Paxos

A Consensus Algorithm for Fault Tolerant Replication
System Model

- **Replicas**
  - identical
  - fail/stop/restart failures
  - stable storage available

- **Messages**
  - possible indefinite delay
  - possible duplication or loss
  - delivered messages not corrupted

**Goal:** insure that all replicas remain identical despite replica failure and message loss.
Safety requirements

- Only a value that has been proposed (by a replica) may be chosen.
- Only a single value is chosen.
- A process never learns that a value has been chosen unless it actually has been.
Within each instance (basic) Praxos is used to arrive at a consensus of the value to be used by all replicas.

The sequence of instances determines a sequence of values accepted by all replicas.
Roles

- Proposer(s): offer proposals of the form [value, number].
- Acceptor(s): accept or reject offered proposals so as to reach consensus on the chosen proposal/value.
- Learner(s): become aware of the chosen proposal/value.

Notes:
- The proposal number is unique
- A single distinguished proposer can be elected to guarantee progress
- A single distinguished learner can be elected
- In practice, all replicas play all roles
- In practice, an elected “master” plays the roles of the distinguished proposer and the distinguished learner
Each proposer makes a proposal to some majority of the acceptors.

A majority of acceptors must accept a proposal for the proposed value to be chosen as the consensus value.

If $P_1$ and $P_2$ are making different proposals, then there must be at least one acceptor that they share in common (and this common acceptor will decide which proposal prevails).
An acceptor will accept the proposal with the largest proposal number.

A value is chosen once a majority of acceptors have accepted a proposal with that value.

Once a proposal/value is chosen all proposals with a higher proposal number are “forced” to have the chosen value.
Key idea

The property:

$P2^b$: If a proposal with value $v$ is chosen, then every higher-numbered proposal issued by any proposer has value $v$.

is guaranteed by maintaining the invariant:

$P2^c$: For any $v$ and $n$, if a proposal with value $v$ and number $n$ is issued, then there is a set $S$ consisting of a majority of acceptors such that either (a) no acceptor in $S$ has accepted any proposal numbered less than $n$, or (b) $v$ is the value of the highest-number proposal among all proposals numbered less than $n$ accepted by the acceptors in $S$. 
Paxos Protocol

**Proposer**

(a) Select proposal number \( n \) and send a *prepare* request with \( n \) to a majority of acceptors.

**Acceptor**

(b) If \( n \) greater than that of any *prepare* request to which it has already responded, then (1) respond with the highest-numbered proposal (if any) it has accepted, (2) do not accept any proposal numbered less than \( n \).

(a) If majority response received, then send *accept* request for proposal \([v,n]\) where \( v \) is the value of the highest-number proposal among the responses or any value it chooses.

(b) Accept the proposal in the *accept* request unless it has already responded to a *prepare* request having a higher number.
Chubby – applying Paxos

- A high-availability lock service
- Stores small files for applications having elected primary servers to advertise their existence and parameters
- Based on replicated architecture with elected master
- Used by GFS, Bigtable
Chubby – Paxos framework

Client app

Paxos framework

submit

value

Paxos protocol

callback

value

Replica 1

Replica 2

Replica 3

callback

value

value

value
Chubby – Replica Architecture

Chubby clients network

Replicas network

Snapshot exchange

File transfer

Paxos protocol

Local file system

Chubby protocol

RPC

Chubby

Fault-tolerant DB

File I/O

Log

Fault-tolerant Log

Snapshot