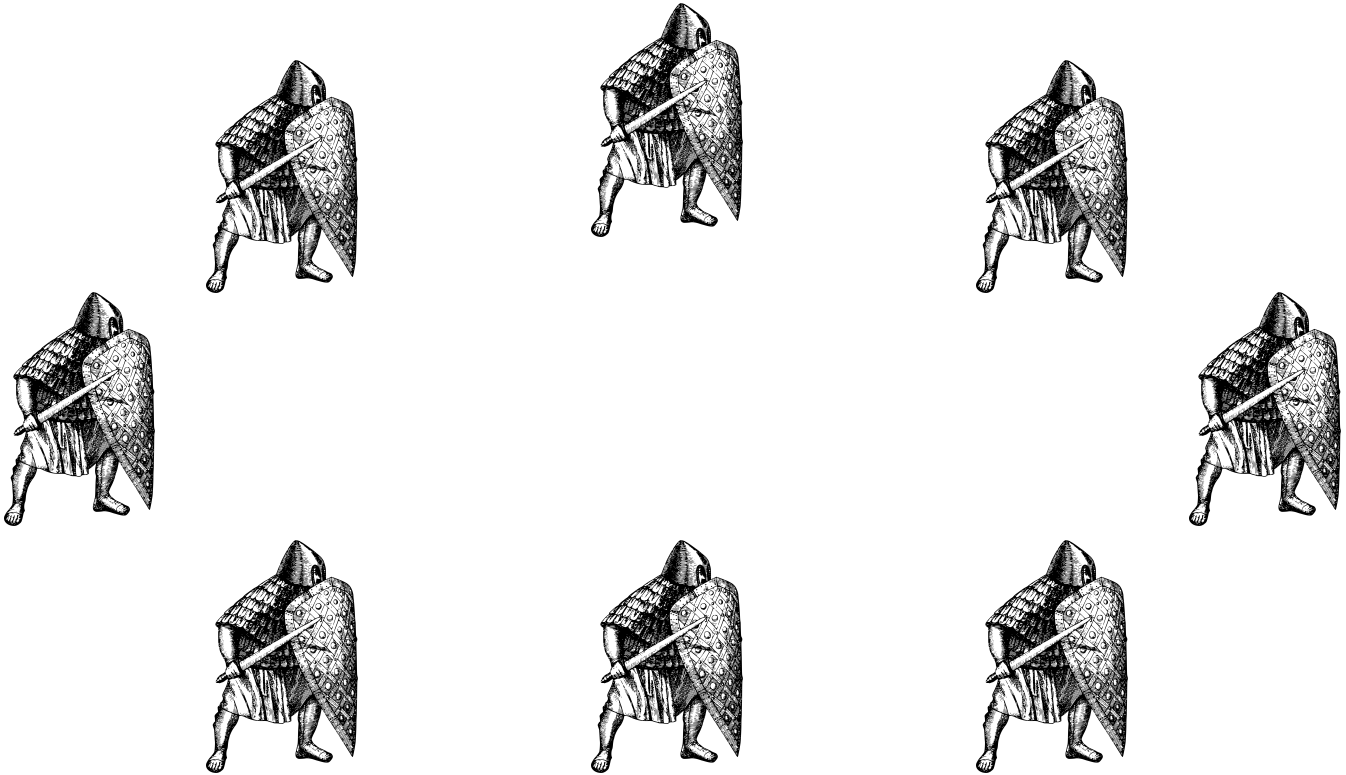
Byzantine Agreement

In Practice

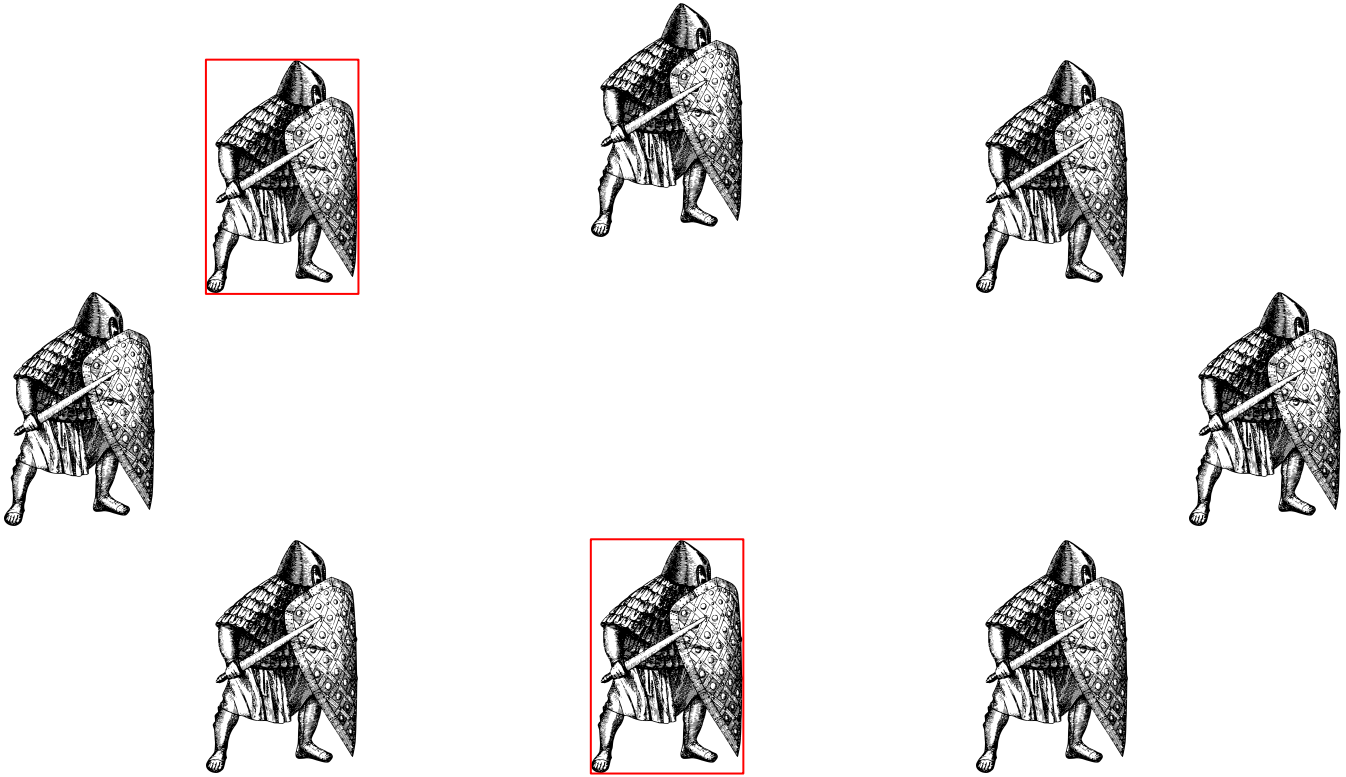
Jack Hessel

Used Slide Outlines from Ion Stoica + Fang Zhang for help

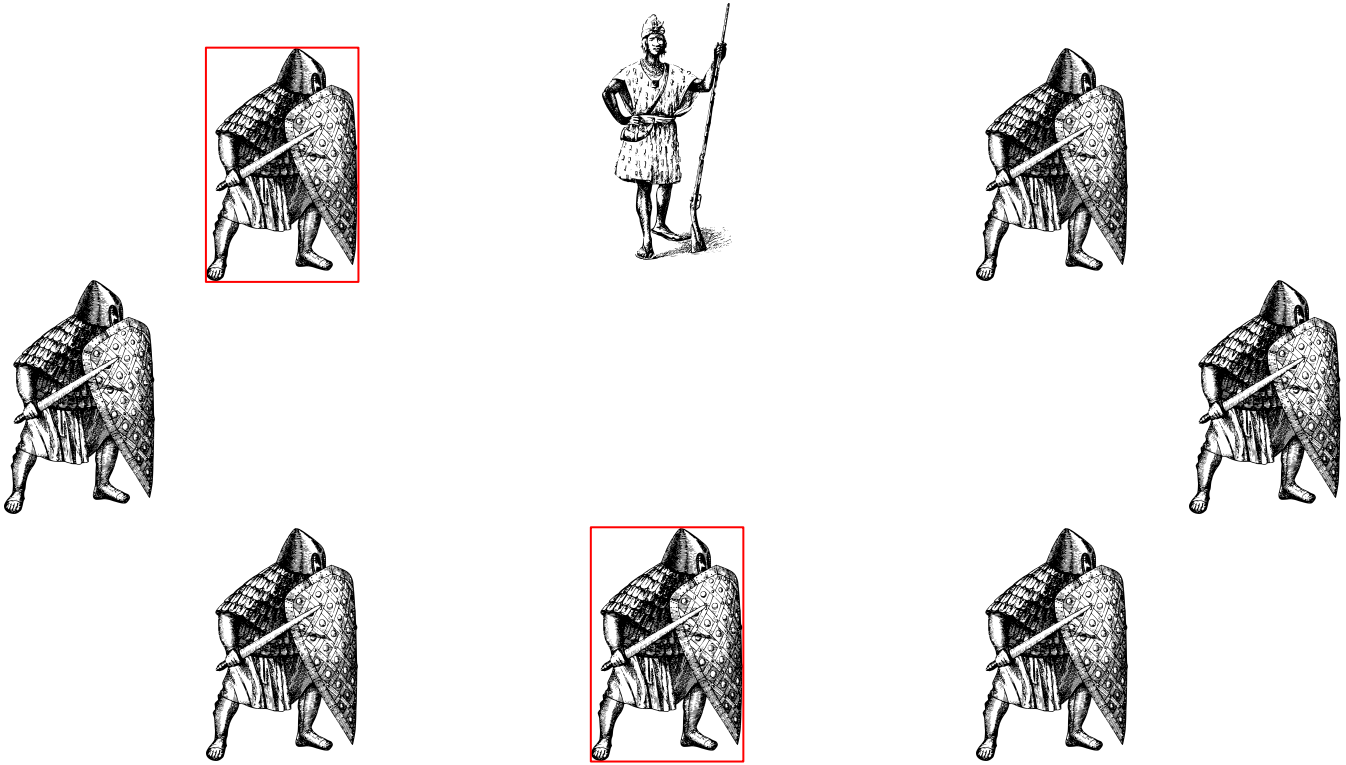
Recall...



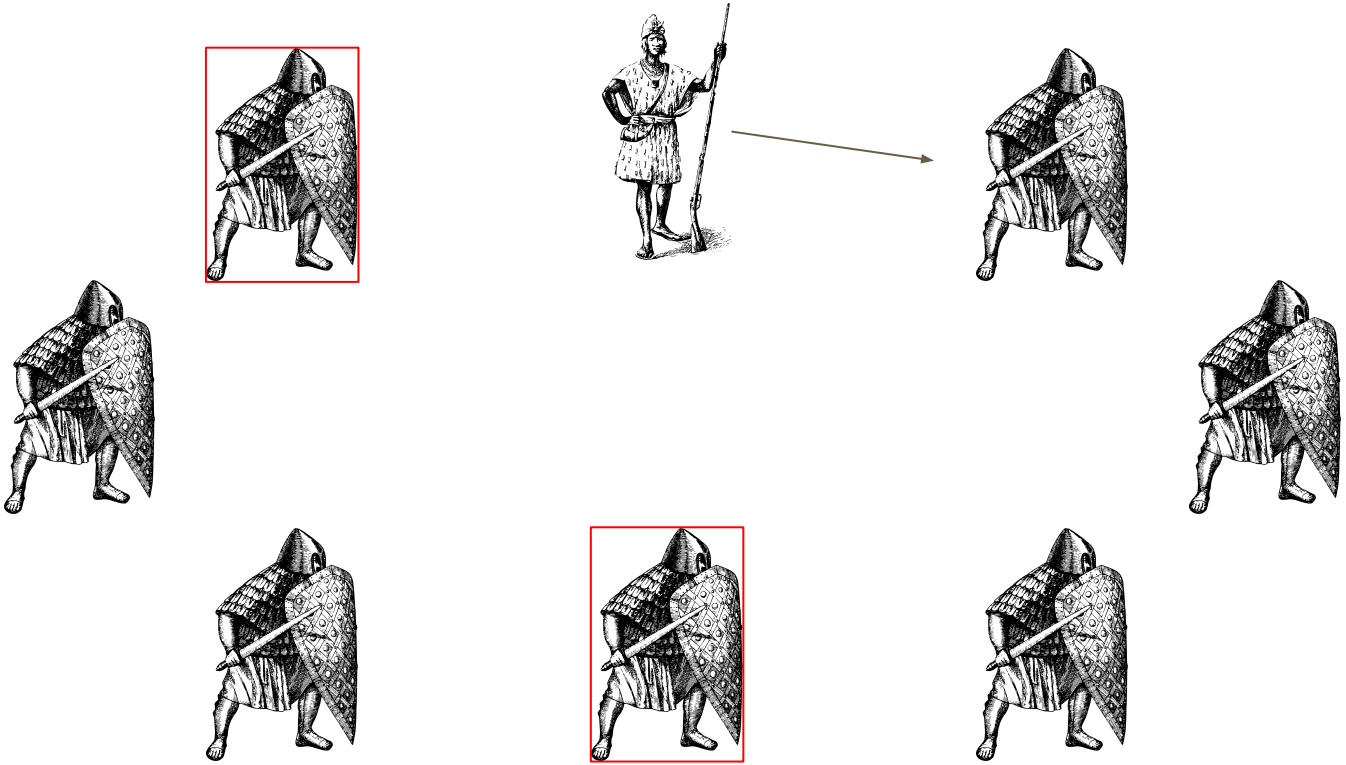
Recall...



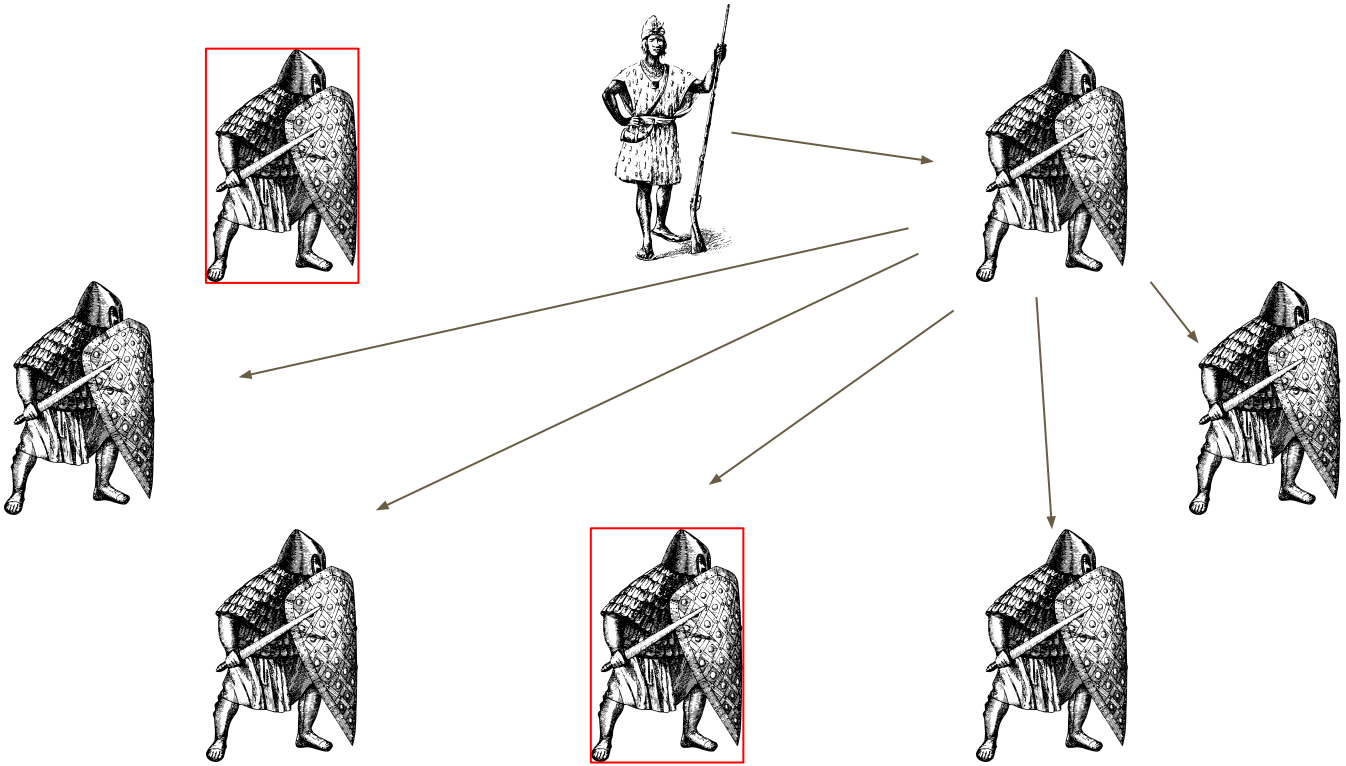
Recall...



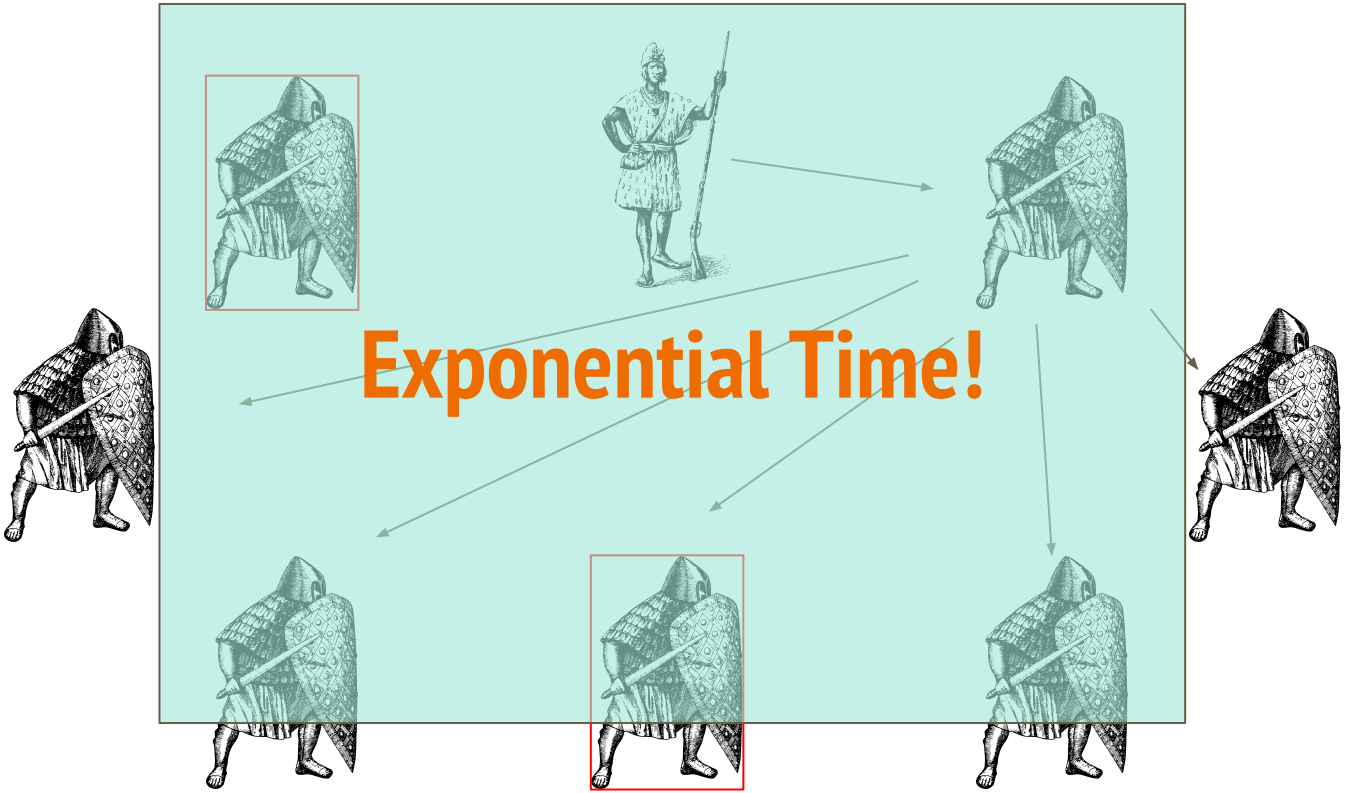
Recall...



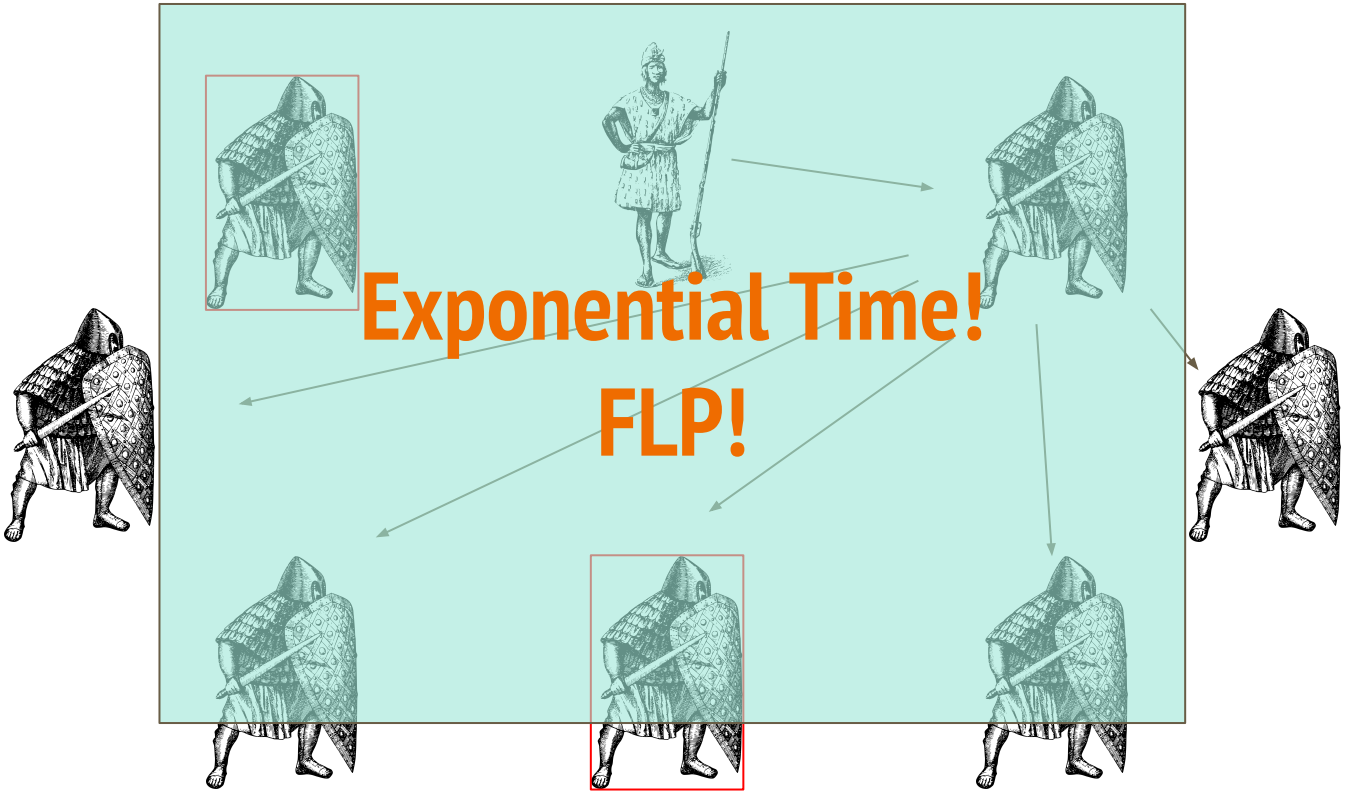
Recall...



Recall...

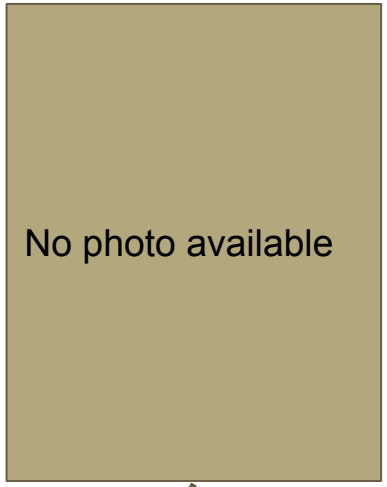


Recall...



- Fully synchronous?
- Network is magically immune to attacks
- Actual implementations were still slow between 1982-1999

Practical Byzantine Fault Tolerance - OSDI 1999



Miguel Castro

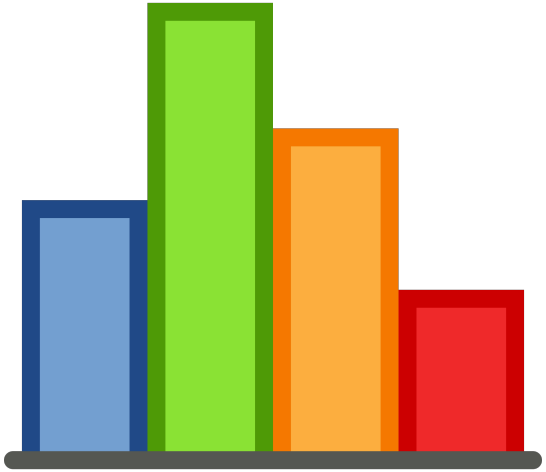
Barbara Liskov

Take-aways…

Take-aways...

Asynchronous
and correct

Take-aways...

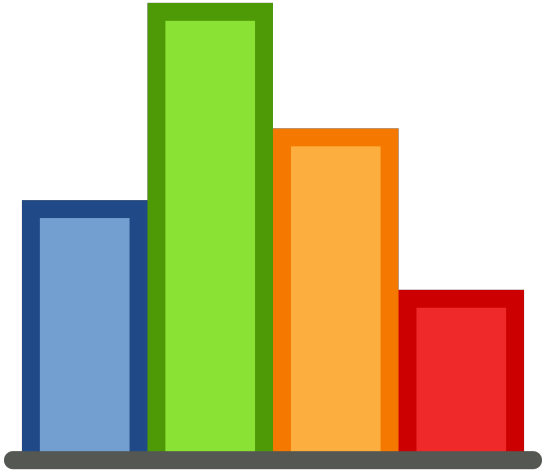


Asynchronous and correct

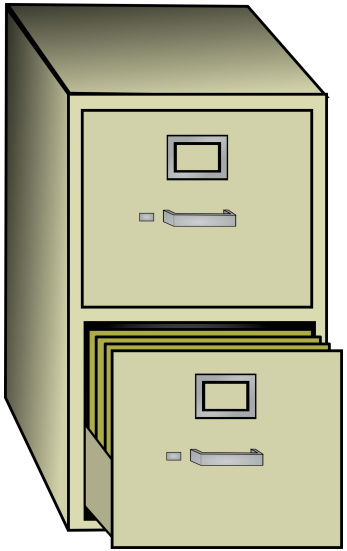
Optimizations (3%* slowdown)

Take-aways...

Asynchronous and correct

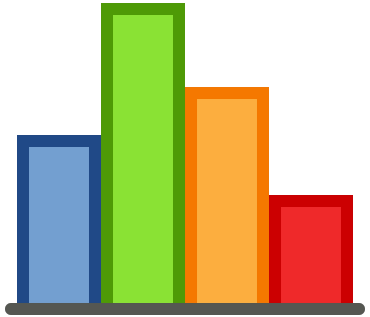


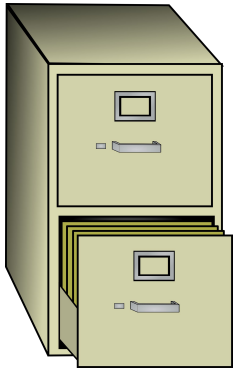
Optimizations (3%* slowdown)



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…

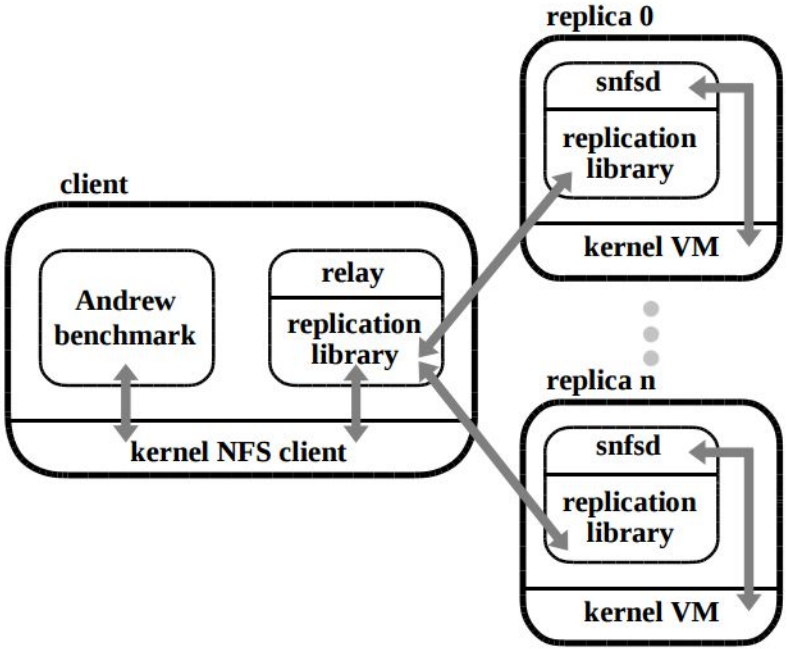
delay(t) doesn’t grow faster than *t* forever.

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)

Directory Creation, Read Everything, etc.

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

Byzantine Fault Tolerant

Existing System

Question

See anything potentially misleading about this table?

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

Question

See anything potentially misleading about this table?

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

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

Did you buy the “3%” claim?

Results (worst case overhead)

arg./res. (KB)	replicated		without replication
	read-write	read-only	
0/0	3.35 (309%)	1.62 (98%)	0.82
4/0	14.19 (207%)	6.98 (51%)	4.62
0/4	8.01 (72%)	5.94 (27%)	4.66

Null Operation Variants

Time in ms (slowdown)

Baseline -- Still Their System

Results (worst case overhead)

arg./res. (KB)	replicated		without replication
	read-write	read-only	
0/0	3.35 (309%)	1.62 (98%)	0.82
4/0	14.19 (207%)	6.98 (51%)	4.62
0/4	8.01 (72%)	5.94 (27%)	4.66

Null Operation Variants

Time in ms (slowdown)

Baseline -- Still Their System

(nice breakdown of where each part comes from!)

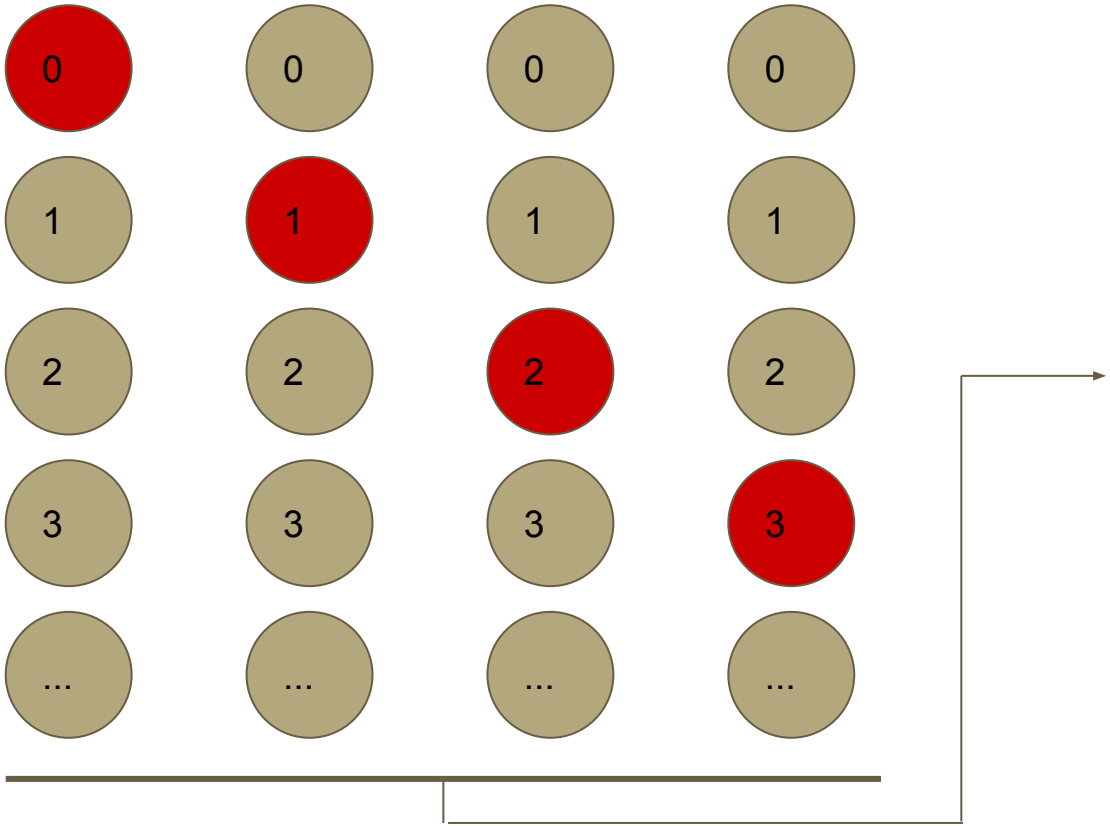
Algorithm Overview

Algorithm Overview

3f+1 State Machines

Algorithm Overview

Algorithm Overview



Cycle through “views” when leader fails

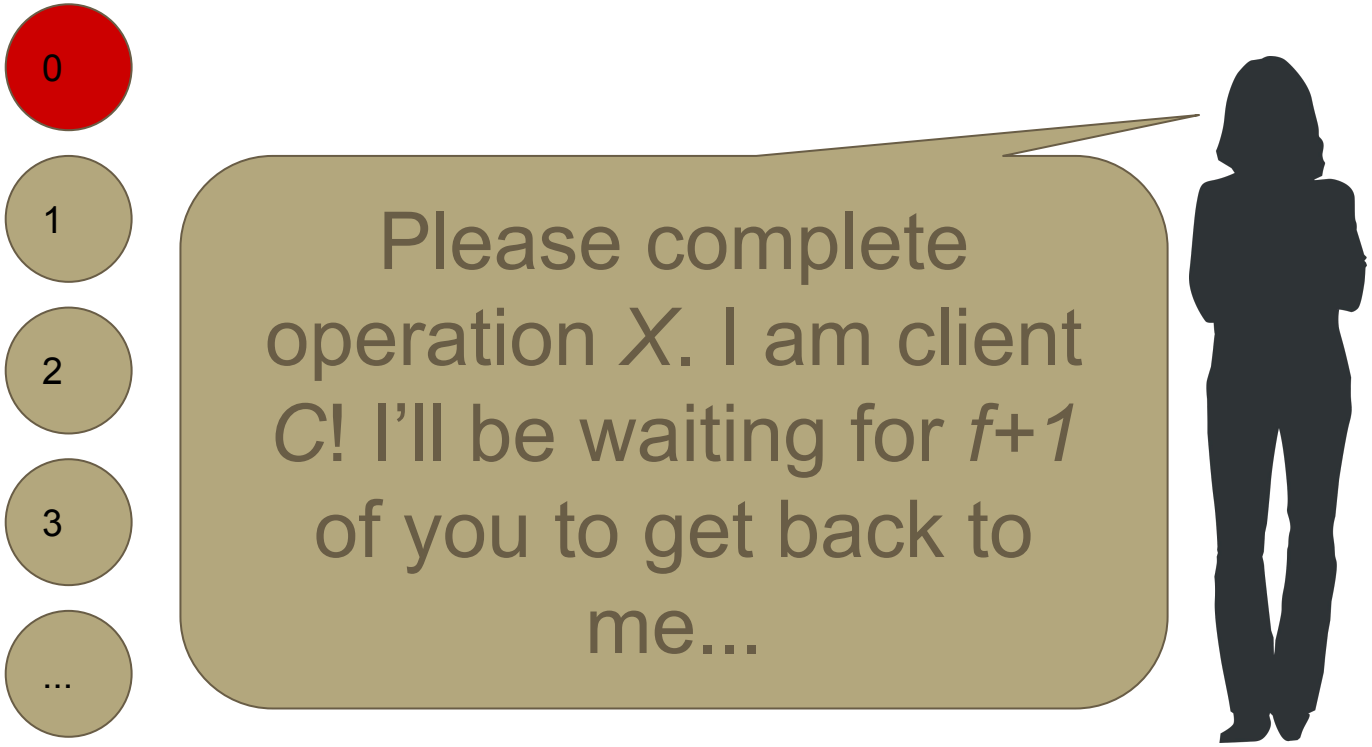
Algorithm Overview

3f+1 State Machines

Algorithm Overview

3f+1 State Machines

Algorithm Overview

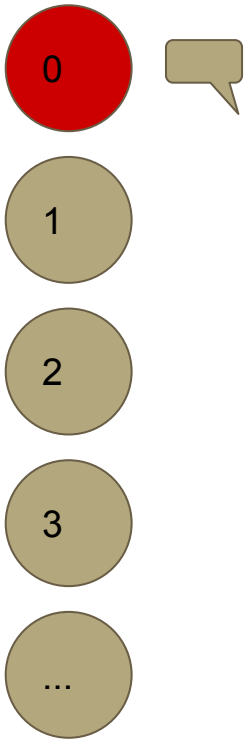


3f+1 State Machines

Algorithm Overview

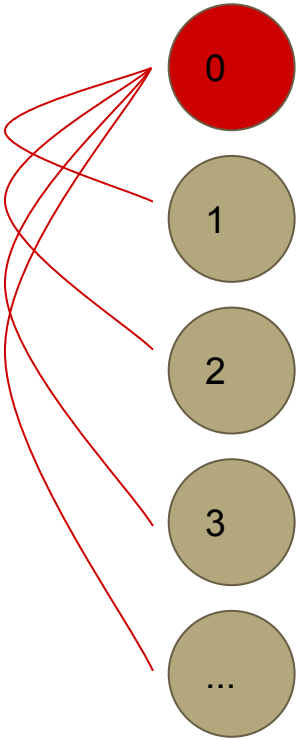
3f+1 State Machines

Algorithm Overview



3f+1 State Machines

Algorithm Overview

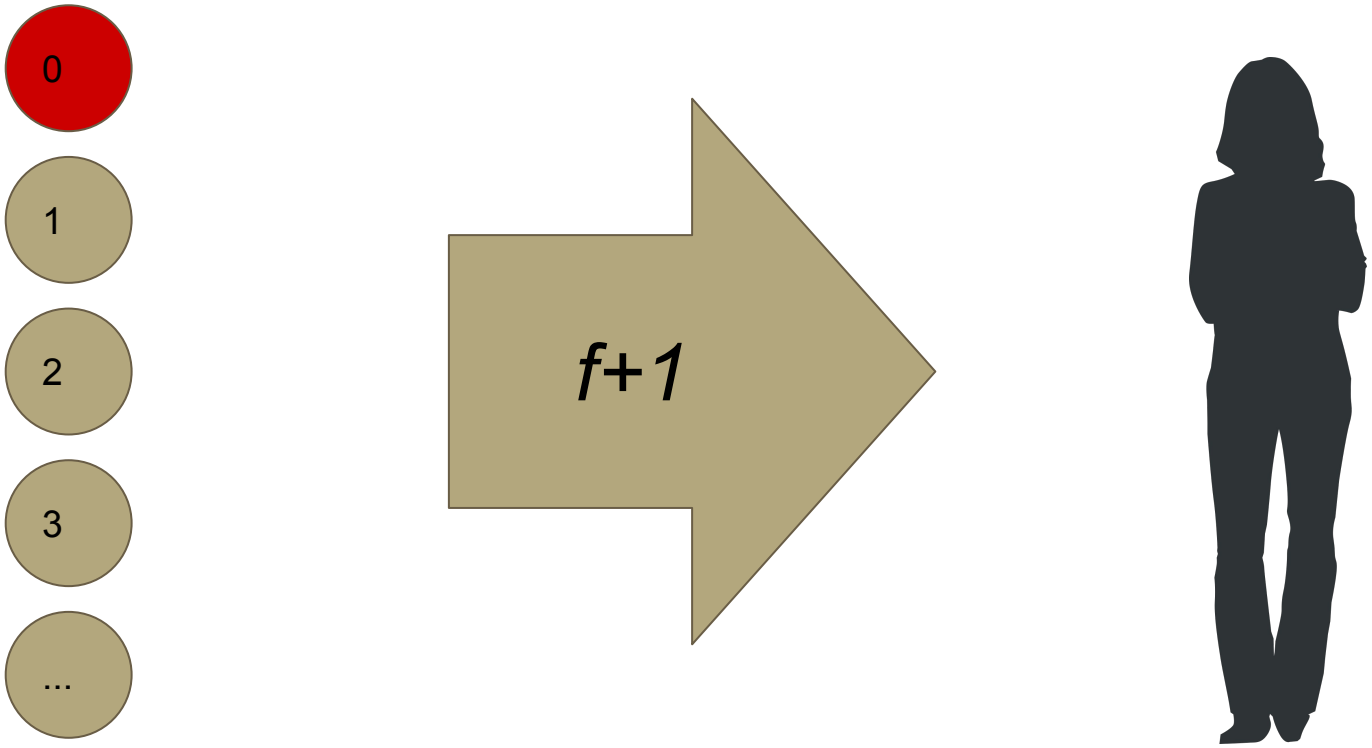


3f+1 State Machines

Algorithm Overview

3f+1 State Machines

Algorithm Overview



3f+1 State Machines

Algorithm Overview

3f+1 State Machines

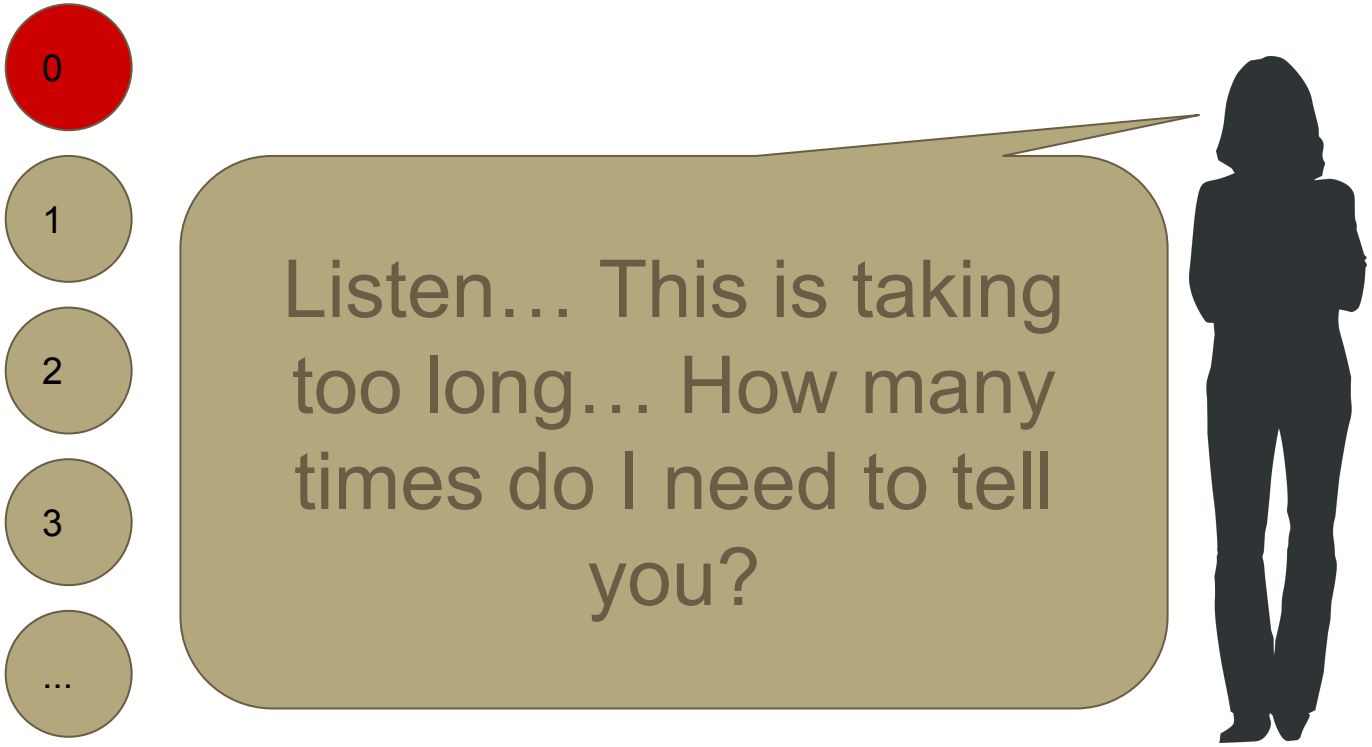
Algorithm Overview

3f+1 State Machines

Algorithm Overview

3f+1 State Machines

Algorithm Overview

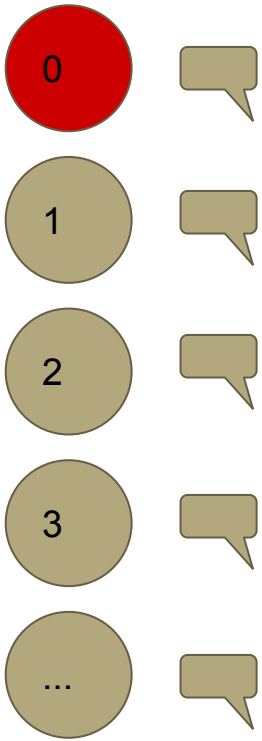


3f+1 State Machines

Algorithm Overview

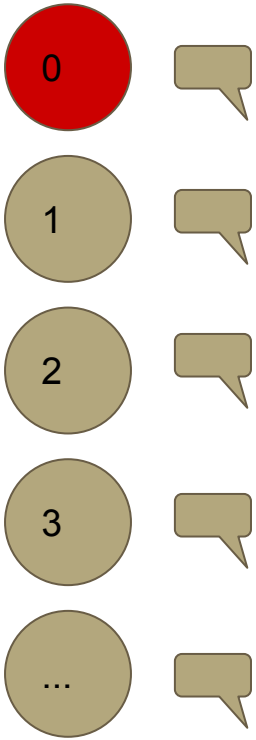
3f+1 State Machines

Algorithm Overview



3f+1 State Machines

Algorithm Overview

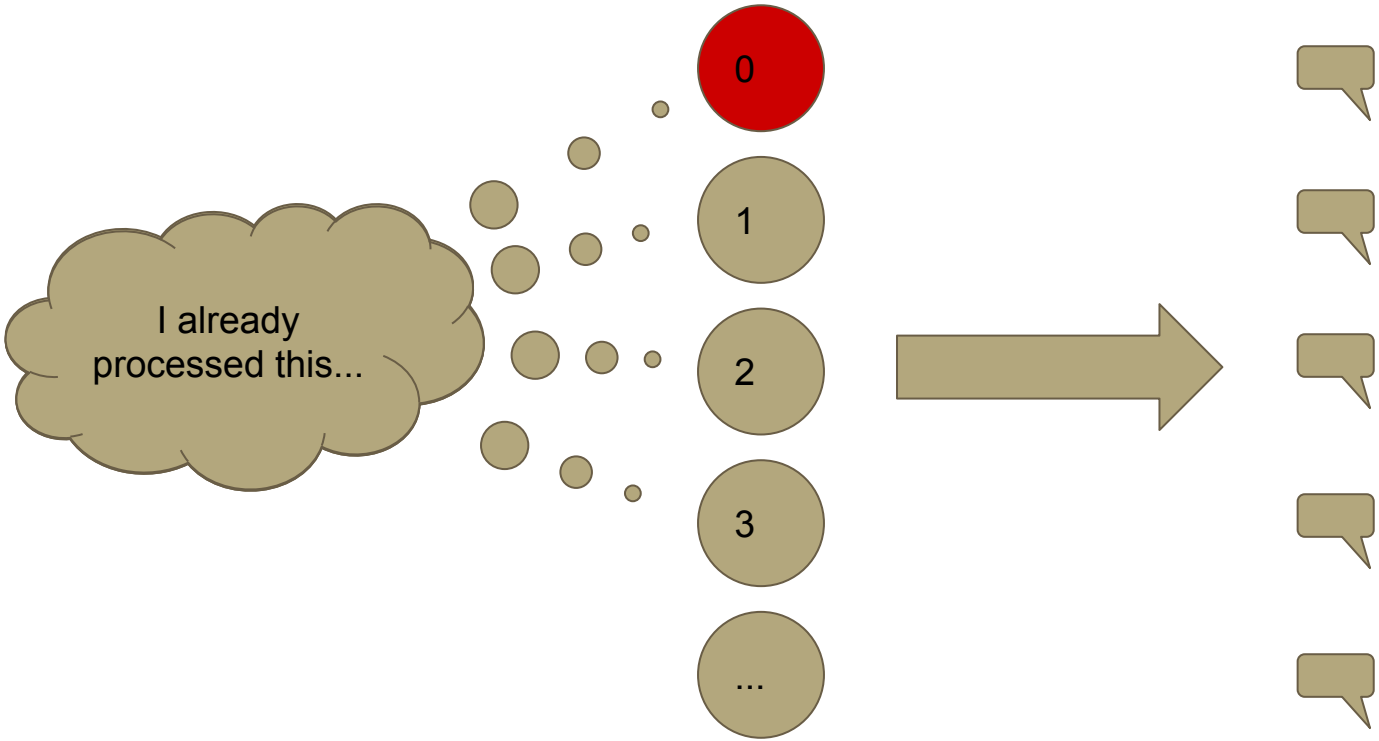


3f+1 State Machines

Algorithm Overview

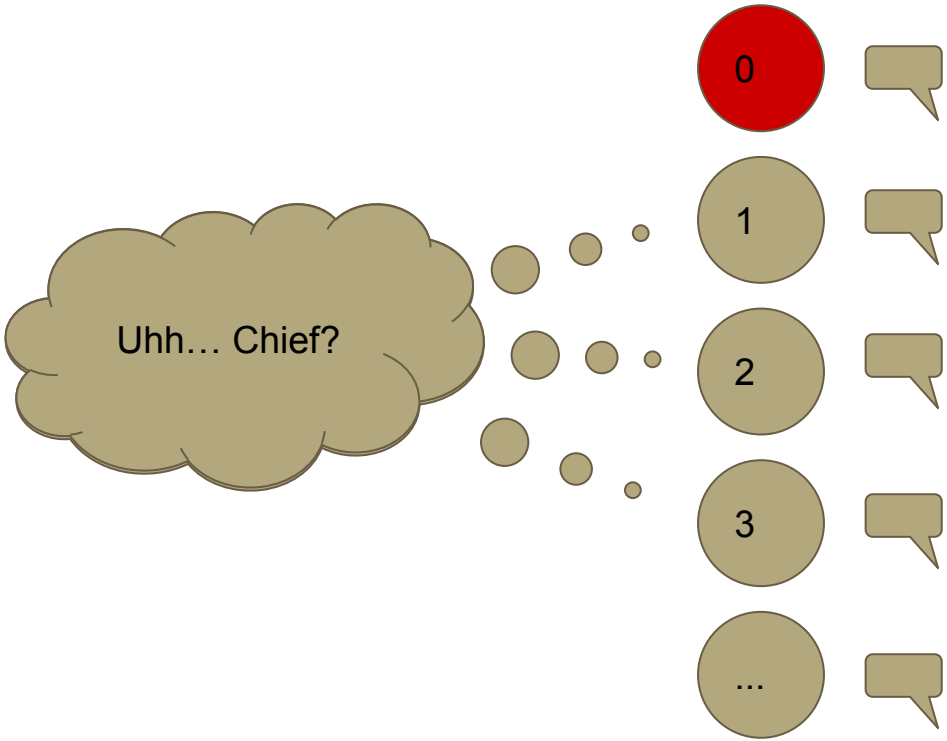
3f+1 State Machines

Algorithm Overview



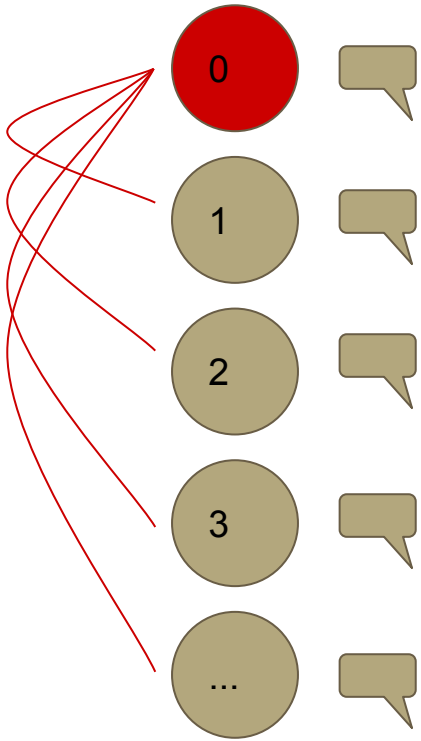
3f+1 State Machines

Algorithm Overview



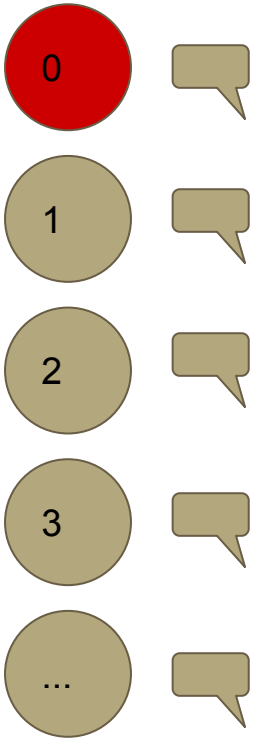
3f+1 State Machines

Algorithm Overview



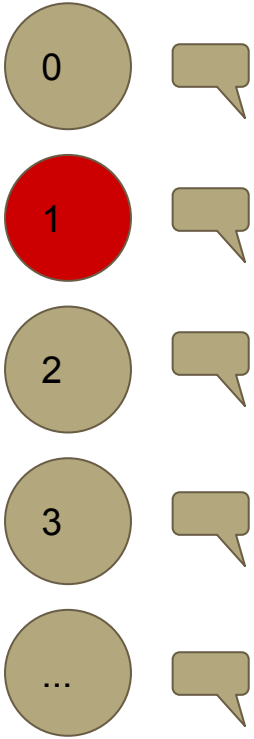
3f+1 State Machines

Algorithm Overview



3f+1 State Machines

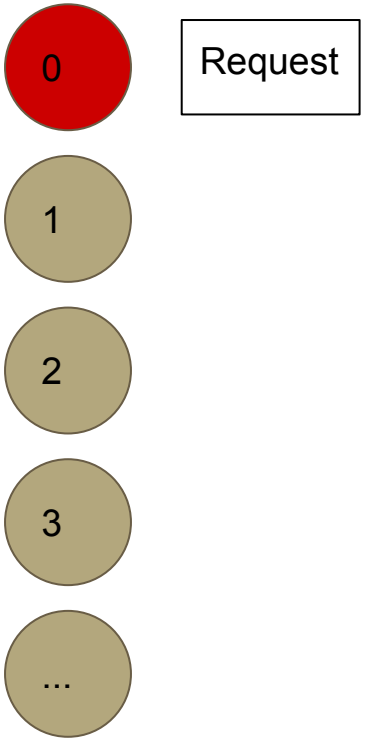
Algorithm Overview



3f+1 State Machines

Some detail...

Some detail...

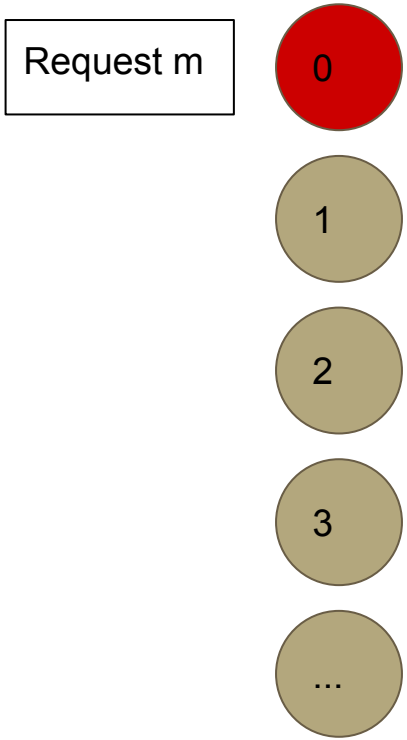


Some detail...

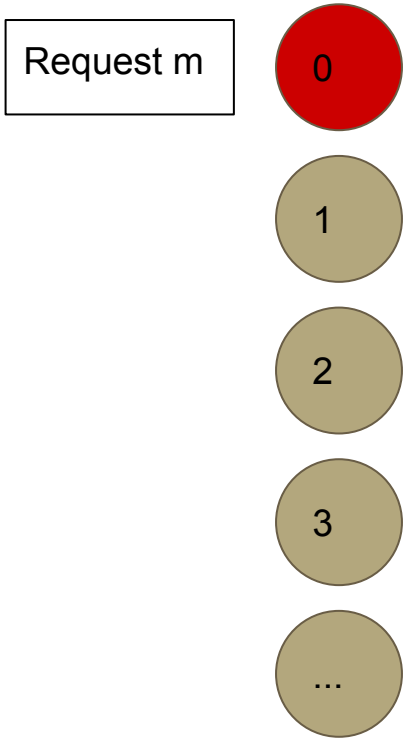
Request:

- Operation requested
- Timestamp of request by client
- Client ID

Some detail...



Some detail...



Some detail...

Request m

Pre-prepare Message

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

Some detail...

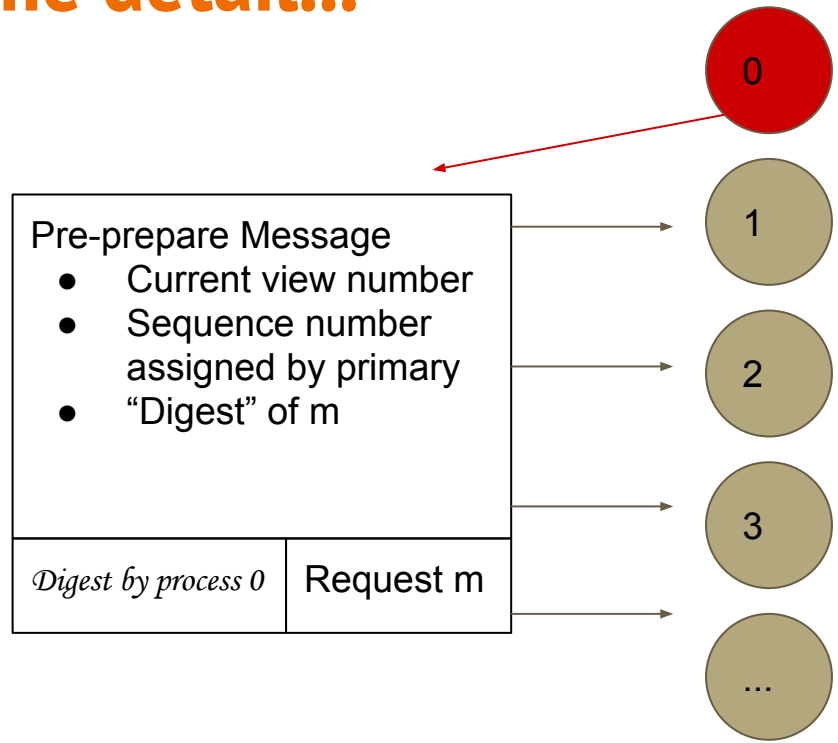
Request m

Pre-prepare Message

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

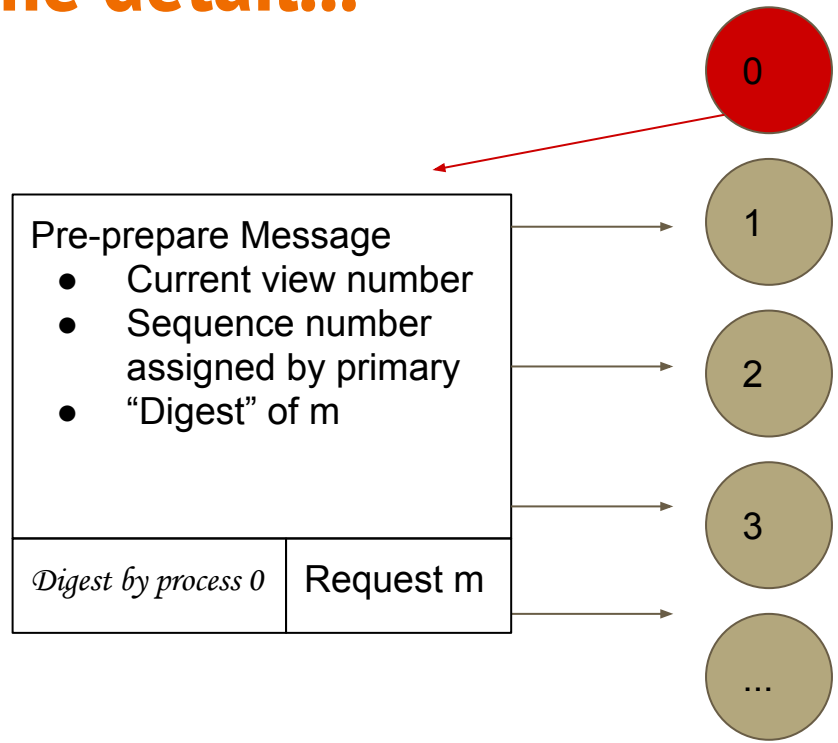
Digest by process 0

Some detail...


0
1
2
3
...
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

Some detail...



State Tracker For Node 2: Pre-prepare

Some detail...

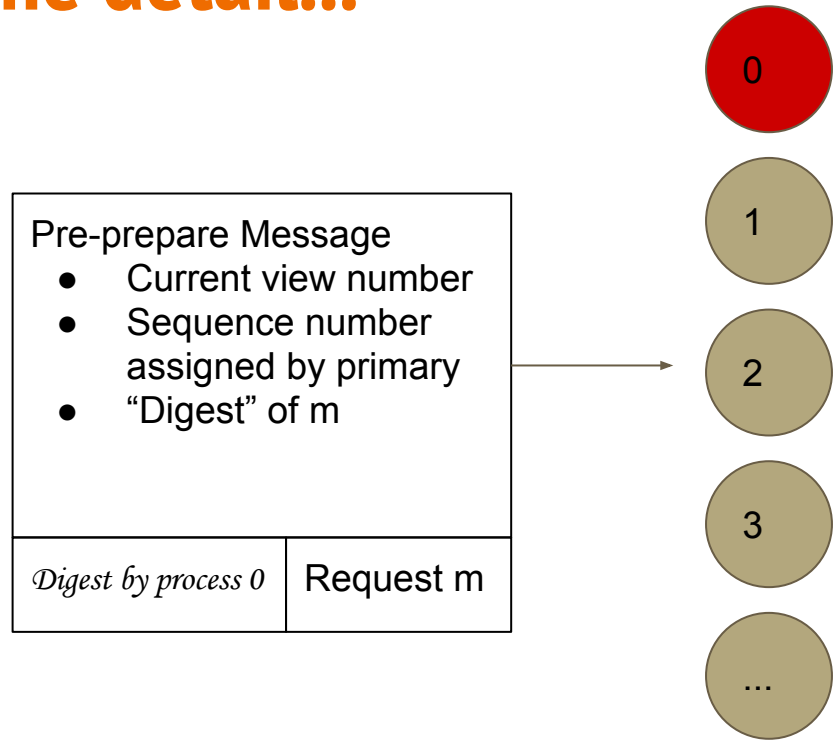
Pre-prepare Message

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

Digest by process 0	Request m

State Tracker For Node 2: Pre-prepare

Some detail...



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)

State Tracker For Node 2: Pre-prepare

Some detail...

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

Digest by process 0	Request m

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 watermarks work out? (will talk about later)

State Tracker For Node 2: Prepare

Some detail...

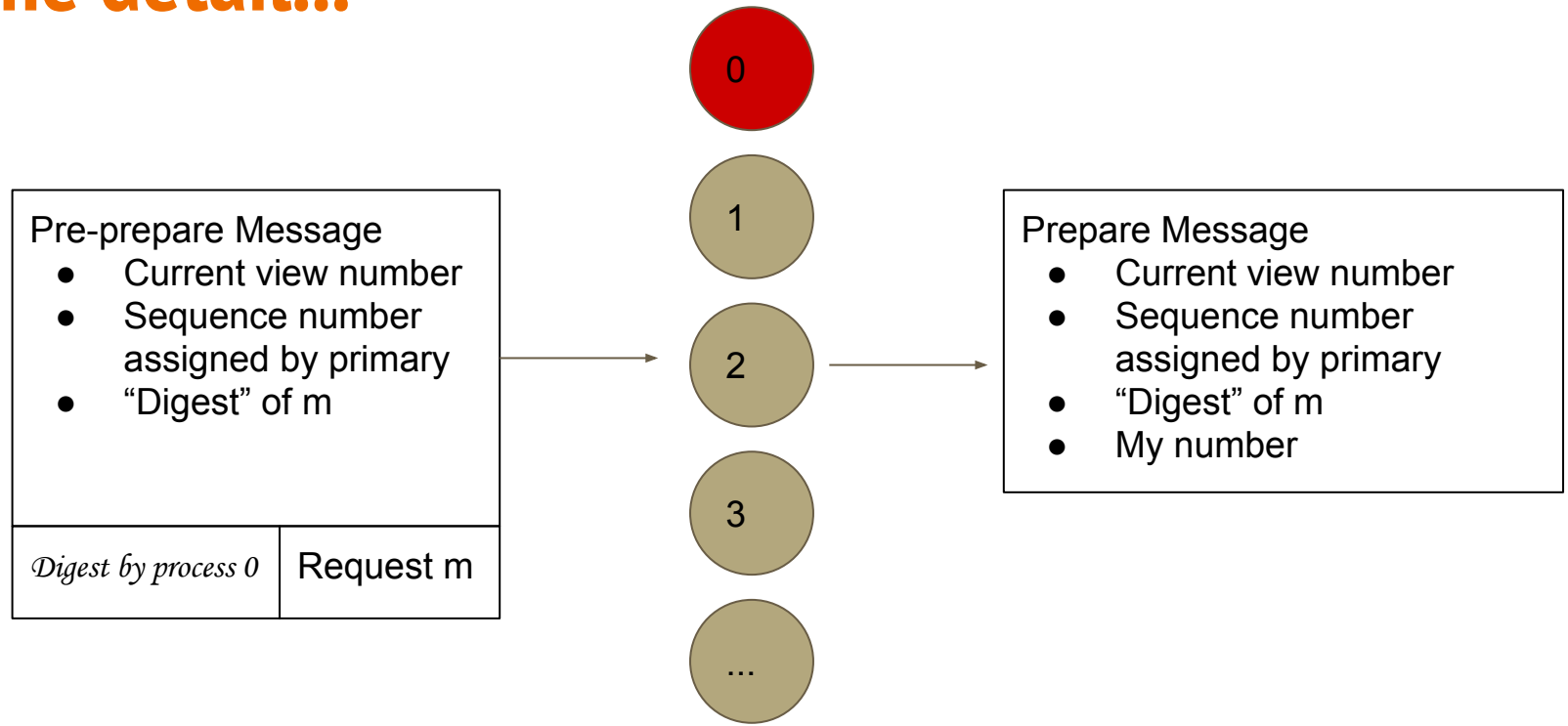
Pre-prepare Message

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

Digest by process 0	Request m

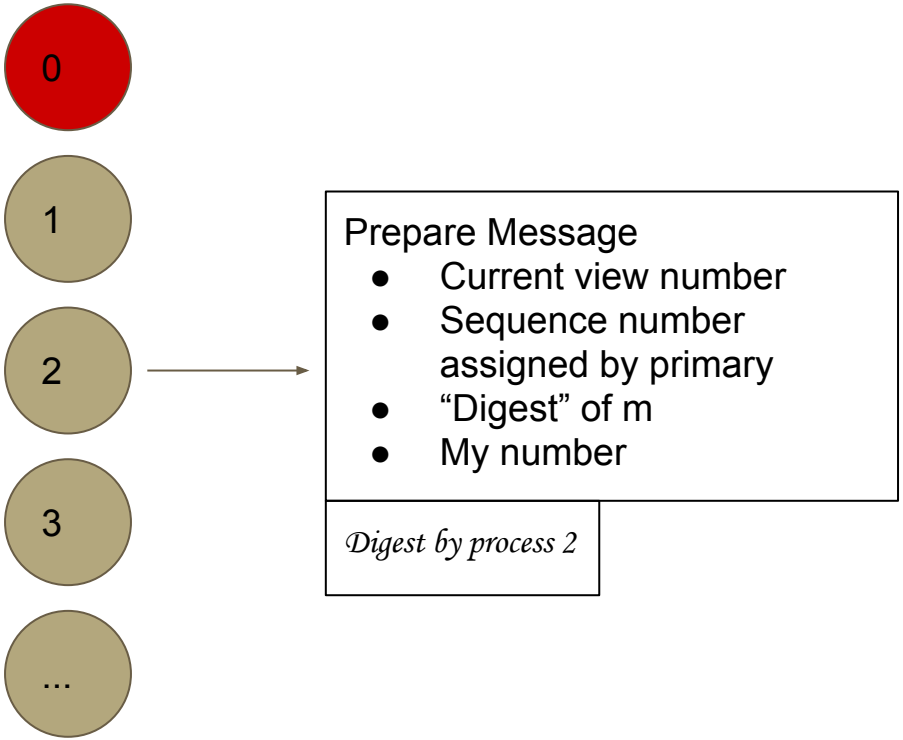
State Tracker For Node 2: Prepare

Some detail...



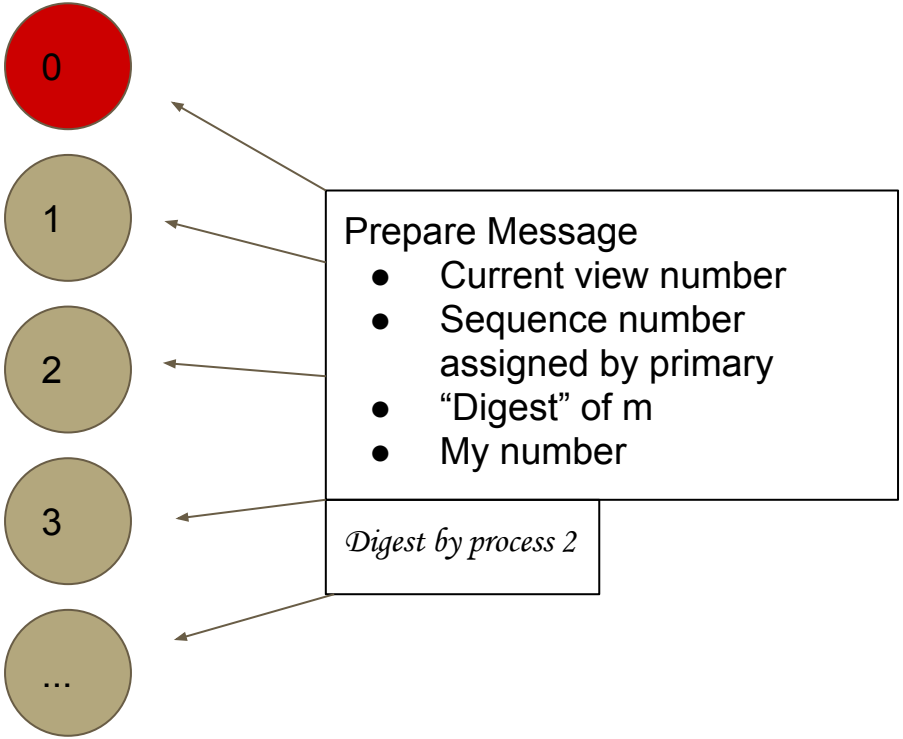
Some detail...

State Tracker For Node 2: Prepare



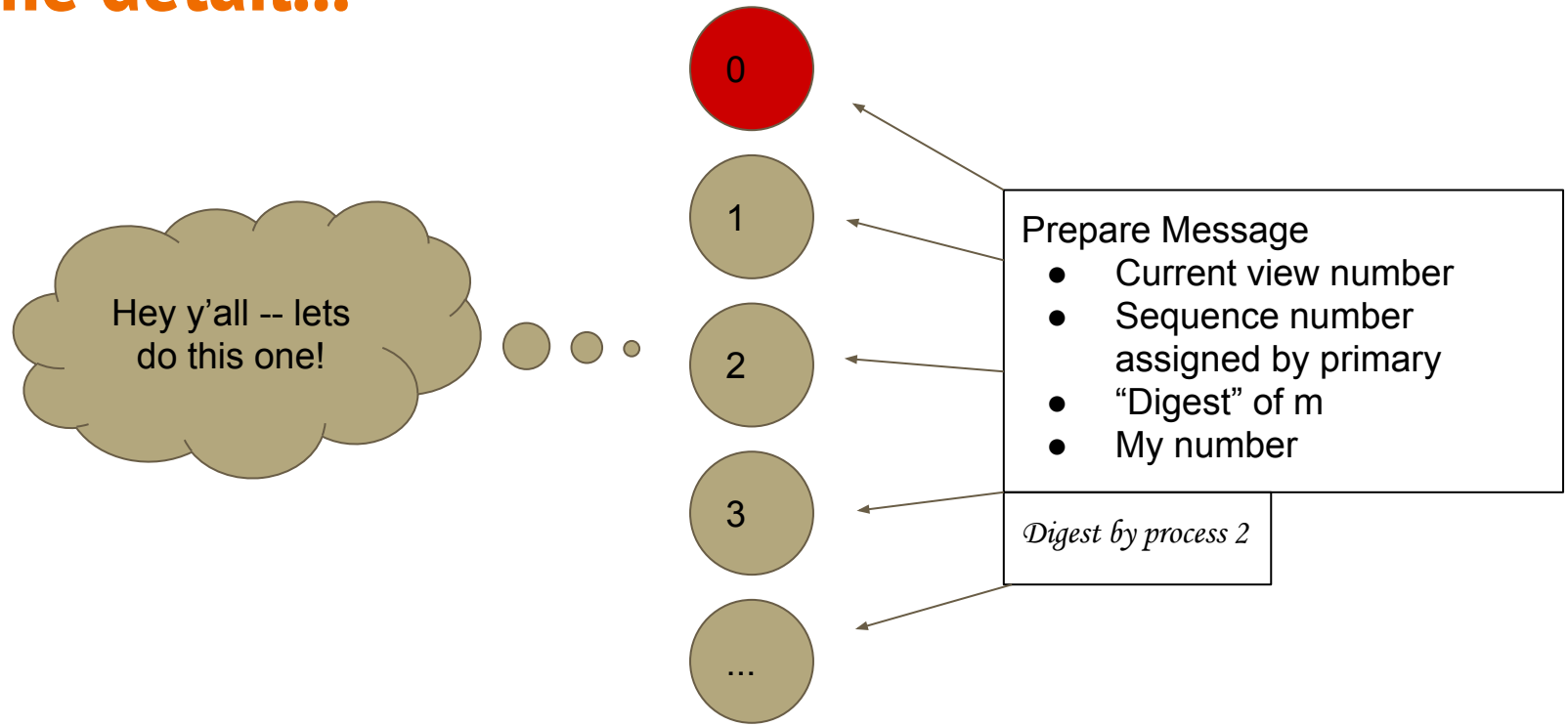
Some detail...

State Tracker For Node 2: Prepare



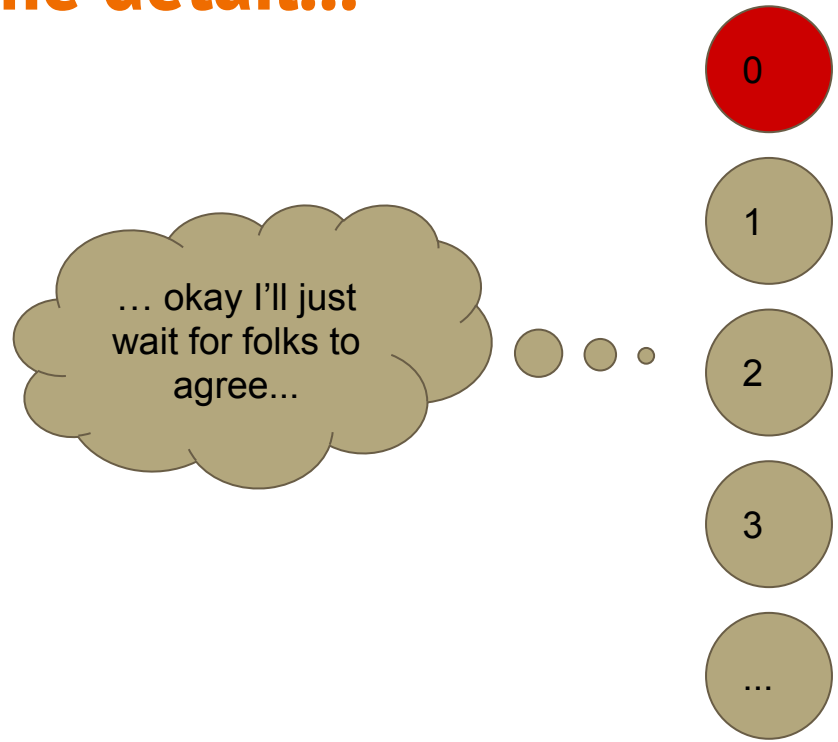
State Tracker For Node 2: Prepare

Some detail...



State Tracker For Node 2: Prepare

Some detail...



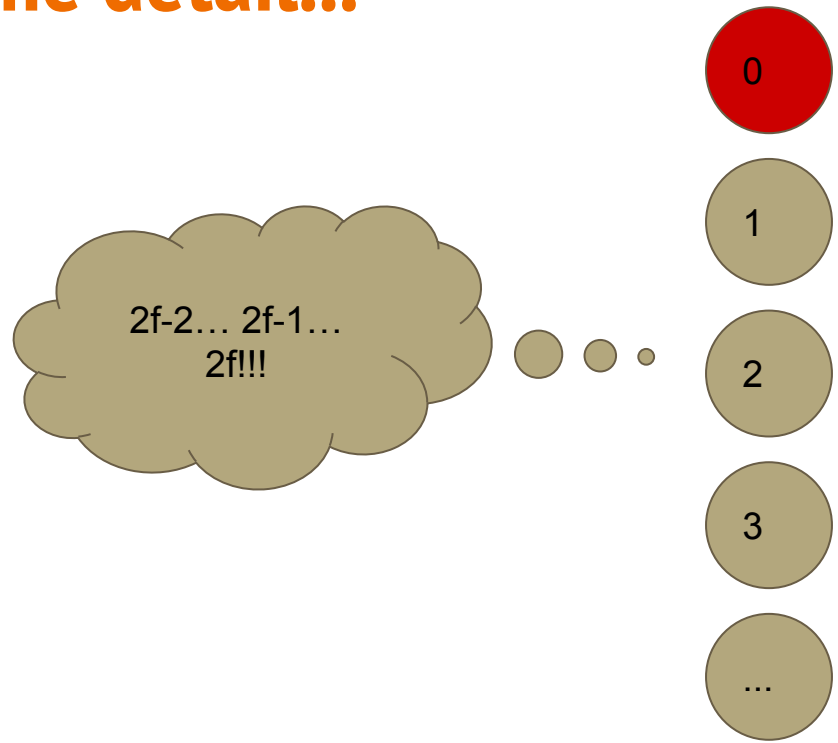
Some time later...

Some time later...

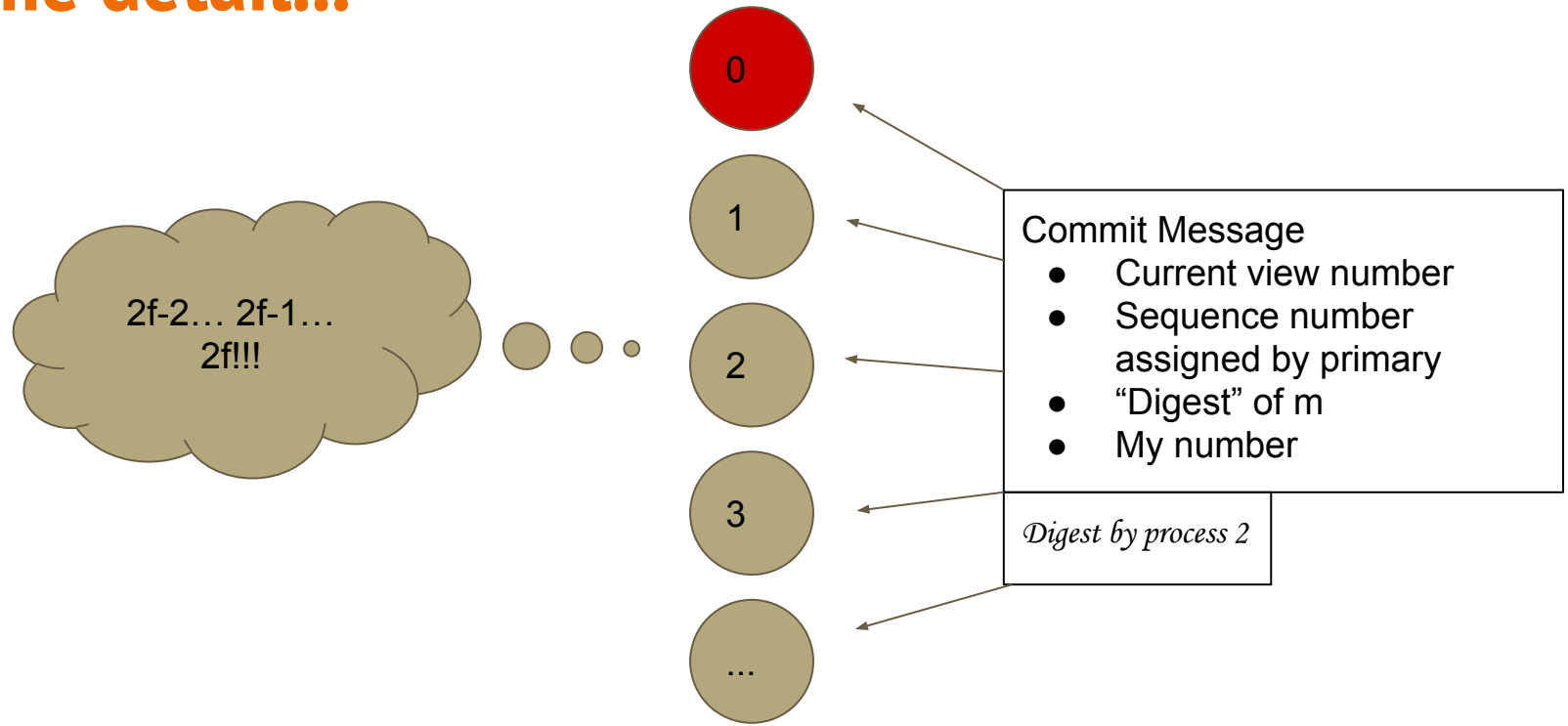
... after process 2 collects enough matching "prepare" statements from other replicas....

State Tracker For Node 2: Prepare

Some detail…

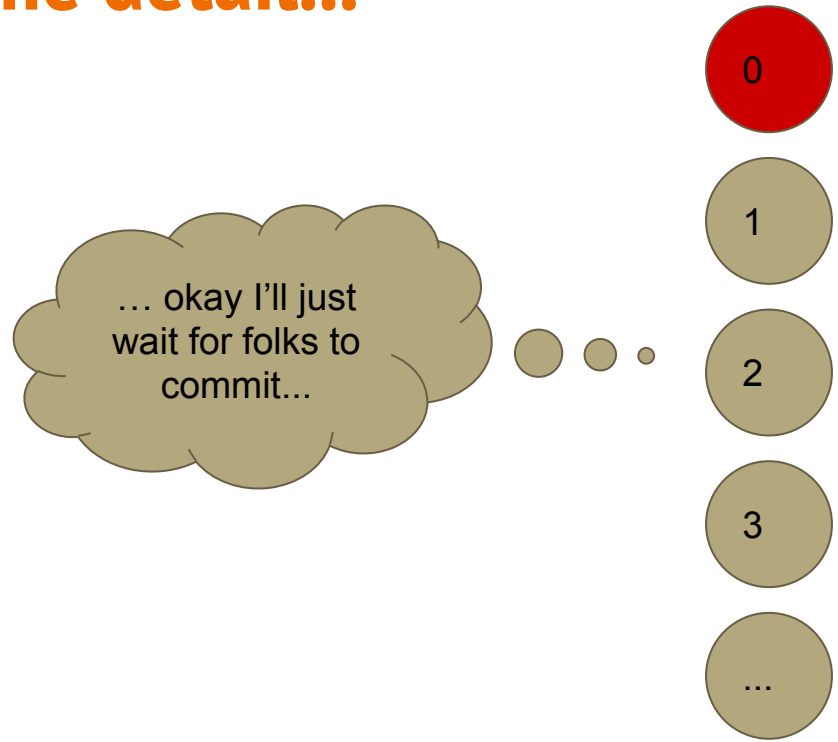


Some detail…


2f-2… 2f-1…
2f!!!
0
1
2
3
…
Commit Message
Current view number
Sequence number assigned by primary
"Digest" of m
My number
Digest by process 2

State Tracker For Node 2: Commit

Some detail...



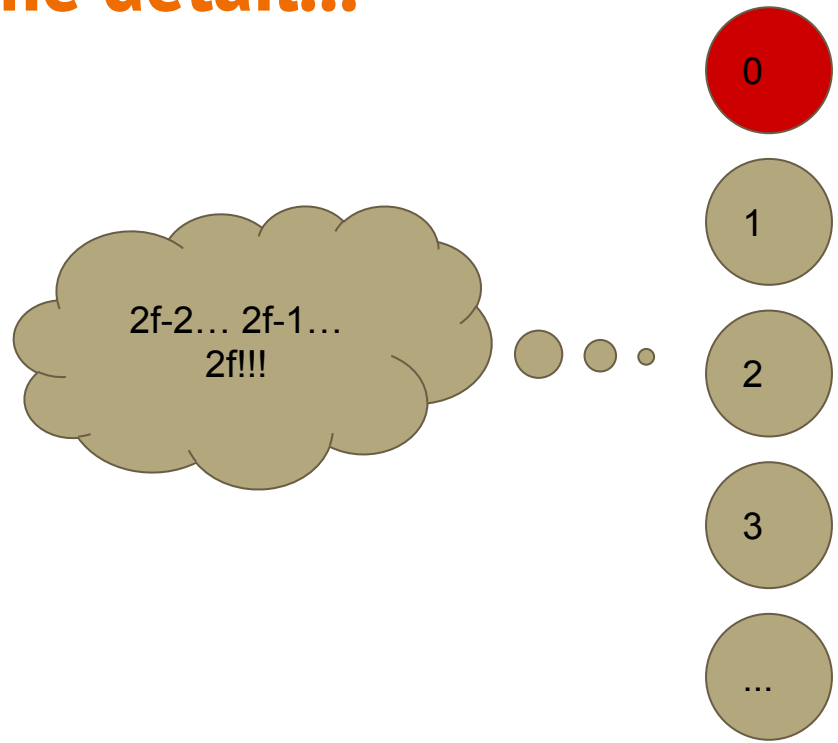
Some time later...

Some time later...

... after process 2 collects enough matching "commit" statements from other replicas....

State Tracker For Node 2: Commit

Some detail…

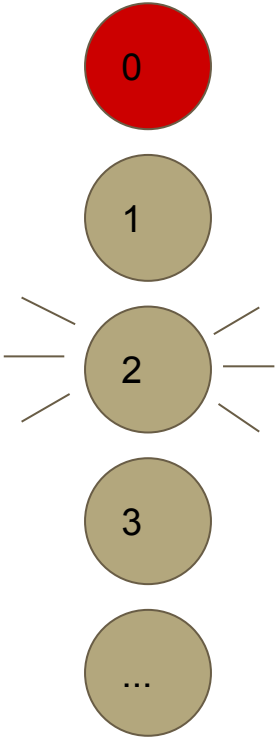


Some detail...

State Tracker For Node 2: Reply

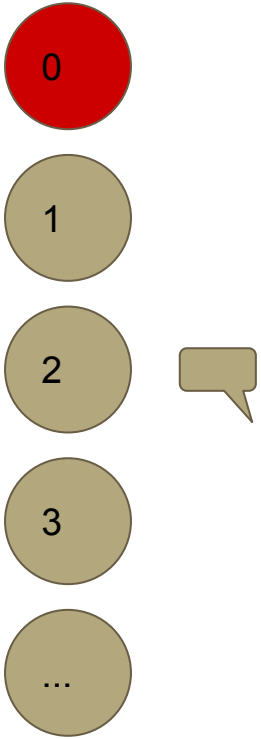
Some detail...

State Tracker For Node 2: Reply



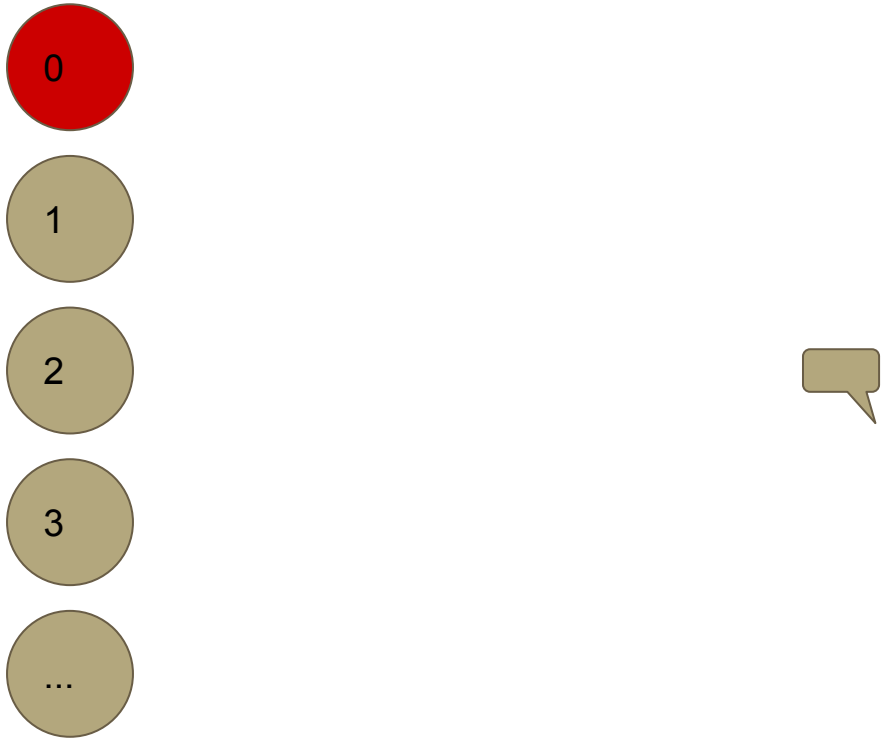
Some detail...

State Tracker For Node 2: Reply



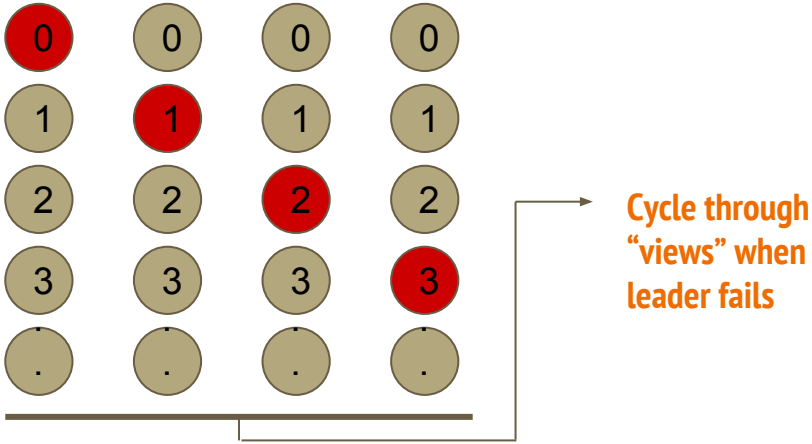
Some detail...

State Tracker For Node 2: Reply



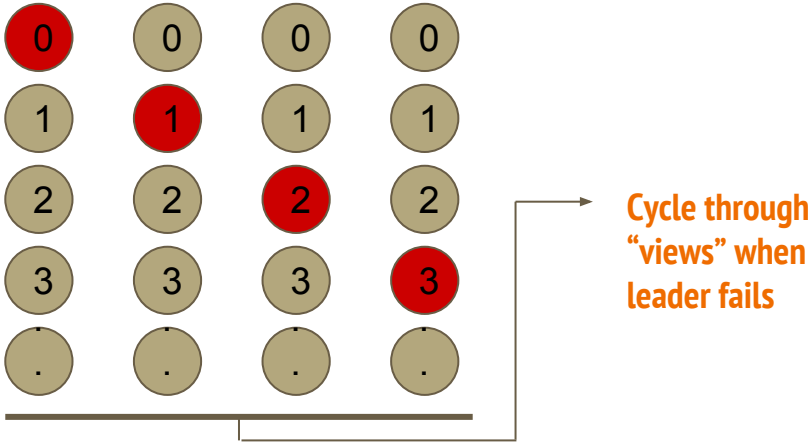
Failure Sketch...

Failure Sketch...



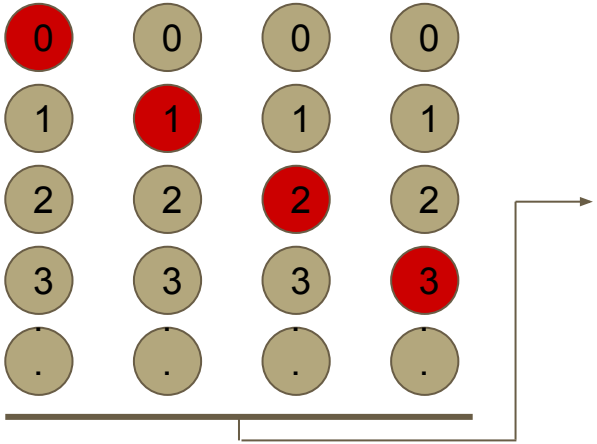
Cycle through "views" when leader fails

Failure Sketch...



Cycle through "views" when leader fails

Failure Sketch...



Cycle through "views" when leader fails

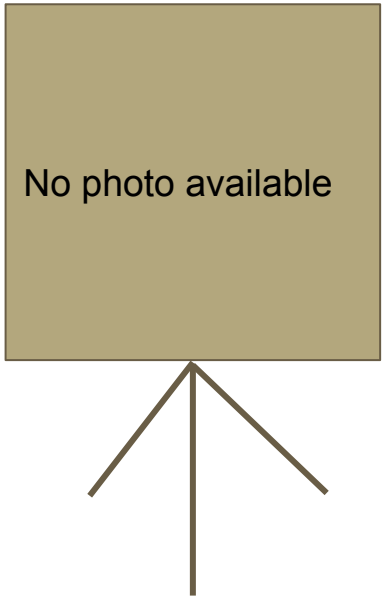
1. Time out
2. Message: View Change Please
3. Message: View Change

Discussion Question

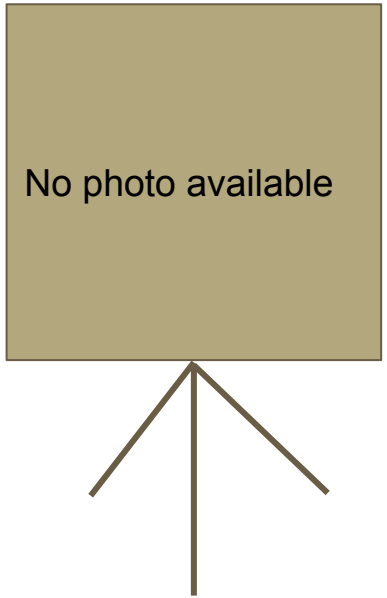
Does Byzantine fault tolerance matter?

Do you buy the motivation?

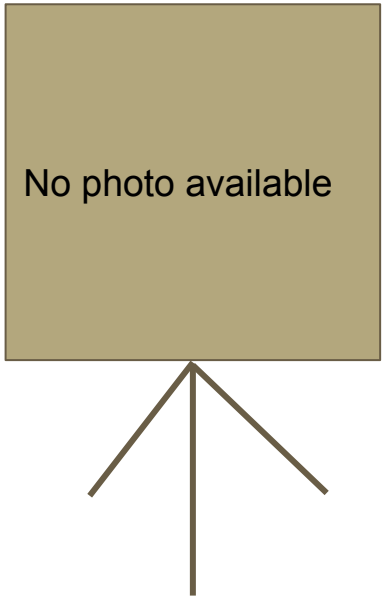
Atomic Broadcast: From Simple Message Diffusion to Byzantine Agreement



Flaviu Cristian



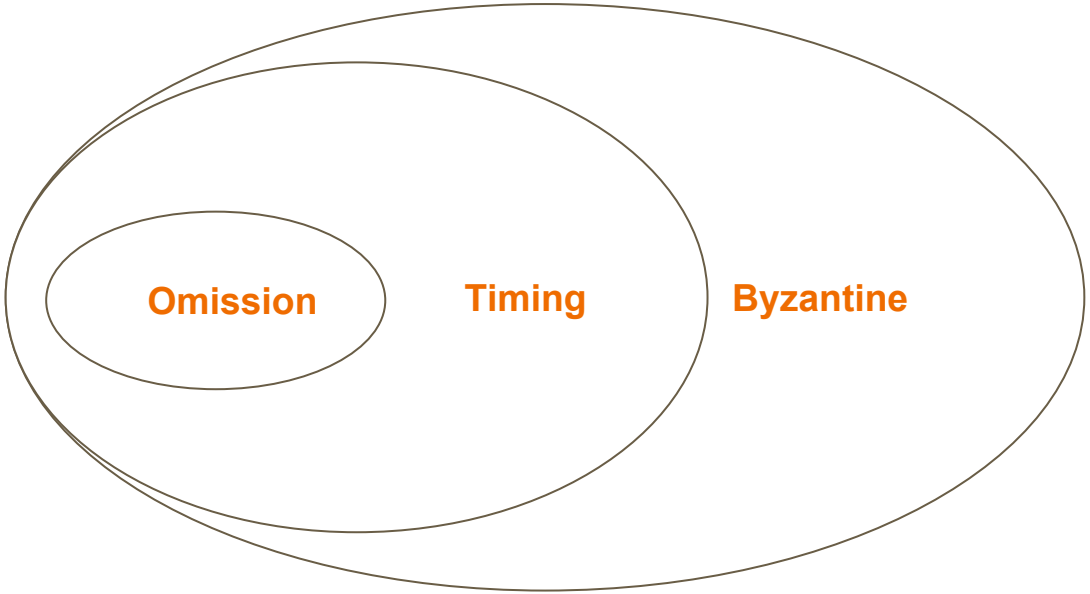
Houtan Aghili



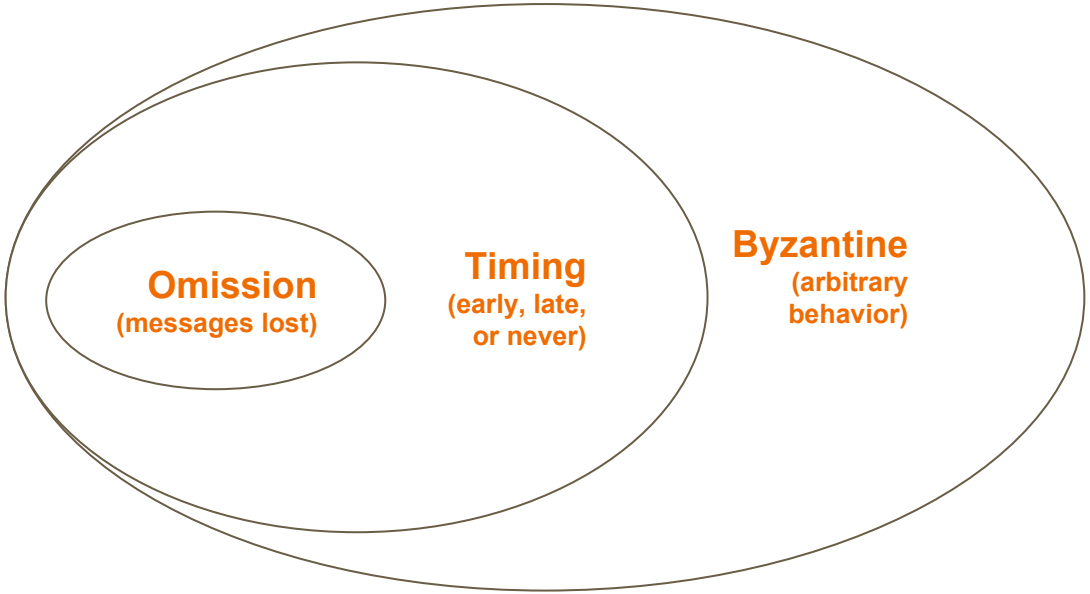
Ray Strong

Danny Dolev

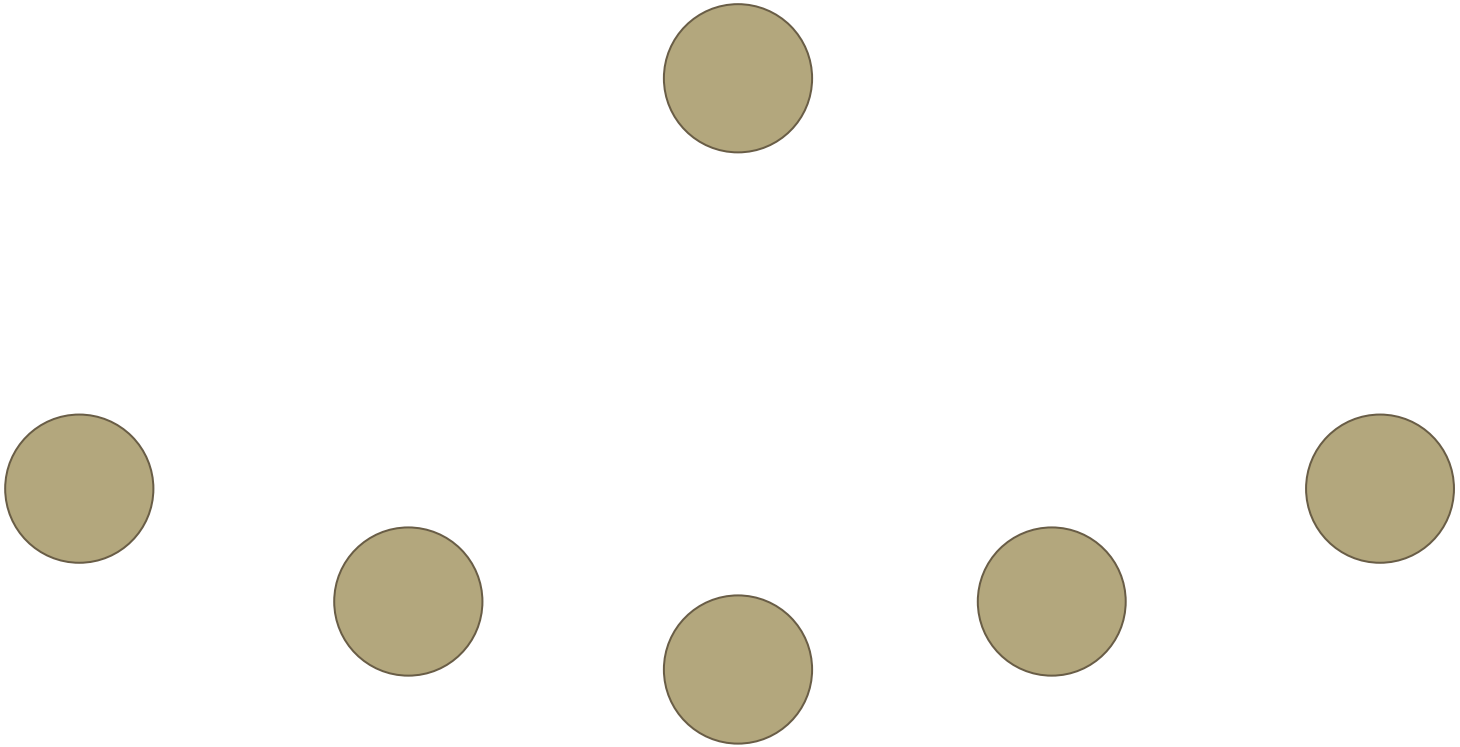
As you already know...



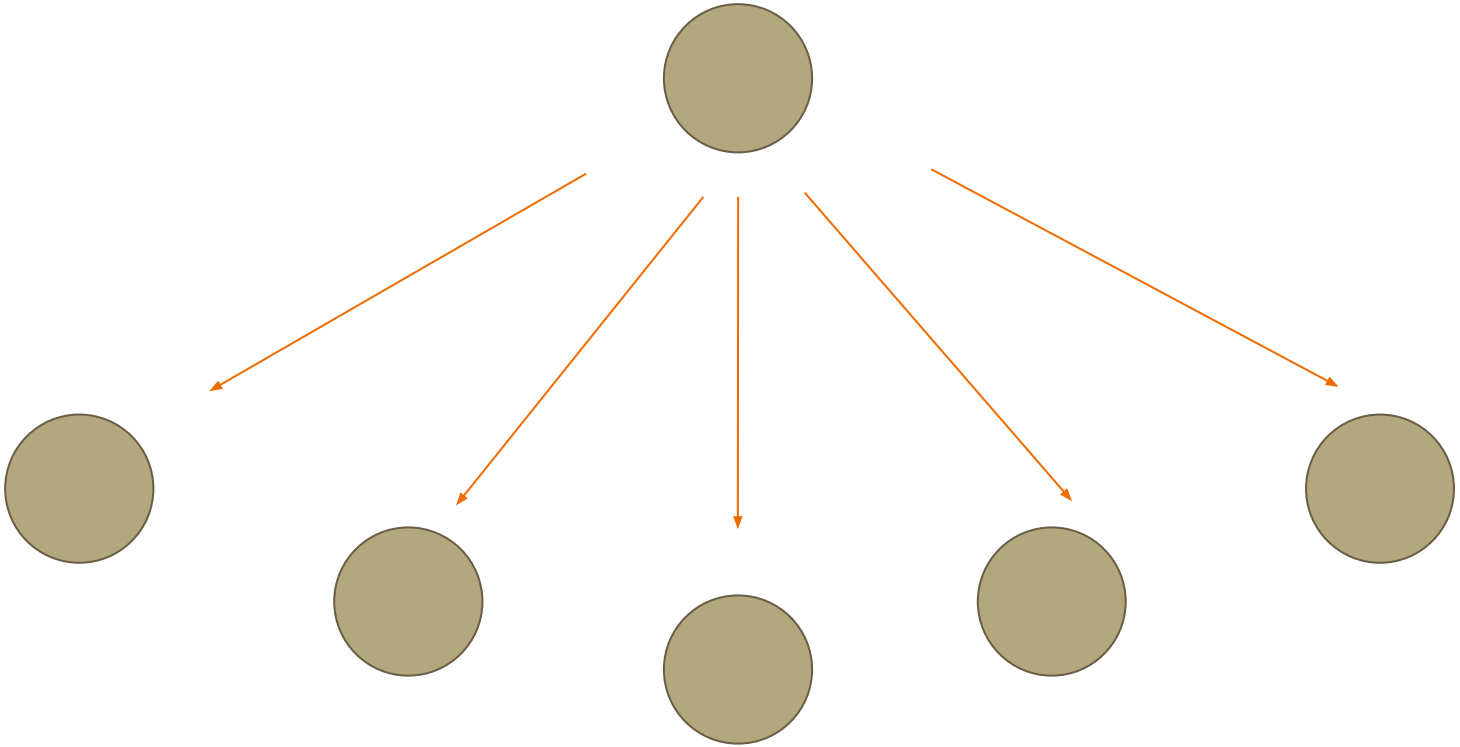
As you already know...



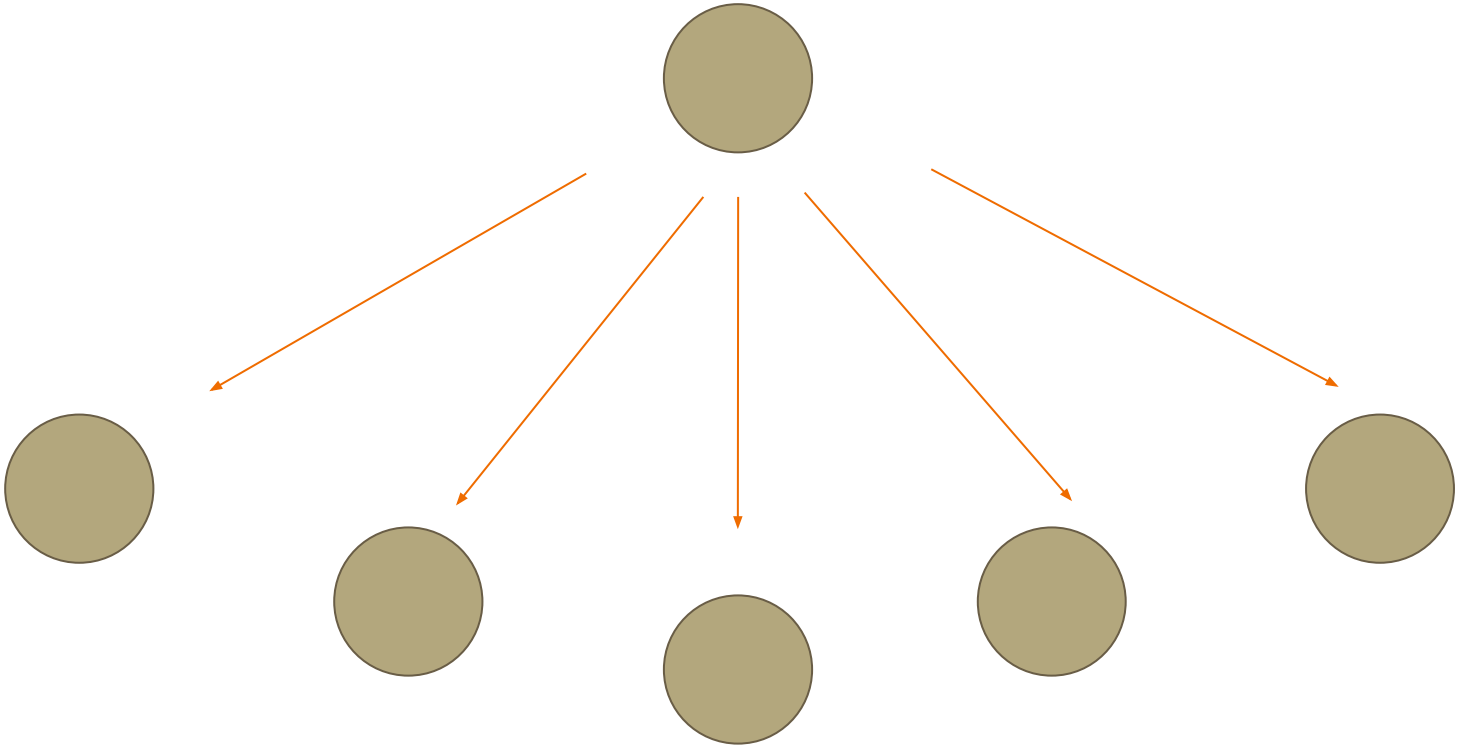
Broadcast



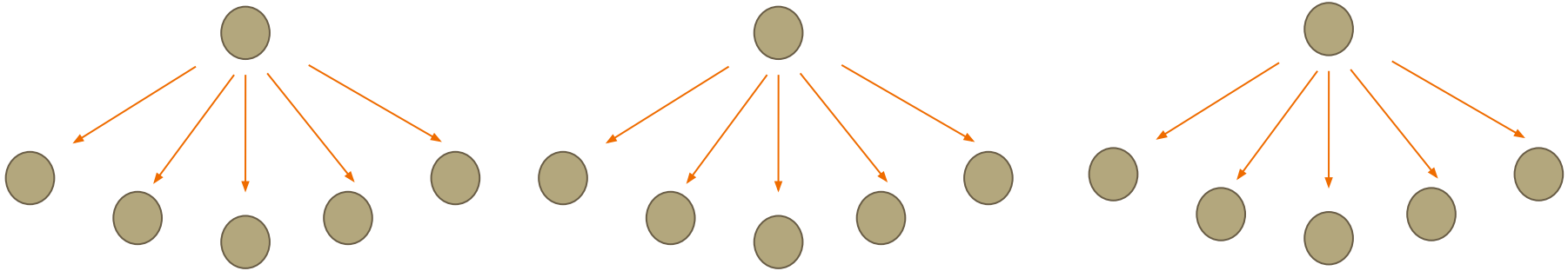
Broadcast



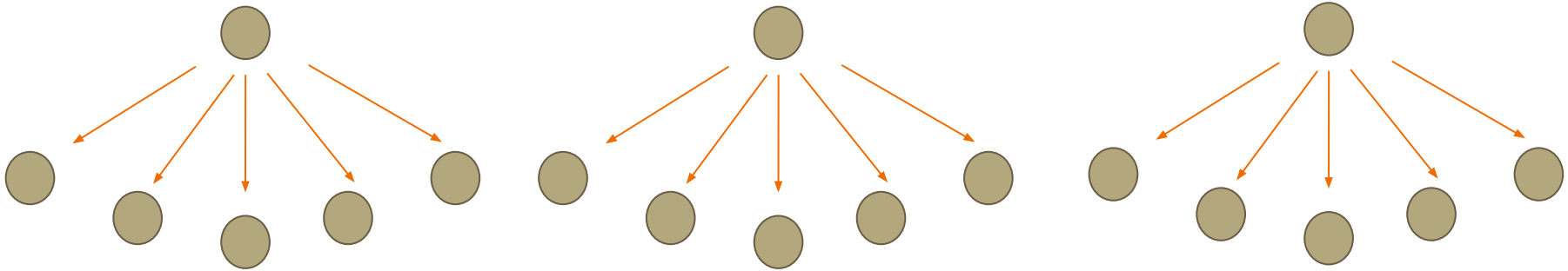
Atomic Broadcast?



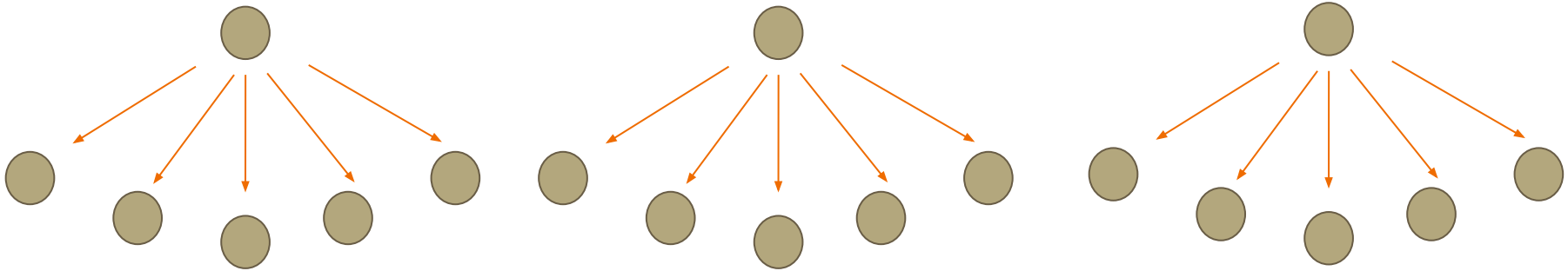
Atomic Broadcast



Atomic Broadcast

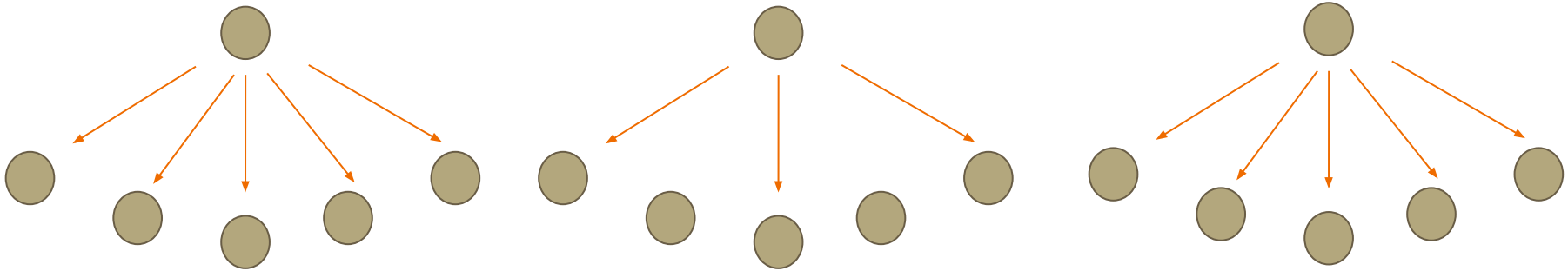


Atomic Broadcast



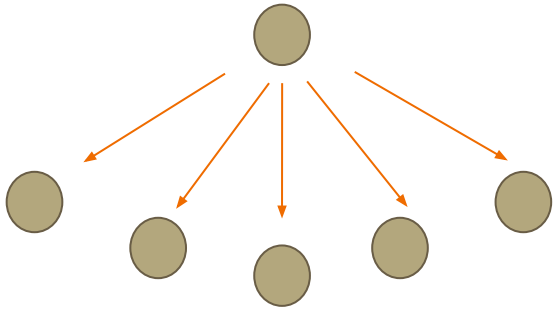
T -> T + Δ

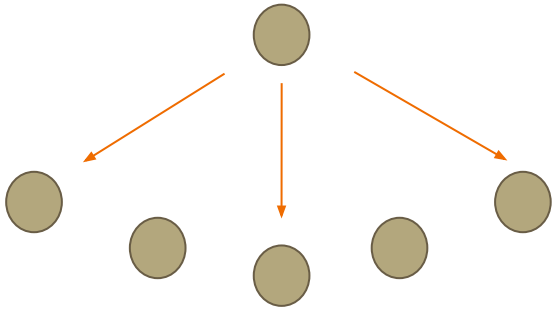
Atomic Broadcast

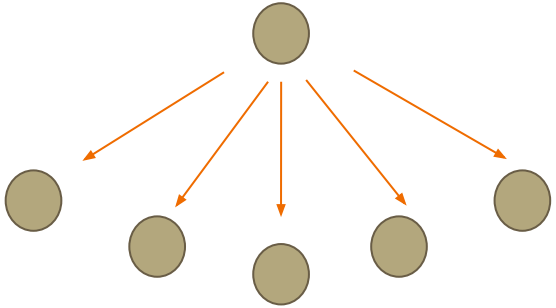


T -> T + Δ

Atomic Broadcast



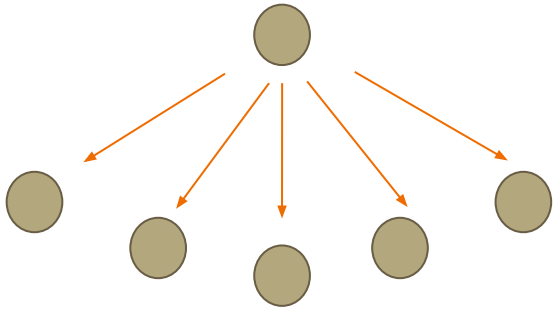


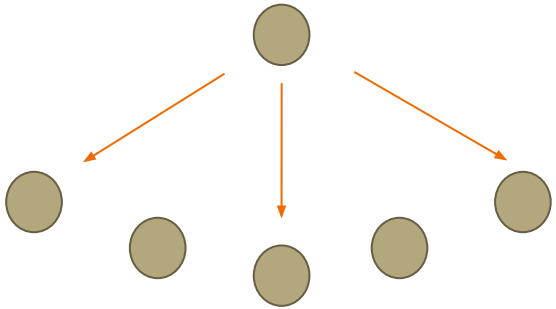


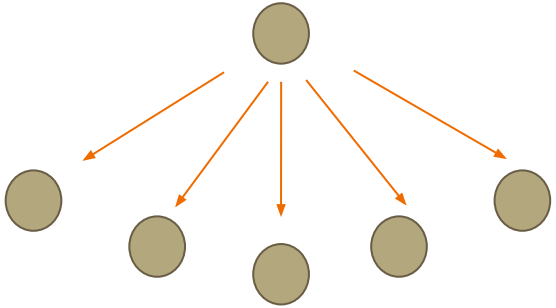
T -> T + Δ

Atomic Broadcast

1. A
2. B
3. C



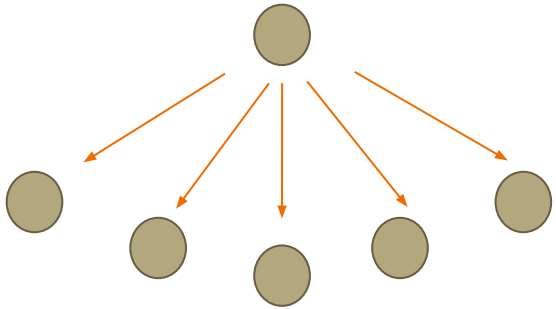


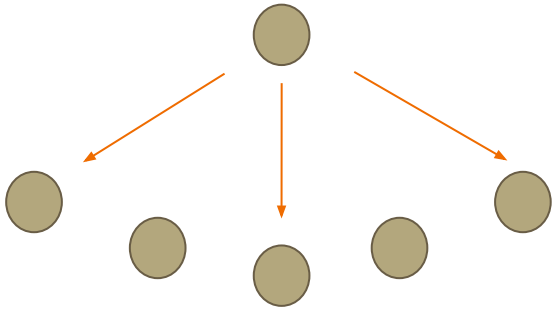


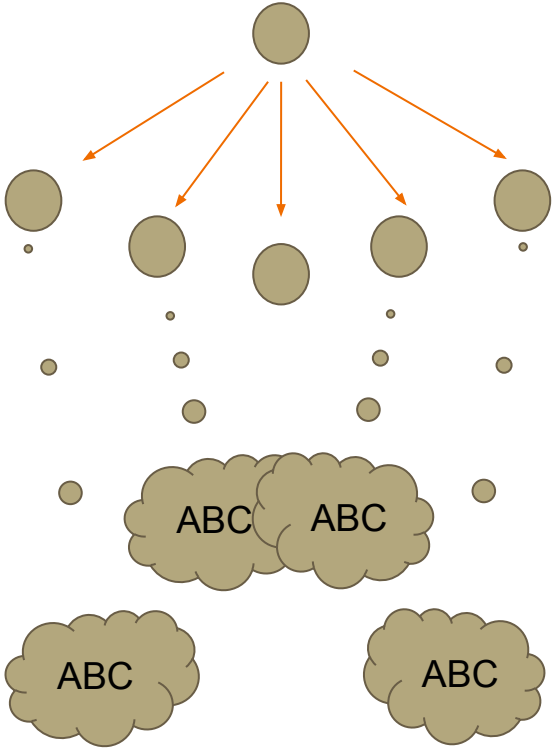
T -> T + Δ

Atomic Broadcast

1. A
2. B
3. C







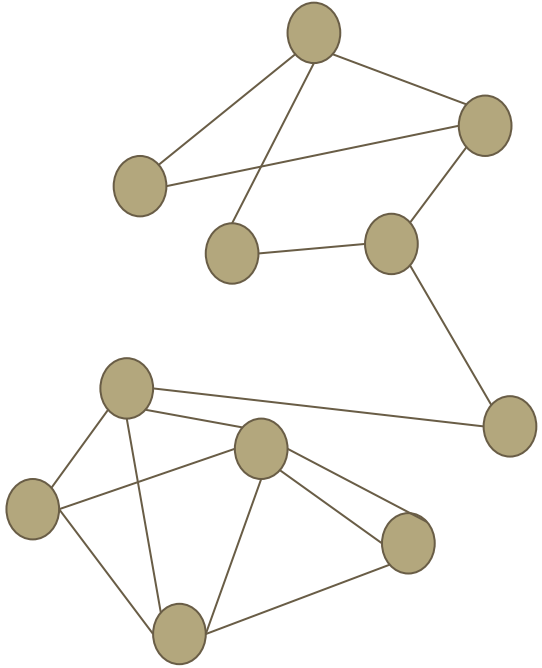
T -> T + Δ

Assumptions

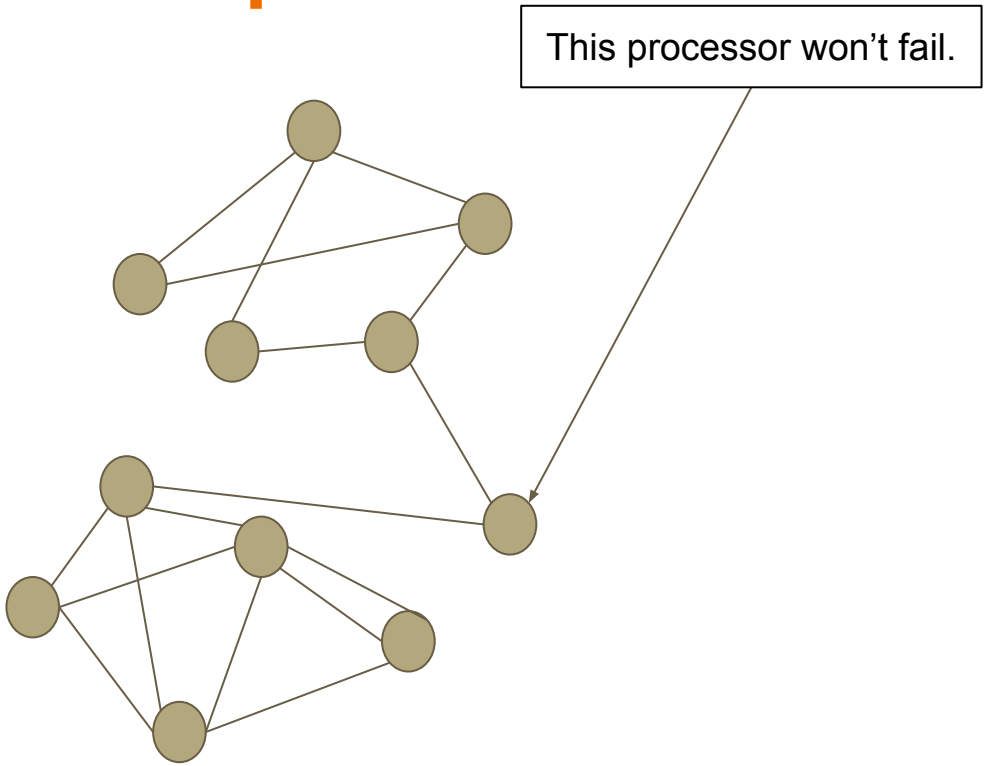
Assumptions (seeing a pattern? few unifying assumptions, assumptions made for ease of proof rather than realism)

(Anyone else feel this way?) /tangent

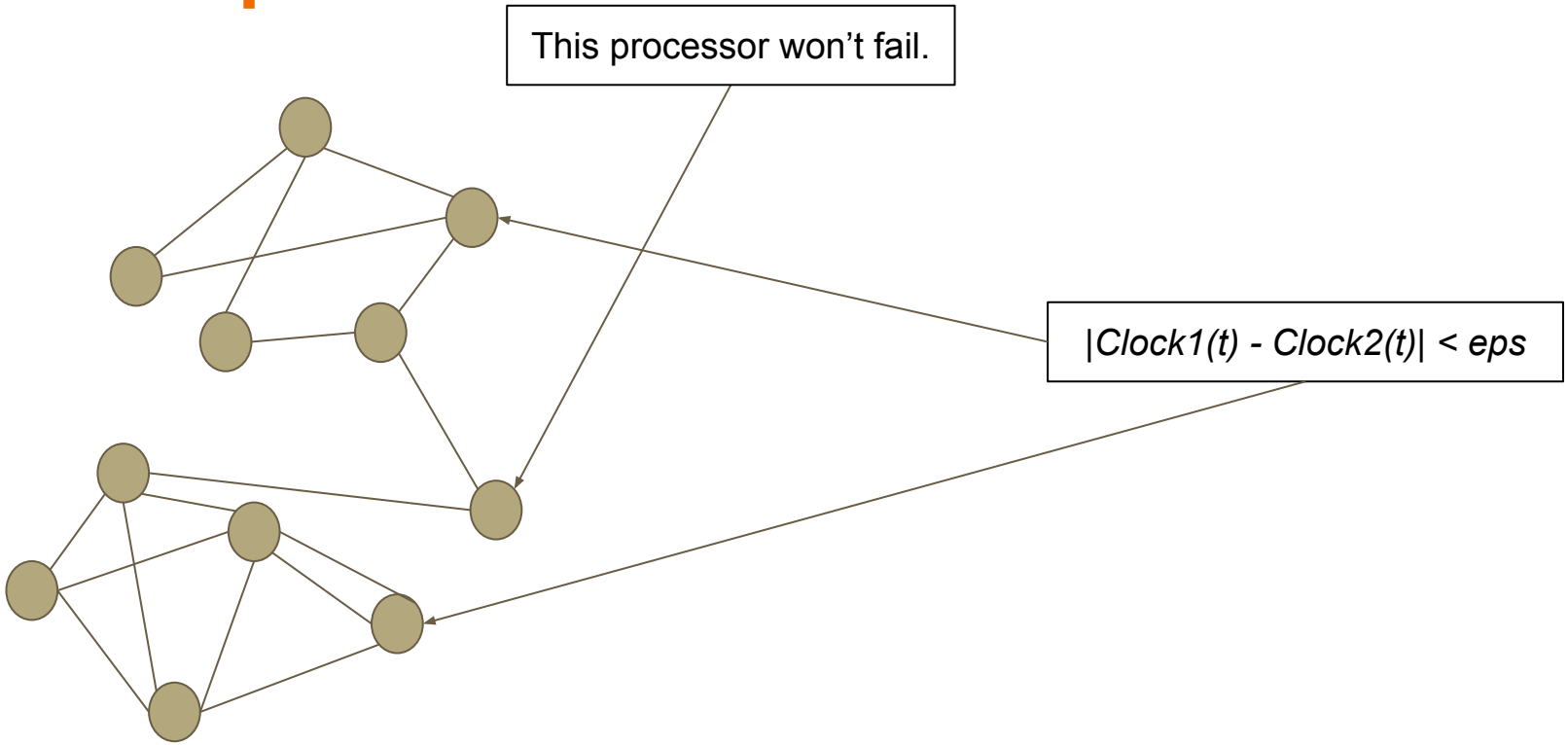
Assumptions



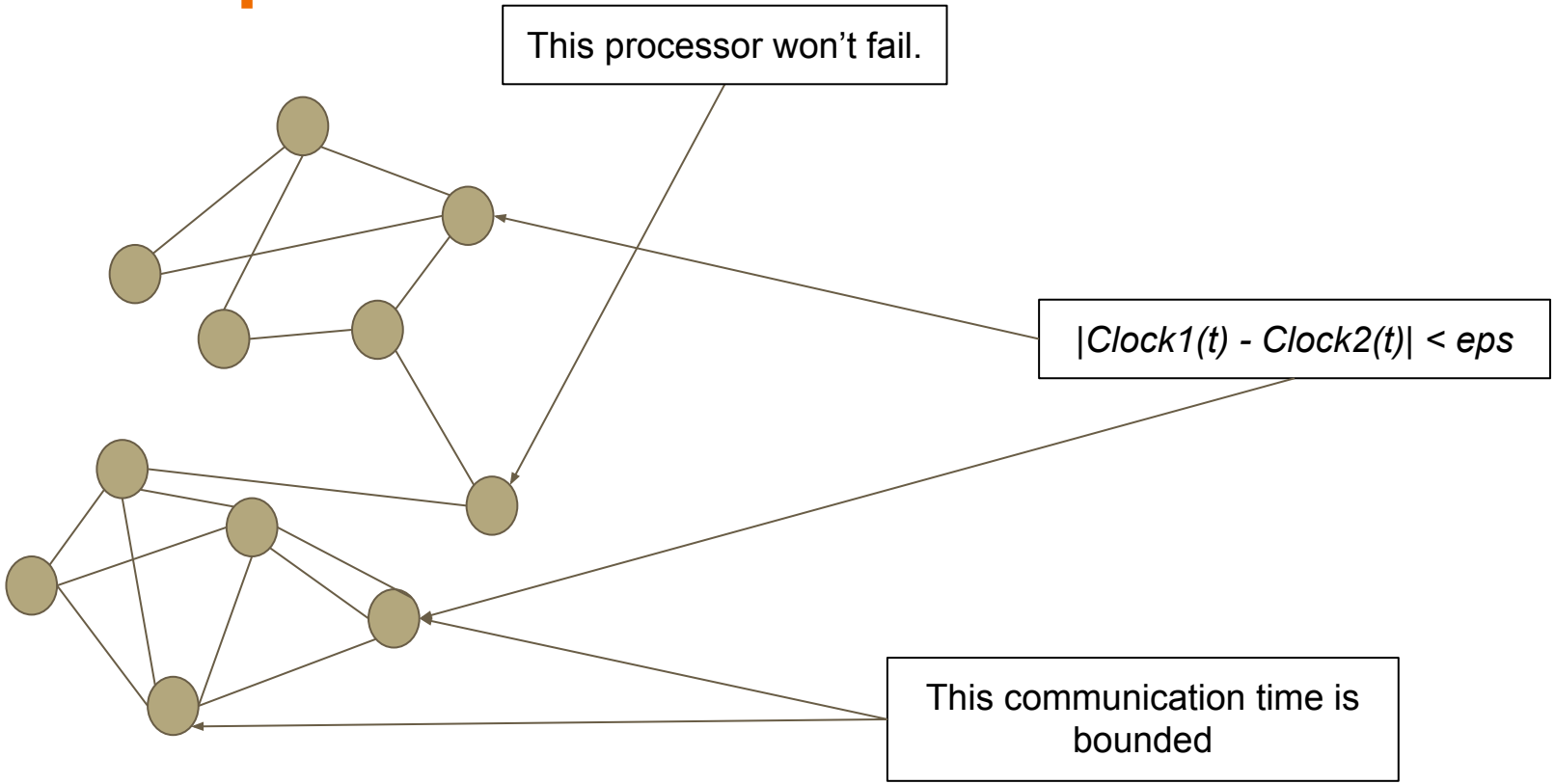
Assumptions



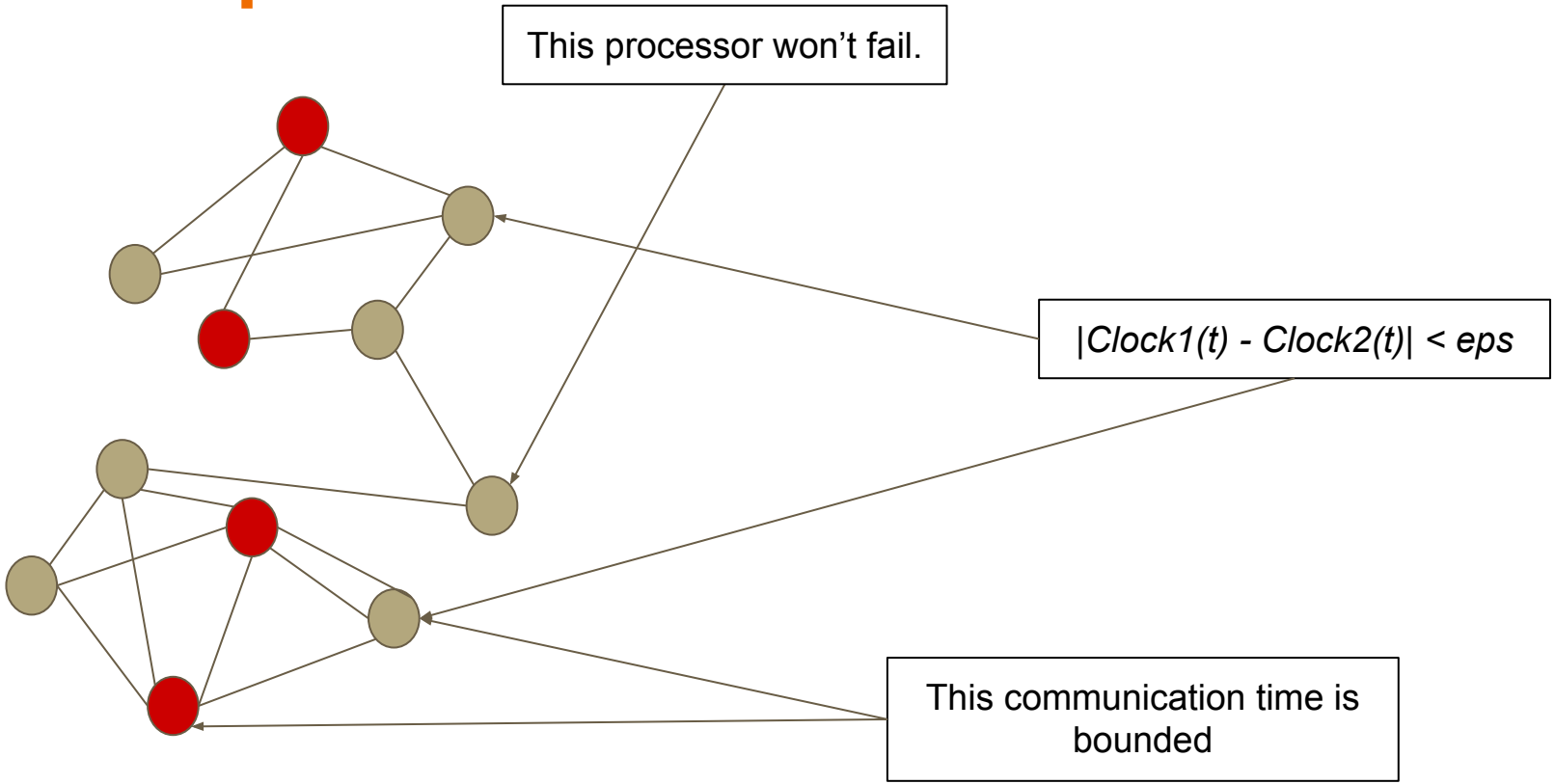
Assumptions



Assumptions



Assumptions



“Diffusion Induction Principle”

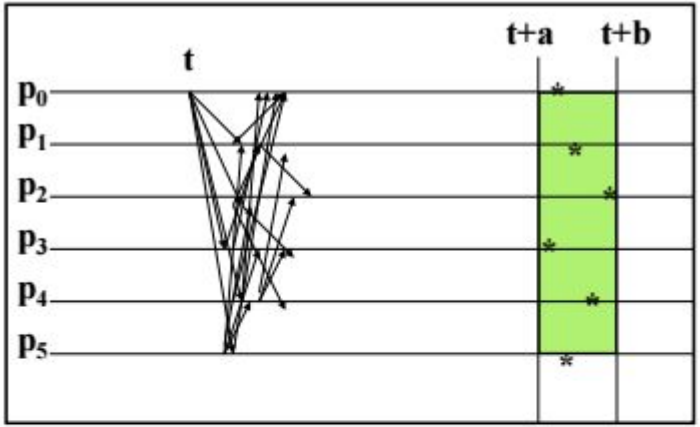
“Diffusion Induction Principle”

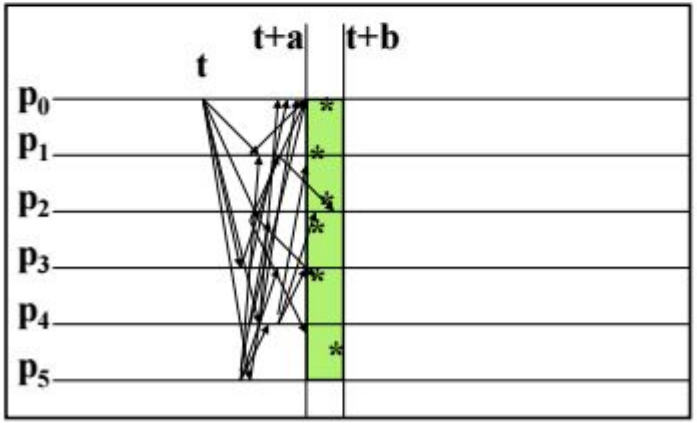
In a connected graph, everyone will eventually get the message.

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


t
t+a
t+b
p0
p1
p2
p3
p4
p5


t
t+a
t+b
p0
p1
p2
p3
p4
p5

CASD for Omission Failures...

Δ = f(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...

Δ = f(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...

Δ = f(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...

Δ = f(Diameter of network, clock skew, processing/messaging time, number of hops for a given message)

CASD for Timing Failures...

Δ_i = f(Diameter of network, clock skew, processing/messaging time, number of hops for a given message)

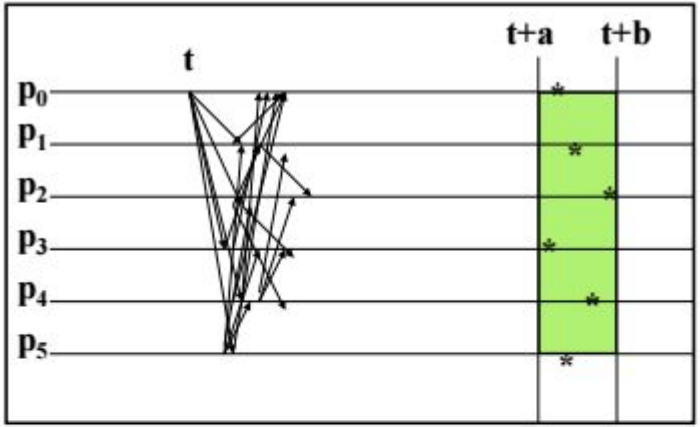
CASD for Timing Failures…

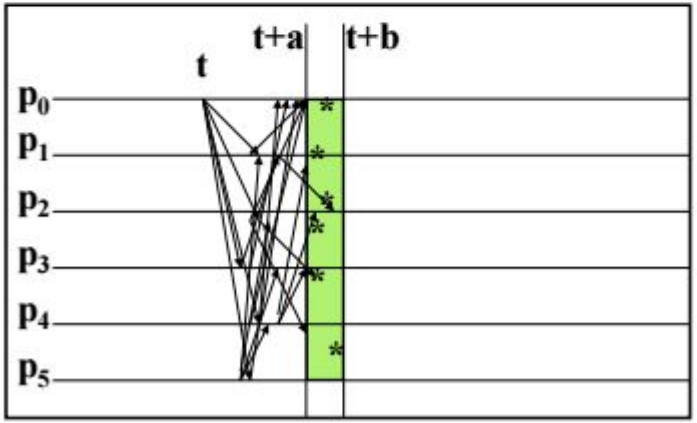
Δ_i = f(Diameter of network, clock skew, processing/messaging time, number of hops for a given message)

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


t
t+a
t+b
p0
p1
p2
p3
p4
p5


t
t+a
t+b
p0
p1
p2
p3
p4
p5

Jack's thoughts...

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