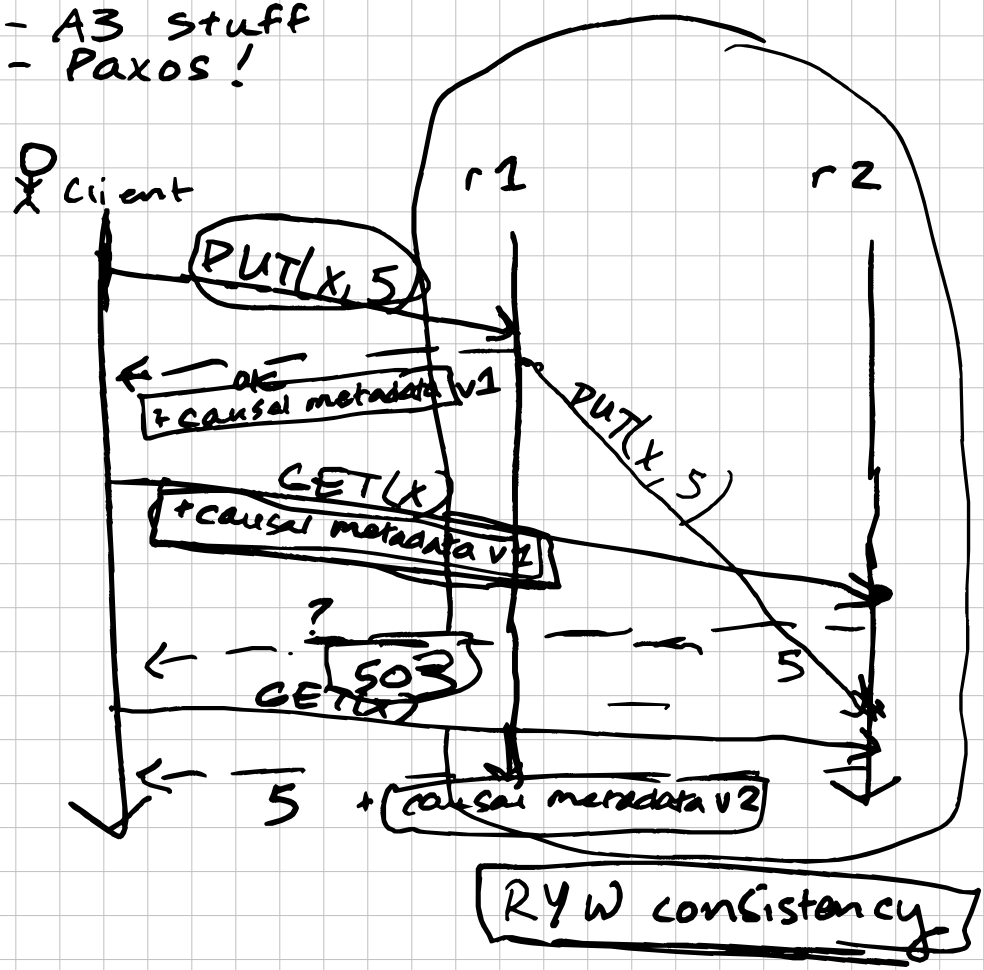
CSE 138 Lecture 12

- announcements
- A3 stuff
- Paxos!



"Strong" (Linearizable)

|

Causal

|

FIFO

|

RYW

Stronger

↕

Weaker

Let's talk Paxos!

Paxos – a family of consensus protocols invented by Leslie Lamport ~~1990~~ 1998

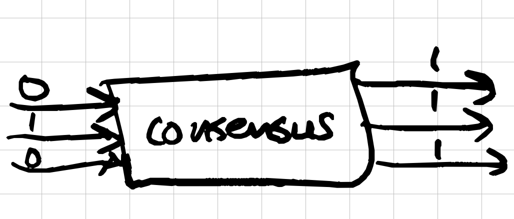
↑ "The Part-Time Parliament"

2001 – "Paxos Made Simple"

Deniz Altinbuken, R. van Renesse – "Paxos Made Moderately Complex" 2015 (?)

3 roles that a process can play:

- **proposers** propose values.
- **acceptors** contribute to choosing from the proposed values.
- **learners** learn the agreed-upon value.

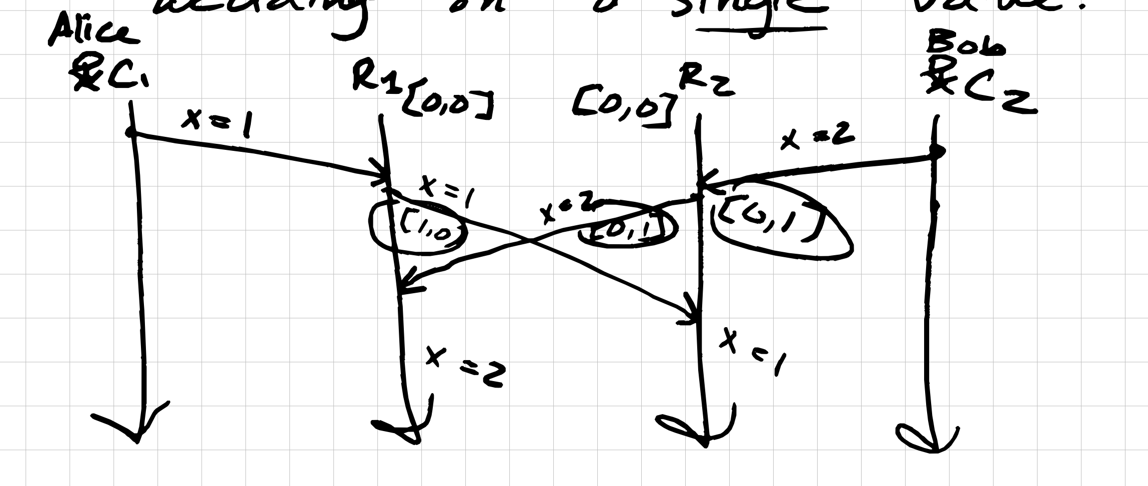


A process can play multiple roles! Any process that plays any role(s) is a Paxos node.

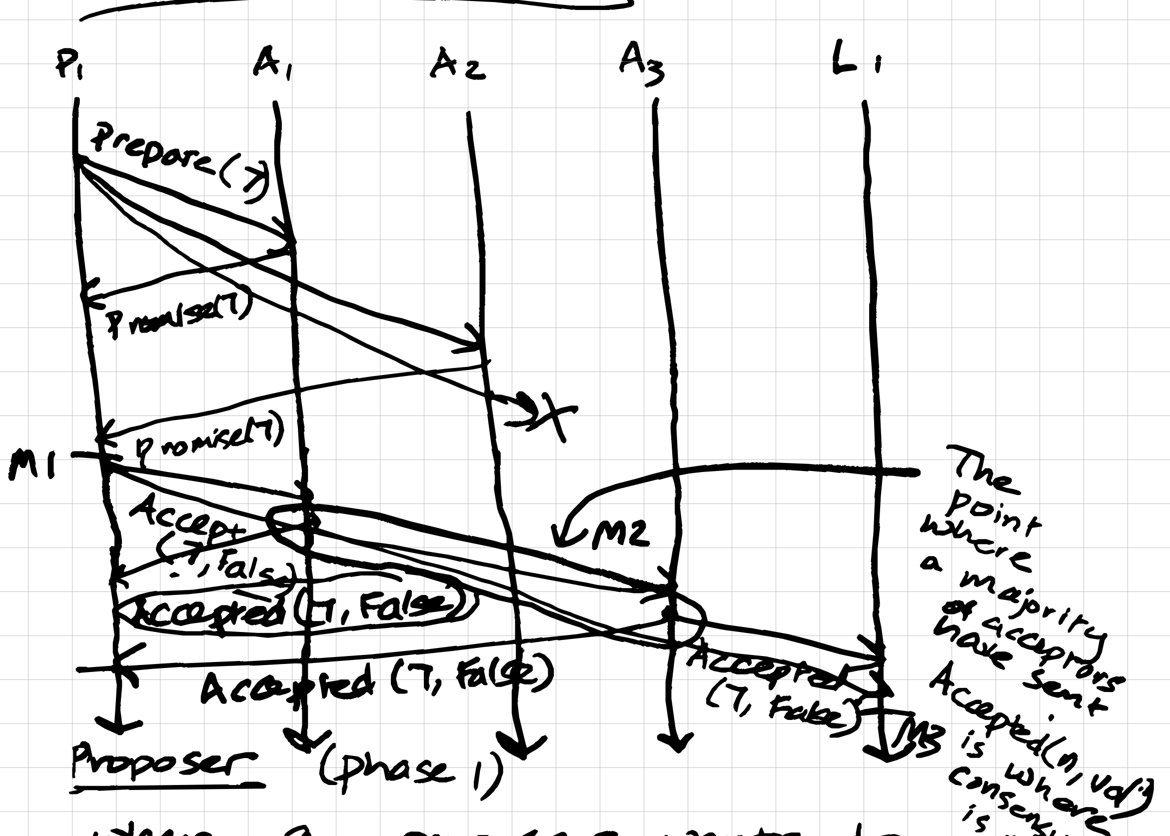
Requirements for a Paxos node:

- Paxos nodes have to know how many acceptors there are. In particular, they have to know what a majority of acceptors is.
- Paxos nodes have to be able to persist data. (in particular, they have to remember stuff like: what have I accepted!)

The version of Paxos we're discussing today is for deciding on a single value.



A run of Paxos! decide: True/False



Proposer (phase 1)

When a proposer wants to propose a value:

- it sends a Prepare(n) message to a majority of acceptors, where n is:
  - higher than any proposal number this proposer has used before
  - unique to this proposer (no one else gets to use this proposal number)

Acceptor (phase 1)

When an acceptor receives a Prepare(n) message, it looks at n, and asks:

"Did I previously promise to ignore requests with this proposal number?"

- if yes: ignores it!
- if no: it responds with Promise(n), * which means, "I will ignore any request with a proposal number lower than n.

* note: it's actually more subtle than this! (we'll come back to this)

Phase 1 ends, Phase 2 begins!

Proposer (phase 2) has received Promise(n) messages from a majority of acceptors for proposal number n.

To propose value val, it sends Accept(n, val) to a majority of acceptors. **

** = actually more complicated than this

Acceptor (phase 2)

When an acceptor receives an Accept(n, val) message, it asks: "Did I previously promise to ignore this proposal number?"

- if yes: ignores it!
- if no: responds with Accepted(n, val) and also sends Accepted(n, val) to all the Learners.

Short summary of phases

Phase 1

Proposer sends Prepare(n)

Acceptors send Promise(n)

Phase 2 (Proposers has heard from a majority w/ Promise(n))

Proposer sends Accept(n, val)

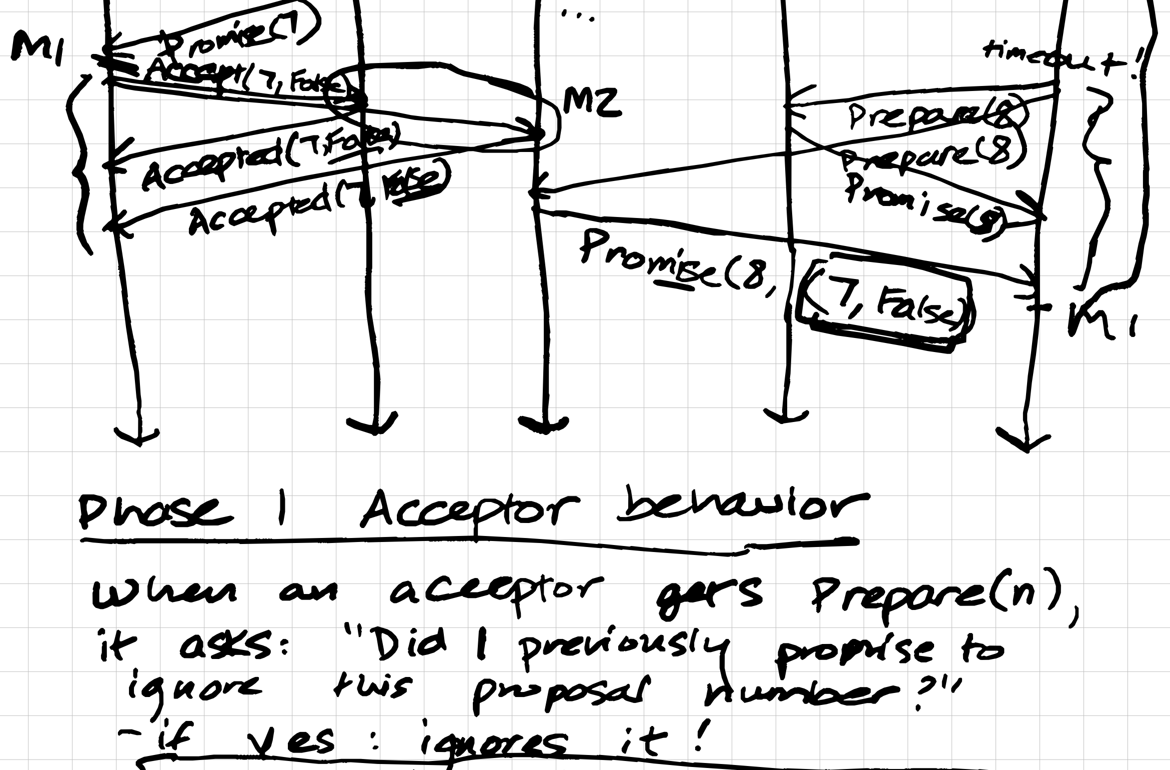
Acceptors send Accepted(n, val)

milestone 1: By getting Promise(n) messages from a majority of acceptors for a given n, A proposer gets a majority of acceptors on board with it.

milestone 2 (consensus is reached) when a majority of acceptors have sent Accepted(n, val) for a particular val.

milestone 3 — happens individually on each learner. When the learner receives Accepted(n, val) for a particular val from a majority of acceptors, that learner knows consensus has been reached.

Let's make things more interesting and have multiple proposers!



Phase 1 Acceptor behavior

When an acceptor gets Prepare(n), it asks: "Did I previously promise to ignore this proposal number?"

- if yes: ignores it!
- if no: it asks: "Have I previously accepted anything?"
  - if yes: responds to the proposor with Promise(n, (n_{prev}, val_{prev}))
    - n_{prev} = highest previously accepted proposal number
    - val_{prev} = previously accepted value.
  - if no: Promise(n).

← the new stuff