Byzantine Agreement

In Practice

Jack Hessel

Used Slide Outlines from Ion Stoica + Fang Zhang for help
Recall...
Recall...
Recall...
Recall...
Recall...
Recall...

Exponential Time!
Recall...

Exponential Time!

FLP!
● Fully synchronous?
● Network is magically immune to attacks
● Actual implementations were still slow between 1982-1999
Practical Byzantine Fault Tolerance - OSDI 1999

No photo available

Miguel Castro

Barbara Liskov
Take-aways...
Take-aways...

Asynchronous and correct
Take-aways...

Asynchronous and correct

Optimizations
(3%* slowdown)
Take-aways...

Asynchronous and correct

Optimizations (3%* slowdown)

BFS
One of these things is not like the other...
One of these things is not like the other...
One of these things is not like the other...

$delay(t)$ doesn't grow faster than $t$ forever.
One of these things is not like the other...

$delay(t)$ doesn’t grow faster than $t$ forever. FLP can be circumvented!
Filesystem Basic Structure...
Results
Results (which I don’t think we talk about enough)

(Anyone else feel this way?) /tangent
## Results (Realistic Use Case Benchmark)

<table>
<thead>
<tr>
<th>phase</th>
<th>strict</th>
<th>r/o lookup</th>
<th>NFS-std</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.55 (-69%)</td>
<td>0.47 (-73%)</td>
<td>1.75</td>
</tr>
<tr>
<td>2</td>
<td>9.24 (-2%)</td>
<td>7.91 (-16%)</td>
<td>9.46</td>
</tr>
<tr>
<td>3</td>
<td>7.24 (35%)</td>
<td>6.45 (20%)</td>
<td>5.36</td>
</tr>
<tr>
<td>4</td>
<td>8.77 (32%)</td>
<td>7.87 (19%)</td>
<td>6.60</td>
</tr>
<tr>
<td>5</td>
<td>38.68 (-2%)</td>
<td>38.38 (-2%)</td>
<td>39.35</td>
</tr>
<tr>
<td>total</td>
<td>64.48 (3%)</td>
<td>61.07 (-2%)</td>
<td>62.52</td>
</tr>
</tbody>
</table>

**Directory Creation, Read Everything, etc.**

**Byzantine Fault Tolerant**

**Existing System**
Question

See anything potentially misleading about this table?

<table>
<thead>
<tr>
<th>phase</th>
<th>BFS strict</th>
<th>r/o lookup</th>
<th>NFS-std</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.55 (-69%)</td>
<td>0.47 (-73%)</td>
<td>1.75</td>
</tr>
<tr>
<td>2</td>
<td>9.24 (-2%)</td>
<td>7.91 (-16%)</td>
<td>9.46</td>
</tr>
<tr>
<td>3</td>
<td>7.24 (35%)</td>
<td>6.45 (20%)</td>
<td>5.36</td>
</tr>
<tr>
<td>4</td>
<td>8.77 (32%)</td>
<td>7.87 (19%)</td>
<td>6.60</td>
</tr>
<tr>
<td>5</td>
<td>38.68 (-2%)</td>
<td>38.38 (-2%)</td>
<td>39.35</td>
</tr>
<tr>
<td>total</td>
<td>64.48 (3%)</td>
<td>61.07 (-2%)</td>
<td>62.52</td>
</tr>
</tbody>
</table>
See anything potentially misleading about this table?

<table>
<thead>
<tr>
<th>phase</th>
<th>BFS</th>
<th>r/o lookup</th>
<th>NFS-std</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.55 (-69%)</td>
<td>0.47 (-73%)</td>
<td>1.75</td>
</tr>
<tr>
<td>2</td>
<td>9.24 (-2%)</td>
<td>7.91 (-16%)</td>
<td>9.46</td>
</tr>
<tr>
<td>3</td>
<td>7.24 (35%)</td>
<td>6.45 (20%)</td>
<td>5.36</td>
</tr>
<tr>
<td>4</td>
<td>8.77 (32%)</td>
<td>7.87 (19%)</td>
<td>6.60</td>
</tr>
<tr>
<td>5</td>
<td>38.68 (-2%)</td>
<td>38.38 (-2%)</td>
<td>39.35</td>
</tr>
<tr>
<td>total</td>
<td>64.48 (3%)</td>
<td>61.07 (-2%)</td>
<td>62.52</td>
</tr>
</tbody>
</table>

Task 3: Examine all files, Task 4: Examine all Bytes

Did you buy the “3%” claim?
## Results (worst case overhead)

<table>
<thead>
<tr>
<th>arg./res. (KB)</th>
<th>replicated</th>
<th>without replication</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>read-write</td>
<td>read-only</td>
</tr>
<tr>
<td>0/0</td>
<td>3.35 (309%)</td>
<td>1.62 (98%)</td>
</tr>
<tr>
<td>4/0</td>
<td>14.19 (207%)</td>
<td>6.98 (51%)</td>
</tr>
<tr>
<td>0/4</td>
<td>8.01 (72%)</td>
<td>5.94 (27%)</td>
</tr>
</tbody>
</table>

Null Operation Variants

Time in ms (slowdown)

Baseline -- Still Their System
## Results (worst case overhead)

<table>
<thead>
<tr>
<th>arg./res. (KB)</th>
<th>replicated</th>
<th>without replication</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>read-write</td>
<td>read-only</td>
</tr>
<tr>
<td>0/0</td>
<td>3.35 (309%)</td>
<td>1.62 (98%)</td>
</tr>
<tr>
<td>4/0</td>
<td>14.19 (207%)</td>
<td>6.98 (51%)</td>
</tr>
<tr>
<td>0/4</td>
<td>8.01 (72%)</td>
<td>5.94 (27%)</td>
</tr>
</tbody>
</table>

Null Operation Variants  

Time in ms (slowdown)  

Baseline -- Still Their System

(nice breakdown of where each part comes from!)
Algorithm Overview
Algorithm Overview

3f+1 State Machines
Algorithm Overview

0
1
2
3
...

Algorithm Overview

Cycle through “views” when leader fails
Algorithm Overview

$3f+1$ State Machines
Algorithm Overview

3f+1 State Machines
Please complete operation X. I am client C! I’ll be waiting for $f+1$ of you to get back to me…
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines

0
1
2
3
...

f+1
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

Listen… This is taking too long… How many times do I need to tell you?

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

I already processed this...

3f+1 State Machines
Algorithm Overview

Uhh… Chief?

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Algorithm Overview

3f+1 State Machines
Some detail...
Some detail...
Some detail...

Request:
- Operation requested
- Timestamp of request by client
- Client ID
Some detail...

Request m

0

1

2

3

...
Some detail...

Request m

0

1

2

3

...
Some detail...

Pre-prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m

Request m
Pre-prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m

Digest by process 0
Some detail...

Pre-prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m

Digest by process 0  Request m

State Tracker: Pre-prepare
**State Tracker For Node 2: Pre-prepare**

- **Pre-prepare Message**
  - Current view number
  - Sequence number assigned by primary
  - “Digest” of m

**Digest by process 0**

**Request m**
State Tracker For Node 2: Pre-prepare

Pre-prepare Message
- Current view number
- Sequence number assigned by primary
- "Digest" of m

<table>
<thead>
<tr>
<th>Digest by process 0</th>
<th>Request m</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
Pre-prepare Message
- Current view number
- Sequence number assigned by primary
- "Digest" of m

<table>
<thead>
<tr>
<th>Digest by process 0</th>
<th>Request m</th>
</tr>
</thead>
</table>

1. Is this digest for m?
2. Am I in the same view?
3. Am I sure I haven’t accepted a similar message for this view and sequence number with a different digest?
4. Do the watermarks work out? (will talk about later)
Some detail...

State Tracker For Node 2: Pre-prepare

1. Is this digest for m?
2. Am I in the same view?
3. Am I sure I haven’t accepted a similar message for this view and sequence number with a different digest?
4. Do the watermarks work out? (will talk about later)

Pre-prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m

Request m

Digest by process 0
Pre-prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m

Digest by process 0
Request m
State Tracker For Node 2: Prepare

Pre-prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m

Prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m
- My number

Digest by process 0  Request m
Some detail...

State Tracker For Node 2: Prepare

Prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m
- My number

Digest by process 2
State Tracker For Node 2: Prepare

- Current view number
- Sequence number assigned by primary
- “Digest” of m
- My number

Digest by process 2
Some detail...

Hey y’all -- let’s do this one!

State Tracker For Node 2: Prepare

Prepare Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m
- My number

Digest by process 2
Some detail...

... okay I’ll just wait for folks to agree...
Some time later...
Some time later...

... after process 2 collects enough matching “prepare” statements from other replicas....
Some detail...

State Tracker For Node 2: Prepare

2f-2… 2f-1… 2f!!!
Some detail...

State Tracker For Node 2: Commit

Commit Message
- Current view number
- Sequence number assigned by primary
- “Digest” of m
- My number

Digest by process 2

2f-2… 2f-1… 2f!!!
Some detail...

... okay I'll just wait for folks to commit...

State Tracker For Node 2: Commit
Some time later...
Some time later...

... after process 2 collects enough matching “commit” statements from other replicas....
Some detail...

State Tracker For Node 2:
Commit

2f-2… 2f-1…
2f!!!

0
1
2
3
...
Some detail...
Some detail...
Some detail...
Some detail...
Failure Sketch...
Cycle through “views” when leader fails
Failure Sketch...

Cycle through “views” when leader fails
Failure Sketch...

1. Time out
2. Message: View Change Please
3. Message: View Change

Cycle through “views” when leader fails
Discussion Question

Does Byzantine fault tolerance matter?

Do you buy the motivation?
Atomic Broadcast: From Simple Message Diffusion to Byzantine Agreement

Flaviu Cristian

Houtan Aghili

Ray Strong

Danny Dolev
As you already know...
As you already know...

- **Byzantine** (arbitrary behavior)
- **Timing** (early, late, or never)
- **Omission** (messages lost)
Broadcast
Broadcast
Atomic Broadcast?
Atomic Broadcast
Atomic Broadcast
Atomic Broadcast

\[ T \rightarrow T + \Delta \]
Atomic Broadcast

T \rightarrow T + \Delta
Atomic Broadcast

\[ T \rightarrow T + \Delta \]
Atomic Broadcast

1. A
2. B
3. C

$T \rightarrow T + \Delta$
Atomic Broadcast

\[ T \rightarrow T + \Delta \]
Assumptions
Assumptions (seeing a pattern? few unifying assumptions, assumptions made for ease of proof rather than realism)

(Anyone else feel this way?) /tangent
Assumptions
Assumptions

This processor won't fail.
Assumptions

This processor won't fail.

$|\text{Clock}1(t) - \text{Clock}2(t)| < \varepsilon$
Assumptions

- This processor won't fail.
- $|\text{Clock1}(t) - \text{Clock2}(t)| < \varepsilon$
- This communication time is bounded
Assumptions

This processor won't fail.

$|Clock1(t) - Clock2(t)| < \epsilon$

This communication time is bounded
“Diffusion Induction Principle”
“Diffusion Induction Principle”

In a connected graph, everyone will eventually get the message.
CASD for Omission Failures...
CASD for Omission Failures...

- Big enough so you don’t time out on normal runs
- Small enough so you actually do what you want
CASD for Omission Failures...

\[ \Delta = f(\text{Diameter of network, clock skew, processing/messaging time}) \]

- Big enough so you don’t time out on normal runs
- Small enough so you actually do what you want
CASD for Omission Failures...

\[ \Delta = f(\text{Diameter of network, clock skew, processing/messaging time}) \]

- Big enough so you don’t time out on normal runs
- Small enough so you actually do what you want
CASD for Timing Failures...

\[ \Delta = f(\text{Diameter of network, clock skew, processing/messaging time}) \]
CASD for Timing Failures...

- New cases: too early and too late
  - Too early: “history log at any correct process is bounded”
CASD for Timing Failures...

- New cases: too early and too late
  - Too early: “history log at any correct process is bounded”

Note on page 16: “This type of faulty behavior may not be very common in practice, but it does fit the definition of early timing failure…” Very weird case!
CASD for Timing Failures...
CASD for Timing Failures...

\[ \Delta = f(\text{Diameter of network, clock skew, processing/messaging time, number of hops for a given message}) \]
CASD for Timing Failures...

\[ \Delta_i = f(\text{Diameter of network, clock skew, processing/messaging time, number of hops for a given message}) \]
CASD for Timing Failures...

\[ \Delta_i = f(\text{Diameter of network, clock skew, processing/messaging time, number of hops for a given message}) \]

Note on page 16: “This type of faulty behavior may not be very common in practice, but it does fit the definition of early timing failure…” Very weird case!
CASD for Byzantine Failures...

- Add signing and authentication
- Slower
Jack’s thoughts...

- Relation to hyperparameters in machine learning
- Framing as an online learning problem?