FLEXIBLE PAXOS:
Quorum Intersection Revisited [1]

Heidi Howard, Dahlia Malkhi, and Alexander Spiegelman

Presented by Jiayi Liu and Pranav Srinivasan
AGENDA

• Brief Review of Traditional Paxos
• A Key Observation
• Definition of Flexible Paxos
• Why Flexible Paxos Works
• Advantages
• Disadvantages
• Enhancements
TRADITIONAL PAXOS
TRADITIONAL PAXOS

- 2 Phases
  - Phase 1: Prepare & Promise
    - Prepare waits for a majority of replicas to promise
  - Phase 2: Propose & Accept

- Values are learned from a majority of accepts
PERFORMANCE OF TRADITIONAL (MULTI)PAXOS

Credit: [1]
A KEY OBSERVATION
A KEY OBSERVATION

- There are actually 2 different types of quorums of acceptors in the protocol
  - Quorum to establish yourself as new leader (Q1)
  - Quorum to learn a proposed value by accepts (Q2)

**Diagram:**
- Proposer
- Learner
- Acceptors: 1, 2, 3
- Q1 = \{2,3\}
- Q2 = \{1,2\}
A KEY OBSERVATION

QUORUM INTERSECTION ISN’T NECESSARY INTRA-PHASE, BUT ONLY ACROSS PHASES
A KEY OBSERVATION

- Traditional Paxos achieves this by requiring majorities for Q1 and Q2
- Any 2 majorities intersect

Majority Quorums aren't the only way to achieve this!
FLEXIBLE PAXOS
DEFINITION OF FLEXIBLE PAXOS

- **Quorum System**: A scheme to ensure Q1 and Q2 intersect
- **Flexible Paxos**: A generalization of Paxos that relies on different quorum systems
SIMPLE QUORUM

• Define the 2 quorum sizes independently, provided they satisfy the following equation

\[ |Q_1| + |Q_2| > N \]

• Note that traditional Paxos meets this criterion
PROOF OF CRITERION

$|Q_1| + |Q_2| > N \Rightarrow |Q_1| + |Q_2| \geq N + 1$

$\Rightarrow |Q_1| \geq (N - |Q_2|) + 1$
APPLICATIONS OF SIMPLE QUORUM

- Can reduce the size of Q2 at the cost of increasing Q1 size (maintains the criterion)
- Common case (phase 2) fast
- An extreme application: With $|Q1| = N$, we can have $|Q2| = 1$
- One ACK to learn!
LIMITATIONS OF THE SIMPLE QUORUM

- As we reduce $|Q_2|$, we improve throughput and decrease latency
- However, our fault tolerance is limited by $|Q_2|$

$$f = \begin{cases} [0, |Q_2| - 1] & \text{normal operation} \\ [|Q_2|, N - |Q_2|) & \text{no new leaders} \\ [N - |Q_2|, \infty) & \text{system down} \end{cases}$$
GRID QUORUMS

- At first glance, Flexible Paxos seems to introduce the following tradeoff

\[ |Q_1| \quad \text{vs.} \quad |Q_2| \]

- With Grid Quorums, we overcome this tradeoff by introducing a new restriction: allowing only specific acceptor quorums
INTUITION: MACHINES IN GRIDS

- Our newly defined quorums must consist of specific, prechosen machines
- Recall, in traditional Paxos:
  \[ Q_1 \cap Q_2 \neq \emptyset \]
  \[ Q_1 \cap Q'_1 \neq \emptyset \]
  \[ Q_2 \cap Q'_2 \neq \emptyset \]
"GRID-PAXOS" TOLERANCE

- Worst case, we can only tolerate
  \[ \min(N_1, N_2) \]

- Best case, we can tolerate
  \[ (N_1 - 1) \times (N_2 - 1) \]
**FLEX PAXOS ON THE GRID**

- With Flexible Paxos, only Q1 and Q2 must intersect
- We can have each row be a Q1 quorum candidate $N_2$
- Each column be a Q2 quorum candidate
- Works like Bingo!
FLEX PAXOS ON THE GRID

- Any Q1 and Q2 quorums intersect!

\[ N_2 \]
\[ N_1 \]
GRID FLEX PAXOS TOLERANCE

- Not all failures are equally bad
- Example: N2 failures

\[ N_2 \]

\[ N_1 \]
TOLERANCE, EXTREME CASES

- Minimally,
  \[ f = N_1 \]
  can destroy the system
  (all learners down)

- Optimistically we can survive

\[ f = (N_1 - 1) \times (N_2 - 1) \]
"PROOF" OF FLEXIBLE PAXOS
Theorem 1: If value $v$ is decided with proposal number $p$ and $v'$ is decided with proposal number $p'$, then $v = v'$

"Once a value is learned, it does not change (no different value can be learned)"

Theorem 2: If value $v$ is decided with proposal number $p$, then for any message propose$(p', v')$ where $p' > p$ then $v = v'$

"Any new leader will always propose the learned value"
WHY IT WORKS

For any learned value, there is a Q2 quorum of processes that have accepted that value.

When a prepare() receives promises from |Q1| acceptors...

\[ Q_1 \cap Q_2 \neq \emptyset \]

Meaning at least one process will promise with the learned value to the new proposer.
ADVANTAGES
As $|Q2|$ increases from FPaxos1 to P, performance decreases.
WE CAN DO BETTER...

- Broadcast vs. signalling specific quorums

\[ |Q_2| = 2 \]
DISADVANTAGES
DRAWBACKS :(

- As we demonstrated, Flexible Paxos can end up relying on specific processes not crashing
- We sacrifice fault tolerance for quickness of the Q2 quorum
- Difficulty in implementation
ENHANCEMENTS
RELAXING THE GOLDEN RULE

\[ Q_1 \cap Q_2 \neq \emptyset \]

Do we need a Q1 to intersect every Q2?
RELAXING THE GOLDEN RULE

- Realistically, in phase 1 we only need a Q1 to intersect with all the Q2s that were previously *used*.
- If a Q2 isn't used, then it contains no info about a learned value.
- Ignore it!
CASE STUDY

\[ N = 100f \]

\[ |Q_2| = f + 1 \quad \text{(specific machines)} \]

Future leaders only need to intersect with one of these \( f + 1 \) machines,

can tolerate up to \( N - f \) failures
Q & A