FLAIR: Accelerating Reads with Consistency-Aware Network Routing

Hatem Takruri, Ibrahim Kettaneh, Ahmed Alquraan, Samer Al-Kiswany
Introduction

Modern cloud applications
• Are read intensive
  • R:W in Google’s F1 advertising system is 380:1 [1]
  • R:W in Facebook’s TAO is 500:1 [2]
• Require data reliability

Main approach: Replication

Strongly consistent replication protocols are popular

Inefficiency of current replication protocols

Modern strongly consistent protocols are inefficient for read-heavy workloads

Main reason: they are leader-based

Raft  Log Cabin  Paxos  Viewstamps  Replication
ZAB
Leader-based Consensus Inefficiency

WriteRequest ➔ Leader
Replicate

Followers
Leader-based Consensus Inefficiency

Inefficient: Only the leader handles read/write requests

Missed opportunity: Utilize the followers to serve reads.
Current Approaches to Utilizing Followers

Read leases
• The leader grants read leases to followers
• With a valid lease: followers serve reads
• On write: leader revokes leases
• **Drawbacks**
  • Complicates lease management
  • Increases write latency
  • Complicates fault tolerance

Eventual consistency
• Replicas serve reads, albeit the possibility of returning stale data
FLAIR: Fast, Linearizable, Network Accelerated Client Reads

A novel approach to serve reads from followers while maintaining linearizability.

A shim layer atop current leader-based protocols.

Main idea
- The network detects read/write conflicts, and
- Load balance reads across consistent replicas

Enabler: Programmable switches
FLAIR in a Nutshell

- Network switches monitor write requests/responses
- Identify which objects are stable, and on which replicas
- Load balance reads across consistent replicas

FLAIR is an in-network consistency-aware load-balancing protocol
Evaluation Summary

• Implemented FLAIR using P4
• Evaluated FLAIR on a cluster with Barefoot Tofino switch

FLAIR achieves
• $1.4 \times$ to $2.1 \times$ higher throughput
• $1.5 \times$ to $2.4 \times$ lower latency
Outline

• Overview of programmable switches
• FLAIR design
• Implementation
• Evaluation
Programmable Switches Overview

Example:
key-based routing pipeline

Client

GET: Key = 200

L2 L3

Key range: [0, 1000)

Key range: [1000, 2000)

Node 1

Node 2
Programmable Switches Overview

Example:
key-based routing pipeline
Programmable Switches Overview

Example: key-based routing pipeline

<table>
<thead>
<tr>
<th>Match</th>
<th>Action</th>
<th>Packet header and metadata</th>
</tr>
</thead>
<tbody>
<tr>
<td>header.key ∈ [0, 1000)</td>
<td>forward(0)</td>
<td>L2 Table</td>
</tr>
<tr>
<td>header.key ∈ [1000, 2000)</td>
<td>forward(1)</td>
<td>IPV4 Table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KV Routing Table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ACL Table</td>
</tr>
</tbody>
</table>
Programmable Switches Overview

Example:
key-based routing pipeline

<table>
<thead>
<tr>
<th>Match</th>
<th>header.key $\in [0, 1000)$</th>
<th>header.key $\in [1000, 2000)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>forward(0)</td>
<td>forward(1)</td>
</tr>
</tbody>
</table>

forward(index):
$$\text{dstIPAddr} = \text{addressArray}[\text{index}]$$

addressArray:
<table>
<thead>
<tr>
<th></th>
<th>IP 1</th>
<th>IP 2</th>
<th>IP 3</th>
</tr>
</thead>
</table>

Facilitates building application-optimized network substrate.
Programmable Switches Challenges

• No loops or recursion
• Restricted pipeline-based programming model
• Limited number of pipeline stages
• Limited computational power
• Restricted memory access model

Can we use programmable switches to build consistency-aware network routing protocol?
Outline

• Overview of programmable switches
• FLAIR design
• Implementation
• Evaluation
Which nodes can serve Read(Key1)?

FLAIR Design

Clients

Network controller

Read(Key1)

FLAIR pipeline

Which nodes can serve Read(Key1)?

Flair modules

Follower1

Follower2

Leader
### FLAIR’s Object Stability Array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Stable</td>
<td>Unstable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>L, F1</td>
<td>All</td>
<td>-</td>
<td>...</td>
</tr>
</tbody>
</table>
FLAIR in Action

Read(KeyHash = 5000)

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Stable</td>
<td>Stable</td>
<td></td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>All</td>
<td>All</td>
<td></td>
</tr>
</tbody>
</table>

Leader
- Key = 5000, Value = A

Follower 1
- Key = 5000, Value = A

Follower 2
- Key = 5000, Value = A
FLAIR in Action

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Stable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>All</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Read(KeyHash = 5000)

Check

Leader

Key = 5000
Value = A

Follower 1

Key = 5000
Value = A

Follower 2

Key = 5000
Value = A
FLAIR in Action

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Stable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>All</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Read(KeyHash = 5000)

ReadResponse(KeyHash = 5000, Val = A)

Key = 5000
Value = A

Leader

Follower 1

Key = 5000
Value = A

Follower 2

Key = 5000
Value = A
FLAIR in Action

Write(KeyHash = 5000, Val = B)

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Stable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>All</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Leader

Key = 5000
Value = A

Follower 1

Key = 5000
Value = A

Follower 2

Key = 5000
Value = A
FLAIR in Action

```
<table>
<thead>
<tr>
<th>Key range</th>
<th>Status</th>
<th>Stable Replicas</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0, 4096)</td>
<td>Stable</td>
<td>All</td>
</tr>
<tr>
<td>[4096, 8192)</td>
<td>Unstable</td>
<td>-</td>
</tr>
<tr>
<td>[8192, 12288)</td>
<td>Stable</td>
<td>All</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
```

Write(KeyHash = 5000, Val = B)

Leader
- Key = 5000
- Value = A

Follower 1
- Key = 5000
- Value = A

Follower 2
- Key = 5000
- Value = A
FLAIR in Action

Write(KeyHash = 5000, Val = B)

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0, 4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Unstable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>-</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Leader

Follower 1

Follower 2
FLAIR in Action

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Unstable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>-</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Read(KeyHash = 5000)

Write(KeyHash = 5000, Val = B)

Leader
- Key = 5000
- Value = A

Follower 1
- Key = 5000
- Value = A

Follower 2
- Key = 5000
- Value = A
FLAIR in Action

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>(4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Unstable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>-</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Read(KeyHash = 5000)

Check

Write(KeyHash = 5000, Val = B)

Leader

Key = 5000
Value = A

Follower 1

Key = 5000
Value = A

Follower 2

Key = 5000
Value = A
FLAIR in Action

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Unstable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>-</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Write(KeyHash = 5000, Val = B)

ReadResponse(KeyHash = 5000)

Leader

Follower 1

Key = 5000 Value = A

Follower 2

Key = 5000 Value = A

Read(KeyHash = 5000)
FLAIR in Action

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0, 4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Unstable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>-</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Write(KeyHash = 5000, Val = B)

Leader

Key = 5000
Value = A

Follower 1

Key = 5000
Value = A

Follower 2

Key = 5000
Value = A
FLAIR in Action

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Unstable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>-</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Leader

Key = 5000
Value = B

Follower 1

Key = 5000
Value = B

Follower 2

Key = 5000
Value = A
FLAIR in Action

WriteResponse(Key1, [L,F1])

Key = 5000
Value = B

Leader

Key = 5000
Value = B

Follower 1

Key = 5000
Value = A

Stale Follower

Follower 2

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Unstable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>-</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>
FLAIR in Action

WriteResponse(Key1, [L, F1])

Update

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Stable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>L, F1</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Leader
- Key = 5000
- Value = B

Follower 1
- Key = 5000
- Value = B

Stale Follower
- Key = 5000
- Value = A

Follower 2
- Key = 5000
- Value = B
FLAIR in Action

WriteResponse(Key1, [L,F1])

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>Status</th>
<th>Stable Replicas</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0,4096)</td>
<td>Stable</td>
<td>All</td>
</tr>
<tr>
<td>[4096, 8192)</td>
<td>Stable</td>
<td>L, F1</td>
</tr>
<tr>
<td>[8192, 12288)</td>
<td>Stable</td>
<td>All</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Key = 5000
Value = B

Leader

Stable Replicas: All

Follower 1

Stable Replicas: L, F1

Follower 2

Stable Replicas: All

Stale Follower
FLAIR in Action

Objects stability array

<table>
<thead>
<tr>
<th>Key range</th>
<th>[0,4096)</th>
<th>[4096, 8192)</th>
<th>[8192, 12288)</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Stable</td>
<td>Stable</td>
<td>Stable</td>
<td>...</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>All</td>
<td>L, F1</td>
<td>All</td>
<td>...</td>
</tr>
</tbody>
</table>

Read(KeyHash = 5000)

Check

Leader

Follower 1

Stale Follower
FLAIR Design

• Concurrent writes to the same object
• Packet reordering
• Failures
  • Switch failure
  • Leader failure
  • Follower failure
  • Network partitioning

Protocol validation
• Detailed proof
• TLA+ model checking
Concurrent Writes

\[ w_1(x) \]

\[ w_2(x) \]

\[ x \text{ is unstable} \]

Time on the switch

\[ w_1 \text{request} \]

\[ w_2 \text{request} \]

\[ w_1 \text{response} \]

\[ w_2 \text{response} \]
Concurrent Writes

\[ \text{\( w_1(x) \)} \quad \text{\( w_2(x) \)} \]

Time on the switch

\( x \) is unstable
Concurrent Writes

- Every write gets a unique sequence number
- Objects stability array stores the sequence number of the last write
Concurrent Writes

- Every write gets a unique sequence number.
- Objects stability array stores the sequence number of the last write.
- Objects remain unstable until last sequence number is acknowledged.

Unstable objects array

<table>
<thead>
<tr>
<th>Hash range</th>
<th>[0,4096)</th>
<th>…</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Unstable</td>
<td>…</td>
</tr>
<tr>
<td>Stable Replicas</td>
<td>-</td>
<td>…</td>
</tr>
<tr>
<td>Seq#</td>
<td>w2_seq#</td>
<td>…</td>
</tr>
</tbody>
</table>

- Every write gets a unique sequence number.
- Objects stability array stores the sequence number of the last write.
- Objects remain unstable until last sequence number is acknowledged.

\[ w_1(x, w_1_{seq#}) \]
\[ w_2(x, w_2_{seq#}) \]
Outline

• Overview of programmable switches
• FLAIR architecture
• Implementation
• Evaluation
Implementation

FlairKV is a key-value store that optimizes Raft using FLAIR
• Implemented using P4 language
• Utilizes 12 registers and 30 tables (only 5% of ASIC memory)
• Implemented consistency-aware load balancing techniques
  • Random
  • Leader avoidance
  • Follower load awareness
Evaluation

Alternatives
• Leader-based (Raft, VR)
• Leader lease (Opt. Raft)
• Unreplicated
• Fast Paxos
• Follower-leases (Fleases)

Metrics
• Throughout
• Latency
• Load balancing efficiency

We varied:
• Number of replicas
• Number of clients
• Read to write ratio
• Workload skewness
• Data set size
Throughput

Workload
- YCSB-B (95% reads)
- Uniform workload

Throughput (1000 ops/sec)

Number of clients

<table>
<thead>
<tr>
<th>1</th>
<th>4</th>
<th>16</th>
<th>64</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>100</td>
<td>200</td>
<td>300</td>
<td>400</td>
</tr>
<tr>
<td>500</td>
<td>600</td>
<td>700</td>
<td>800</td>
<td>900</td>
</tr>
</tbody>
</table>
Throughput

Workload
YCSB-B (95% reads)
Uniform workload
Workload
YCSB-B (95% reads)
Uniform workload
Workload
YCSB-B (95% reads)
Uniform workload
Throughput

Workload
YCSB-B (95% reads)
Uniform workload

42% higher throughput
2.1X higher throughput
Read Latency

Workload
YCSB-B (95% reads)
Uniform workload
Read Latency

Workload
YCSB-B (95% reads)
Uniform workload
Read Latency

Workload
YCSB-B (95% reads)

Uniform workload
Read Latency

Workload
YCSB-B (95% reads)

Uniform workload

FLAIR reduces latency
• Avoid inconsistent replicas
• Avoid the leader

50% lower latency
Conclusion

• FLAIR a shim layer atop leader-based consensus protocol
  • Exploits programmable switches
  • Builds in-network consistency-aware load balancing
  • Maintains linearizability

• FlairKV achieves
  • 2.1× higher throughput than classical approaches, and up to 1.4× than leases
  • 2.4× lower latency than classical approaches, and up to 1.5× than leases

Despite their limitations, programmable switches can be leveraged to accelerate complex system protocols.