More

RSNs

Where we are

- **STATE MACHINES & STATE MACHINE REPLICATION**

- **RAFT**
  - **OVERALL STRUCTURE**
    - *LEADER* Synchronizes log with replicas
    - Decides when commands are committed
    - *An entry at index* \( I \) *once committed*
      - *is at log index* \( I \) *at all future leaders*
    - Log completeness.
  
  \( \Rightarrow \) Protocol
Leader Election

- Term

- Requirements to vote for a Candidate

- Requirement for a Candidate → Leader

When is an Entry Committed?

- Command Replication?
0. Interactions with Leader Election

<table>
<thead>
<tr>
<th></th>
<th>A log</th>
<th>B log</th>
<th>C log</th>
<th>D log</th>
<th>E log</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

\[(\text{lastLogTerm} > \text{node.lastLogTerm}) \text{ OR } (\text{lastLogTerm} == \text{node.lastLogTerm} \text{ AND } \text{lastLogIndex} > \text{node.lastLogIndex})\]

Reconfiguration

Problem: Want to add a/or remove nodes

OLD

\[\{0, 1, 2\}\]

NEW

\[\{3, 4\}\]
Need to make sure no commands are lost.

Avoid “_SPLIT BRAIN”

How:
- Store configuration in log
  \[ \text{set of active nodes} \]

- Transition
  \[ \text{OLD} \rightarrow \text{OLD+NEW} \rightarrow \text{NEW} \]
  \[ \text{Majority in OLD + Majority in New} \]

\[ \text{NEW} \]
\[ \text{OLD} \]
\[ 0 \]
\[ 1 \]
\[ 2 \]
\[ 3 \]
\[ 4 \]

\[ \text{OLD} \]
\[ 0 \]
\[ 1 \]
\[ 2 \]
\[ 3 \]
\[ 4 \]

\[ \text{NEW} \]
\[ 3 \]

Multi Paxos

- _CLIENTS_/ _REPLICAS_/ _LEADERS_/ _ACCEPTORS_

SIGH!
- Really not all that different from Raft as we will see

\[ \text{Replica} \quad \text{Leader} \quad \text{Acceptors} \]

\[ f = 1 \]

\[ \geq 2f+1 \text{ acceptors} \]

\[ \geq f+1 \text{ replicas, leaders} \]

\[ \\]

\[ \text{SYNOD Protocol} \]

\[ \text{Leader} \quad \text{Candidate} \quad \text{Leader} \quad \text{Leader} \]

\[ \text{P1A}(\text{ProcID}, \text{Ballot}) \]

\[ \text{Acceptors} \]

\[ \text{PIBC}(\text{Ballot, Cur Ballot}), \text{Accepted} \]

\[ \text{Acceptors} \]

\[ \text{Acceptors} \]

\[ \langle \text{P1A, P1B} \rangle \equiv \text{Leader Election/View Change} \]
Differences

- No log completeness check.

- Implication: Must ensure chosen leader has committed entries. How?

- When is P1A triggered & leader stability?

Scouts
Phase 2 of Replication

LEADER

P2A(\omega, \langle \text{ballot, slot} \rangle)

Acceptor

LEADER

P2B(\omega, \text{Acceptor, ballot})

CHECK WHETHER P2A SUCCEEDED OR NOT

LEADER

Acceptor

REPLICA

Adopted (slot, command)

LEADER

Count # of P2B successes:
\[ t \geq \frac{n}{2} + 1 \]

REPLICA

LEADER

REPLICA

Can execute command in slot \( i \) once:
* Command in \( i-1 \) adopted \& executed \&
* Command in slot \( i \) ADOPTED

ALTERNATE VIEW
MPaxos

Distributed Systems

Raft

* Leader logic
  * AppendEntry Request

* Candidate logic

* Follower Logic

* State Machine/Replication Logic

Some things left unspecified

* Failure Detection Logic

My contention is Raft is designed for a particular use

Multi-Paxos provides a design pattern that can be applied to different deployments

Examples (more in the paper)

Disk Paxos - Acceptors are disks

* Why?
* What Changes?

Acceptors Interface: Read Block, Write Block

Not Active

Mencius/E Paxos: Consensus in the Wide Area

Consensus in Practice: Lock Services