Paxos

Johan Montelius

KTH

HT15

Paxos

- or how to decide the price of olive oil.

the problem

How do we reach a consensus when:

- nodes can crash
- messages get lost
- we have no failure detectors

We might have failure detectors but we can not trust them completely.

the environment

Nodes can crash,

- but are restarted and
- will remember where in the protocol they were.

Messages can:

- take arbitrary long time to be delivered,
- get lost or get duplicated,
- but not corrupted.

Actors

- Proposers:
  - drive the execution
  - want to find a consensus
  - will inform the *learners* if consensus is reached
- Acceptors:
  - vote for proposals
- Learners:
  - wait for a consensus to be reached

outline

- Proposer: sends request with *unique sequence numbers*
- Acceptors: promise not to vote for a proposal with lower sequence number
- Proposer: collect promises and *initiate a ballot* with a proposal
- Acceptors: vote for the proposal **if they have not promised not to vote in the sequence number**
- Proposer: collect votes and if a quorum vote for the proposal then we're done

the proposer

Operates in rounds, each round using **a unique sequence number**.

In a round:
- send a request to all acceptors
- collect a quorum of promises
- **keep the proposal with highest sequence number**
- request votes for **the proposal**
- if a quorum vote for the proposal, we have reached consensus

When you're tired of waiting you start a new round.

the acceptor

Keeps track of:
- a sequence number below which it has promised not to vote
- the **accepted value** with the highest sequence number that it has voted for

If requested to promise:
- promise and
- return **accepted value and the sequence number of your vote**

If requested to vote for a proposal:
- vote, if not promised otherwise

messages

Request to promise:

> *Please do not vote in any sequence number less than 42:a.*

Promise:

> *Ok - but I have voted for 8 euros in sequence number 37:b.*

Request to vote:

> *Please vote for 8 euros in sequence number 42:a.*

Vote:

> *ok*

failures

- An acceptors need never reply on anything; the protocol will never end in more than one value being selected by a quorum.
- A proposer can abort and restart anytime; must select unique sequence number.
- Any message can get lost; which also means that you can ignore any message.
- Progress is not guaranteed; two proposers can fight forever over a quorum.

If a consensus is reached, it is the only consensus that will ever be reached.

why

Why does this work?

- Assume that one proposer has a quorum for 8 euros and another proposer has a quorum for 10 euros.
- Prove that we have a contradiction. .

- Assume that one proposer has gained a quorum for 8 euros in sequence number k.
- Assume that each quorum formed in sequence numbers k, $k+1$, .. $n-1$ has also voted for 8 euros.
- Prove that if a quorum is formed in sequence number n it will also be for 8 euros.

hmm, sounds simple

40pt Let's try.