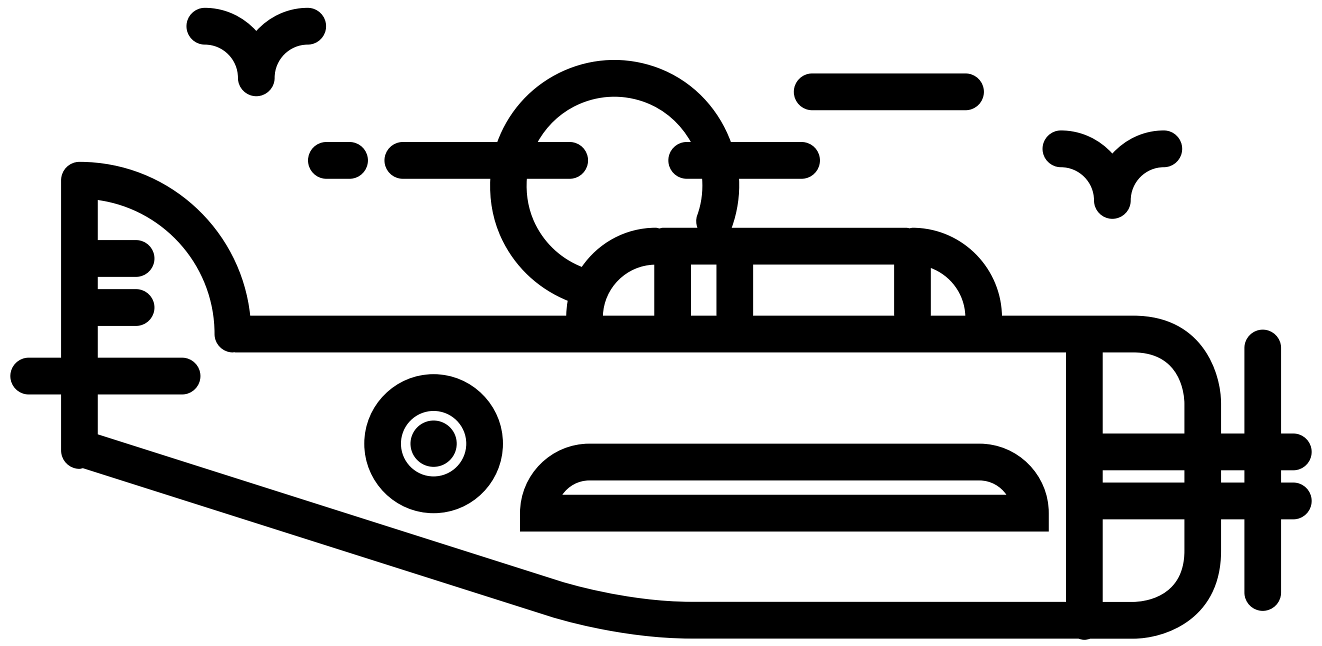
In Search of an Understandable Consensus Algorithm



What is Raft

- Another protocol for building replicated state machines.

What is Raft

- Another protocol for building replicated state machines.
- Different level of abstraction compared to **Paxos made Simple**
    - Mainly describes a protocol to keep logs consistent.
    - Describes something closer to view-stamped replication than Paxos.

What is Raft

- Another protocol for building replicated state machines.
- Different level of abstraction compared to **Paxos made Simple**
    - Mainly describes a protocol to keep logs consistent.
    - Describes something closer to view-stamped replication than Paxos.
- Claim: Easier to understand (really?)
    - Will return to this later.

System Model

System
Raft
System
Raft
System
Raft
Client

System Model

System
Raft
System
Raft
System
Raft
Request
Client

System Model

System
Raft
System
Raft
System
Raft
Request
Client

System Model

System
Raft
On Commit
System
Raft
On Commit
System
Raft
Client

System Model

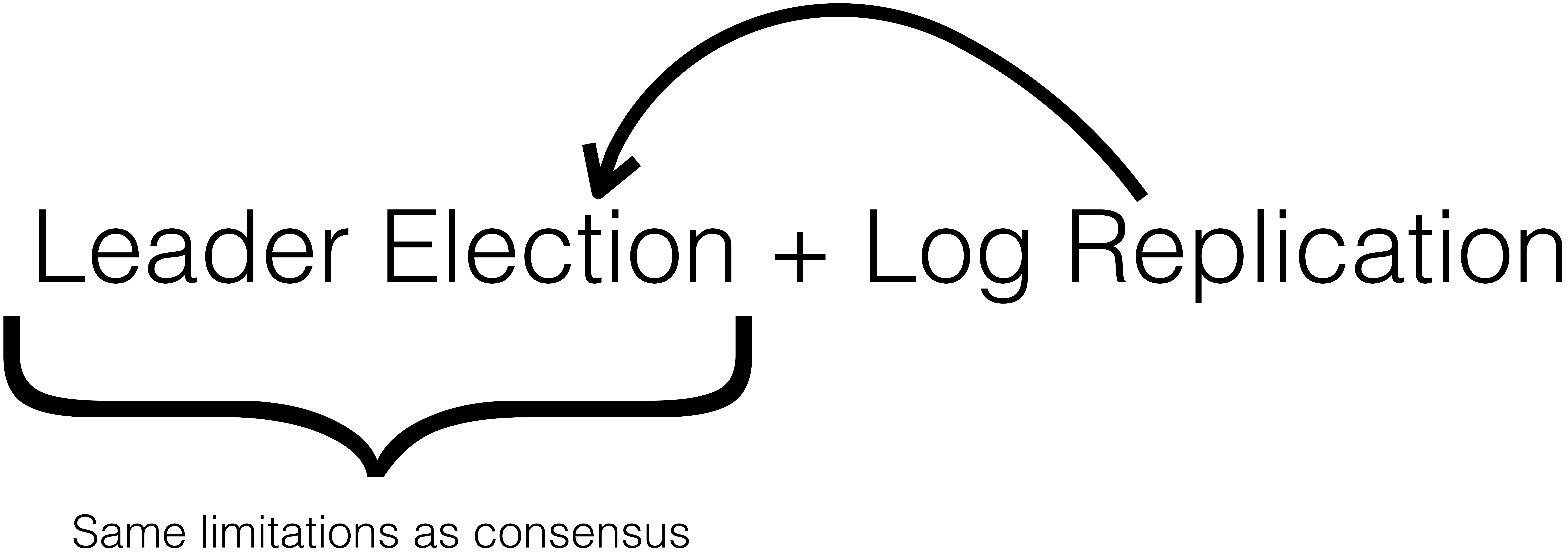
System
Raft
On Commit
System
Raft
On Commit
System
Raft
Client

Safety Requirements

- **Log Matching:** if the kth log entry for 2 logs match, so do all previous entries.
- **State Machine Safety:** all server's logs agree on committed entries.

Leader Election + Log Replication

$$\underbrace{\text{Leader Election}}_{\text{Same limitations as consensus}} + \text{Log Replication}$$



Leader Election + Log Replication

Same limitations as consensus

Log Semantics

Process 1 … (2, 7)

Process 2 … (2, 7)

Process 3 … (2, 7)

Entry (Term, Index)

Committed Log Entry

Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8)

Process 2 … (2, 7)

Process 3 … (2, 7)

Entry (Term, Index) Committed Log Entry Uncommitted Log Entry

Log Semantics

Process 1 ··· (2, 7) (3, 8)

Process 2 ··· (2, 7) (3, 8)

Process 3 ··· (2, 7)

Entry (Term, Index) Committed Log Entry Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8)

Process 2 … (2, 7) (3, 8)

Process 3 … (2, 7) (3, 8)

Entry (Term, Index) Committed Log Entry Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8)

Process 2 … (2, 7) (3, 8)

Process 3 … (2, 7) (3, 8)

Entry (Term, Index)

Committed Log Entry

Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8)

Process 2 … (2, 7) (3, 8)

Process 3 … (2, 7) (3, 8)

Entry (Term, Index)

Committed Log Entry

Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8) …

Process 2 … (2, 7) (3, 8) …

Process 3 … (2, 7) (3, 8) … (3, 11)

Entry (Term, Index) Committed Log Entry Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8) …

Process 2 … (2, 7) (3, 8) …

Process 3 … (2, 7) (3, 8) … (3, 11)

Entry (Term, Index) Committed Log Entry Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8) … (3, 11)

Process 2 … (2, 7) (3, 8) …

Process 3 … (2, 7) (3, 8) … (3, 11)

Entry (Term, Index) □ Committed Log Entry ⬚ Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8) … (3, 11)

Process 2 … (2, 7) (3, 8) …

Process 3 … (2, 7) (3, 8) … (3, 11)

Entry (Term, Index) □ Committed Log Entry ⬚ Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8) … (3, 11)

Process 2 … (2, 7) (3, 8) …

Process 3 … (2, 7) (3, 8) …

Entry (Term, Index)

Committed Log Entry

Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8) … (3, 11)

Process 2 … (2, 7) (3, 8) … (4, 11)

Process 3 … (2, 7) (3, 8) …

Entry (Term, Index) Committed Log Entry Uncommitted Log Entry

Log Semantics

Process 1 ··· (2, 7) (3, 8) ··· (3, 11)

Process 2 ··· (2, 7) (3, 8) ··· (4, 11)

Process 3 ··· (2, 7) (3, 8) ··· (4, 11)

Entry (Term, Index) □ Committed Log Entry ⬚ Uncommitted Log Entry

Log Semantics

Process 1 … (2, 7) (3, 8) … (4, 11)

Process 2 … (2, 7) (3, 8) … (4, 11)

Process 3 … (2, 7) (3, 8) … (4, 11)

Entry (Term, Index) Committed Log Entry Uncommitted Log Entry

Leader Election

- Each process has a random election timer.
    - On timeout declares itself a candidate and requests votes.
- Any follower votes for a candidate if
    - Requested term > current term
    - Candidate’s log is at least as up to date as follower’s.

More Understandable?

- What is the point of understandability?
- Easier to show correctness?
    - What is the state of the log at any step?
    - What entries are uncommitted?
    - What entries will survive?
    - Verdi spent a year on proving correctness, uncertain results.

More Understandable?

- Easier to use?
    - Is it that much easier than ZooKeeper, Chubby, etc.?

More Understandable?

- Easier to implement correctly?
    - Lots of implementations, a fair number of bugs.
        - See recent work by Colin Scott and me.

Some More Thoughts

- Raft is more “directly” usable as described in the paper.
    - Higher-level abstraction (RSM), as opposed to an algorithm
    - However, algorithm is (perhaps) easier to fit into different settings.
- Helped by when it was released
    - Distributed systems were hot, “practitioners” weren’t looking at old papers.