An Understandable Consensus Algorithm

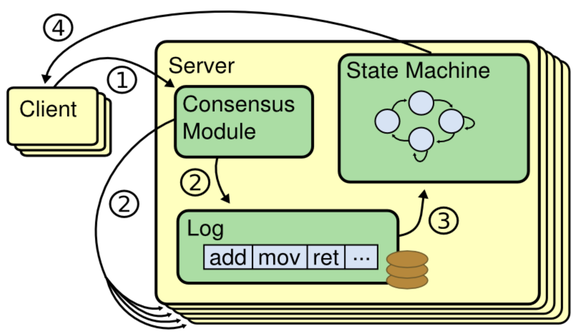
Diego Ongaro and John Ousterhout

What is consensus?

- Consensus algorithms can replicate any deterministic program to make it fault-tolerant
- Play a key role in reliable, large-scale systems
- Practical consensus algorithms are:
  - Live as long as any majority of the cluster is up (assume servers fail by stopping)
  - Efficient: require one round of communication to guarantee durability in the normal case
  - Resilient to faulty clocks and message delays

How does consensus work?

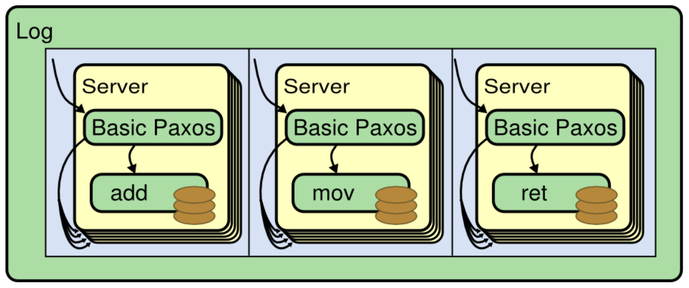
- Consensus algorithms order requests into a replicated log
- State machines on each server process same sequence of requests from replicated log



Seems easy enough?

- Paxos is the dominant consensus algorithm
- Unfortunately, it's hard to understand:
  - Difficult to teach and learn (but nice for theory)
  - Poor choice for building systems
  - Difficulty stems from its poor decomposition
- We developed Raft to be easier to understand:
  - Decomposes into logical components
  - Reduces non-determinism and state space complexity

Paxos overview



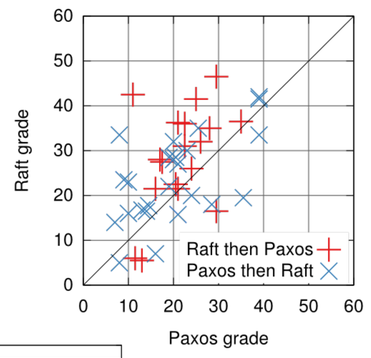
- Basic Paxos: consensus on a single log entry
- Multi-Paxos: executes one instance of Basic Paxos for each log entry
  - But doing so naïvely is inefficient
- Complete algorithm looks nothing like Basic Paxos foundation

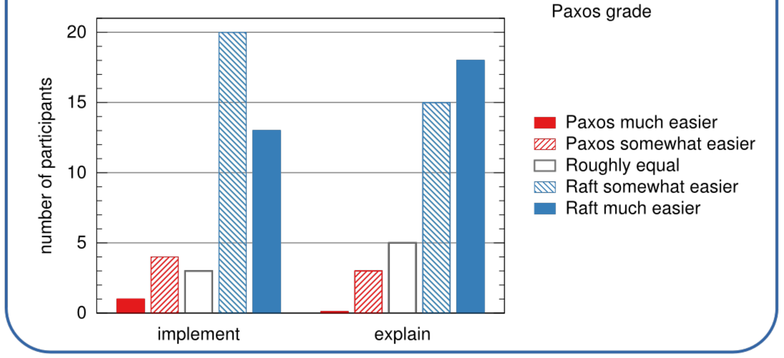
Raft overview

1. **Elect a leader:** the only server which creates new entries in the replicated log
   - Elect a replacement when the leader fails
2. **Replication**: the leader makes other servers' logs match its own, and notifies them when it's safe to execute commands
3. **Safety:** leader election is rigged to allow only servers with sufficiently up-to-date logs to win,
   - It's safe to execute commands once every future leader must have them

Which is more understandable?

- Taught students both Paxos and Raft
- 1 hour videos, 1 hour quizzes, short survey





ramcloud.stanford.edu/raft.pdf