The Part-time Parliament

Karol Kański
(kk248362@students.mimuw.edu.pl)

on the base of:

November 10, 2010
Long, long time ago...

“Early in this millennium, the Aegean island of Paxos was a thriving mercantile center. Wealth lead to political sophistication, and the Paxons replaced their ancient theocracy with a parliamentary form of government. But trade came before civic duty, and no one in Paxos was willing to devote his life to Parliament. The Paxon Parliament had to function even though legislators continually wandered in and out of the parliamentary Chamber.”
... and today

- Problem of implementing fault tolerant distributed systems.
- The state machine approach proposed by Lamport in 1978.
- The first algorithm for implementing arbitrary state machine was also by Lamport.
- The Paxon Parliament’s protocol provide another way to do that.
Parliament main task was to determine the law, which was defined by the sequence of decrees it passed. Passed decrees were recorded in legislators ledgers (each legislator maintained one ledger). Two main requirements was:

- Consistency of ledgers;
- Progress.
Assumptions

Legislators had at their disposal:

- sturdy ledgers with two sections: list of decrees (entry was never changed), notes (entry can be crossed out);
- slips of paper (may be lost outside the Chamber);
- hourglass timers;
- messengers.

Legislators and messengers were honest but was able to leave the Chamber (even forever).
Evolution of the protocol

The Single-decree Synod:
- The Preliminary Protocol
- The Basic Protocol
- The Complete Protocol

The Multi-decree Parliament:
- Multiple instances of the Complete Synod Protocol.
Synod of priests, which was ancestor of the Parliament, was choosing only one decree. The decree was obtained by a series of numbered ballots. Formally, a ballot $B$ consisted of the following four components:

- $B_{dec}$ – A decree (the one being voted on).
- $B_{qrm}$ – A nonempty set of priests (the ballot’s quorum).
- $B_{vot}$ – A set of priests (the ones who casts votes for the decree).
- $B_{bal}$ – A ballot number.

A ballot be was successful if and only if $B_{qrm} \subseteq B_{vot}$.
Let $\mathcal{B}$ be the set of ballots and define the following three conditions:

$B1(\mathcal{B})$ Each ballot in $\mathcal{B}$ has a unique ballot number.

$B2(\mathcal{B})$ The quorums of any two ballots in $\mathcal{B}$ have at least one priest in common.

$B3(\mathcal{B})$ For every ballot $B$ in $\mathcal{B}$, if any priest in $B$’s quorum voted in an earlier ballot in $\mathcal{B}$, then the decree of $B$ equals the decree of the latest of those earlier ballots.
It can be proved that if $B_1(\mathcal{B})$, $B_2(\mathcal{B})$ and $B_3(\mathcal{B})$ hold, then:

$$((B_{qrm} \subseteq B_{vot}) \land (B'_{bal} > B_{bal}) \Rightarrow (B'_{dec} = B_{dec})), $$

which leads to:

$$((B_{qrm} \subseteq B_{vot}) \land (B'_{qrm} \subseteq B'_{vot}) \Rightarrow (B'_{dec} = B_{dec})), $$

which means that if set $\mathcal{B}$ of ballots that was hold satisfies $B_1(\mathcal{B})$, $B_2(\mathcal{B})$ and $B_3(\mathcal{B})$, then the consistency is guaranteed.
The following statement can be also proved.

**Theorem**

Let $b$ be a ballot number, and let $Q$ be a set of priests such that $b > B_{bal}$ and $Q \cap B_{qrm} \neq \emptyset$ for all $B \in \mathcal{B}$. If $B1(\mathcal{B})$, $B2(\mathcal{B})$ and $B3(\mathcal{B})$ hold, then there is ballot $B'$ with $B'_{bal} = b$ and $B'_{qrm} = B'_{vot} = Q$ such that $B1(\mathcal{B} \cup B')$, $B2(\mathcal{B} \cup B')$ and $B3(\mathcal{B} \cup B')$ hold.

Which means that a balloting protocol based on B1-B3 will not deadlock.
Steps I

1. Priest p chooses a new ballot number b and sends a \textit{NextBallot}(b) message to some set of priests.

2. A priest q responds to the receipt of a \textit{NextBallot}(b) message by sending a \textit{LastVote}(b,v) message to p, where v is the vote with the largest ballot number less than b that q has cast, or his null vote null q if q did not vote in any ballot numbered less than b.

3. After receiving a \textit{LastVote}(b,v) message from every priest in some majority set Q, priest p initiates a new ballot with number b, quorum Q, and decree d, where d is chosen to satisfy B3. He then records the ballot in the back of his ledger and sends a \textit{BeginBallot}(b,d) message to every priest in Q.
Upon receipt of the $\text{BeginBallot}(b,d)$ message, priest $q$ decides whether or not to cast his vote in ballot number $b$ (he may not cast the vote if doing so would violate a promise implied by a $\text{LastVote}(b,v)$ message he has sent for some other ballot). If $q$ decides to vote for ballot number $b$, then he sends a $\text{Voted}(b,q)$ message to $p$ and records the vote in the back of his ledger.

If $p$ has received a $\text{Voted}(b,q)$ message from every priest $q$ in $Q$ (the quorum for ballot number $b$), then he writes $d$ (the decree of that ballot) in his ledger and sends a $\text{Success}(d)$ message to every priest.

Upon receiving a $\text{Success}(d)$ message, a priest enters decree $d$ in his ledger.
Each priest should store:

- $\textit{lastTried}[p]$ – the number of the last ballot that $p$ tried to initiate, or $-\infty$ if there was none;

- $\textit{prevVote}[p]$ – the vote cast by $p$ in the highest-numbered ballot in which he voted, or $-\infty$ if he never voted;

- $\textit{nextBal}[p]$ – the largest value of $b$ for which $p$ has sent a $\textit{LastVote}(b,v)$ message, or $-\infty$ if he has never sent a such message.
Steps I

1. Priest $p$ chooses a new ballot number $b$ greater than $lastTried[p]$, sets $lastTried[p]$ to $b$, and sends a $NextBallot(b)$ message to some set of priests.

2. Upon receipt of a $NextBallot(b)$ message from $p$ with $b > nextBal[q]$, priest $q$ sets $nextBal[q]$ to $b$ and sends a $LastVote(b,v)$ message to $p$, where $v$ equals $prevVote[q]$ (a $NextBallot(b)$ message is ignored if $b < nextBal[q]$).

3. After receiving a $LastVote(b,v)$ message from every priest in some majority set $Q$, where $b = lastTried[p]$, priest $p$ initiates a new ballot with number $b$, quorum $Q$, and decree $d$, where $d$ is chosen to satisfy B3. He then records the ballot in the back of his ledger and sends a $BeginBallot(b,d)$ message to every priest in $Q$. 
Steps II

4. Upon receipt of the $\text{BeginBallot}(b,d)$ message with $b = nextBal[q]$ priest $q$ cast his vote in ballot number $b$, sets the $prevVote[q]$ to this vote and sends a $\text{Voted}(b,q)$ (A $\text{BeginBallot}(b,d)$ message is ignored if $b \neq nextBal[q]$).

5. If $p$ has received a $\text{Voted}(b,q)$ message from every priest $q$ in $Q$ (the quorum for ballot number $b$), where $b = lastTried[p]$, then he writes $d$ (the decree of that ballot) in his ledger and sends a $\text{Success}(d)$ message to every priest.

6. Upon receiving a $\text{Success}(d)$ message, a priest enters decree $d$ in his ledger.
The basic protocol maintains consistency, but it cannot ensure any progress because it does not require any priest to do anything. To help achieve progress the Complete Protocol:

- requires that priests perform steps (2) - (6) of the Basic Protocol as soon as possible;
- determines when a priest should initiate a ballot.
Initiating a new ballot

Never initiating ballot will prevent progress. Initiating too many ballots will also do. In the Complete Protocol we need to know how long it took messengers to deliver messages (4 minutes) and priests to respond (within 7 minutes of the event that caused that response).

If only single priest p was initiating ballots and he was locked with majority of priests in the Chamber, then the decree would be passed and recorded in all ledgers within 99 minutes.
Choosing the president

Presidential selection requirement

If no one entered or left the Chamber, then after $T$ minutes exactly one priest in the Chamber would consider himself to be the president.

If that requirement was met, then after $T+99$ minutes every priest in the Chamber would have a decree written in his ledger.
The Parliament had to pass a series of numbered decrees. The Parliament used separate instances of the Complete Synod Protocol with the following modifications and assumptions:

- There was only one president for all instances.
- The first two steps was performed only once.
- The president immediately performs step (3) of the protocol for those instances which decree is determined by B3.
- The president would fill any gaps in his ledger by passing a meaningless decree.
Properties of the Protocol

Protocol satisfies the following property:

**Decree-ordering property**

If decrees A and B are important and decree A was passed before decree B was proposed, then A has a lower number than B.

Complexity of passing a decree:

- 3 message delays;
- about 3N messages (assuming parliament of N legislators and a quorum of about N/2);
- in busy parliament *BeginBallot* and *Success* messages can be combined leading to 2N messages.
Areas of interest

- Picking a president.
- Long ledgers.
- Bureaucrats.
- Learning the Law.
- Dishonest legislators and honest mistakes.
- Choosing new legislators.
The state machine consists of:

- set of states;
- set of commands;
- set of responses;
- function from states*commands to states*responses.

The Parliament Protocol being implementation of state machine:

- the current law is the current state of machine
- executing a command is passing a decree
Pros & cons

Advantages of implementing the state machine with the Parliament Protocol:

- less expensive;
- consistency is maintained despite the failure of any number of processes.

Disadvantages of implementing the state machine with the Parliament Protocol:

- less robust;
- does not tolerate arbitrary, malicious failures;
- does not guarantee bounded-time response.