Paxos

Lamport. Paxos Made Simple. 2001

(After many people told Leslie that he needs to rewrite his original paper).

Butler Lampson convinced Leslie + helped to popularize Paxos
What’s new

• Project schedule is posted
  • Thursday we will run a 30m speed project dating in class
  • Post your project ideas to piazza/slack
• Reading schedule finalized
• Sign up for advocate/skeptic roles!
• Grading rubric is posted
Paxos and RSMs

• **What is** the relationship between Paxos and the previous paper: replicated state machines?

• Paxos provides *agreement* and *order* from the previous paper => use Paxos to build an RSM

• Paxos/Raft/VR/… is the dominant way to build RSMs
Paxos

• What is the system model? (What is our world?)
  • Roles: Proposers, Acceptors, Learners
  • Physical processes for each, and they can be co-located arbitrarily
  • Network assumptions: async, msgs can be lost, delays arbitrarily long, msgs cannot be corrupted

• What types of failures are in scope?
  • Non-byzantine, nodes can fail, other nodes don’t necessarily find out that nodes fail (halting)
Paxos

• A pragmatic algorithm

• Guarantees that the algorithm gives us
  
  • **Safety**: agreement and order
  
  • Not necessarily **liveness** (progress): NOT necessarily “eventually a value will be “committed” ~ chosen”

• **FLP impossibility result**: you can’t have both in an async network
  
  • Either system never responds
  
  • Or system responds, but is incorrect

• Pragmatic choice in Paxos reflects the best-effort network that we have in practice
Paxos

• See whiteboard for message flow and liveness issue

• Remember that proposers must wait for promises from at least a majority before proceeding.
  
  • Learn a value that is chosen (if there is any)
  
  • Because every majority overlaps
Paxos safety guarantees

- P1: acceptor must accept first proposal it receives
  - If there is just one proposer, alg should continue (make progress)
- P1a: acceptor can accept a proposal number n iff it has not promised to a prepare with a number greater than n
  - P1a => P1
  - P1a stronger than P1
Paxos safety guarantees

- **P2**: If proposal with val v *chosen* (accepted by majority of acceptors), then higher numbered proposal *chosen* must have value v
  - Basic consensus safety: only one value is chosen (can’t undo our choice)

- **P2a**: if proposal with val v *chosen*, then every higher numbered proposal *accepted* by any acceptor has value v
  - P2a => P2
  - For a value to be chosen, it must be accepted by a majority of acceptors => accepted by at least one acceptor

- **P2b**: if proposal with val v *chosen*, then every higher numbered *proposal issued by any proposer* has value v
  - P2b => P2a => P2
  - For a value to be accepted, it must be proposed

- **P2c** (invariant): For any v and n, if a proposal with value v and n is issued, then there is a set S consisting of a majority of acceptors, such that either (a) no acceptor in S accepted any proposal less than n, or (b) v is the value of the highest-numbered prop among all props numbered less than n accepted by the acceptors in S
  - P2c => P2b (it’s a constraint on issuing a proposal with value v and number n)
  - P2c => P2b => P2a => P2

- **P2c + P2b**: constraints on proposers; **P2a + P2**: constraints on acceptors
Paxos + RSM

- RSM = deterministic FSM
- Agree on order of commands
- For every command slot $i$, run Paxos to agree on which command is executed in slot $i$
- Possibly concurrently
- A matrix of commands $X$ proposals
- Use no-ops to fill in missing commands
Paxos and liveness

• How does an acceptor catch up?
• What about fairness (btw/ proposers)?
• How to optimize the system?
• What’s actually on disk?
Next paper: ZooKeeper

- A practical RSM-based system
- Open source and used widely! https://zookeeper.apache.org/
- Under the hood, will look similar in some ways by building on Paxos and RSM paper, but different in others. Relies on an atomic broadcast protocol.
- Our first big systems paper (long!)
  - Motivation, Design, Implementation, Evaluation
  - Focus on design and think critically about evaluation