Distributed Algorithms 60009 Coursework - Multi-Paxos

Szymon Kubica, CID: 01871147

February 26, 2023

Architecture

Liveness

My initial implementation of the section 3 followed closely the algorithm described in the paper. Each leader after receiving a `:PREEMPTED` message becomes inactive and starts pinging the leader that has preempted it, instead of picking a higher ballot number and spawning a scout immediately. That pinging functionality is achieved by spawning a failure detector module alongside each leader. When preempted, a leader sends a request to its failure detector to start pinging the leader who preempted it with the ballot number `b`, then it deactivates itself and proceeds to process other messages.

```
1       {:PREEMPTED, b} ->
2         send(self.failure_detector, {:PING, b})
3         self |> deactivate |> next
```

Listing 1: Super fancy code

The failure detector belonging to this reader maintains the highest ballot number that it has ever pinged and on receipt of the message `{:PING, b}` it checks if the incoming ballot number is higher than its stored, highest one. If so the ballot number gets updated. After that the failure detector starts pinging the leader associated with its current ballot number.

```
1  defmodule FailureDetector do
2    #  ... Setters, start function
3    defp next(self) do
4      receive do
5        {:PING, ballot_num} ->
6            if BallotNumber.greater_than?(ballot_num, self.ballot_num),
7              do: self |> update_ballot_number(ballot_num) |> ping,
8              else: self |> ping
9      end |> next
10   end
11   defp ping(self) do
12     preempting_leader = self.ballot_num.leader
13     send(preempting_leader, {:RESPONSE_REQUESTED, self()})
14     receive do
15       {:STILL_ALIVE, current_ballot, timeout} ->
16         Process.sleep(self.timeout)
17         self =
```

```
18            if BallotNumber.greater_than?(current_ballot, self.ballot_num),
19              do: self |> update_ballot_number(current_ballot),
20              else: self
21          self |> update_timeout(timeout) |> ping
22      after
23        self.timeout ->
24          send(self.leader, {:PREEMPT, self.ballot_num})
25          self|> next
26      end
27    end
28  end
```

Listing 2: Super fancy code

Evaluation