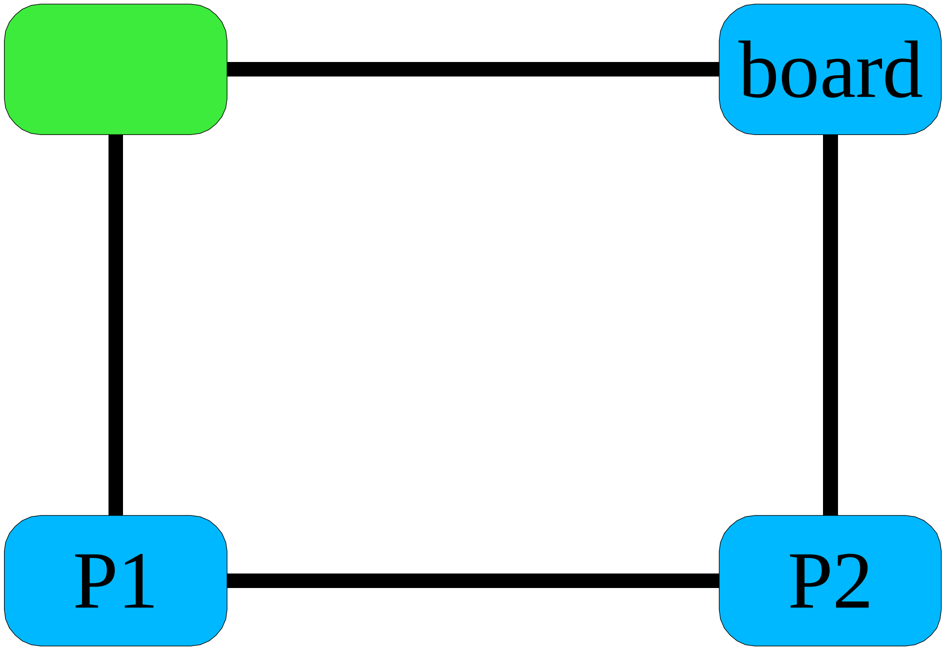
Meeting for Beer

- P1 transmits “Panther Hollow Inn” to blackboard
- P1 transmits to P2
  - Hey, P2, let's go have a beer.
  - I have transmitted the bar's name to the blackboard.
  - See you there!
- P2 receives P1's message
- P2 queries blackboard

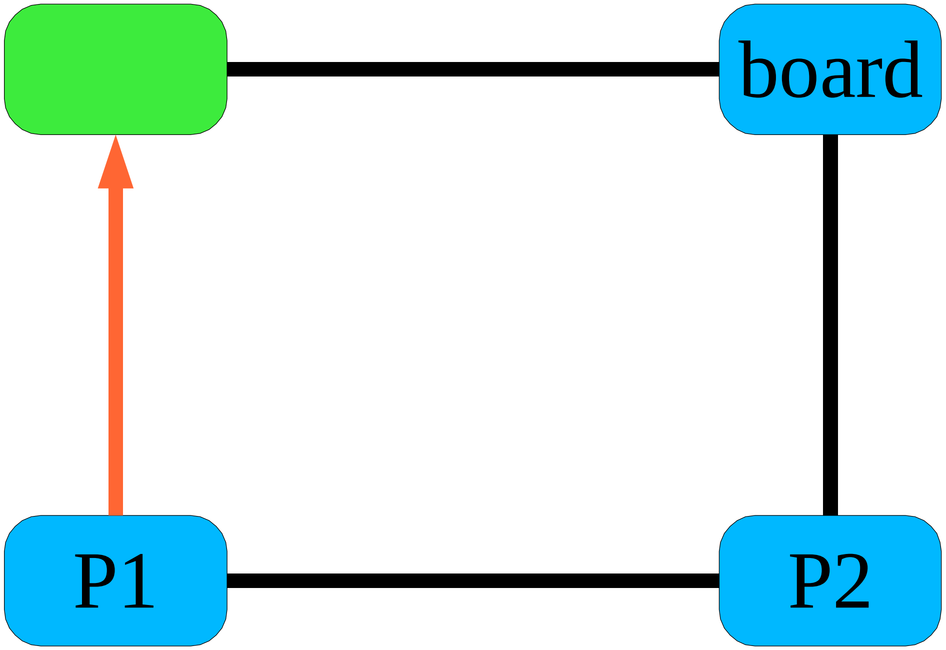
Meeting for Beer

- P1 transmits “Panther Hollow Inn” to blackboard
- P1 transmits to P2
  - Hey, P2, let's go have a beer.
  - I have transmitted the bar's name to the blackboard.
  - See you there!
- P2 receives P1's message
- P2 queries blackboard
- It says “Squirrel Cage” - ***how???***

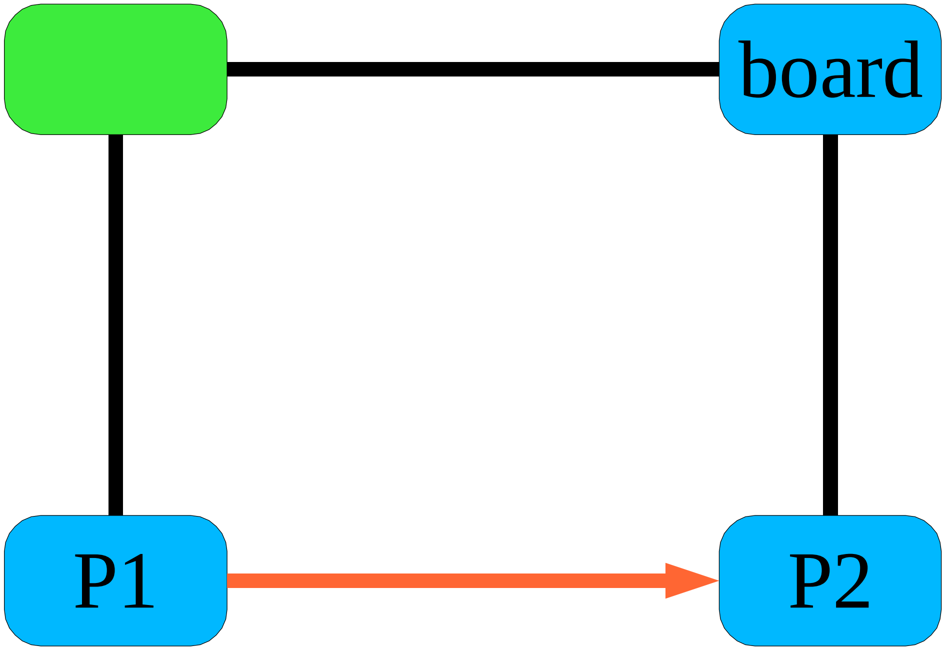
Meeting for Beer



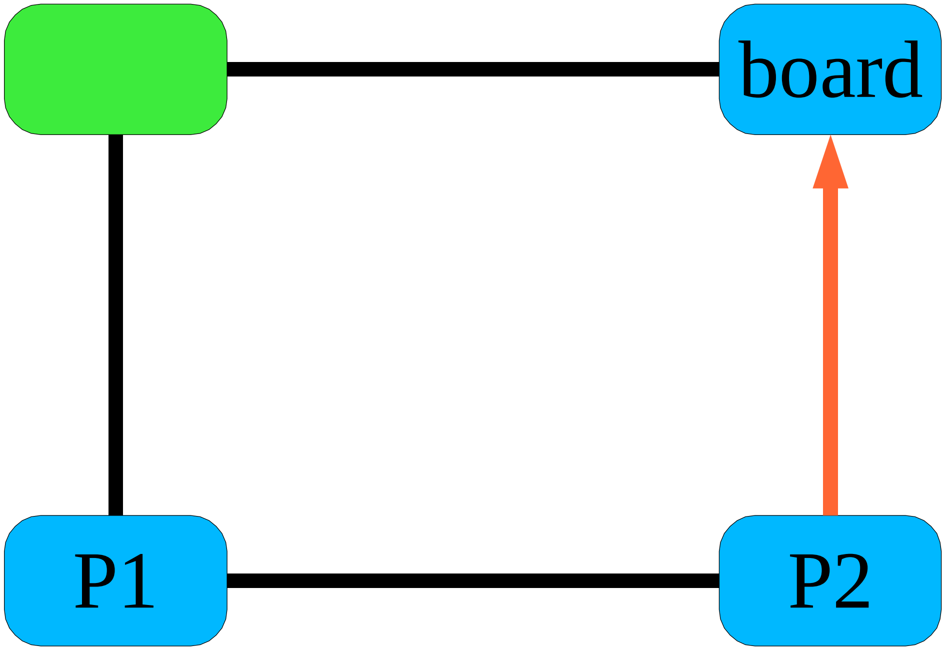
Meeting for Beer



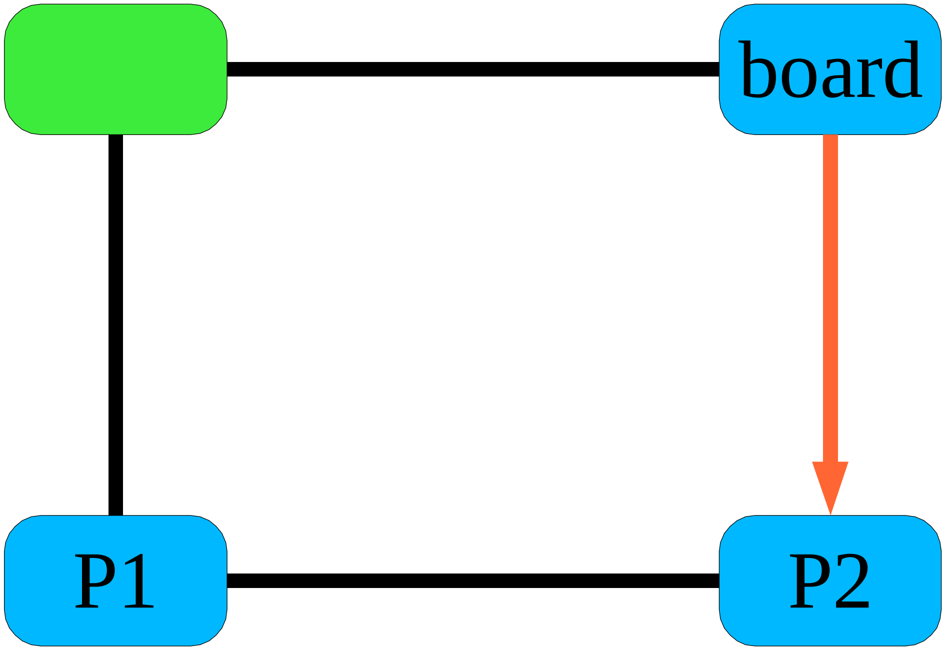
Meeting for Beer



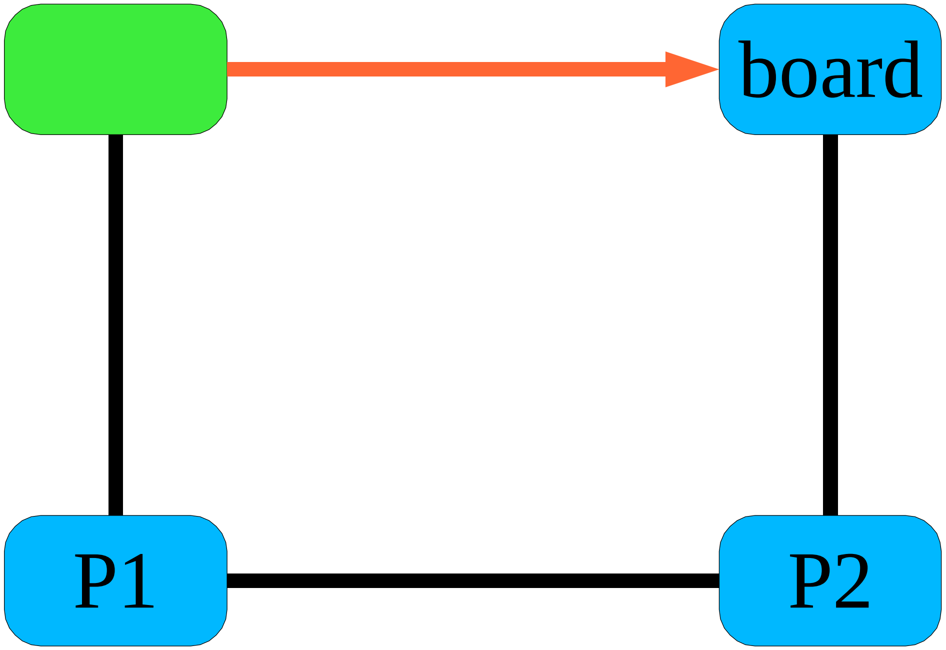
Meeting for Beer



Meeting for Beer



Meeting for Beer



What went wrong?

- P1 thought
  - Blackboard update ***happened before*** invitation
- P2 thought
  - Invitation ***happened before*** blackboard update
- When does an event “happen”?
  - When its effects propagate “everywhere relevant”
- What does “happen before” mean?
- Could that green node really be so slow?

Universe Model

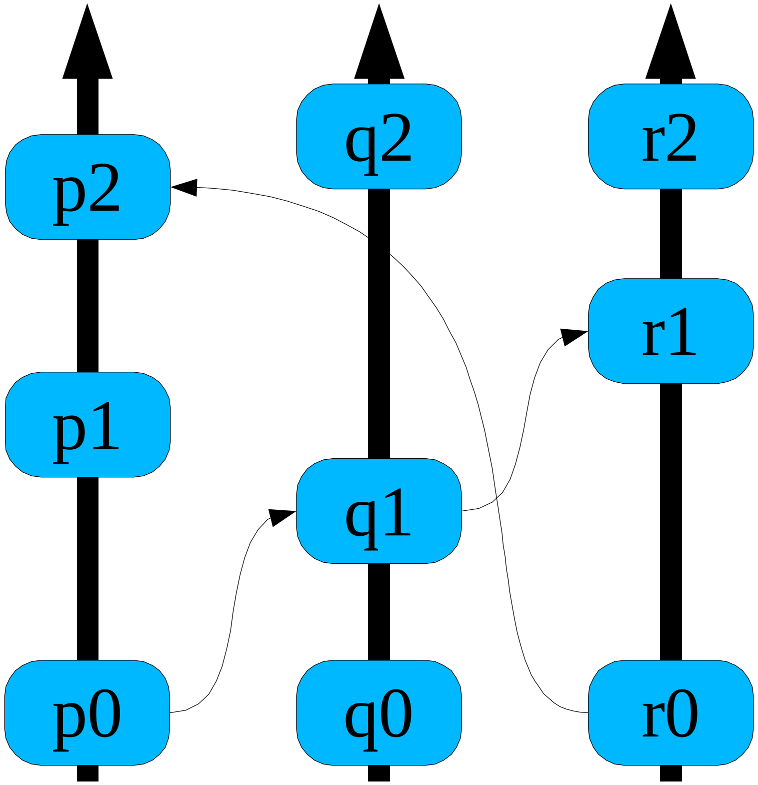
- System = set of processes
- Process = sequence of events
- Event
  - Internal: ++x;
  - Message transmission
  - Message reception

“Happened before” partial order

- A ***happens before*** B (A → B)
  - If A and B happen inside a process, in that order
  - A = transmission, B = reception, of same message
  - If A → B and B → C, then A → C
- A and B are ***concurrent*** when
  - A !→ B and B !→ A
- Observe A !→ A

Space-time Diagram

- →
  - inside a process, or
  - follow a message
- p0 → r2
- concurrent
  - p0, q0, r0
  - p1, q1
  - q1, r0
  - p1, r0



$\rightarrow$ means "possibly causes"

- p0 possibly causes p1
  - ...by storing something in P's memory
- p0 possibly causes q1
  - Message could trigger q1
- Concurrent events
  - ...cannot cause each other

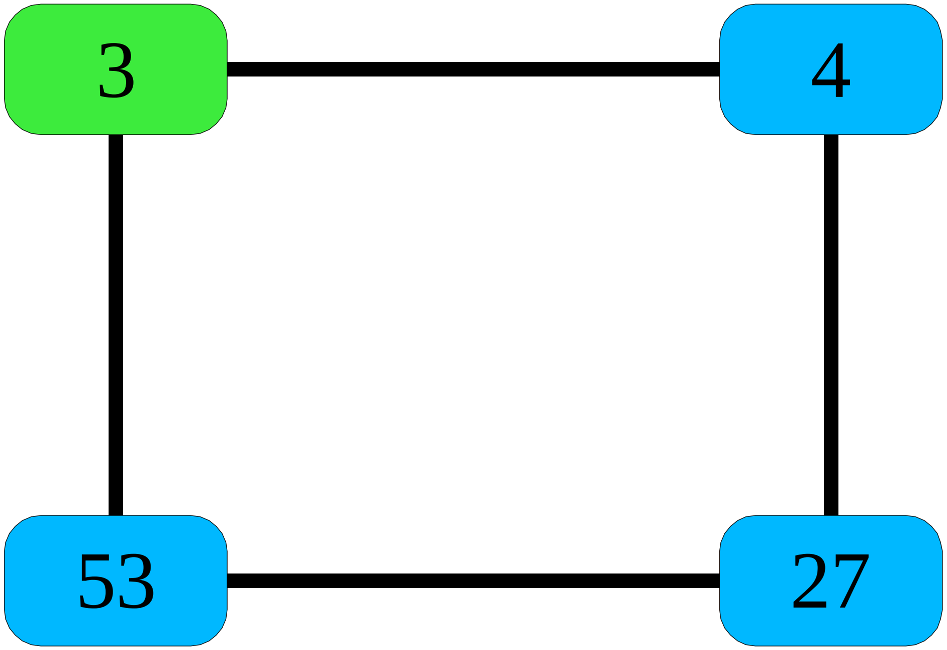
Logical clocks

- Can we assign timestamps to events?
- Want
  - If $A \rightarrow B$ then $C(A) < C(B)$
- Events inside P_i
  - $a \rightarrow b \Rightarrow C_i(a) < C_i(b)$
- Message from P_i to P_j
  - $a = P_i$'s send, $b = P_j$'s receive $\Rightarrow C_i(a) < C_j(b)$

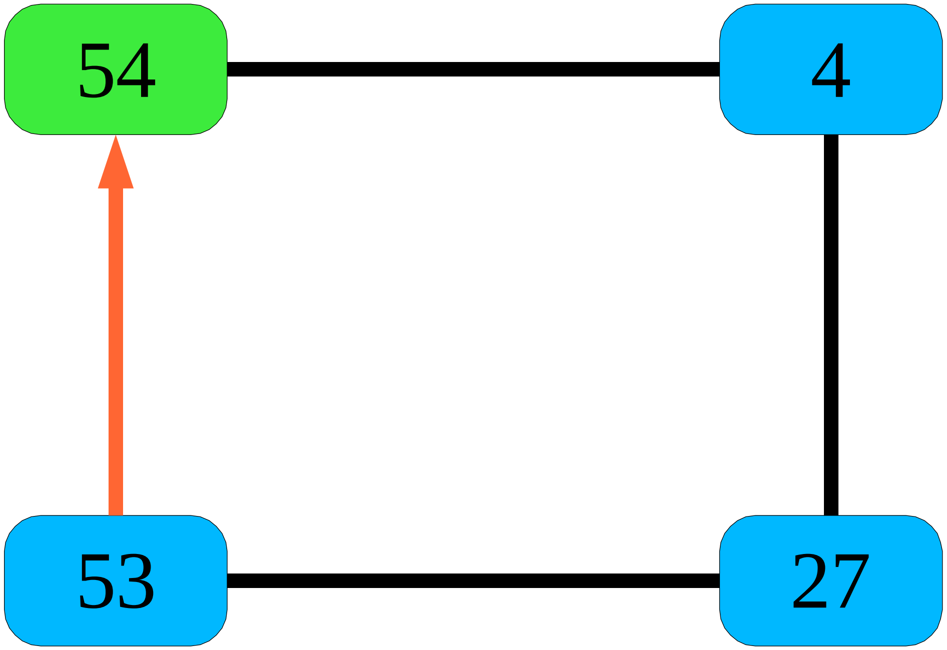
Logical clocks

- Events inside P_i
  - Increment $C_i()$ between successive events
- Message from P_i to P_j
  - Sender: place ***timestamp*** T in message: $C_i(\text{send})$
  - Receiver: ensure $C_j(\text{receive}) > T$

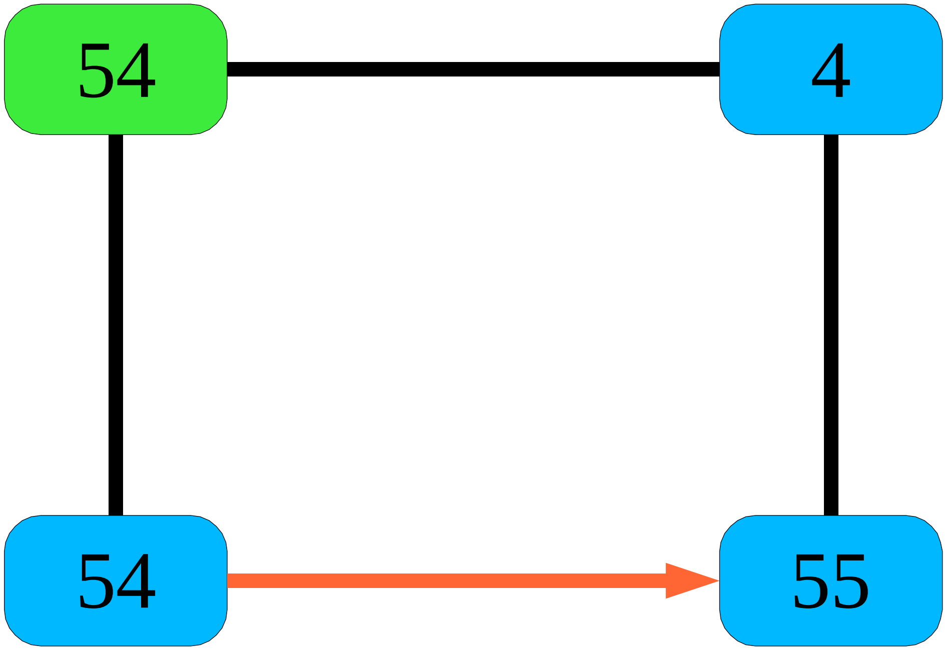
Meeting for Beer



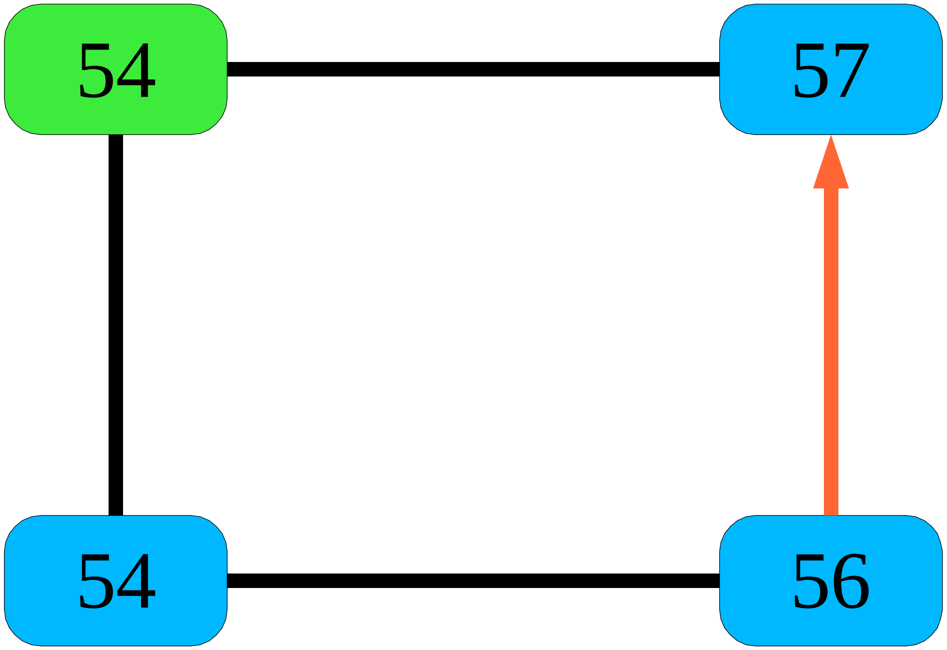
Meeting for Beer



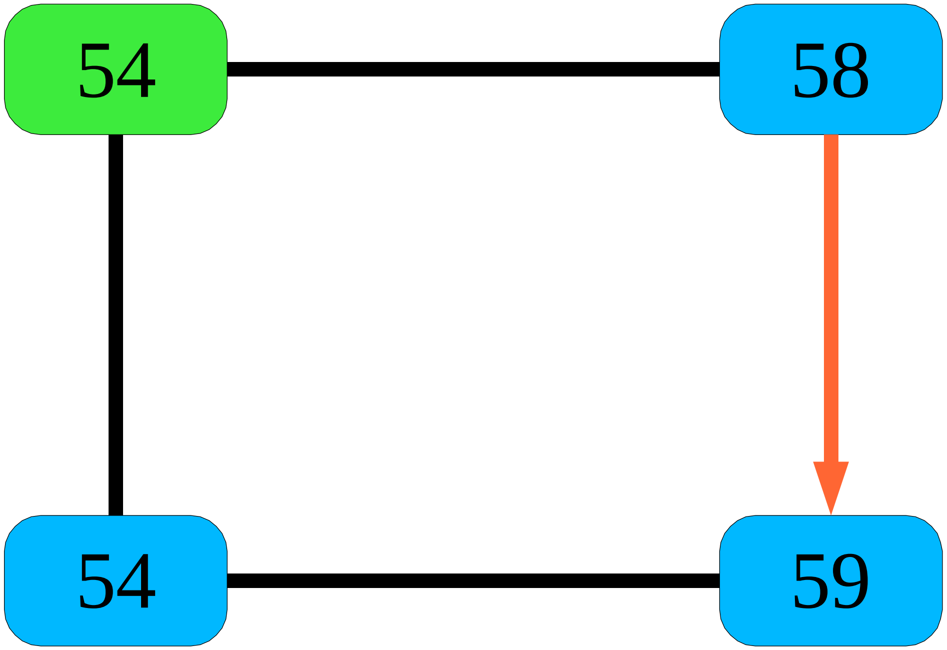
Meeting for Beer



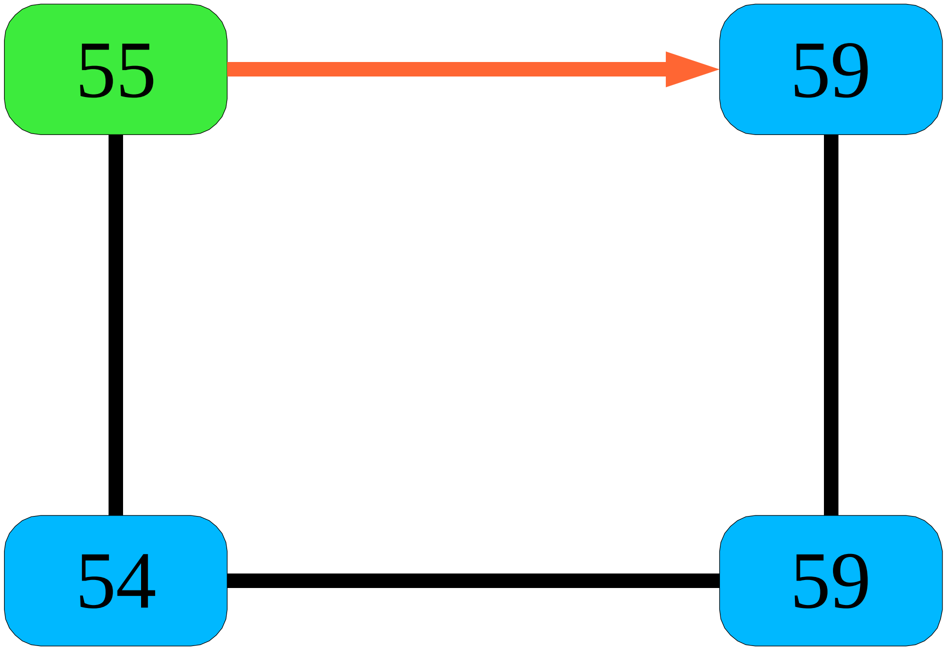
Meeting for Beer



Meeting for Beer



Meeting for Beer



What this means

- P1 wants
  - <“PHI” on board> ***happened before*** <P2 read board>
- Equivalent to “59 < 58” (oops)
- The events were ***concurrent***
- “PHI” ***could not cause*** P2's bar trip

Fixing the problem

- P1 should wait for board to acknowledge
- “PHI” causes ACK
- ACK causes “Meet me at...”
- “Meet me at...” causes bar trip
- Then: “PHI” causes bar trip

Extensions

- Define ***total ordering*** of system events
  - Typical (timestamp, process #) tuple comparison
    - Process # used to break timestamp ties
- Distributed agreement algorithms
  - Such as "***fair*** distributed mutual exclusion"
    - Requests must be granted "in order"
    - See text: 17.2
- Adding physical (real-time) clocks

Summary

- Light cones
- “Happened before" partial order
- Potential causality
- Another definition of concurrency
- Timestamps track message causality