RAFT: An RSM Protocol

Final Project Proposals

- Group Size
- Topics
- Proposal Length

Midterm Observations
Succinctness:

Don't tell me about safety & liveness
In a question about fairness

On proofs

→ Can assume anything covered in class
Don't need to prove FLP

→ But must show why
Why is WFD consensus?

→ Must be logically consistent

Where we last left off

- State machines

- Replicated state machines

- Consensus

→ Impossibility & workarounds

Today: An RSM protocol
Invariants

1. Leader's Log Is "Authoritative"

   Protocol makes sure replica logs == leader logs

2. If a majority of replicas have log entry E at index I, then all future leaders will have log entry E @ I

   Safe to execute command at index I once
Leader Election Protocol Ensures Stability of a Sufficiently Replicated Entry

Leader Counts # of Times an Entry is Replicated & Decides When Commands Are Executed

RAFT Specific: Log Completeness

If Index I is Committed
Then Index I-1 Must Be Committed

Split Protocol into Four Portions

1. Log Synchronization
   How the Leader Adds to the Log

2. Failure Detection
   How Followers Detect Leader Failures

3. Failure Recovery
   How a New Leader Is Chosen
Adding New Machines & Removing Existing Ones

Log Replication

Log Entry
- Index
- Term
- Command
- ...

\[ \text{A log} \]
\[ \text{B log} \]
\[ \text{C log} \]

\[ \begin{array}{cccccccc}
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
2 & 2 & 2 & 3 & & & & & \\
2 & 2 & 2 & & & & & & \\
2 & 2 & & & & & c & c & c \\
\end{array} \]

AER \( t=3 \), \( pli=4 \)
Entries: \[ \{c_0, c_2, c_7\} \]

\[ \begin{array}{cccccccc}
A & B & C \\
A & & & & & & & \\
B & & & & & & & \\
C & & & & & & & \\
\end{array} \]

Time

AER \( t=3 \), \( pli=3 \)
Entries: \[ \{c_0, c_8, c_{12}, c_{17}\} \]

\[ \begin{array}{cccccccc}
A & B & C \\
A & & & & & & & \\
B & & & & & & & \\
C & & & & & & & \\
\end{array} \]

Time

Do I have to execute \( c_0 \)?
Q. When can A execute transmission?

After C

Q. When can A respond to C?

After redaction/after C

(2) FAILURE DETECTION

Problem: Leaders initiate all communication.

Silence → Failure

A

B

C

Time
Solution: Don't let leaders be silent for too long $\Rightarrow$ Heartbeats

Q$: Heartbeat frequency?

Q$: How long should a follower wait before assuming leader has failed? (ELECTION TIMEOUT)

Randomized Timers

Q$: How accurate is this process? Can a follower incorrectly suspect a live 'leader'?

Yes
Failure Recovery/Leader Election

If a majority of replicas have log entry E at index I, then all future leaders will have log entry E @ I

Requirements

(R1) Any follower can initiate leader election
Randomized election timeout

(R2) Must hold for any follower that becomes a leader
Candidate must compare log with a majority of replicas

(R3) At most 1 leader at a time.
R3o Term Numbers.

- Replicas remember highest term number they have sent or received.
- Reject any messages with lower term.
- Observation: Replica's log can only be changed by leader with correct term.
- Must ensure at most one leader per term.

R1 + R2 = at most one leader per term

Any follower can become candidate
  (on election timeout)

\( \rightarrow \) increments term number
  \((\text{term} + 1)\)

Implications?

(2) broadcasts Request Vote Message to all replicas.

Request Vote (term, id, lastLogIndex, term)
Claim: (Index, Term) completely identifies a log entry. Why?

3) When a replica receives ReqVote
   1) Update Term if necessary
   2) Ask if already voted for this term (why)
   
   If Voted → Do NOT Grant Vote

3) \( \begin{cases} \text{if } \text{LastLogTerm} > \text{Node.LastLogTerm} \text{ OR } \\
\text{Log Matching} \text{ } \\
\text{if } \text{LastLogTerm} \equiv \text{Node.LastLogTerm} \\
\text{AND } \text{LastLogIndex} \geq \text{Node.LastLogIndex} \end{cases} \) Vote yes

else vote no

3) Wait for response from a
3. Waits for response from majorities. If a majority vote yes $\Rightarrow$ candidate becomes leader.

Requirements

1. At most one leader per term. Why?

2. If a majority of replicas have log entry $E$ at index $I$, then all future leaders will have log entry $E \geq I$.

Claim 0: $$(\text{LastLogTerm} > \text{Node.LastLogTerm}) \lor$$ $$(\text{LastLogTerm} = \text{Node.LastLogTerm} \land \text{LastLogIndex} > \text{Node.LastLogIndex})$$
If a majority of replicas have log entry E at index i, then all future leaders will have log entry E @ T.

Why?

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>A log</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B log</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C log</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Other issues with leader election:
- Entries from previous terms.
Problem: Larger term dominates.

4 Reconfiguration

- Add or remove nodes.

Problem: What constitutes a majority?

Solution: Require majority in both old and new configuration.
Commit configuration changes using Raft.