Distributed Systems

CS425/ECE428

May 1st 2023

Instructor: Radhika Mittal

Logistics

- Final exam on: May 4-11
  - Reservation via PrairieTest.
  - Same format as your midterm, but longer.
  - Unless you have approved accommodations, you have 1 hour 50mins to complete the exam from the start time.
  - **Comprehensive:** includes everything covered in the course.
    - Higher weightage assigned to materials that were not covered in midterm syllabus (i.e. Raft and beyond).

PrairieLearn

- Exam format:
  - Multiple choice questions and True/False
    - For questions with multiple choices correct, there is negative marking for selecting incorrect choices to discourage guesswork (the minimum score per question is capped at zero).
  - Numerical questions
    - No step marking!

- If a subpart is not attempted or has invalid format, entire question will be left ungraded.

Exam Syllabus

- All topics covered so far
  - Midterm content
  - Post-midterm content (higher weightage, > 65%)
    - Starting from Raft, up until distributed datastores.

Exam Syllabus

- **Midterm content (included in finals)**
  - System model and Failures
  - Failure Detection
  - Clock Synchronization
  - Event ordering and Logical Timestamps
  - Global Snapshot
  - Multicast
  - Mutual Exclusion
  - Leader Election
  - Synchronous Consensus and Paxos

Exam Syllabus

- **Remaining topics (included in finals)**
  - Raft
  - Blockchains
  - Transaction Processing and Concurrency Control
  - Distributed Transactions
  - ~~External consistency and Spanner~~
  - Distributed Hash Tables (Chord)
  - MapReduce
  - Distributed Datastores

Disclaimer

- Quick reminder of the relevant concepts we covered in class.
- Not meant to be an exhaustive review!
- Go over the slides for each class.
  - Refer to lecture videos and textbook to fill in gaps in understanding.

System model and Failures

- What is a distributed system?
- Relationship between processes
- Synchronous and Asynchronous Systems
- Types of failures

Failure Detection

- Ping-ack and Heartbeats
  - what are appropriate timeout values?
- Correctness of failure detection algorithms
  - accuracy and completeness
  - synchronous vs asynchronous systems
- Performance of failure detection algorithms
  - bandwidth usage and worst-case failure detection times
- Extending to a system of N processes.

Clock Synchronization

- Clock skew and drift rates
- External vs Internal Synchronization
- Clock synchronization in synchronous systems
- Clock synchronization in asynchronous systems
    - Cristian Algorithm
    - Berkeley Algorithm
    - NTP symmetric mode synchronization

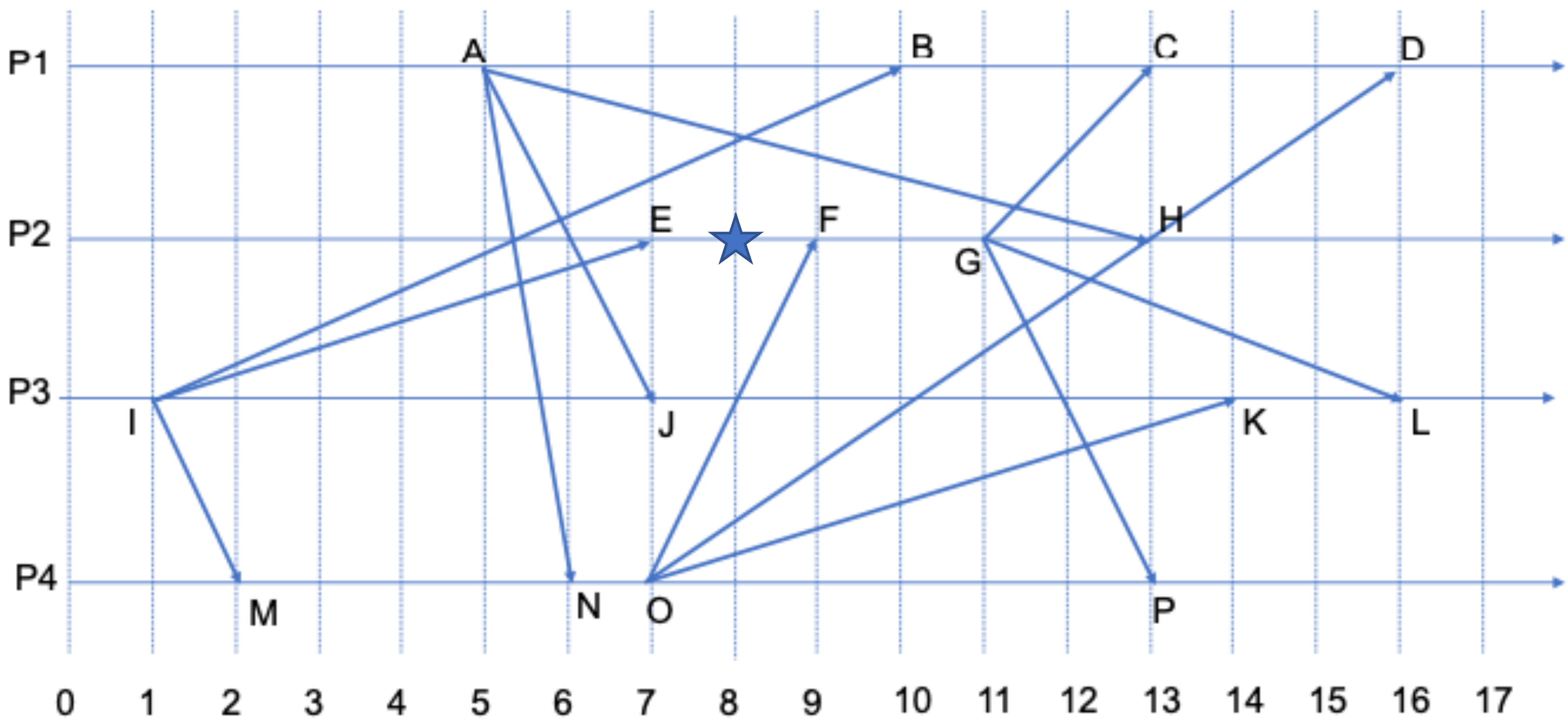
Event ordering and Logical Timestamps

- Happened before relationship
- Lamport Clocks
- Vector Clocks

Global Snapshots

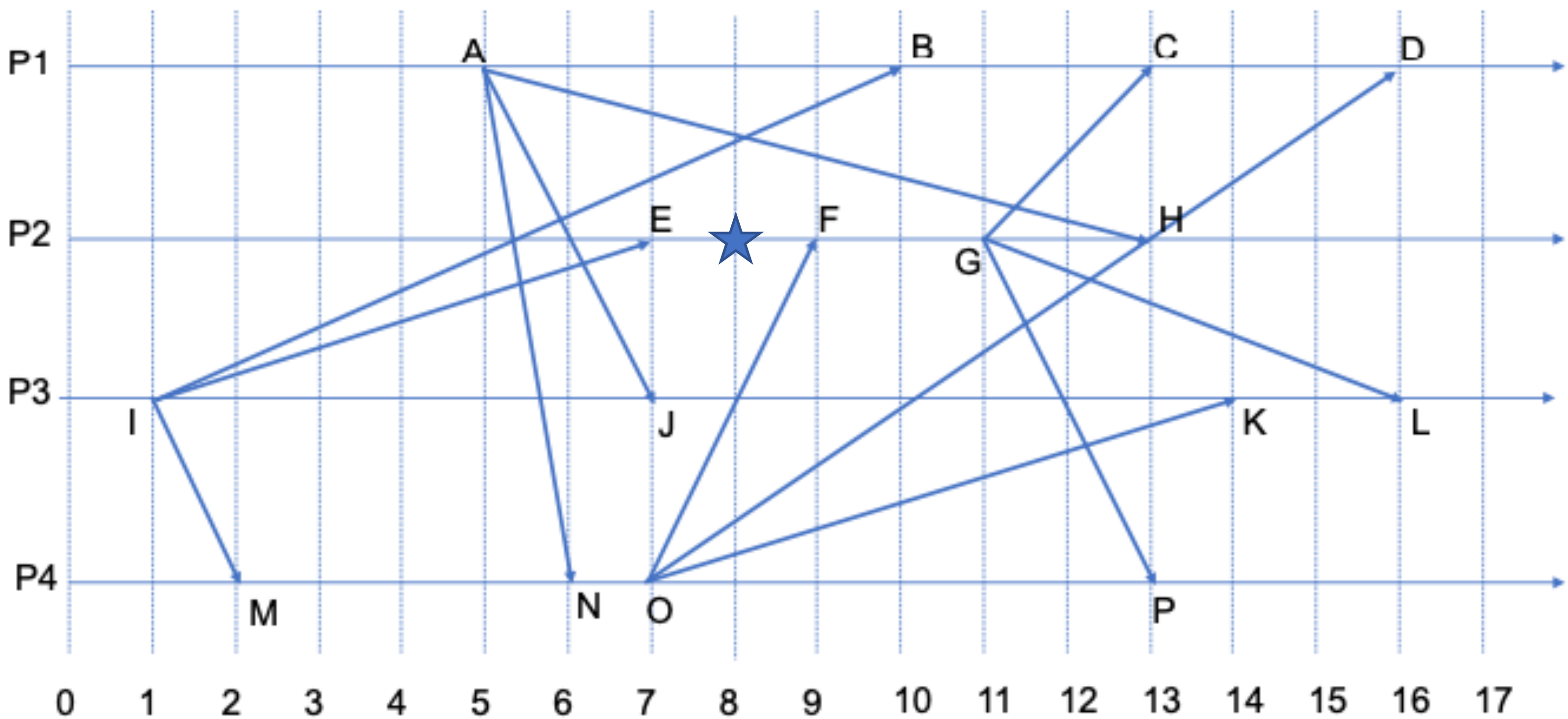
- Process and channel states
- Consistent cuts
- Chandy-Lamport algorithm
- Runs and Linearizations
- Safety and liveness properties, stable global predicates

Global Snapshots



Set of consistent cuts?

Global Snapshots



Incoming channels at P1?

Multicast

- Basic multicast
- Reliable multicast
- Ordered multicast: FIFO, Causal, Total
    - Implementing FIFO ordered multicast
    - Implementing causal ordered multicast
    - Implementing total ordered multicast
        - centralized (sequencer) algorithm
        - ISIS algorithm

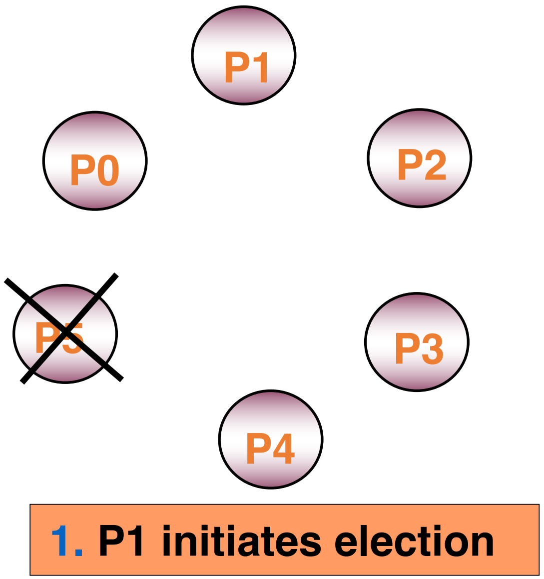
Mutual Exclusion

- Central server algorithm
- Ring-based algorithm
- Ricart Agrawala algorithm
- Maekawa algorithm (breaking deadlock not in your syllabus)
- Analyzing these algorithms:
  - Safety, liveness, and ordering
  - Client delay, Synchronization delay, and Bandwidth.

Leader Election

- Ring election algorithm (Chang and Roberts algorithm)
- Bully algorithm
- Analyzing these algorithms:
  - Safety and liveness for synchronous and asynchronous systems
  - Turnaround time and bandwidth

Bully Algorithm



To begin with,
only P1 knows of P5's failure.
No other failures.

Number of messages sent by P2?

Number of messages received by P2?

Turnaround time?

Synchronous vs Asynchronous Consensus

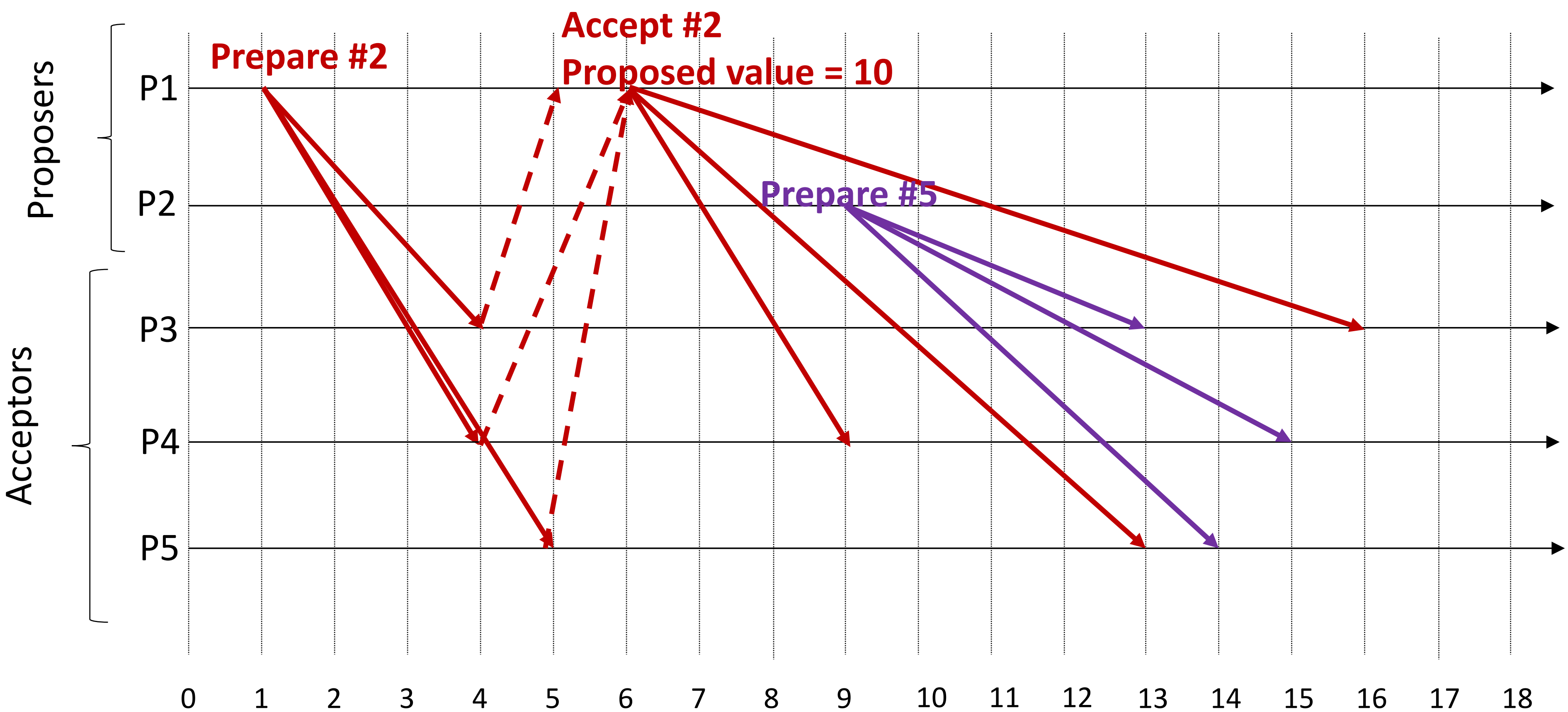
- Round-based algorithm for synchronous consensus
  - how many rounds are needed to tolerate up to f failures?

- Impossibility of consensus in asynchronous systems
  - cannot achieve both safety and liveness for consensus in an asynchronous system.
  - proof not in your syllabus.

Paxos

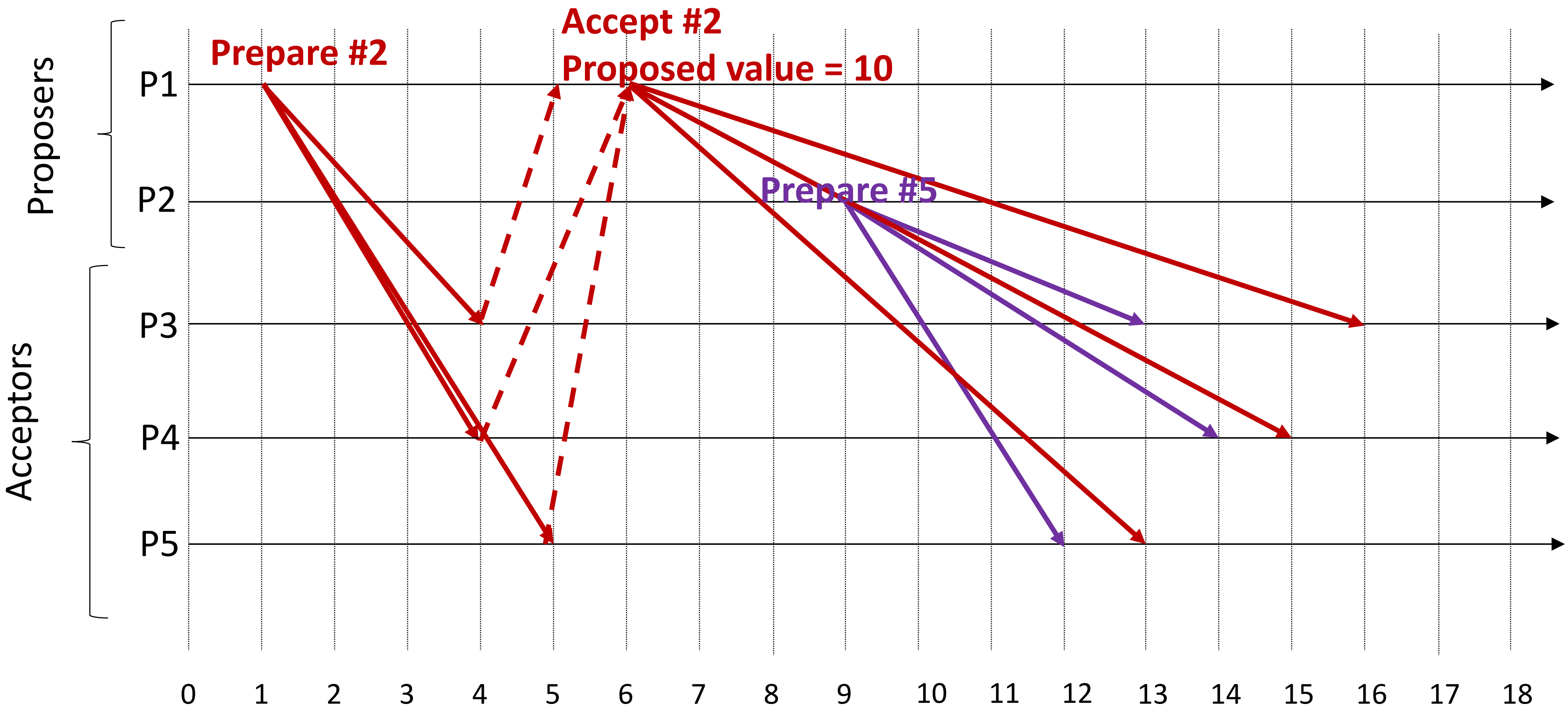
- Three roles: proposer, acceptor, learner.
- Phase 1: *prepare* request and response.
  - When will an acceptor respond with a promise?
  - What are the contents of the promise?
- Phase 2: *accept* request (if applicable)
  - When will an accept request be sent?
  - What will be the proposed value?
- When is a value implicitly decided? How is the value shared with the learners? What is required to guarantee safety?

Paxos



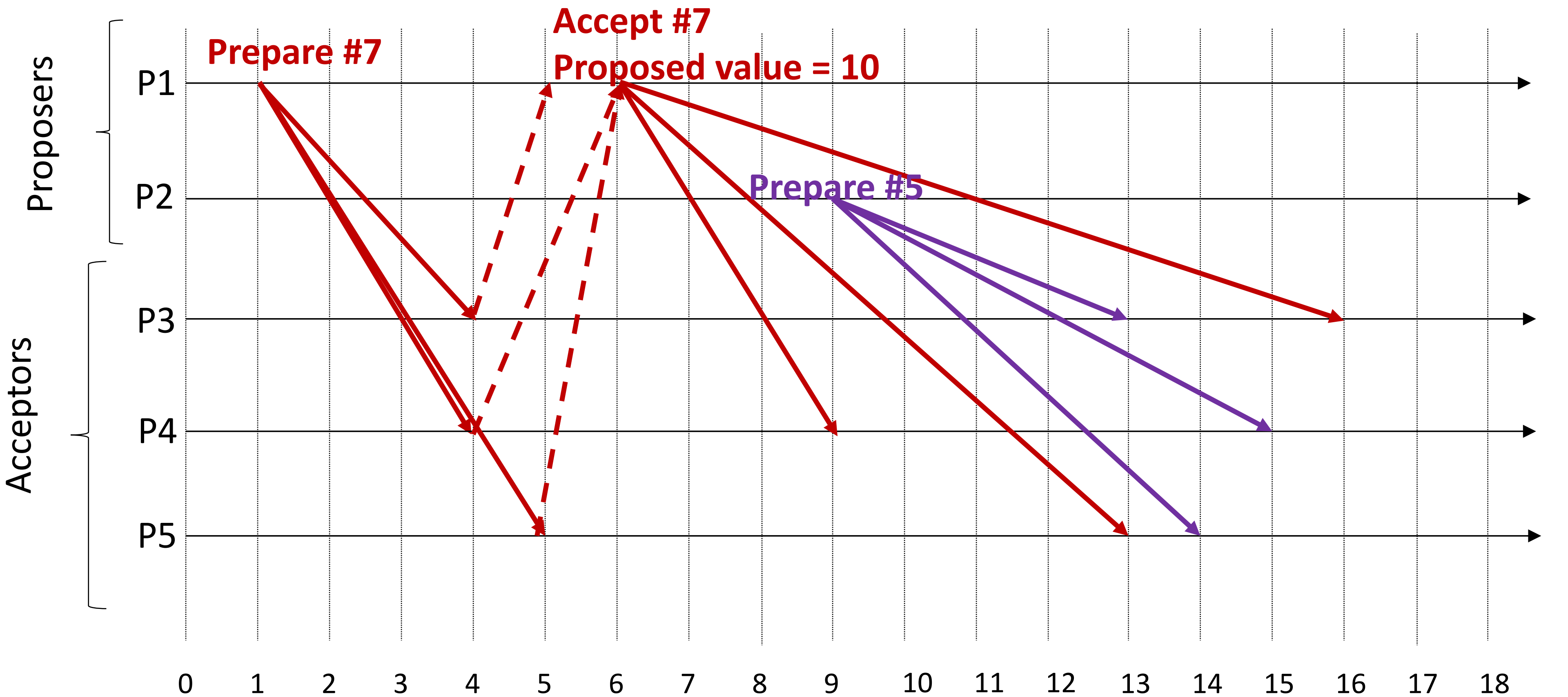
P2 intends to propose a value 15.
Will P2 send an accept request? What value will it propose?

Paxos



How about now?

Paxos



How about now?

Raft

- Algorithm for log consensus. Designed for simplicity.

- What are the guarantees provided by Raft and how?
- How is leader elected?
    - Under what conditions will a process refuse to grant vote?
- What happens when a leader fails or gets disconnected?
- How are log entries appended?
- What leads to missing / extra entries in a server's log?
- When can log entries be overwritten?
- When can log entries be committed?

Raft

- Valid or not?
  - S1: 1, 1, 1
  - S2: 1, 2, 2
  - S3: 1, 2, 3

  - S1: 1, 1, 1
  - S2: 1, 1, 2
  - S3: 1, 1, 3

  - S1: 1, 1, 3, 3
  - S2: 1, 2, 2
  - S3: 1, 1, 3

Bitcoin / Blockchains

- How is a new transaction added to the log?
    - How is a block mined, and added to a chain?
- What factors determine the rate at which a block is mined?
- What happens if two nodes mine different versions of a block?
- How is information propagated in a Bitcoin network?

Transaction Processing

- What are the ACID properties?
  - How is atomicity achieved?
  - What does consistency mean in this context?
  - What does isolation mean, and how is it achieved?
  - What is durability?

Concurrency Control

- What could go wrong if we don't have isolation?
  - Lost update problem
  - Inconsistent retrieval problem
- What are conflicting operations?
- What is serial equivalence?
- How can we check if an interleaving is serially equivalent?

Concurrency Control

- Pessimistic Concurrency Control
  - Global lock vs per-object locks vs per-object read/write locks
  - Two-phase locking
  - Deadlocks
- Optimistic Concurrency Control
  - Timestamped ordering

Concurrency Control

$T1$	$T2$
read A	
read B	
write A	
	read D
	write C
	read A
read D	
	write B
write B	
	write E

- Is this serially equivalent?

Concurrency Control

$T1$	$T2$
read A	
read B	
write A	
	read D
	write C
read D	
write B	
	read A
	write B
	write E

- What about this?
- Can it be achieved with strict two-phase locking?

Concurrency Control

$T1$	$T2$
read A	
read B	
write A	
	read D
	write C
read D	
write B	
	read A
	write B
	write E

- What about this?
- Can it be achieved with strict two-phase locking?
- Can it be achieved with timestamp ordering?

Concurrency Control

$T1$	$T2$
read A	
read B	
write A	
	read D
	write C
	read A
read D	
write B	
	write B
	write E

- What about this?
- Can it be achieved with strict two-phase locking?
- Can it be achieved with timestamp ordering?

Distributed Transactions

- Meeting ACID requirements for distributed transaction:
  - Two-phase commit for atomicity
  - Distributed deadlock detection with two-phase locking.

Distributed Hash Tables (Chord)

- What determines the placement of nodes in a Chord ring with m-bit key space?
- Which node is responsible for storing a given key?
- What are the routing table entries maintained by each node:
  - Finger tables
  - r successor entries
- What is the key lookup protocol in Chord?
- How does Chord handle churns?
  - Stabilization protocol.

MapReduce

- Map: creates intermediate key-value pairs
- Reduce: aggregate by key, and run some computation across all values for the key.
- A MapReduce chain comprises of multiple map-reduce pairs.
- Allows easier parallelization.
  - Multiple map/reduce tasks scheduled in parallel across the servers in a cluster.
- Barrier between a map stage and a reduce stage.
  - No reduce task starts before all map tasks are finished.

Distributed Datastores (Cassandra)

- What is CAP theorem?
  - Can only achieve two out of consistency, availability, and partition-tolerance.
- Cassandra: chooses availability, with *eventual* consistency
  - Key partitioning and replication strategies.
  - How is cluster membership updated?
  - How is a write query executed?
  - How is a read query executed?
  - What are the different consistency levels?
  - What is hinted-handoff and read repair?

Exam Syllabus

- **Pre-midterm**
  - System model and Failures
  - Failure Detection
  - Clock Synchronization
  - Event ordering and Logical Timestamps
  - Global Snapshot
  - Multicast
  - Mutual Exclusion
  - Leader Election
  - Synchronous Consensus
  - Paxos
- **Post-midterm (more weight)**
  - Raft
  - Blockchains
  - Transaction Processing and Concurrency Control
  - Distributed Transactions
  - ~~External consistency and Spanner~~
  - Distributed Hash Tables
  - MapReduce
  - Distributed Datastores

Good luck!