Question 1 (Paxos)
Why can an acceptor not necessarily accept the first value it receives but must sometimes accept different values?

(arrow from the video)
There might not be a majority of proposed values to determine a winner.

For example:
- 2 acceptors might have value A
- 2 acceptors might have value B
- 1 acceptor might have value C

Therefore, there won’t be one value that all servers can agree on as the majority value. An acceptor has the right to change its mind.

A value that has been accepted doesn’t mean it is chosen. It is only chosen once we have a majority of acceptors.

First check for existing proposed values. Reject older proposals (each proposal has a proposal number) received after newer ones.

October 30, 2017 © 2014 - 2016 Paul Krzyzanowski

Question 1 – Discussion

Why can an acceptor not necessarily accept the first value it receives but must sometimes accept different values?

If each acceptor just accepts a proposed value, it is possible that no acceptors get a majority of any proposed value:
- Acceptors therefore have to be able to accept different values — they may have to change their mind.
- They cannot accept every proposed value because then multiple values could be chosen.
- Once a value has been chosen, a new proposer has to abandon its value and use a previously chosen value.
  - We need a 2 phase protocol: phase 1 asks the acceptor for chosen values before proposing a value.
  - Any competing proposals have to be aborted.
  - This is done by forcing an order: newer proposals will take precedence over older proposals.

October 30, 2017 © 2014 - 2016 Paul Krzyzanowski

Paxos summary

Two-phase protocol: started by a proposer
- Phase 1: (prepare)
  - Broadcast prepare request to all acceptors.
  - Acceptors respond to a prepare request from a proposer with the highest numbered proposal that they accepted (if another proposal has already been accepted).
  - They cannot accept every proposed value because then multiple values could be chosen.
  - Will promise it will never accept a proposal number with a lower request (blocks older proposals).
- Phase 2: Send accept
  - If a majority of acceptors respond that they agree on this value, then it is chosen.

October 30, 2017 © 2014 - 2016 Paul Krzyzanowski

Question 2 (Paxos)

When does a proposer have to change the value that it is proposing during the Paxos consensus protocol?

- A proposer sends a value to an acceptor (= prepare message).
- Multiple proposers may do this concurrently.
- Acceptors respond to a prepare request from a proposer with the highest numbered proposal that they accepted (if another proposal has already been accepted).
  - Usually, this will be the same value as in the prepare message.
  - However, if multiple requests came in concurrently, an acceptor may have seen a higher number. It responds to each proposer with that higher number.
- A proposer must ask for that value to be accepted even if it initially proposed a different value.
- This does not violate the requirement of consensus since the algorithm selects one of the proposed values.

October 30, 2017 © 2014 - 2016 Paul Krzyzanowski

Question 3 (Raft)

Raft uses a single leader (one server is elected as a leader). Explain how Raft performs leader election.

Short answer:
Each candidate starts a random timer before proposing itself as a leader & sending election messages to the group.

If you receive a leader proposal message and you have not yet proposed yourself, you will acknowledge that candidate and not vote for yourself.

If a candidate gets majority votes, it becomes the leader.
Raft uses a single leader (one server is elected as a leader). Explain how Raft performs leader election.

To start an election, a candidate votes for itself and sends a request vote message to all other servers. Other servers that have not yet voted and receive the request acknowledge the candidate to be the leader. Each server that receives a request will vote for at most one candidate.

If the candidate receives a majority of acknowledgements, it becomes the leader.

If the candidate does not win or lose an election, it times out and starts a new election. Randomized timeouts are used to ensure that split votes happen rarely.

To support recovery and avoid state change, a term number is incremented after each election.

If the candidate receives a heartbeat from another server and that leader’s term # is at least as large as the candidate’s current term, then the candidate recognizes the leader as legitimate and becomes a follower.

A log entry is committed once the leader that created the entry has replicated it on a majority of the servers. Committed means that the log entry is applied to the state machine.