HoneyBadgerBFT

Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, Dawn Song

Honey Badger of BFT Protocols
Presented By

Akanksha Kulkarni
Hemang Singh
Varun Singh
Preyash Yadav
Sanjana Mali

Flow of a typical weakly synchronous protocol

- Start with leader and timeout parameter
- 2 workflows - fast path and view change
- Good performace - if leader is non faulty, messages are delivered within Δ
- if leader is faulty, re-elect and increase timeout Δ

Problems of timing assumptions

Small Timeout:
- Leads to premature assumptions of failure
- Causes unnecessary view-synchronizations, reducing efficiency

Large Timeout:
- Increases latency, as processes wait longer than necessary before proceeding
- Reduces responsiveness to actual faults, delaying fault recovery

Selecting a timeout parameter is tricky!
Asynchronous protocols do not have those limiting assumptions

No parameter to tune!

When Weak Synchrony Fails

- Attack on Weakly Synchronous
- Messages sent when faulty node is elected as leader
- Delaying messages from honest leaders
- Cycle of repeated view changes
- Progress is stopped!

Honey Badger of BFT Protocols

Presented By

Akanksha Kulkarni
Hemang Singh
Varun Singh
Preyash Yadav
Sanjana Mali

Flow of a typical weakly synchronous protocol

- Start with leader and timeout parameter
- 2 workflows - fast path and view change
- Good performace - if leader is non faulty, messages are delivered within Δ
- If leader is faulty, re-elect and increase timeout Δ

Problems of timing assumptions

Small Timeout:

- Leads to premature assumptions of failure
- Causes unnecessary retransmissions, reducing efficiency

Large Timeout:

- Increases latency, as processes wait longer than necessary before proceeding
- Reduces responsiveness to actual faults, delaying fault recovery

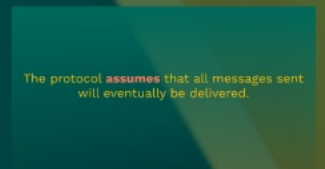
Selecting a timeout parameter is tricky!
Asynchronous protocols do not have these timing assumptions

No parameter to tune!

When Weak Synchrony Fails

- Attack on Weakly Synchronous
- Messages sent when faulty node is elected as leader
- Delaying messages from honest leaders
- Cycle of repeated view changes
- Progress is stopped!

High-Level Flowchart of the HoneyBadger Protocol



Overview

Why Is It Called the Honey Badger Protocol?

- **Resilience:** Handles faults and attacks like a fearless honey badger.
- **Adaptability:** Thrives in asynchronous, unpredictable environments.
- **Security:** Ensures consensus despite malicious participants.

The protocol **assumes** that all messages sent will eventually be delivered.

Overview

Overview

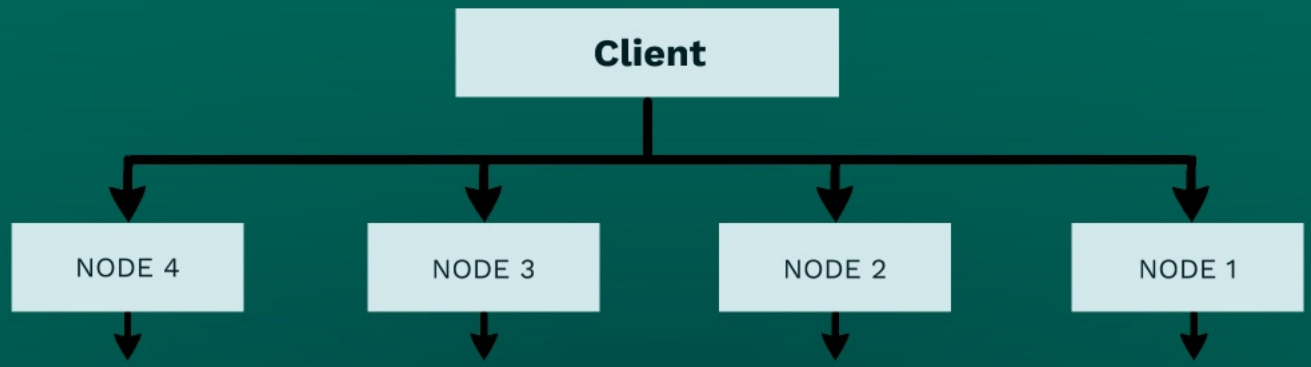
Overview

Client

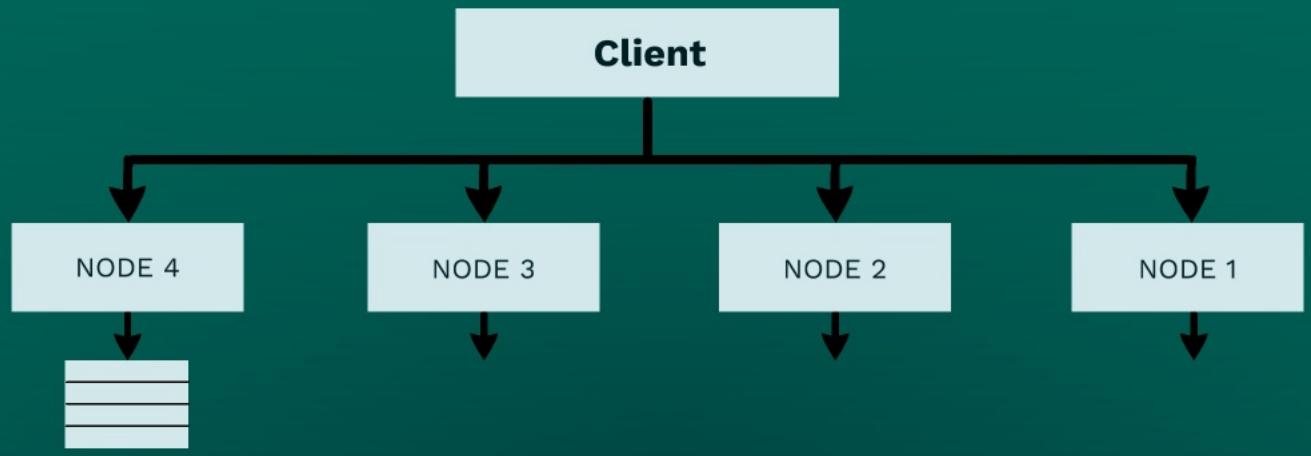
Overview

Overview

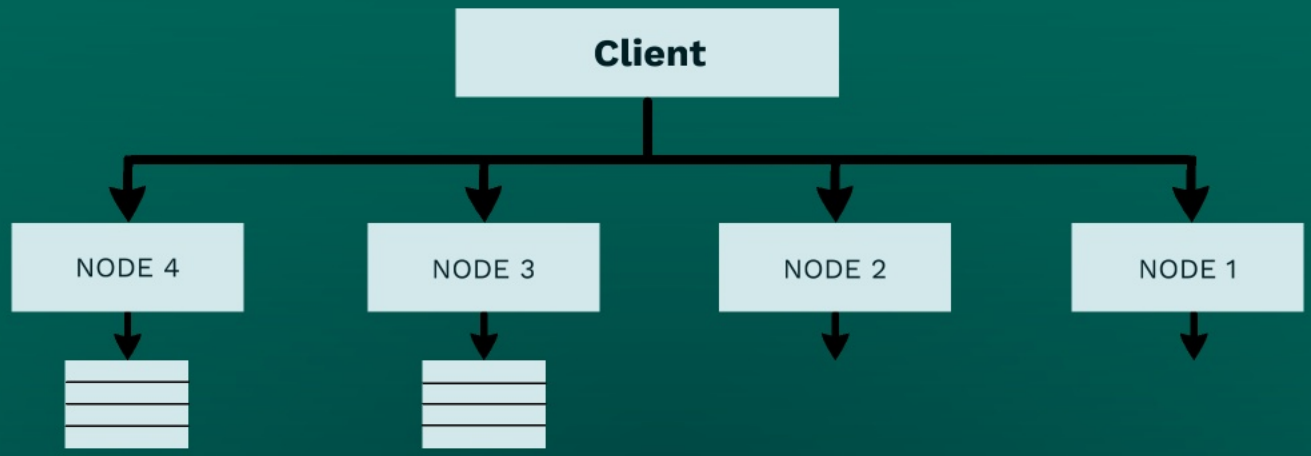
Overview



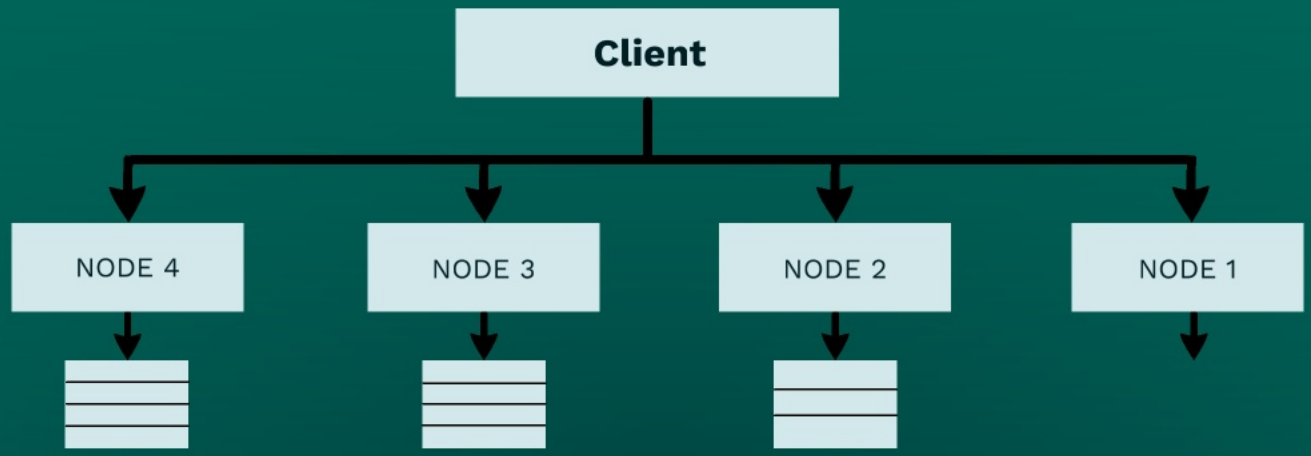
Overview



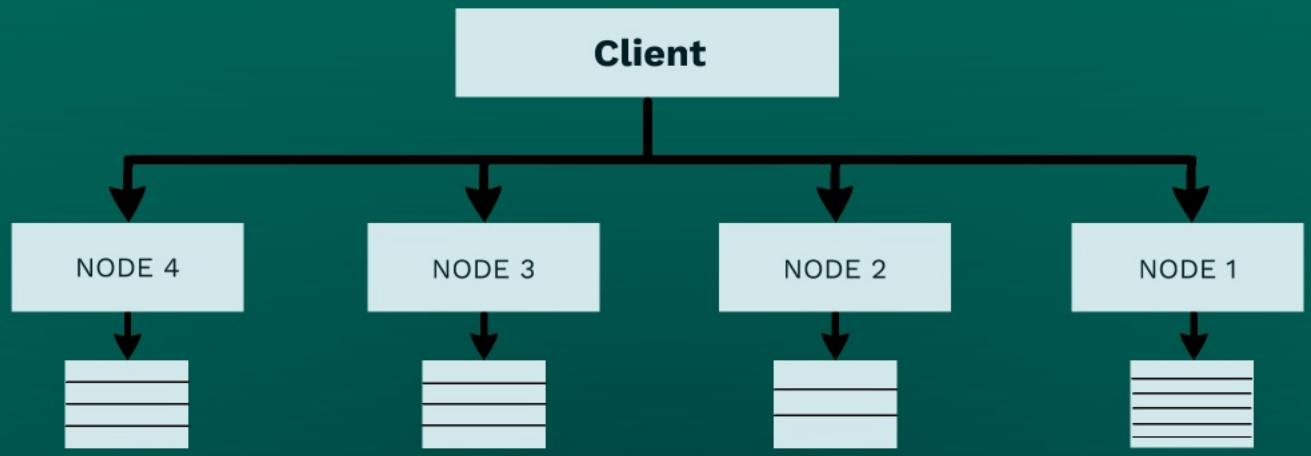
Overview



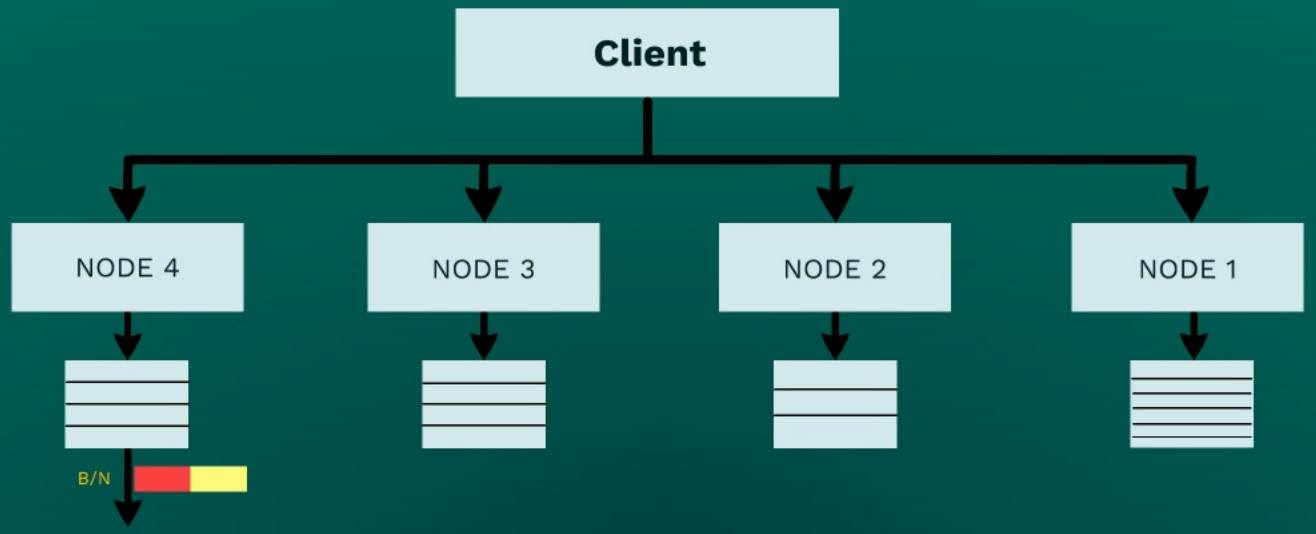
Overview



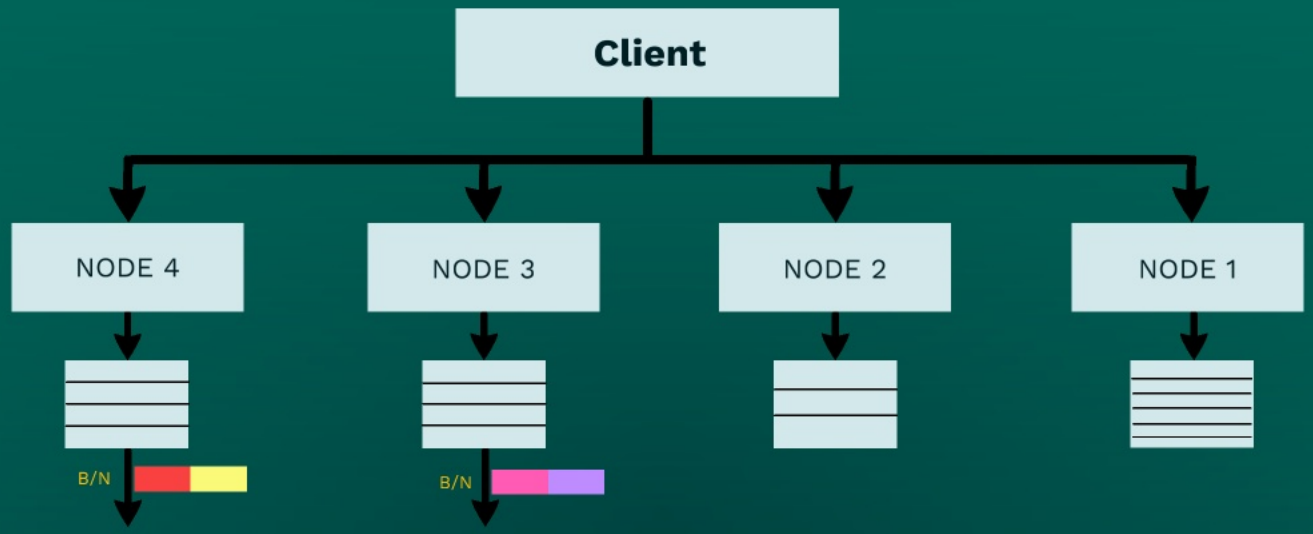
Overview



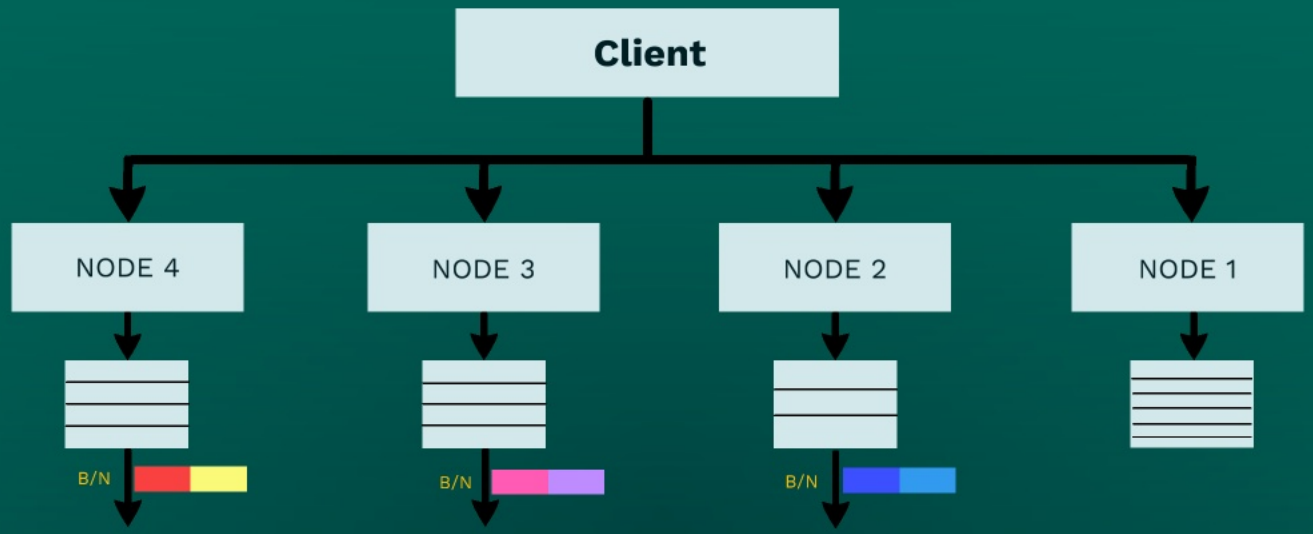
Overview



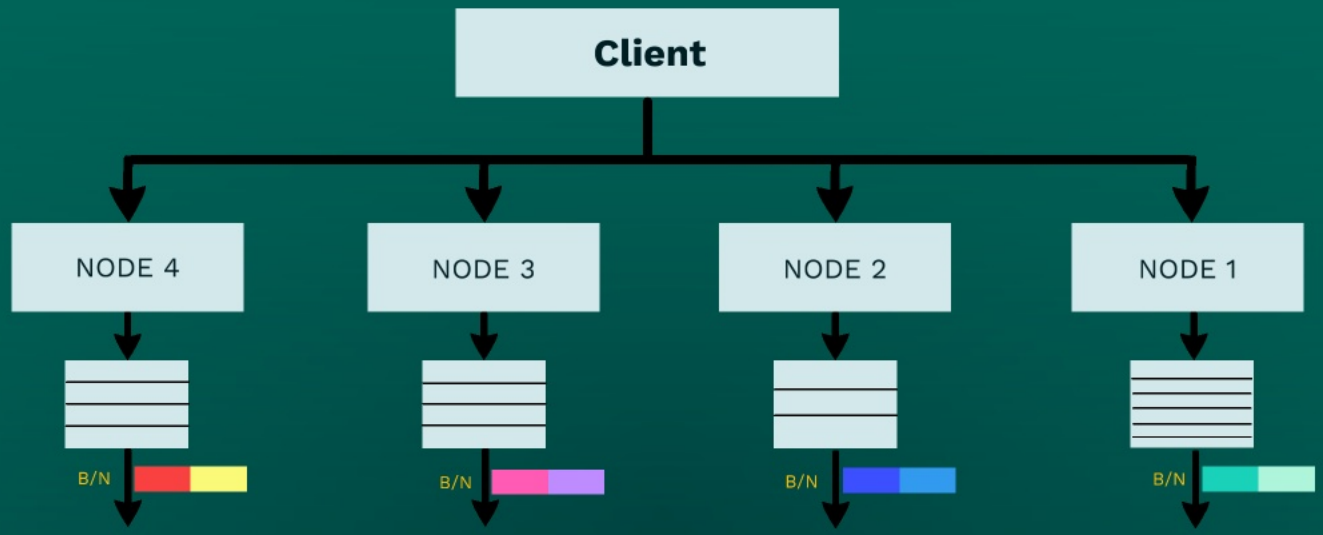
Overview



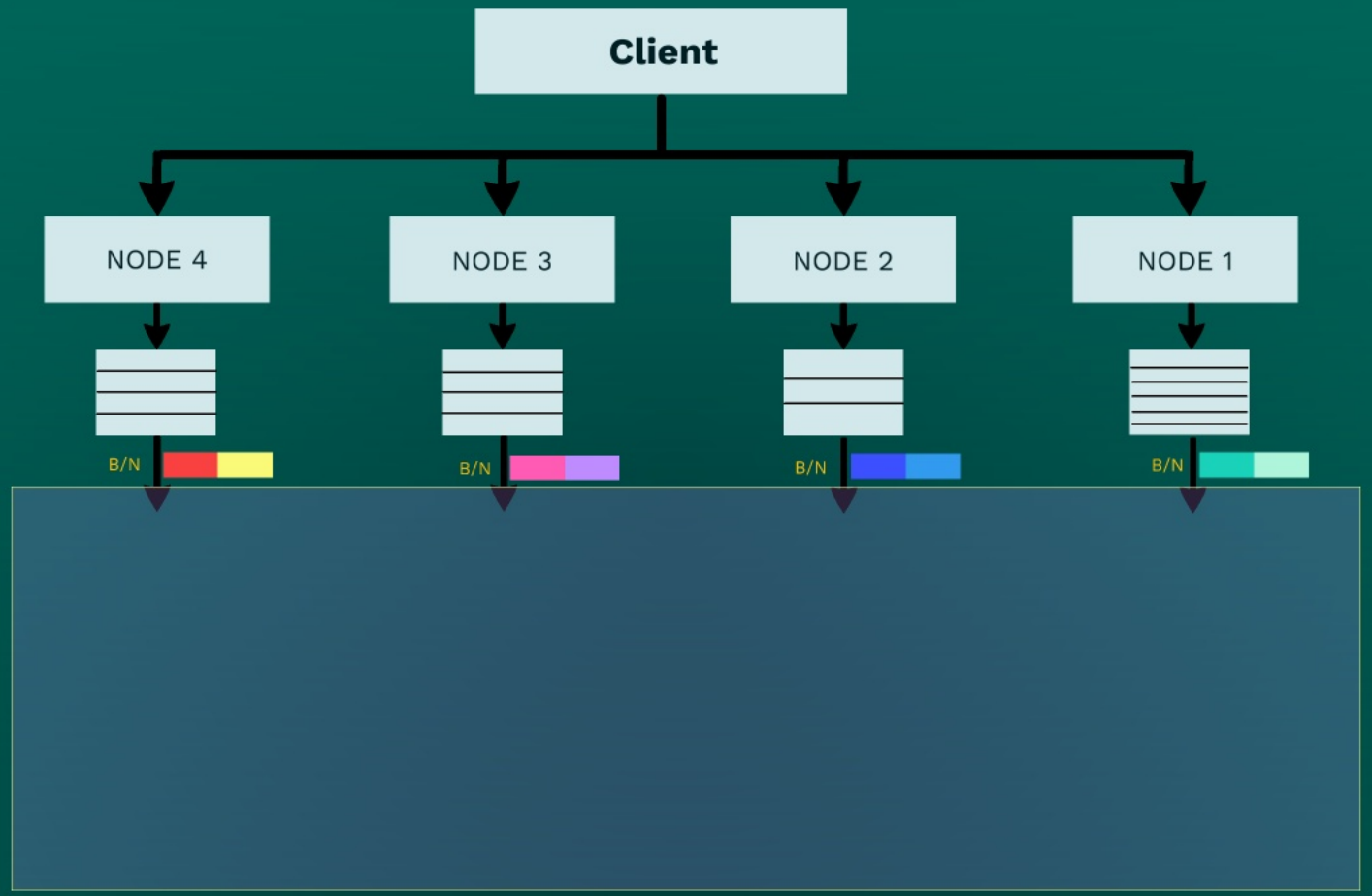
Overview



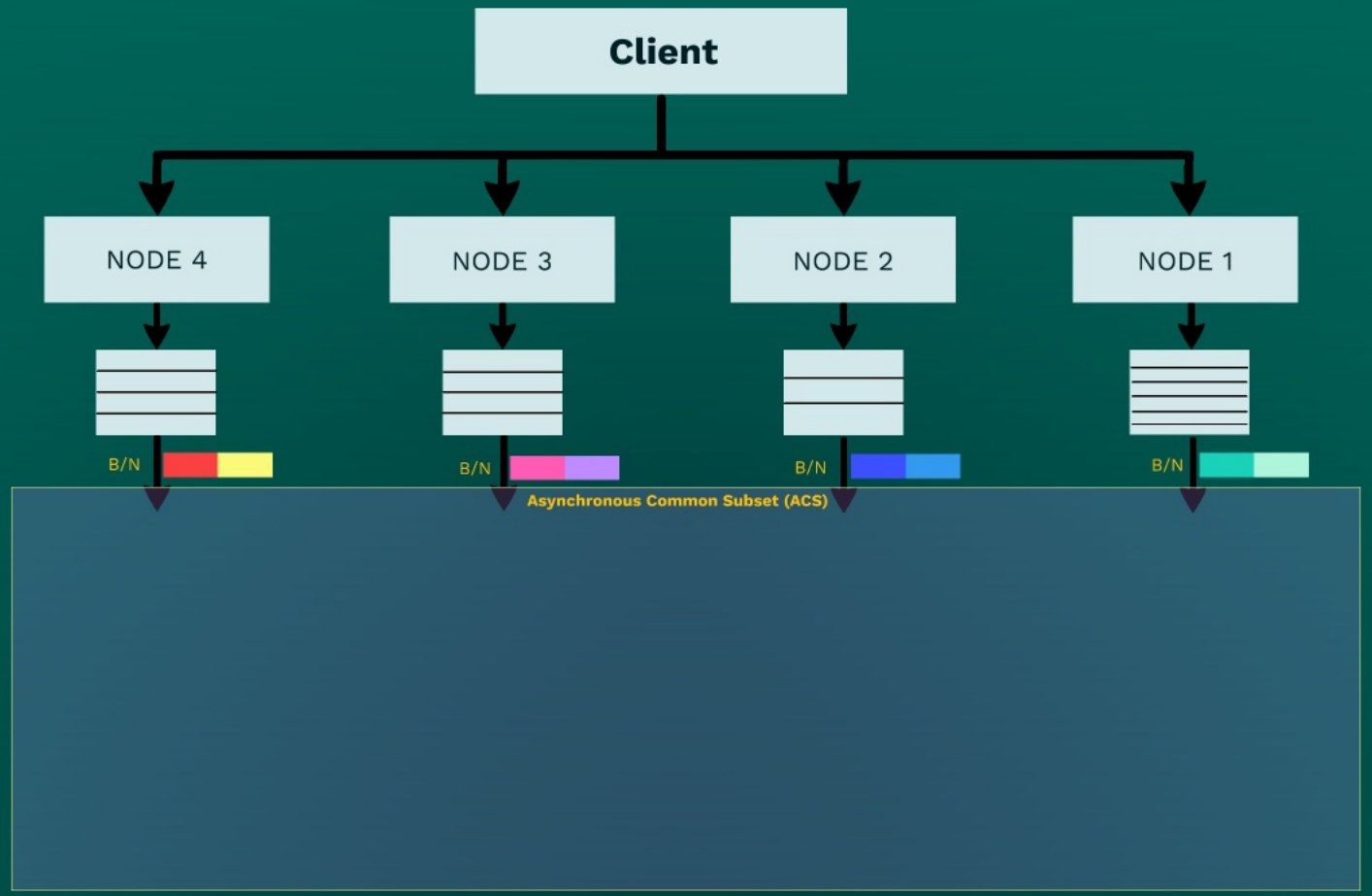
Overview



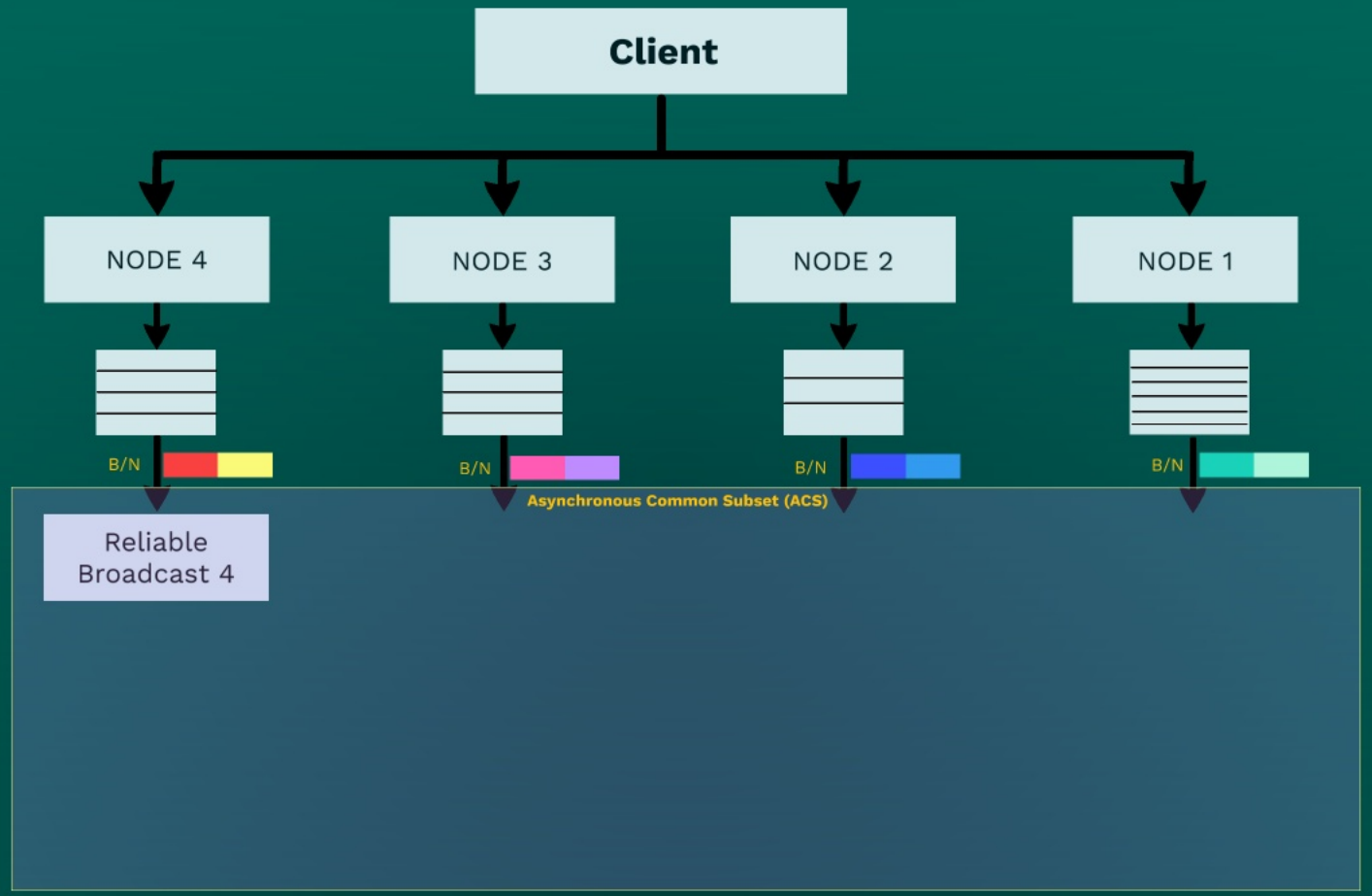
Overview



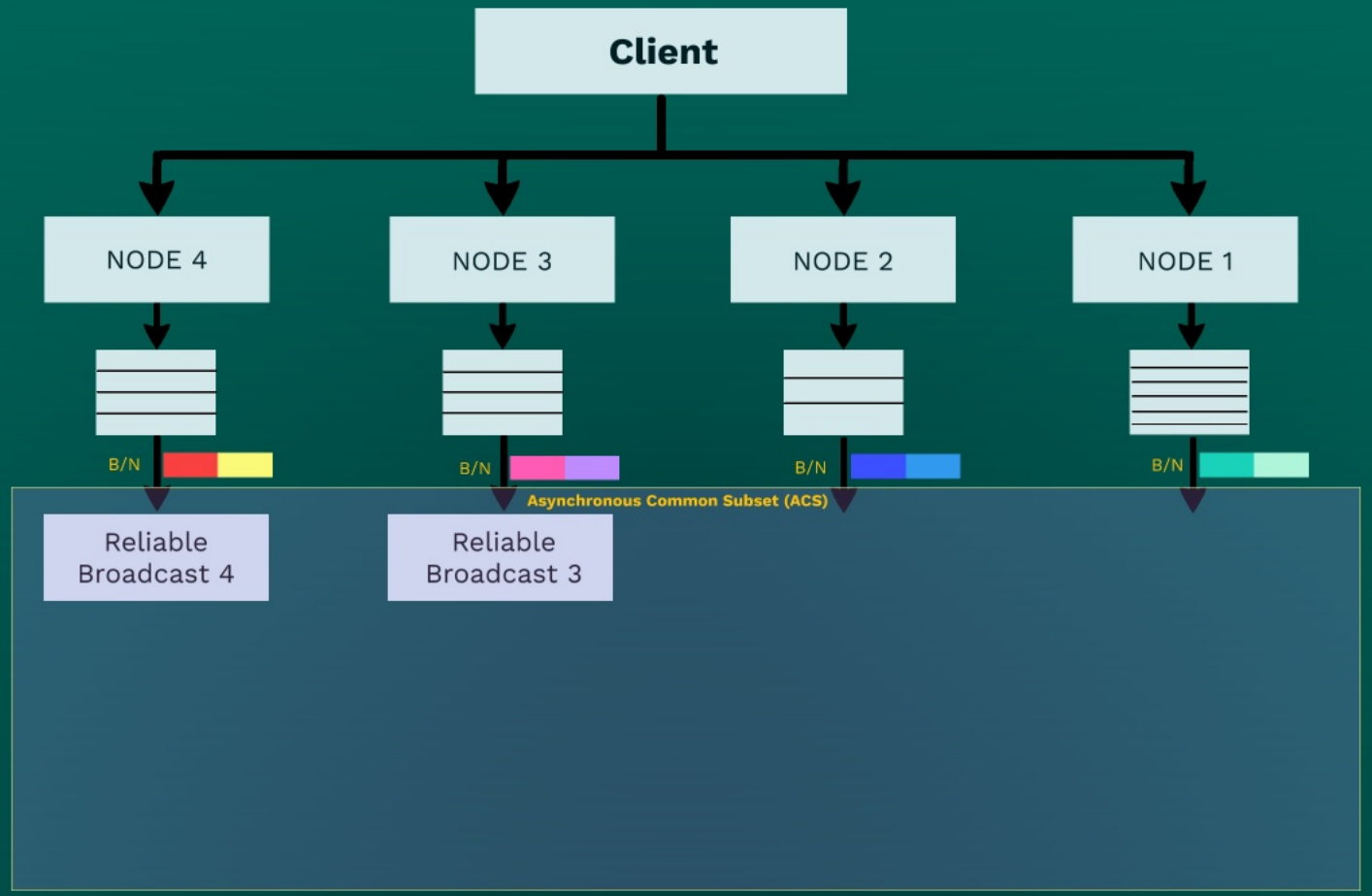
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)

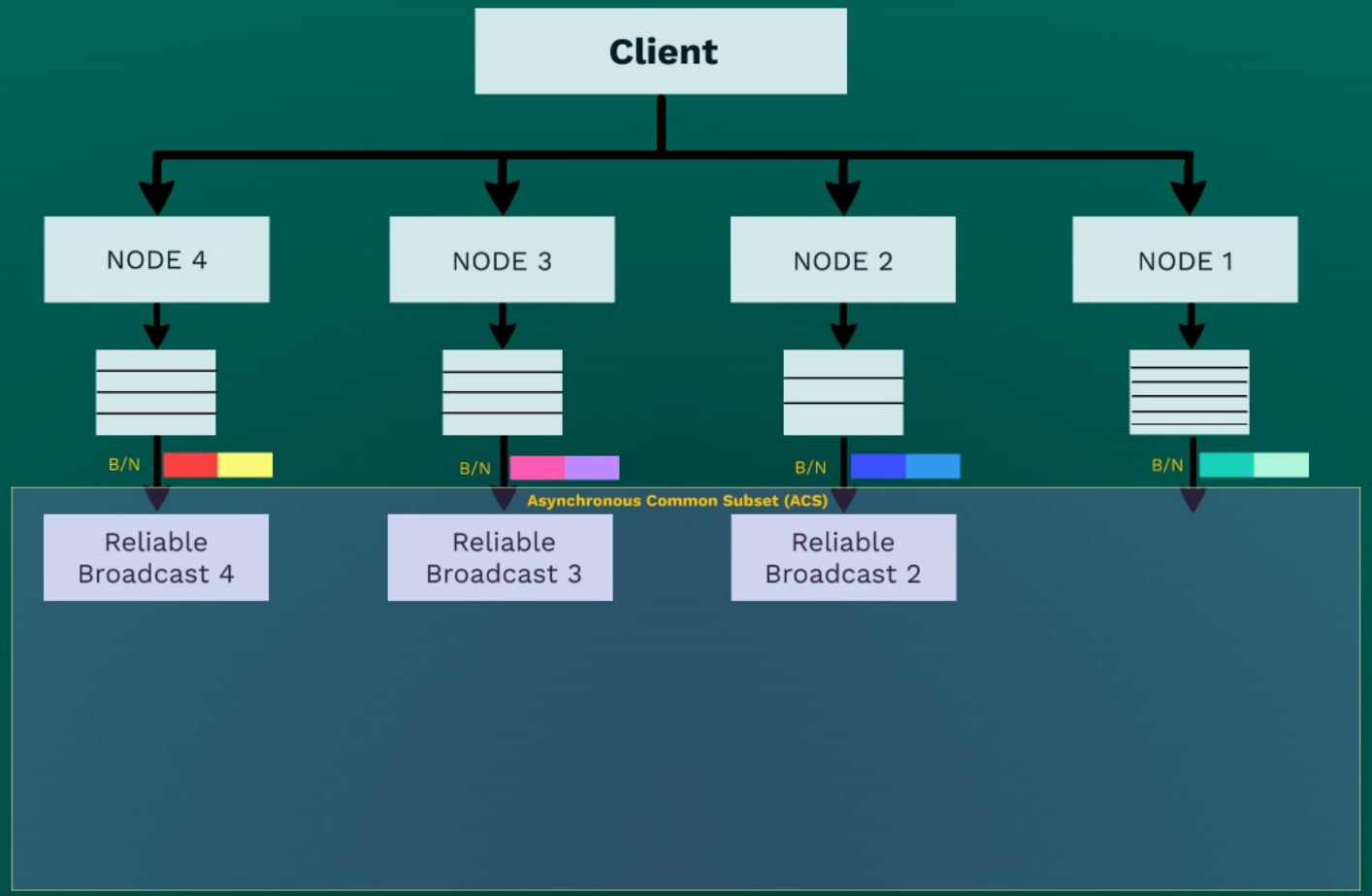
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4

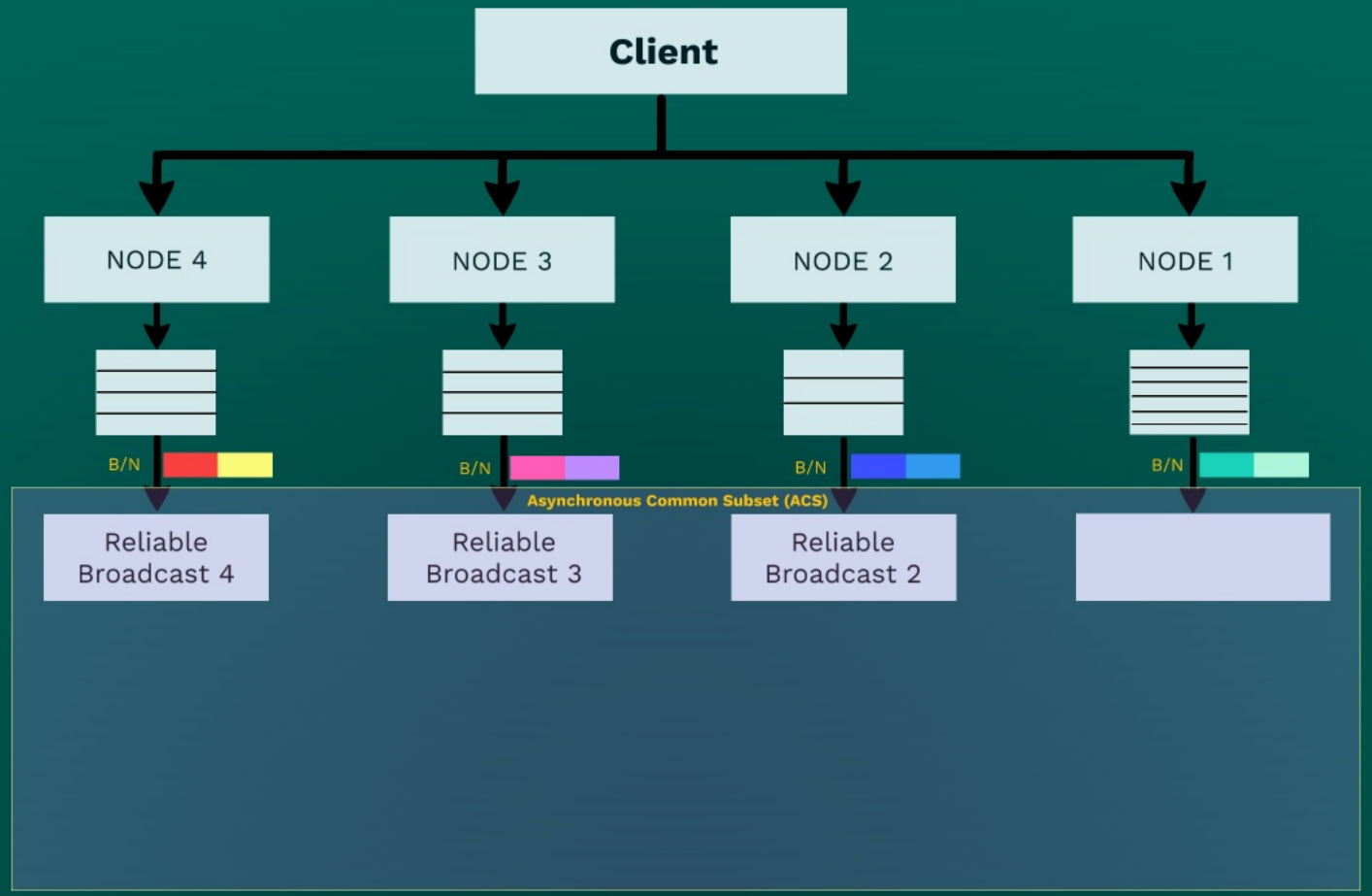
Overview

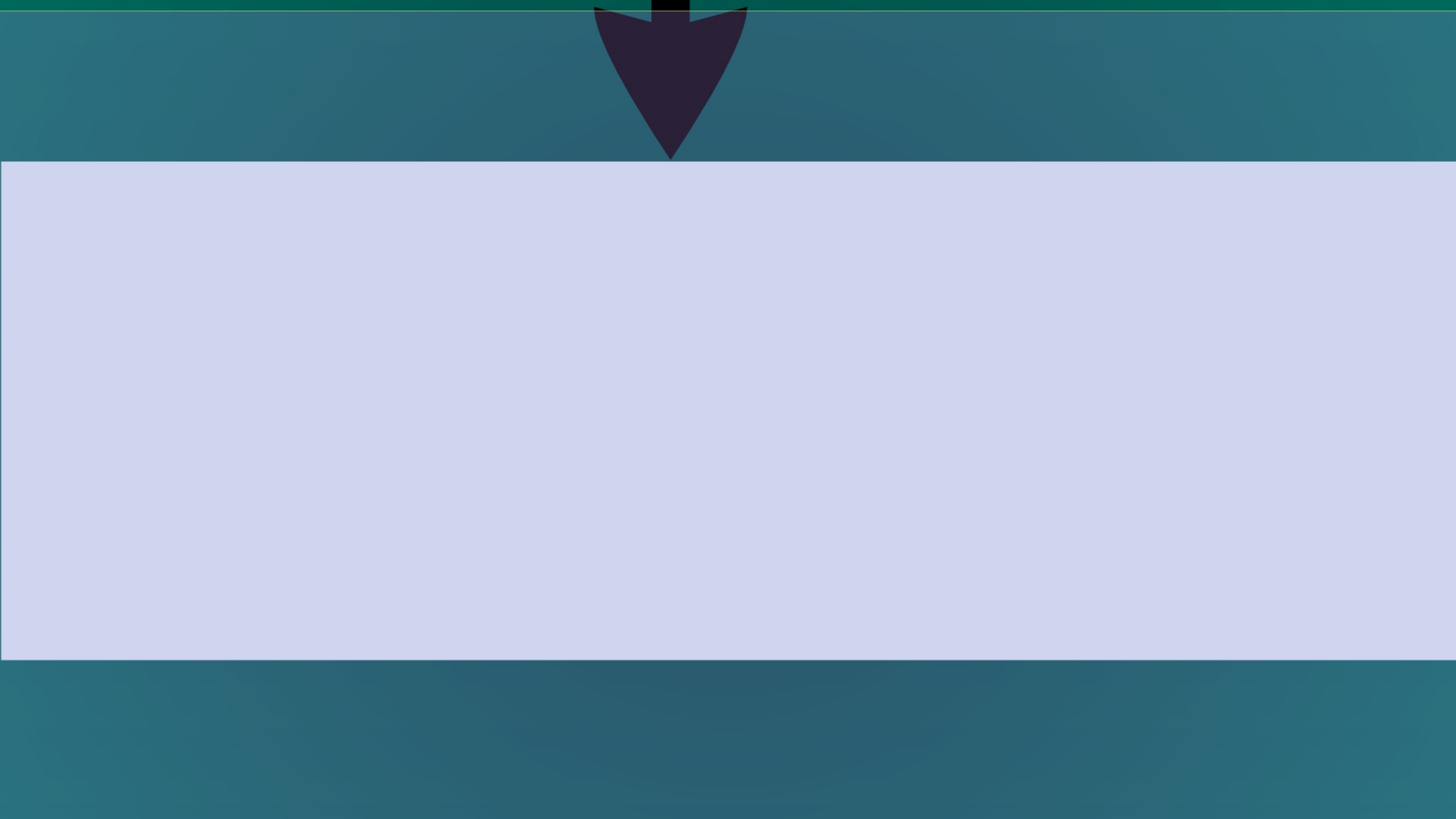

Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3

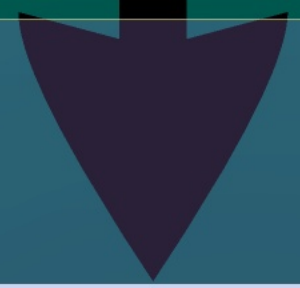
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2

Overview

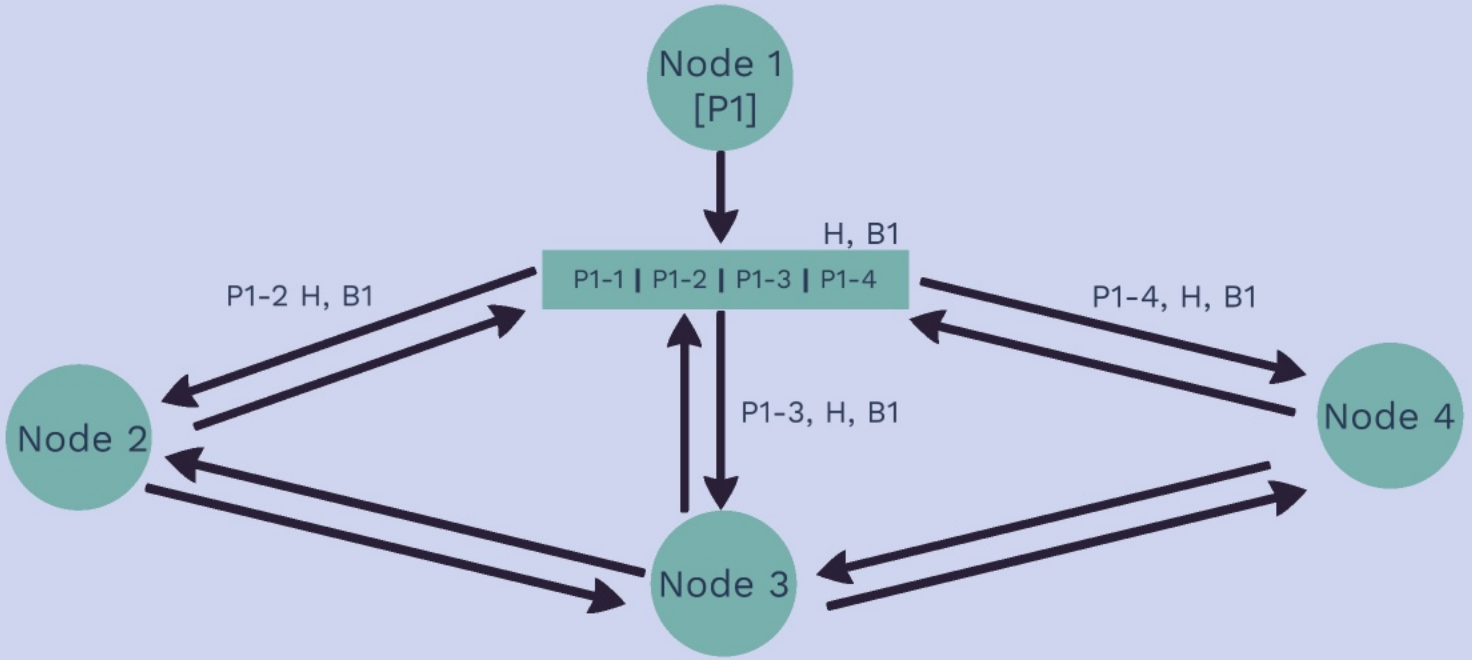

Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2



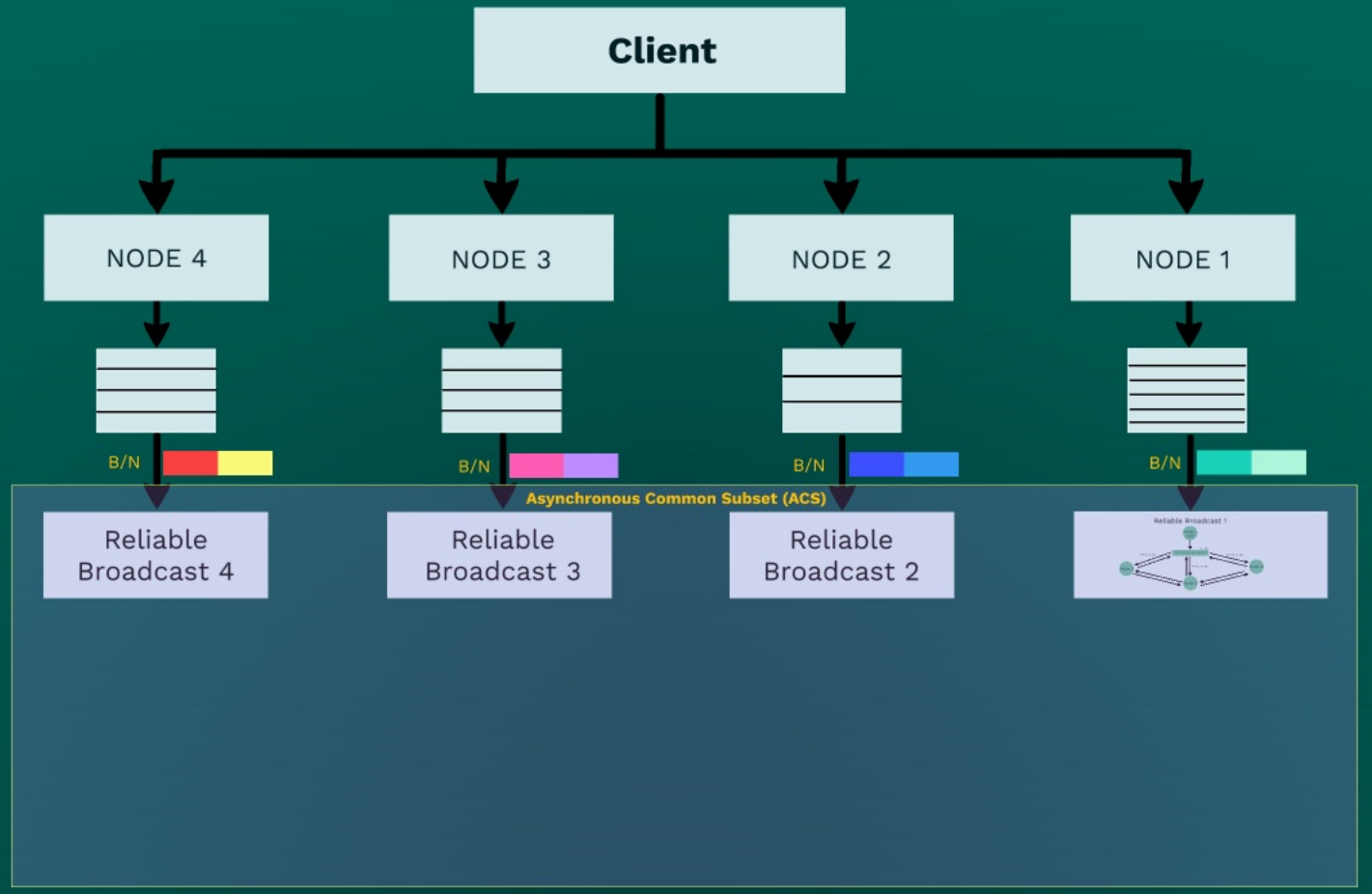


Reliable Broadcast 1

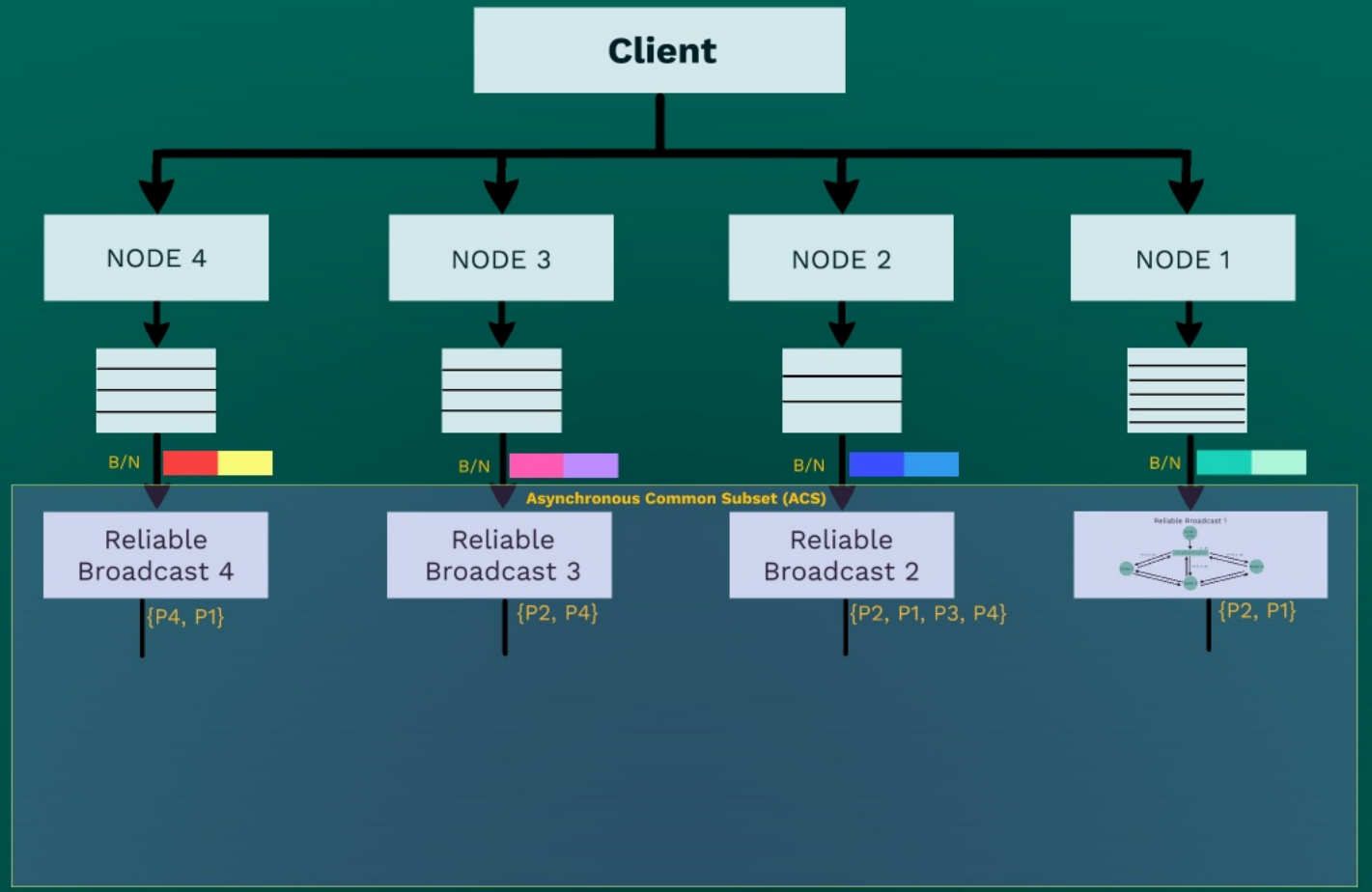
Reliable Broadcast 1



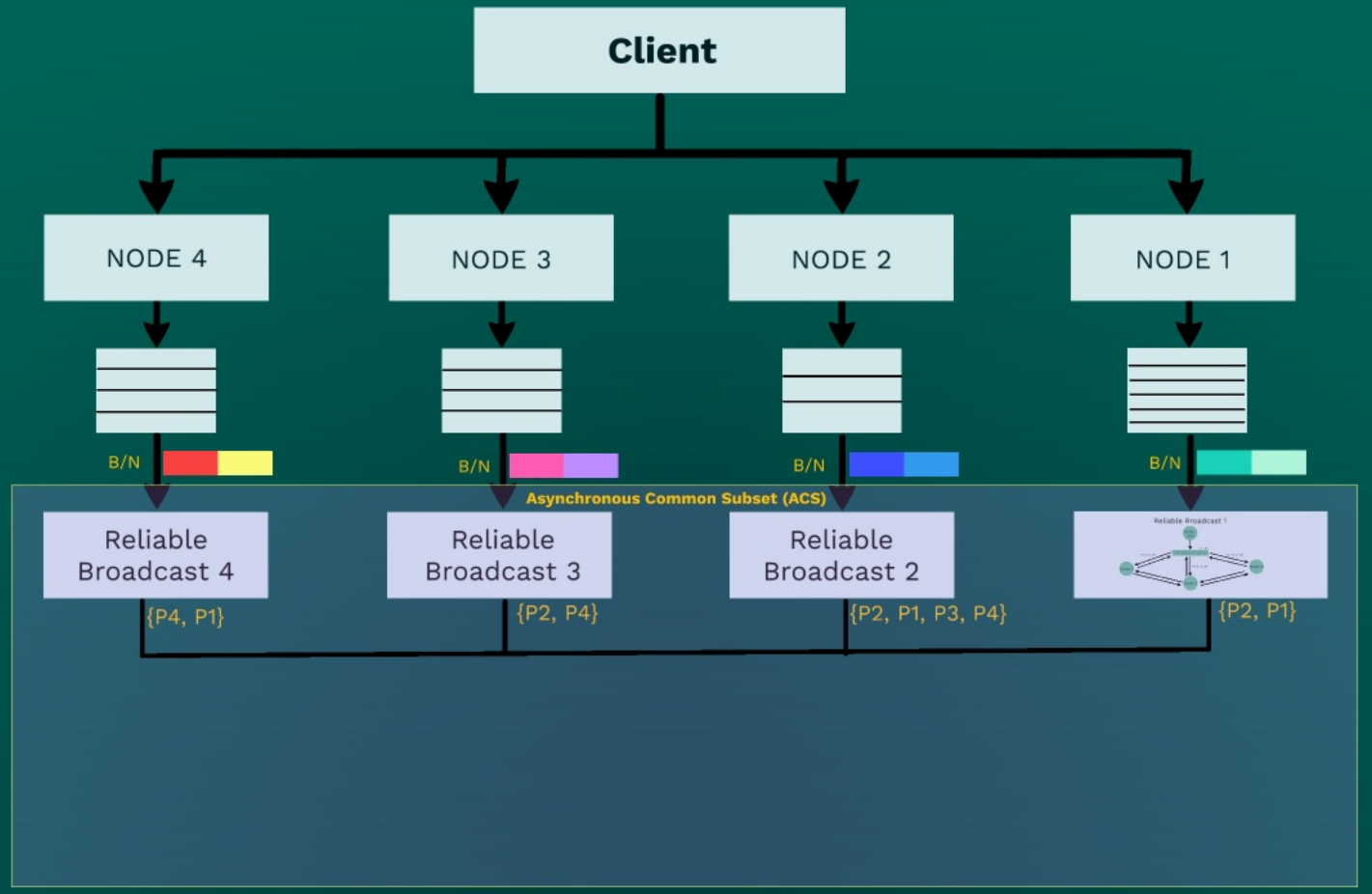
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2

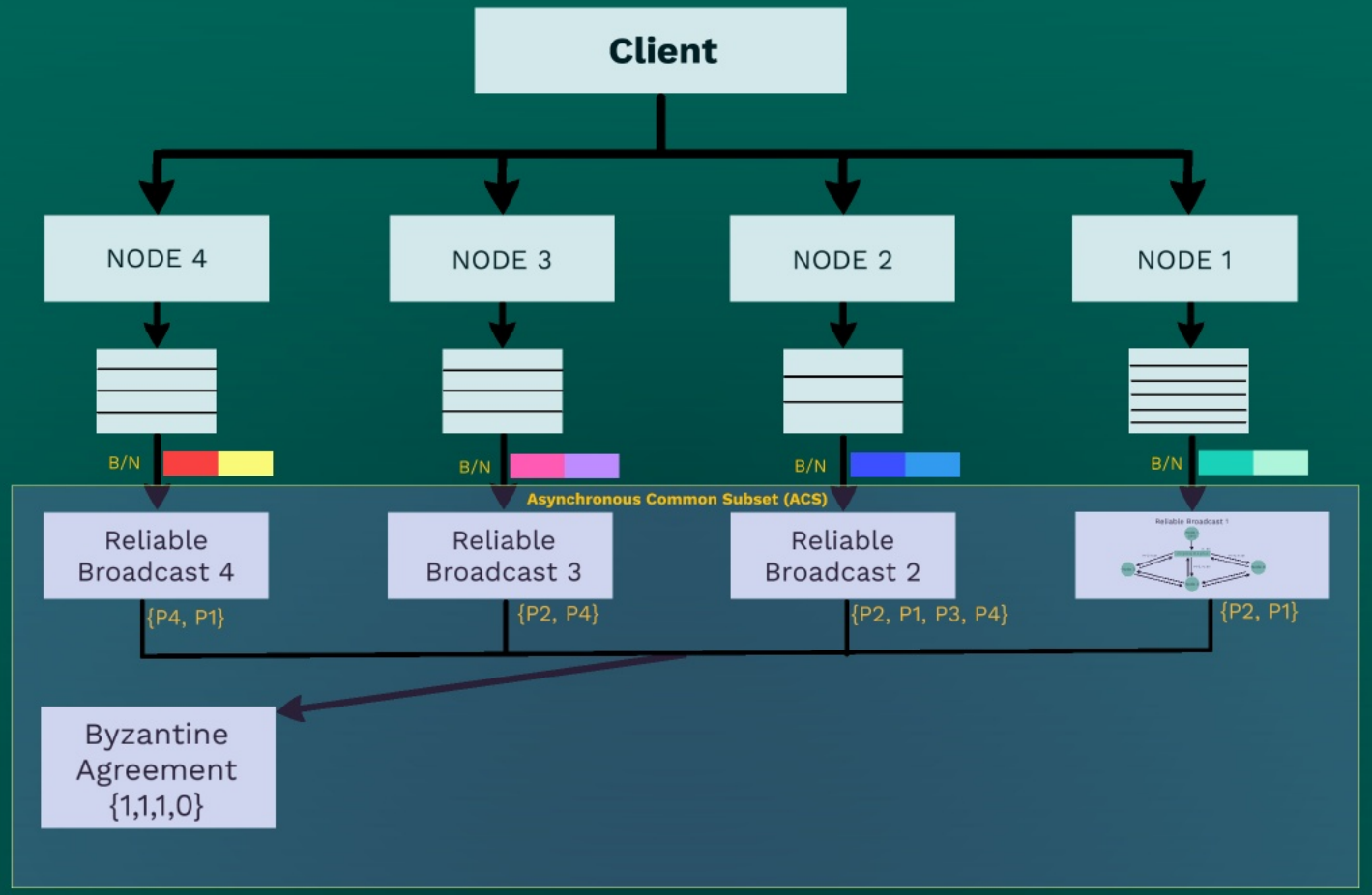
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}

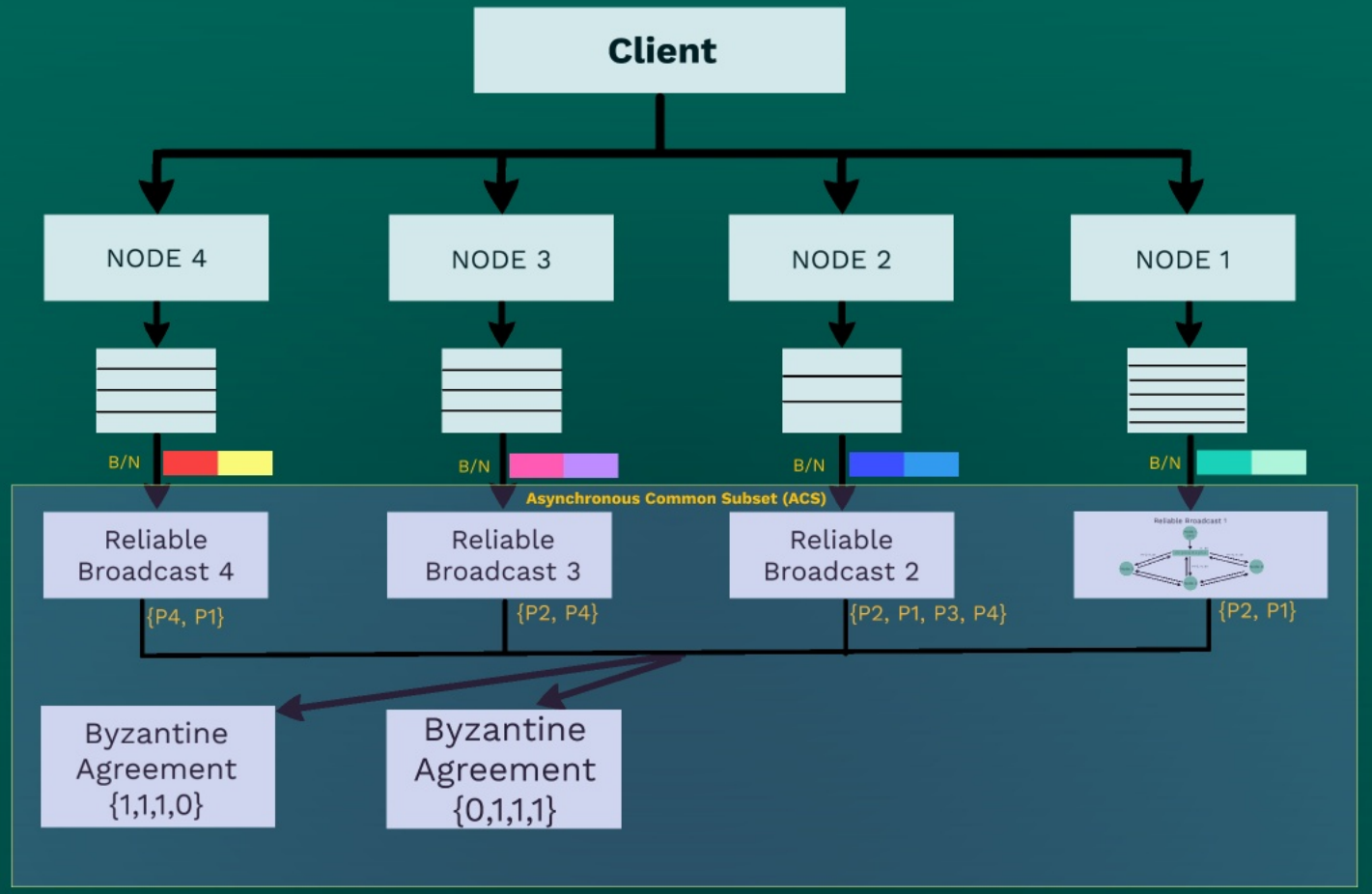
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}

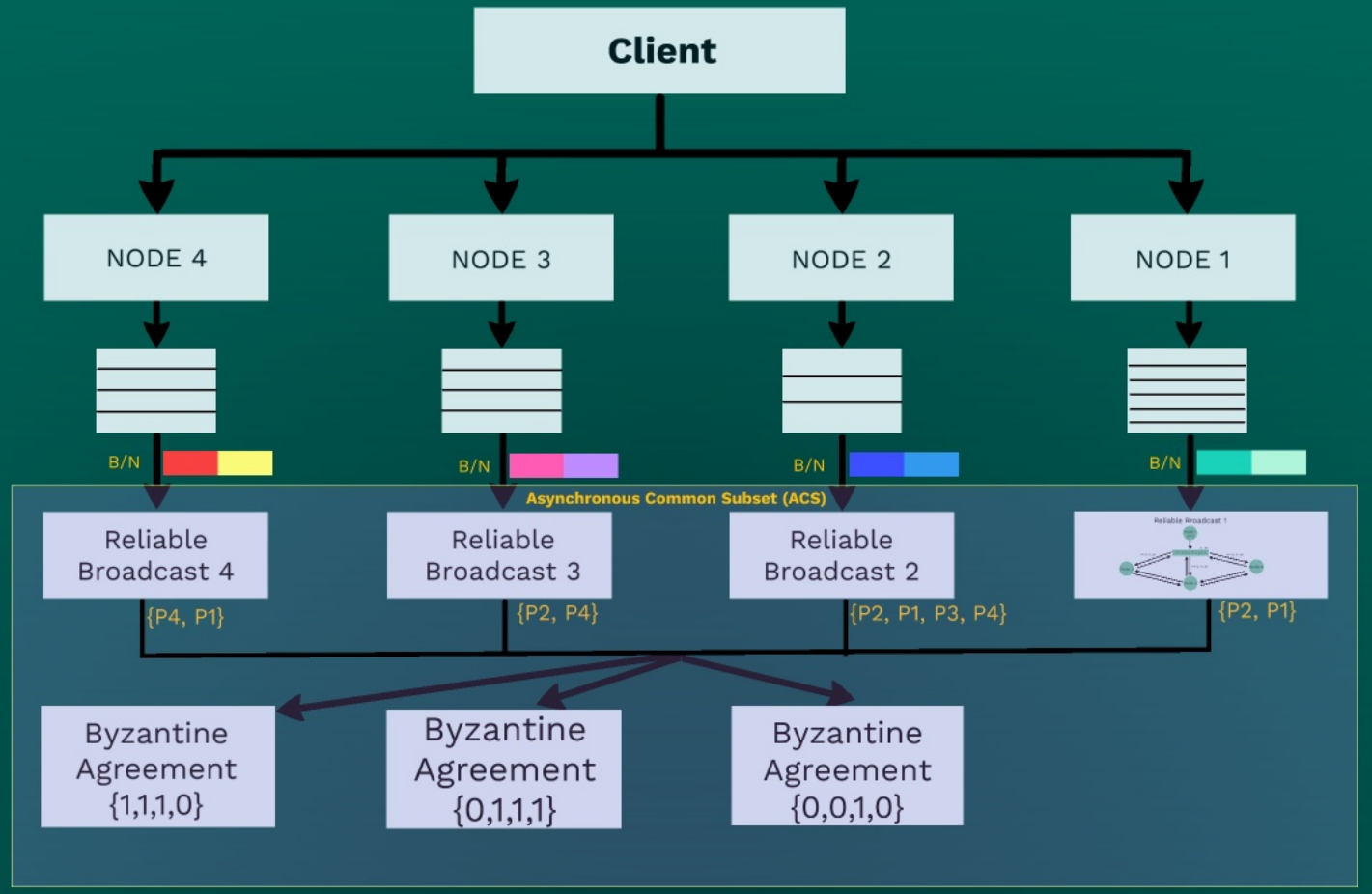
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}

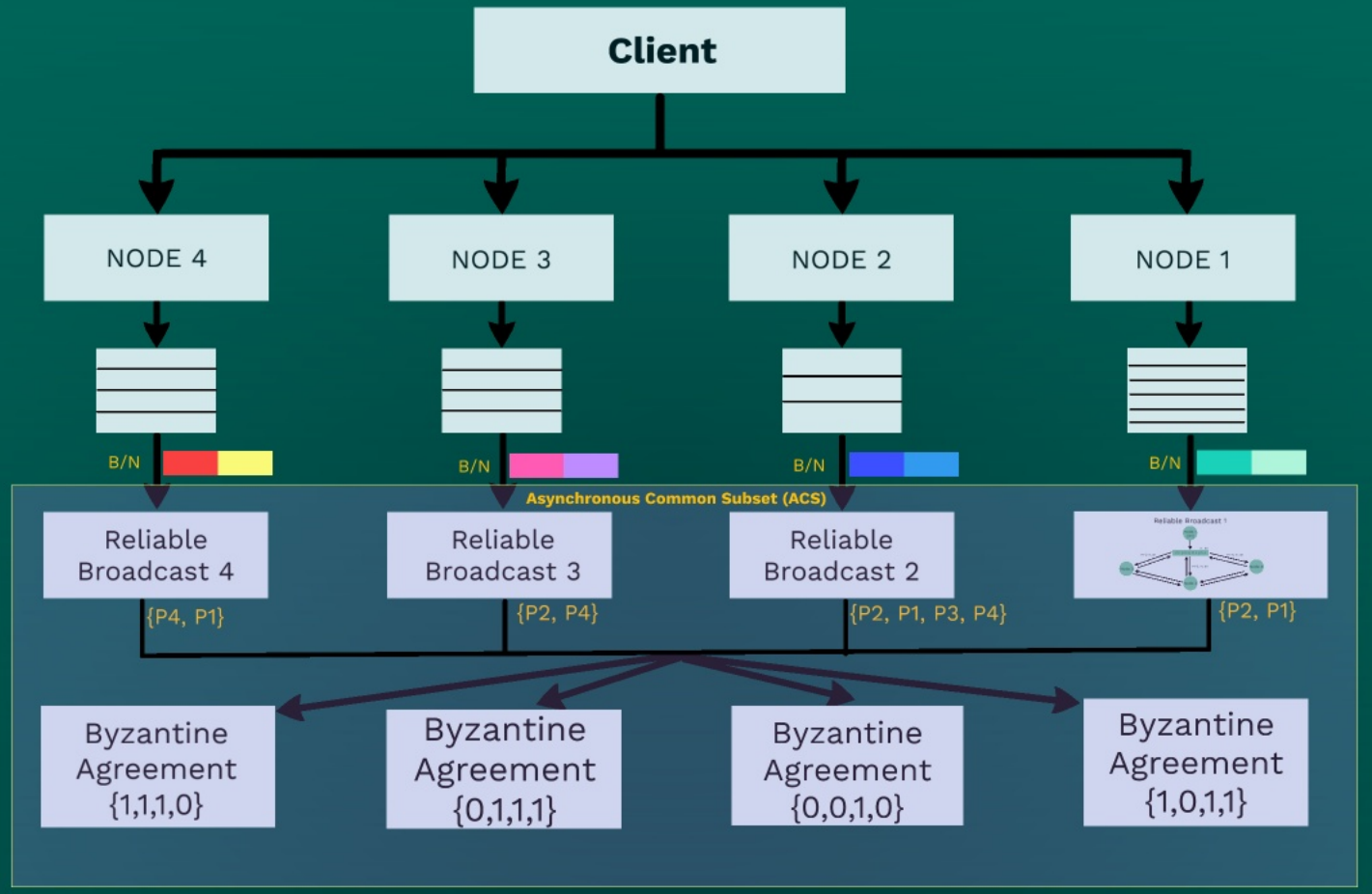
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}

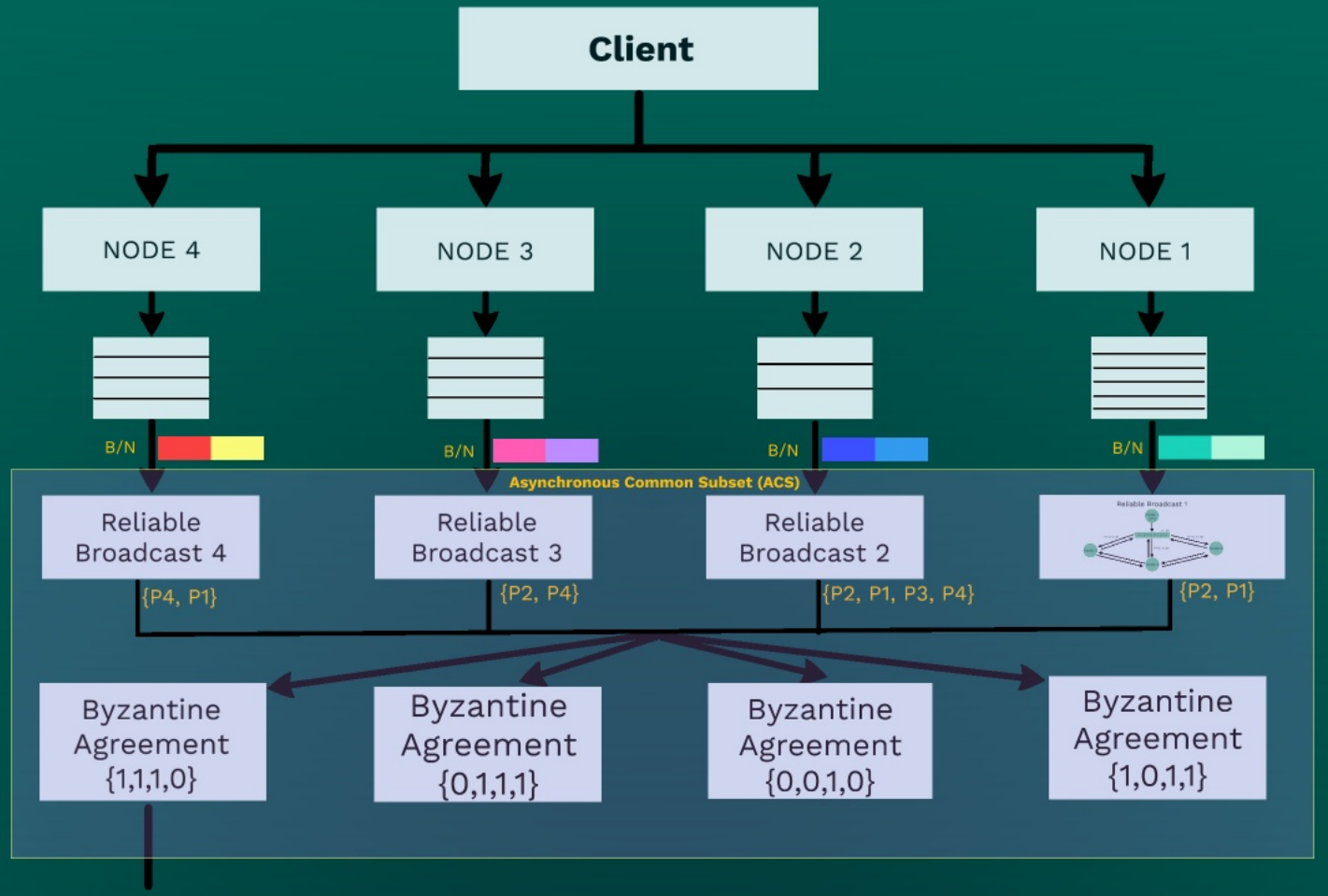
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}

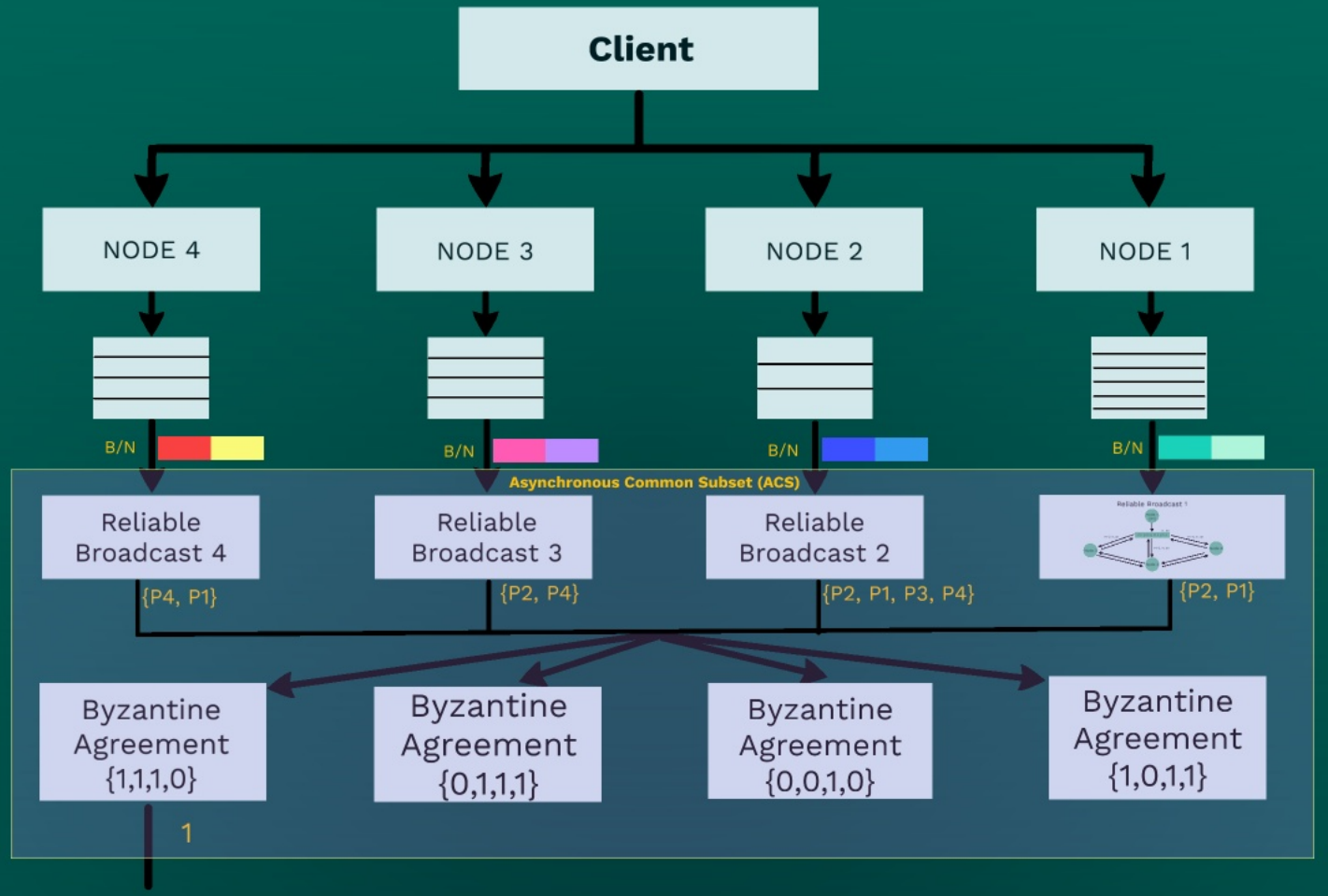
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}

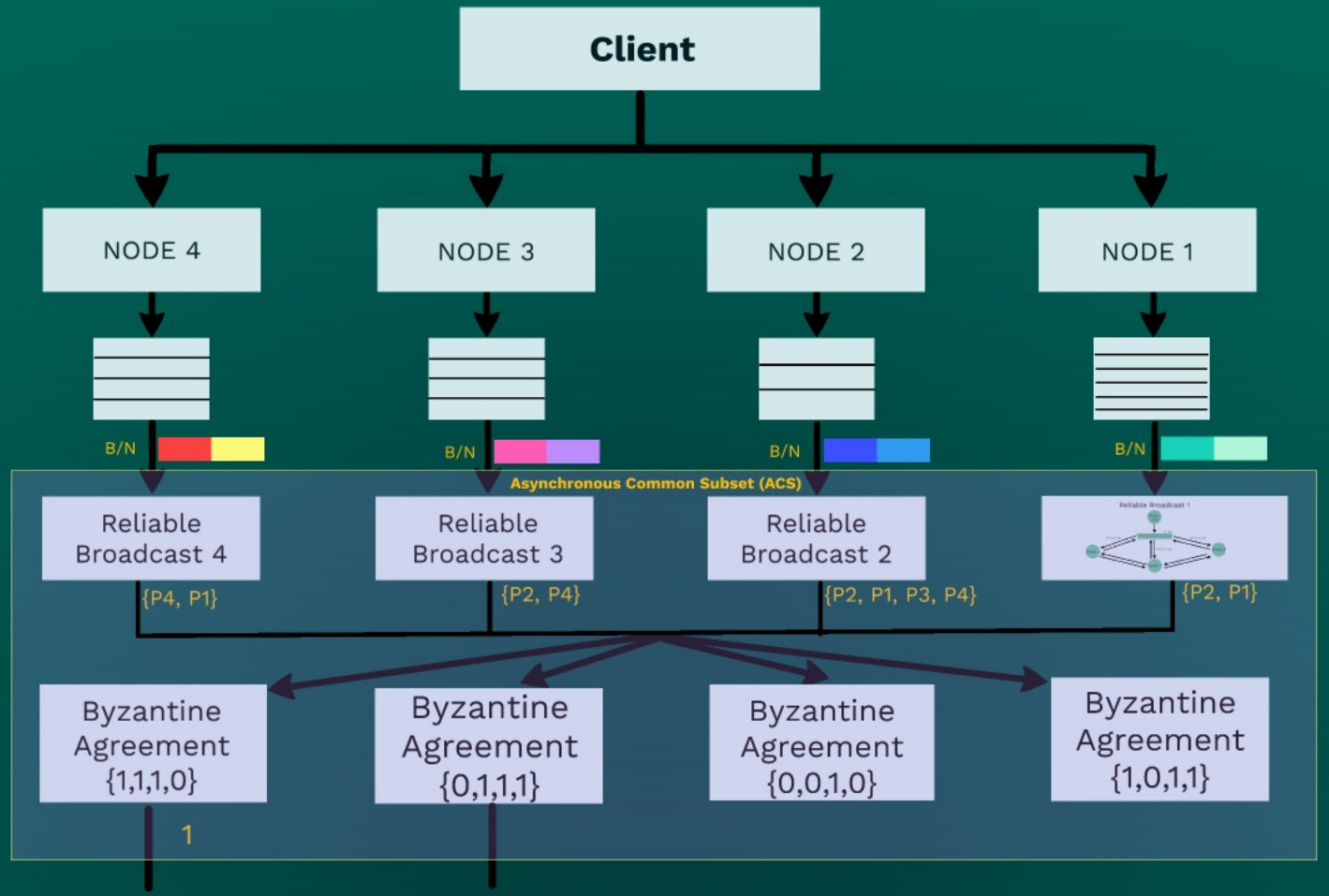
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}

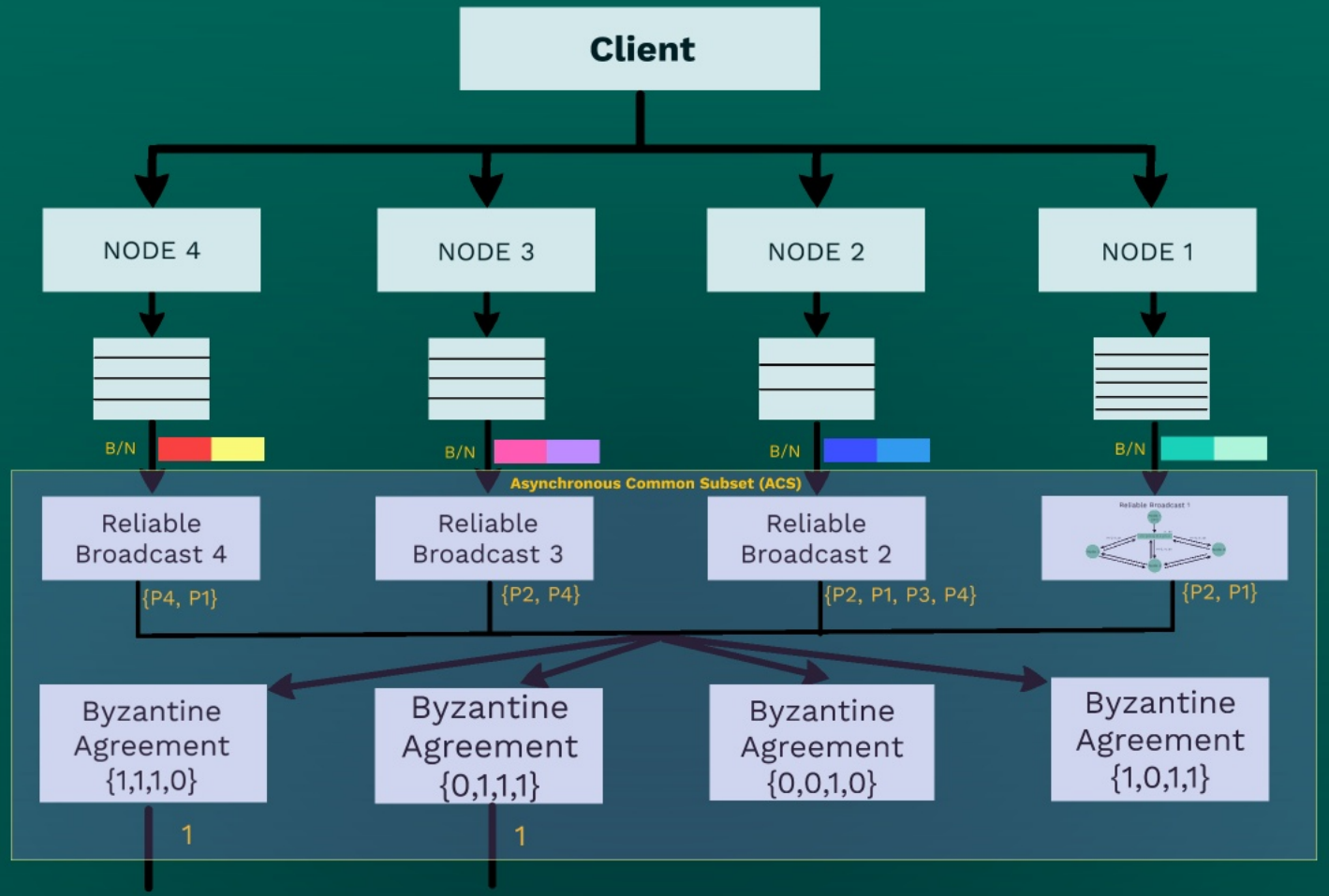
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1

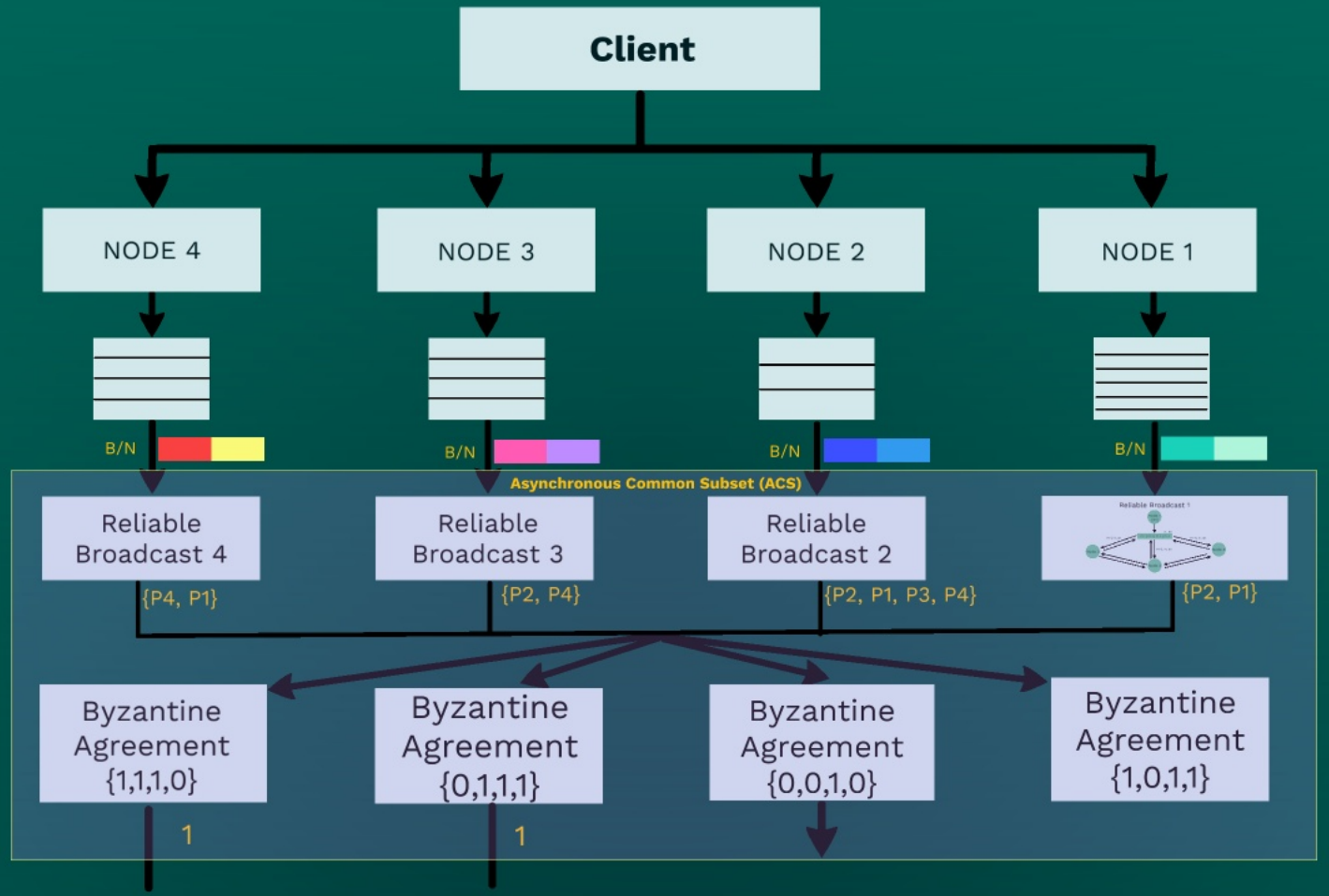
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1

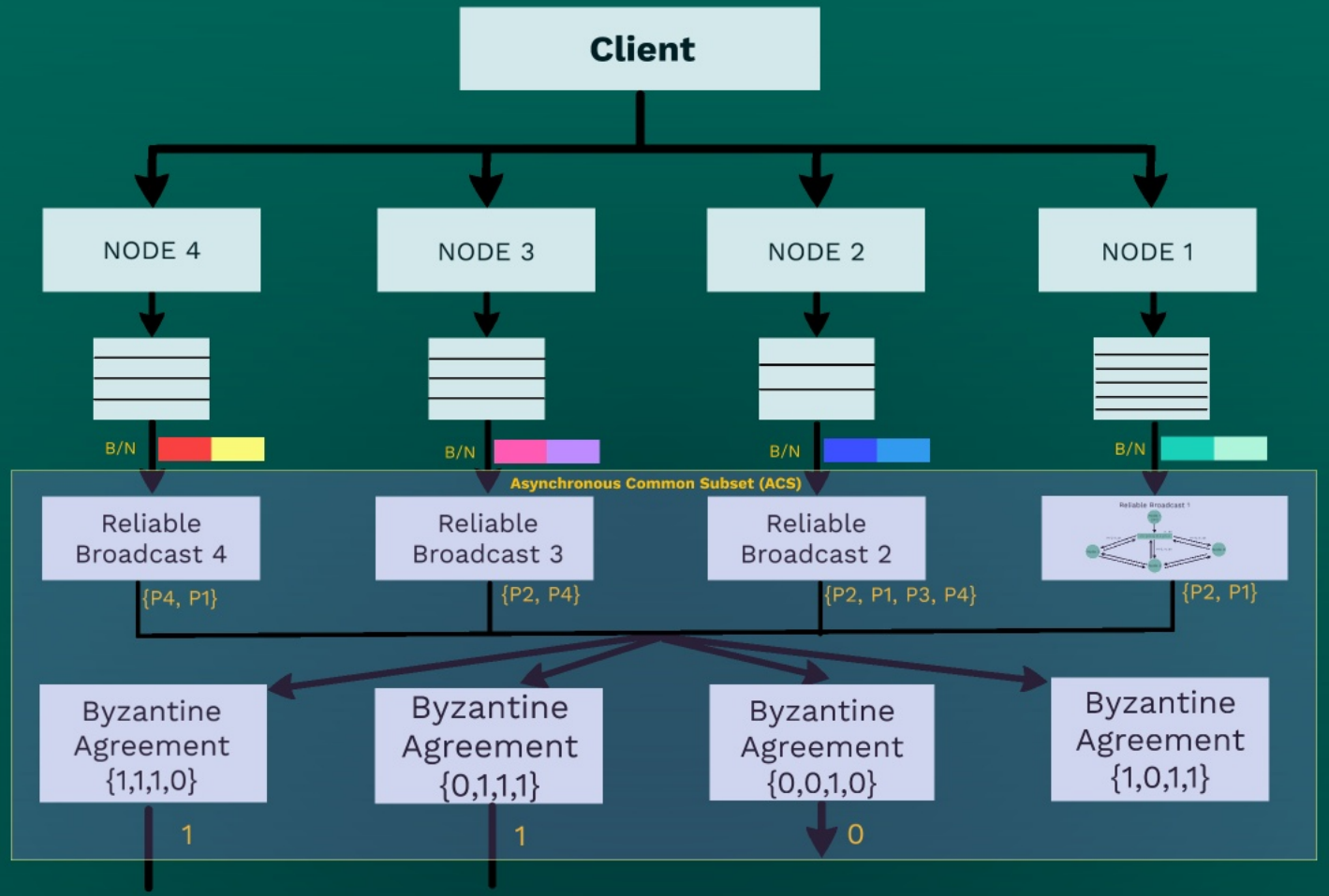
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1

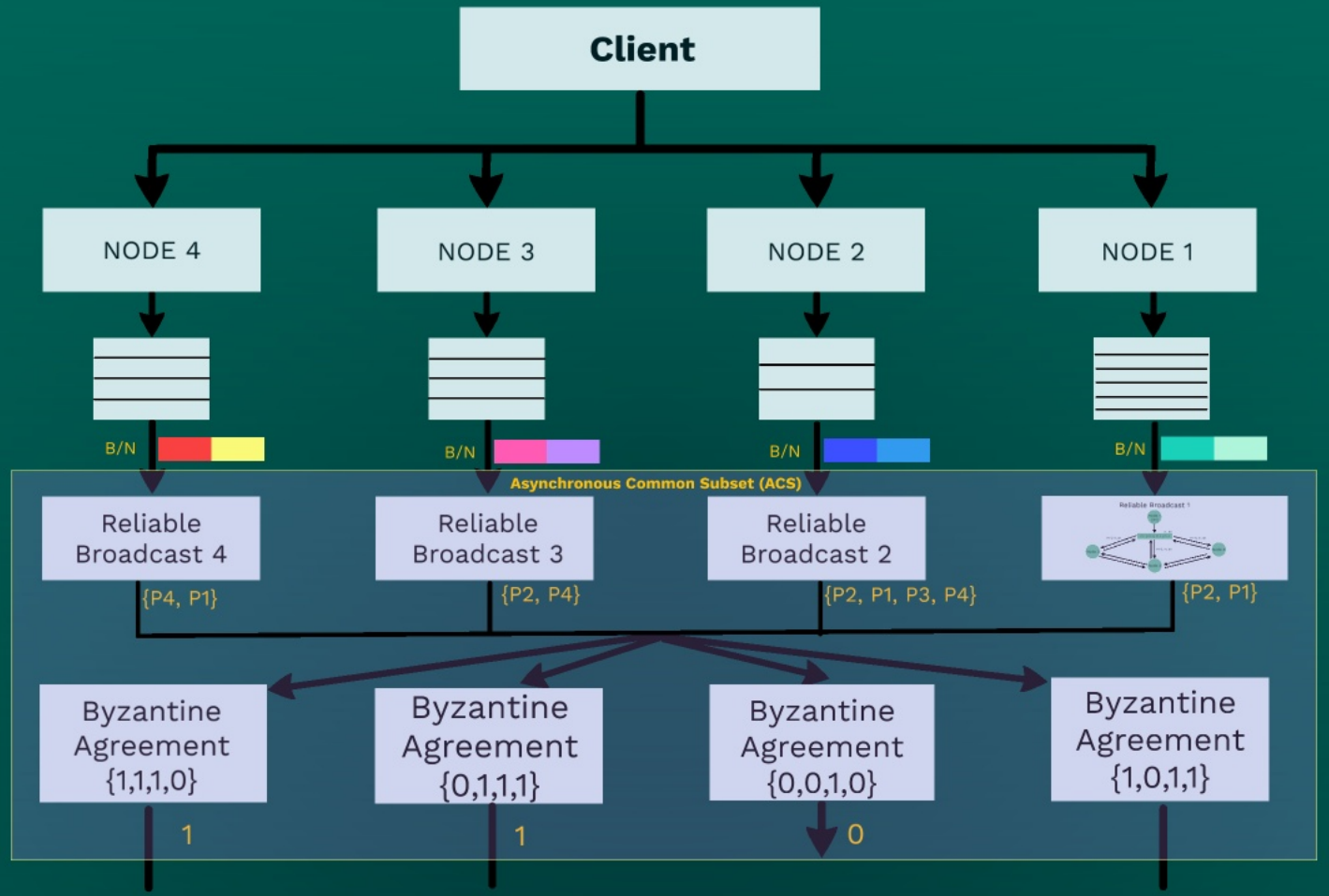
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1

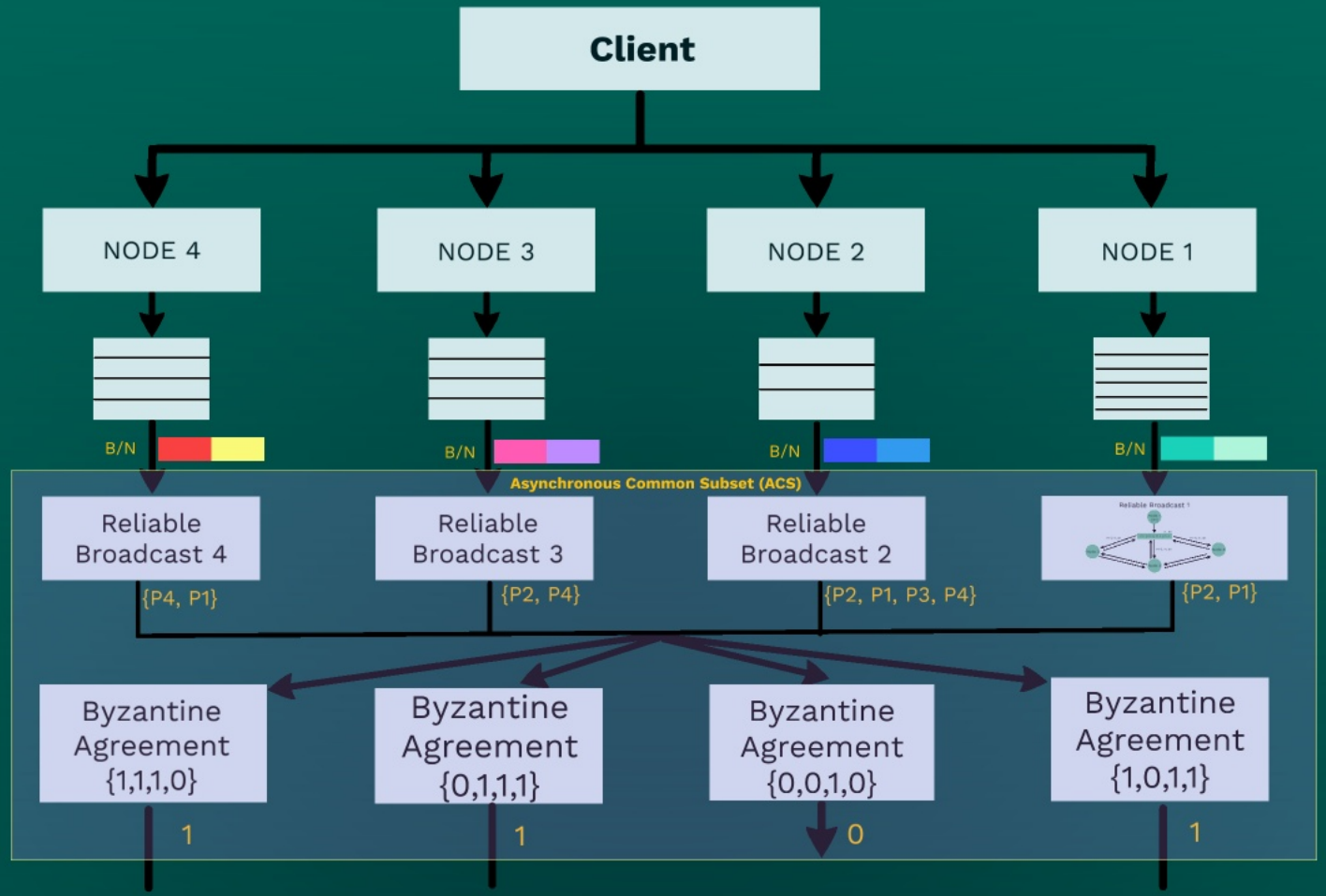
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1
0

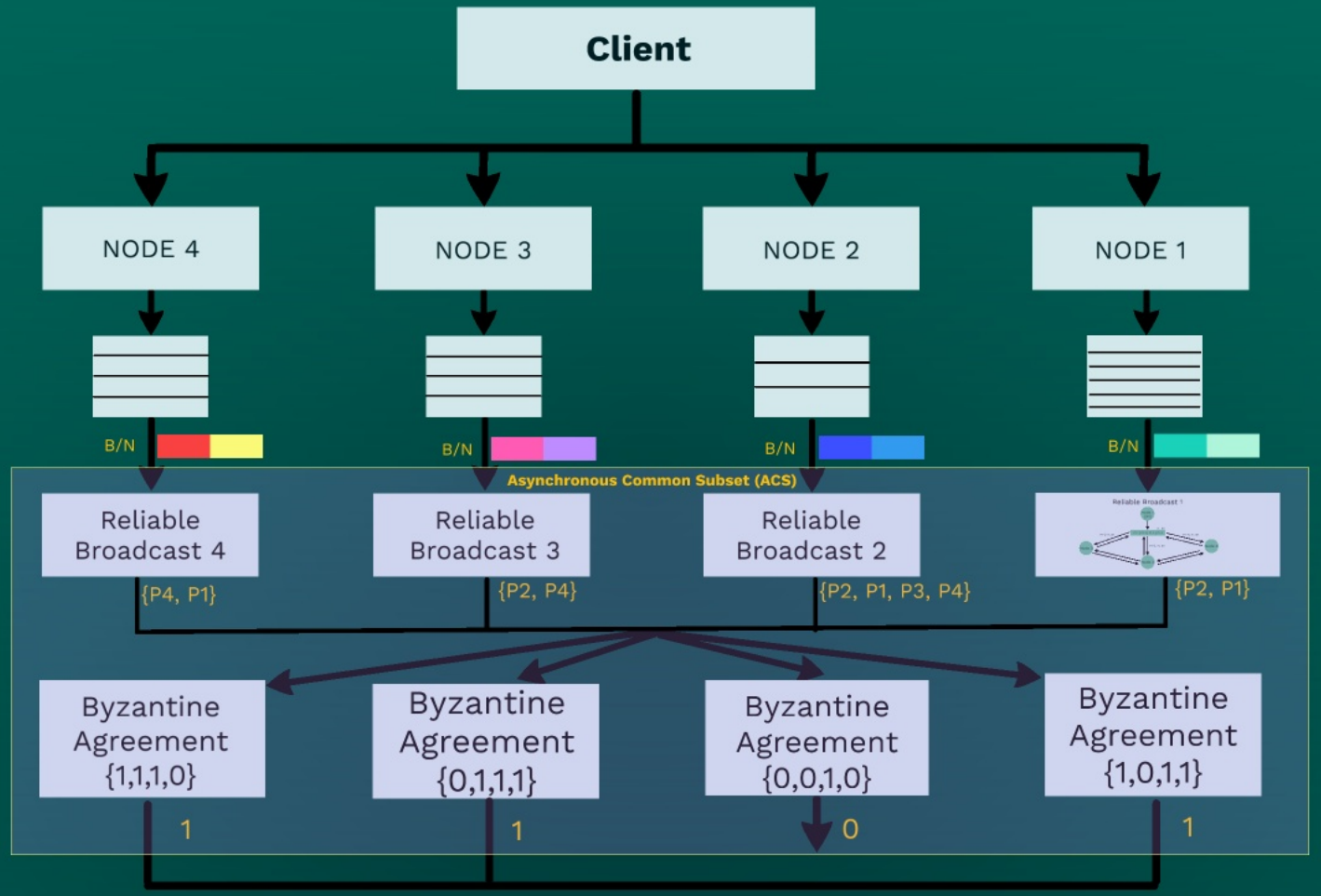
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1
0

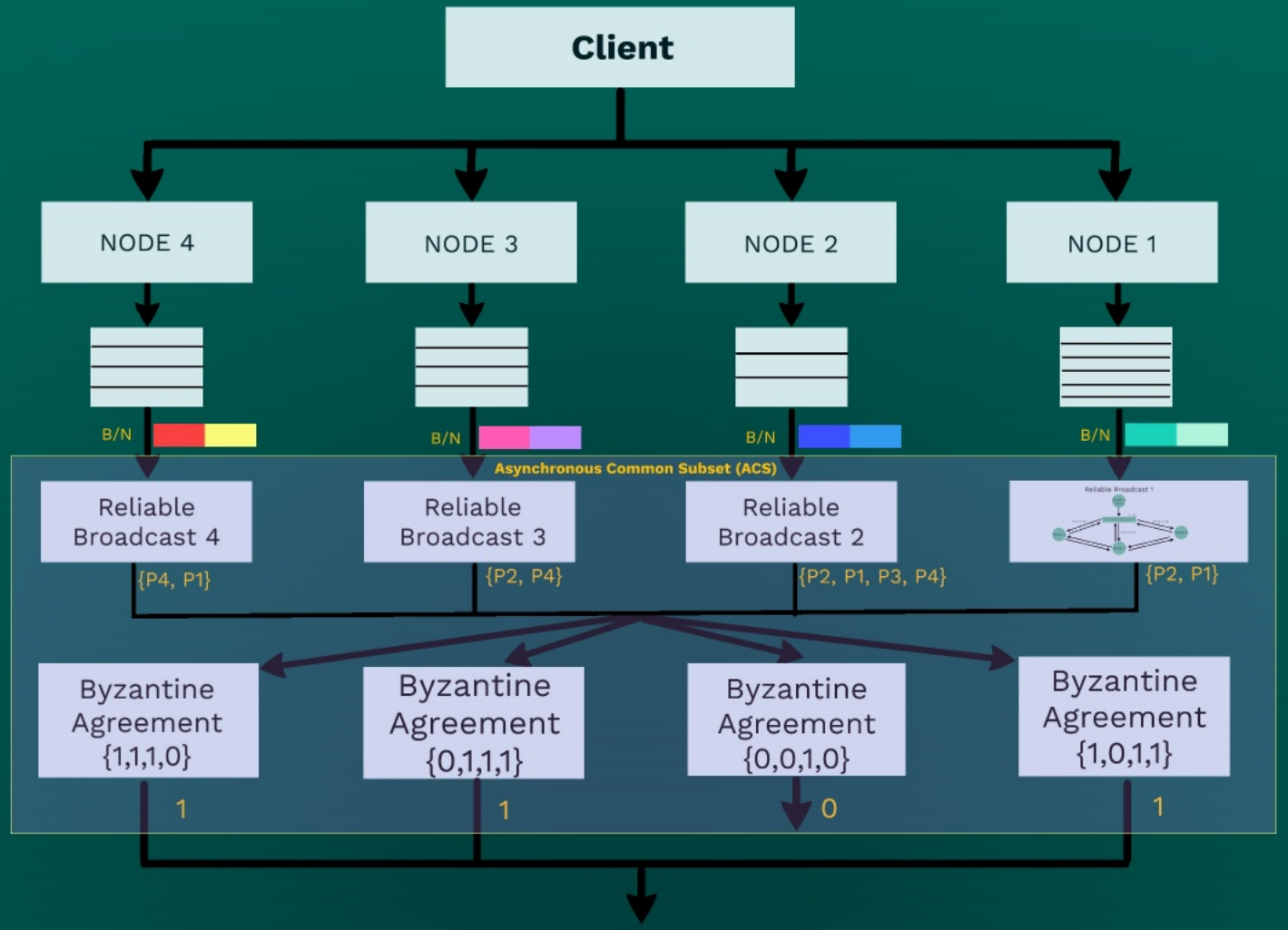
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1
0
1

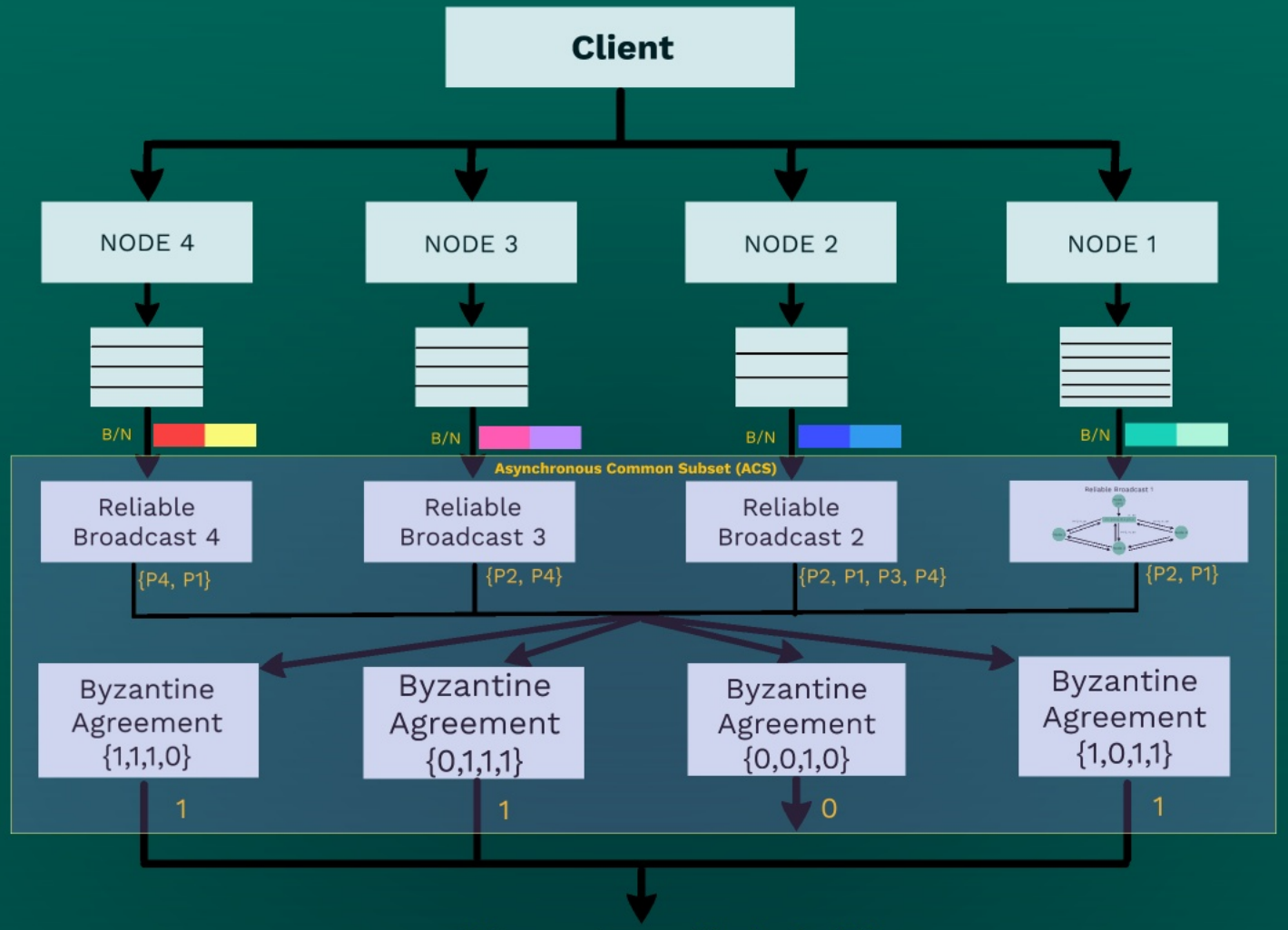
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1
0
1

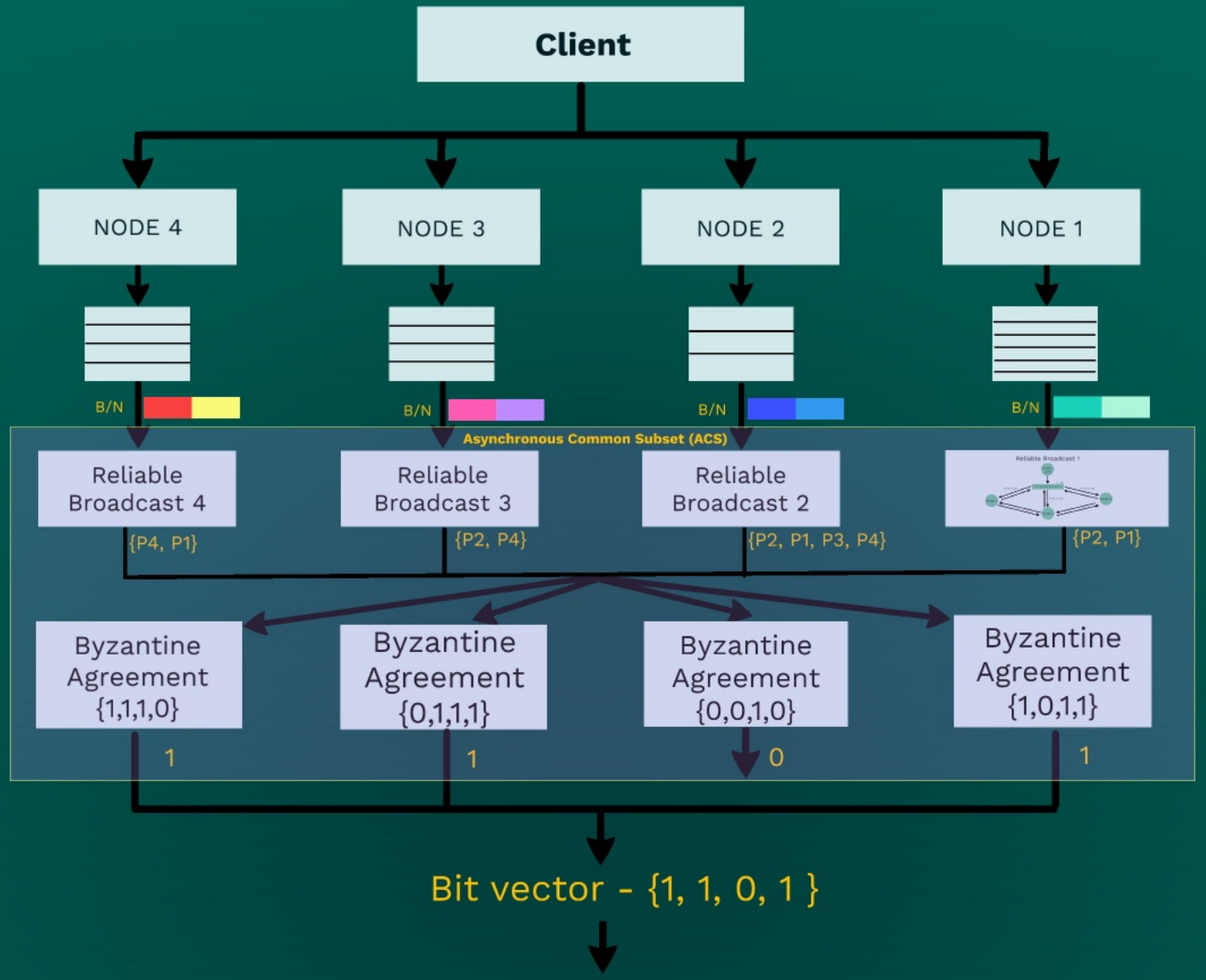
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1
0
1

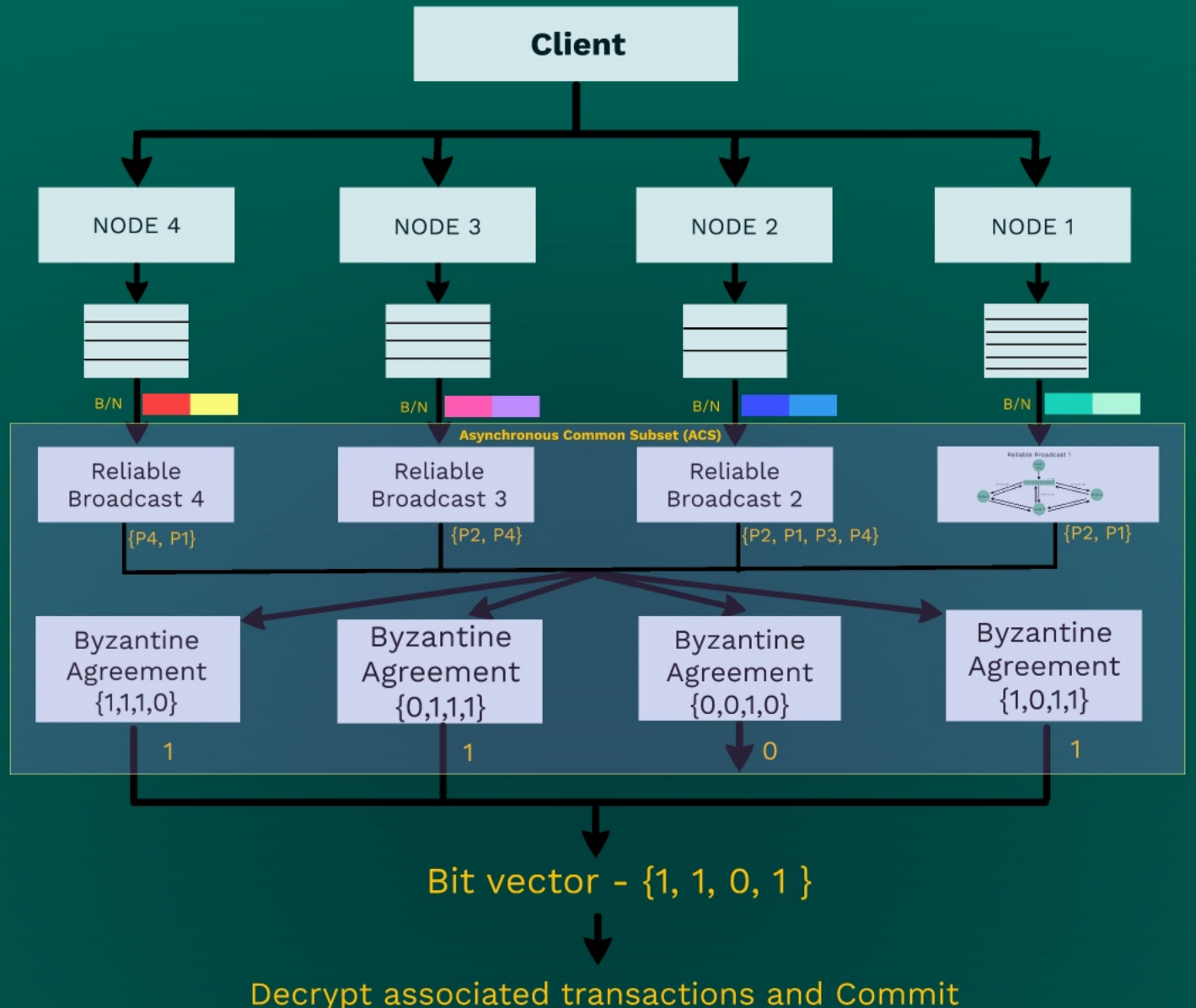
Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1
0
1
Bit vector - {1, 1, 0, 1 }

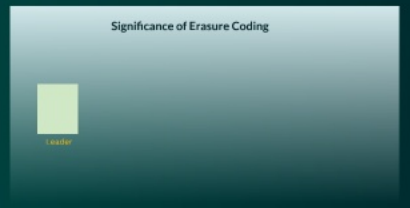
Overview

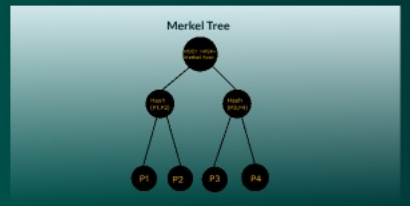

Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1
0
1
Bit vector - {1, 1, 0, 1 }

Overview


Client
NODE 4
NODE 3
NODE 2
NODE 1
B/N
B/N
B/N
B/N
Asynchronous Common Subset (ACS)
Reliable Broadcast 4
Reliable Broadcast 3
Reliable Broadcast 2
{P4, P1}
{P2, P4}
{P2, P1, P3, P4}
{P2, P1}
Byzantine Agreement {1,1,1,0}
Byzantine Agreement {0,1,1,1}
Byzantine Agreement {0,0,1,0}
Byzantine Agreement {1,0,1,1}
1
1
0
1
Bit vector - {1, 1, 0, 1 }
Decrypt associated transactions and Commit

RBC Optimisation Techniques-Erasure Coding, Merkle Trees





Standard Reliable Broadcast

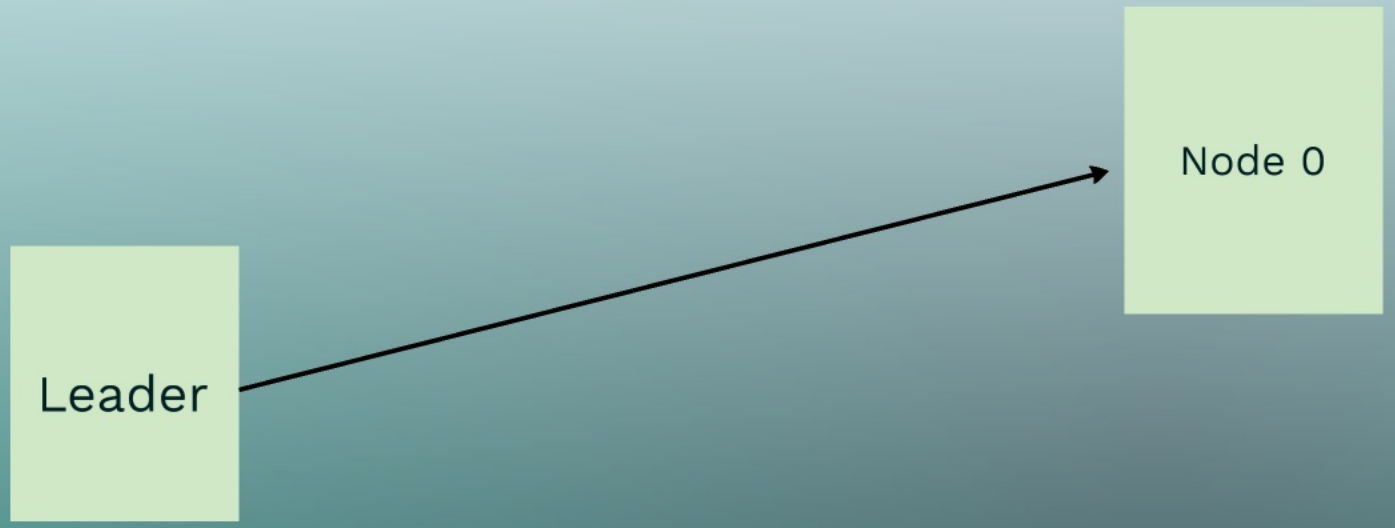
Leader's Bandwidth: O(N*B)

Standard Reliable Broadcast

Leader

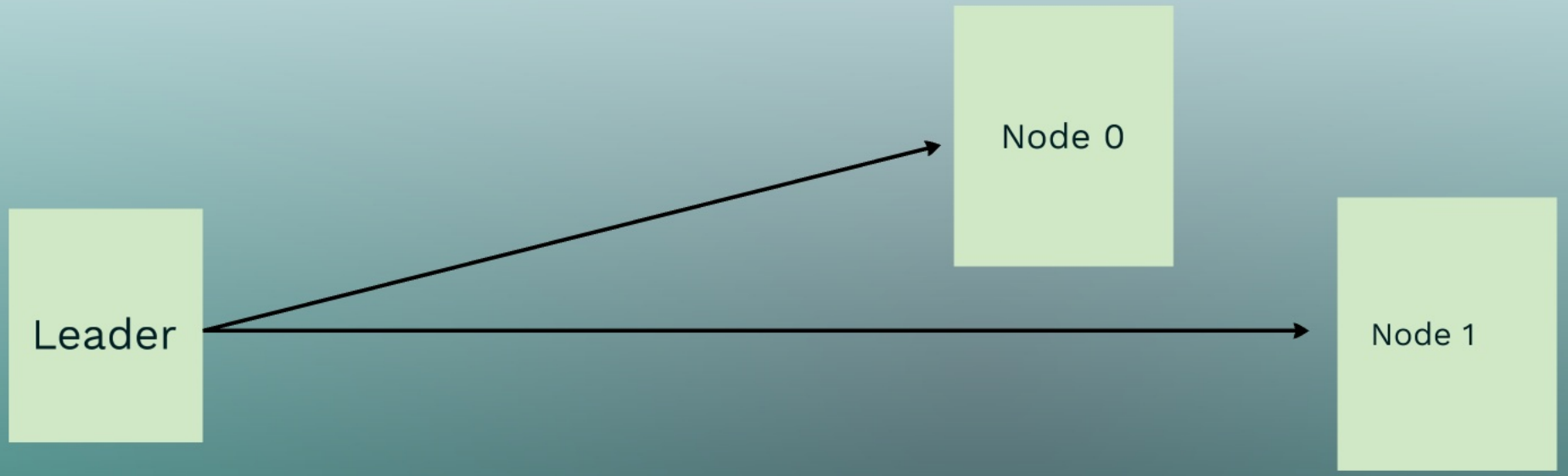
Leader's Bandwidth: O(N*B)

Standard Reliable Broadcast



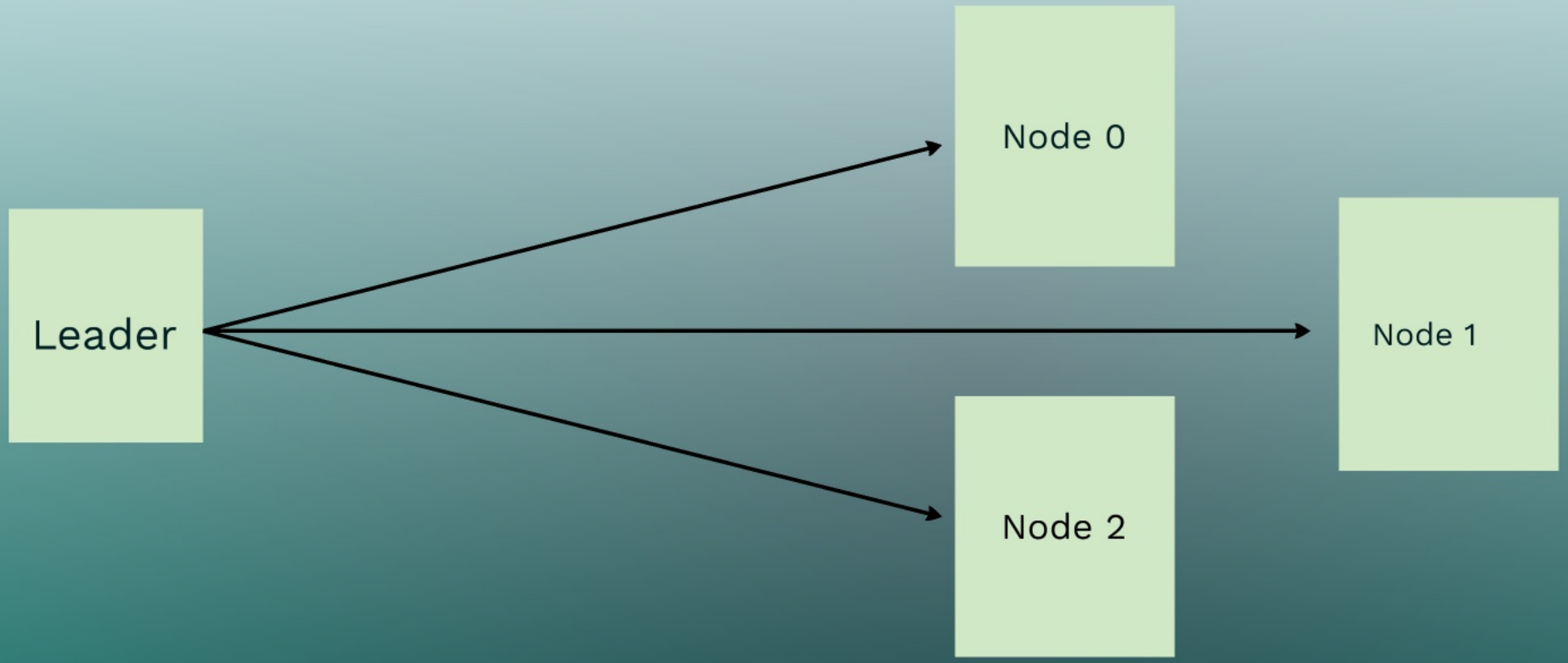
Leader's Bandwidth: O(N*B)

Standard Reliable Broadcast



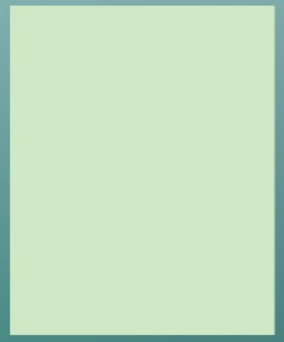
Leader's Bandwidth: O(N*B)

Standard Reliable Broadcast



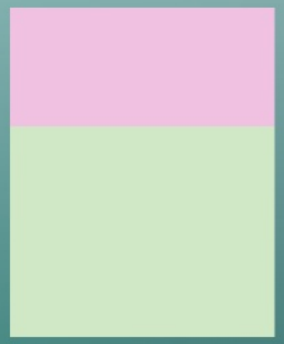
Leader's Bandwidth: O(N*B)

Significance of Erasure Coding



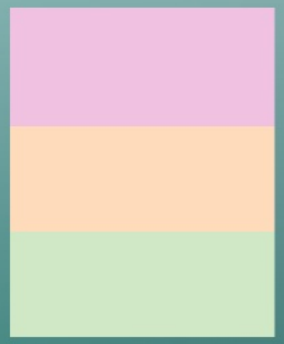
Leader

Significance of Erasure Coding



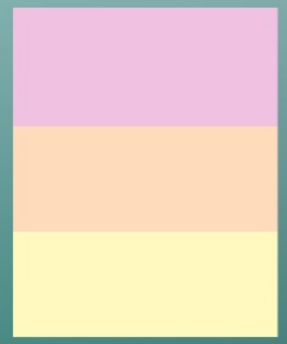
Leader

Significance of Erasure Coding



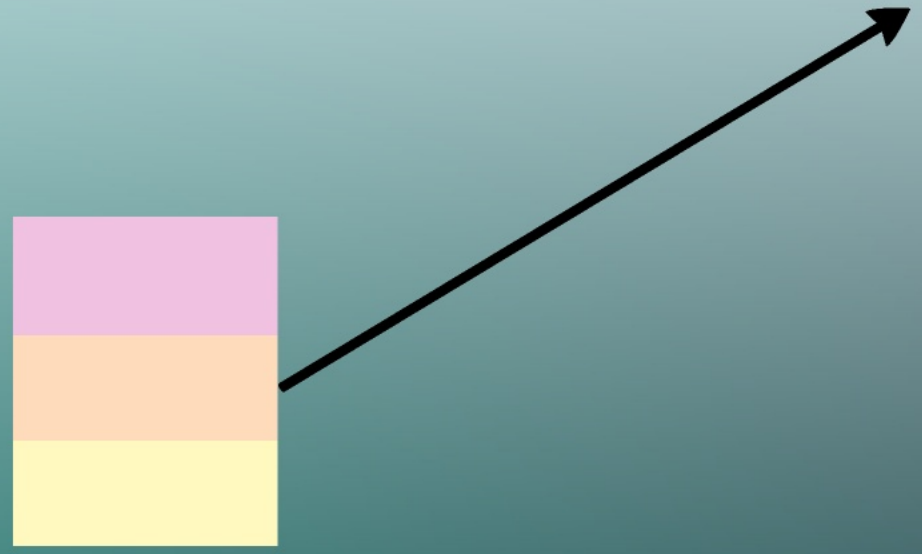
Leader

Significance of Erasure Coding

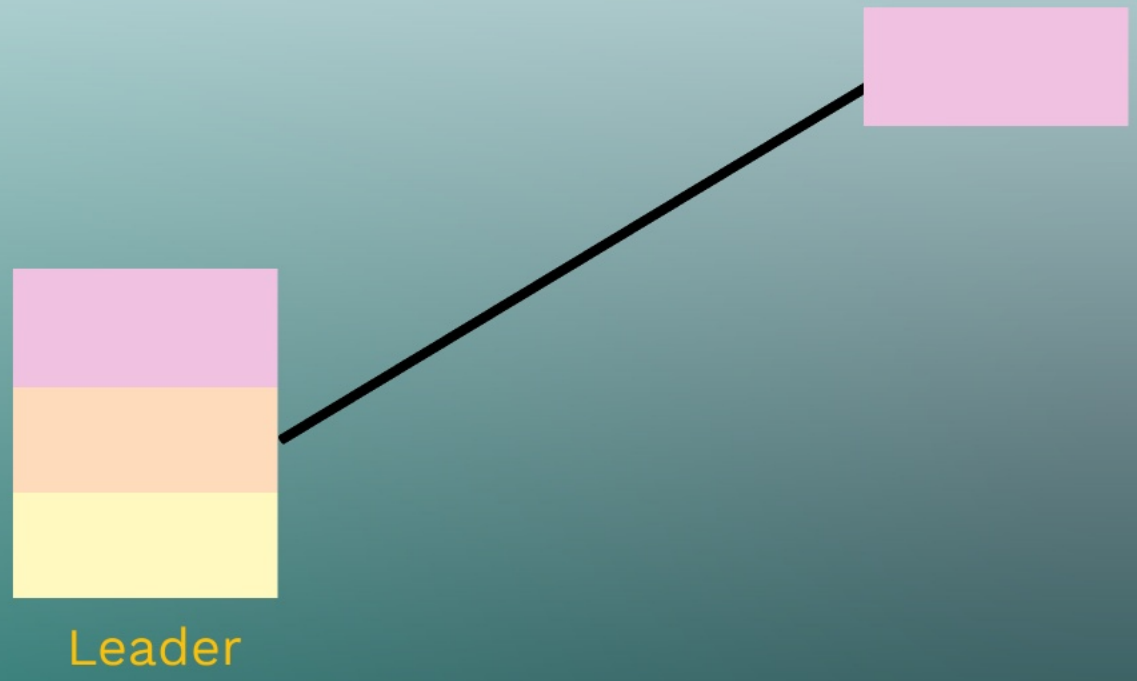


Leader

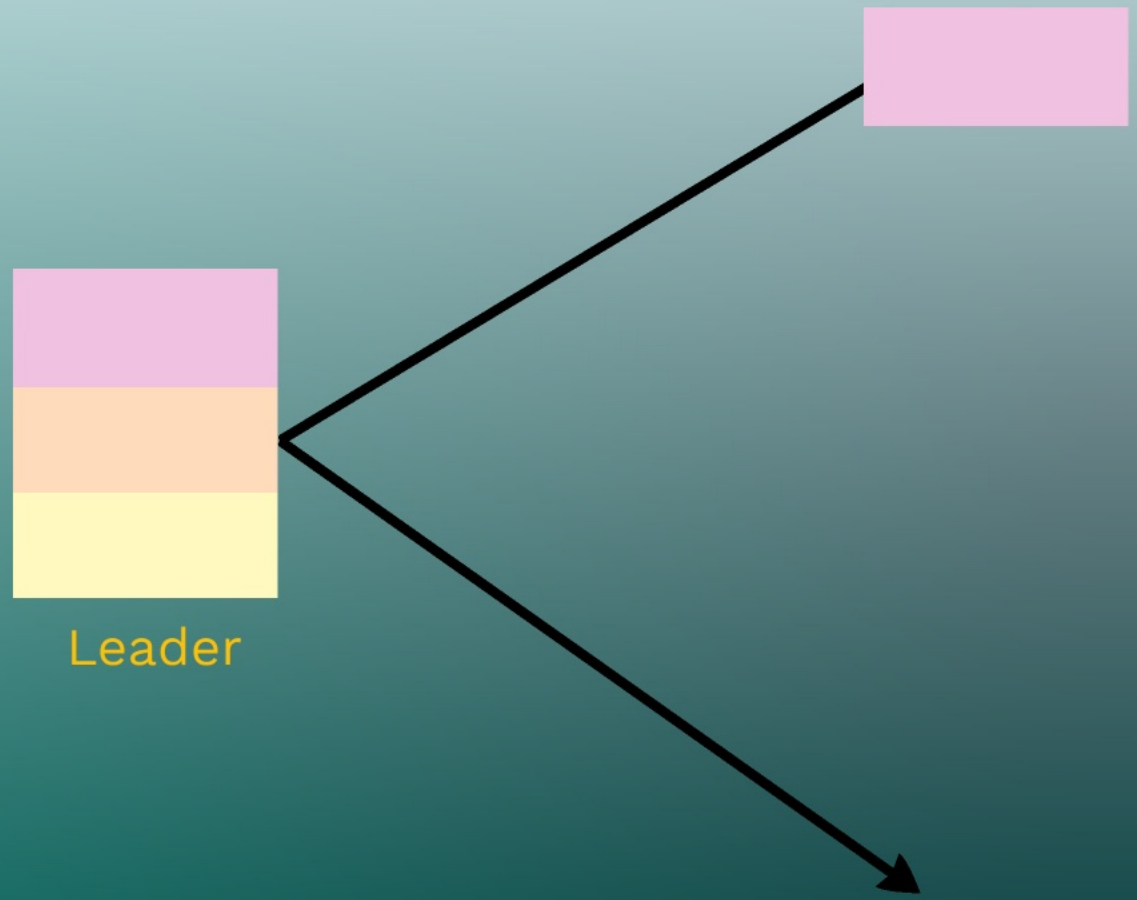
Significance of Erasure Coding



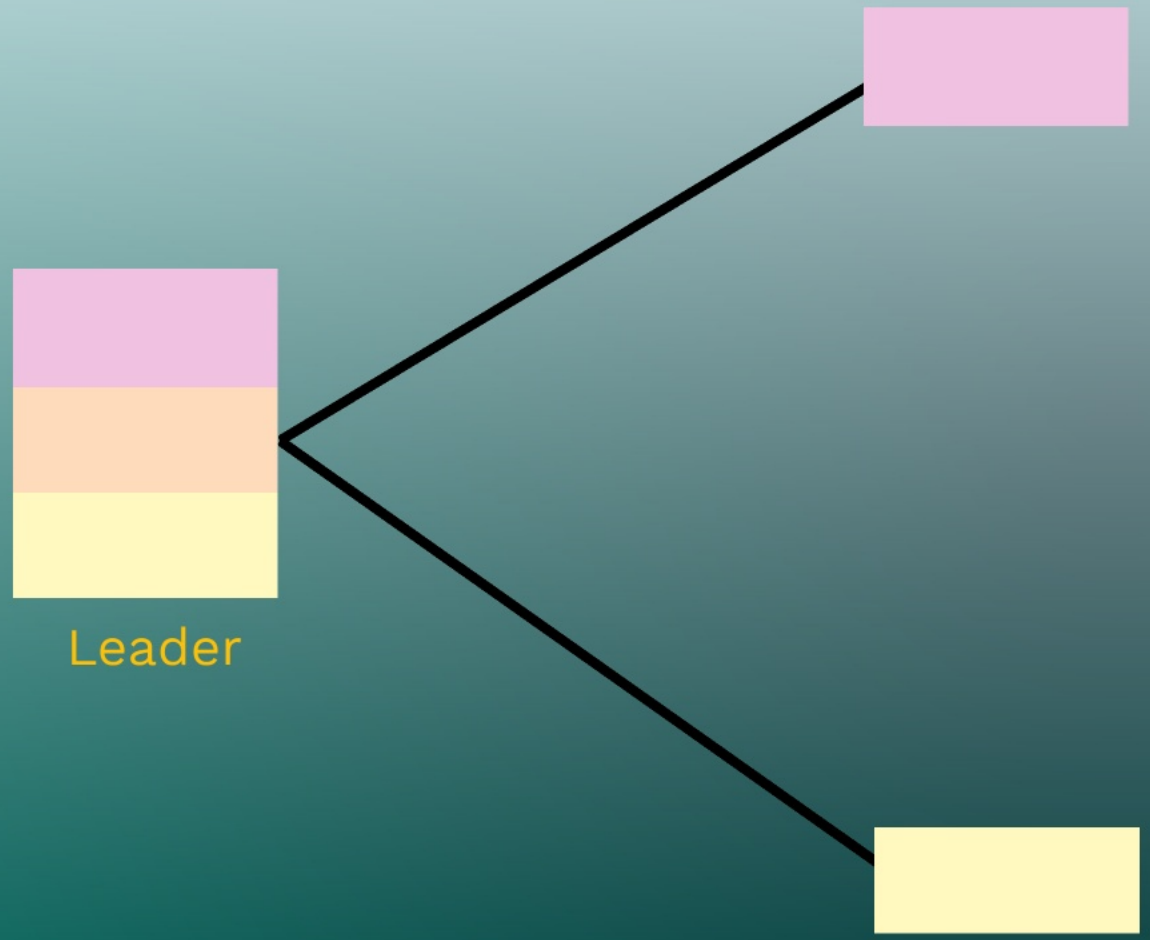
Significance of Erasure Coding



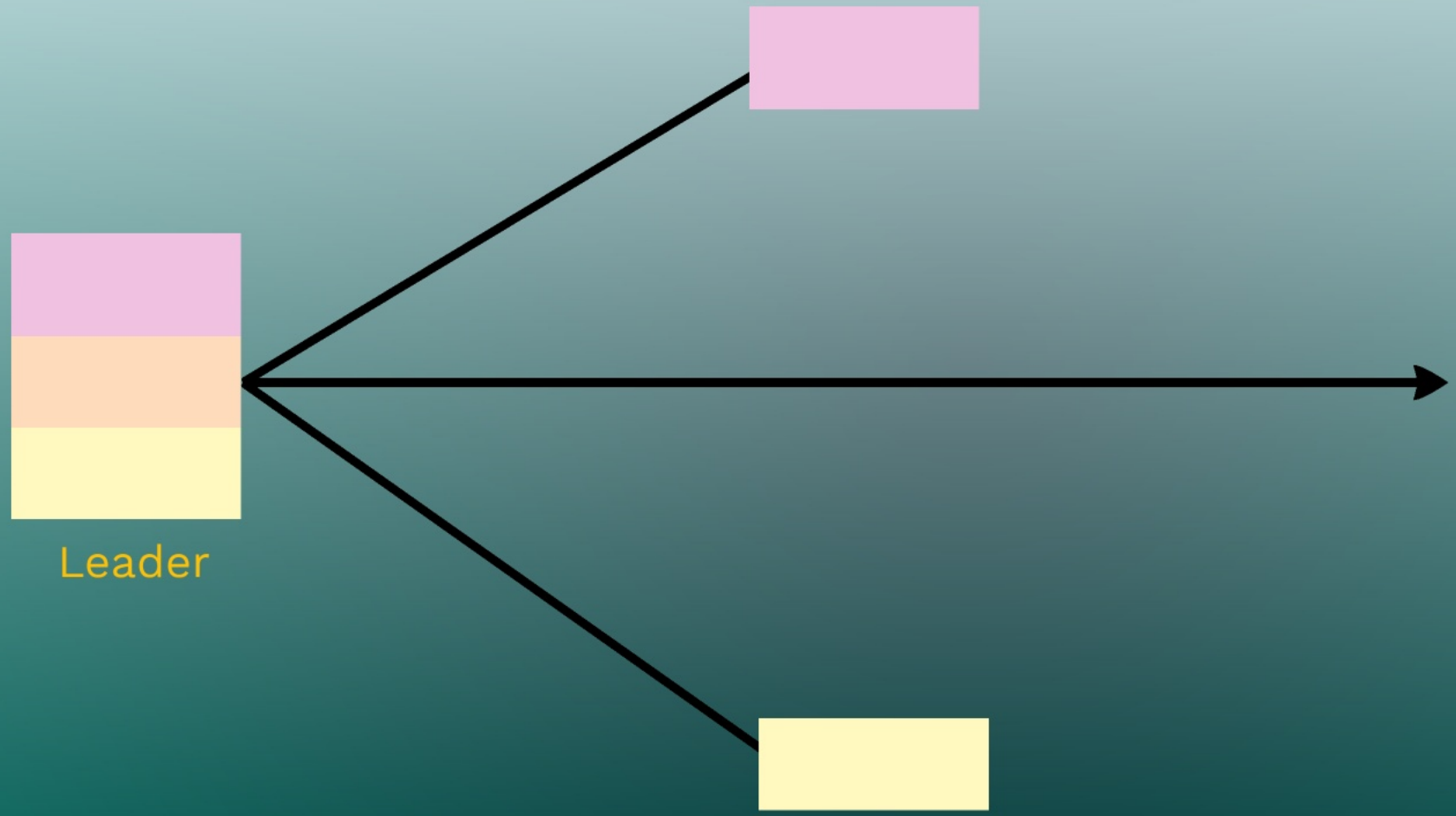
Significance of Erasure Coding



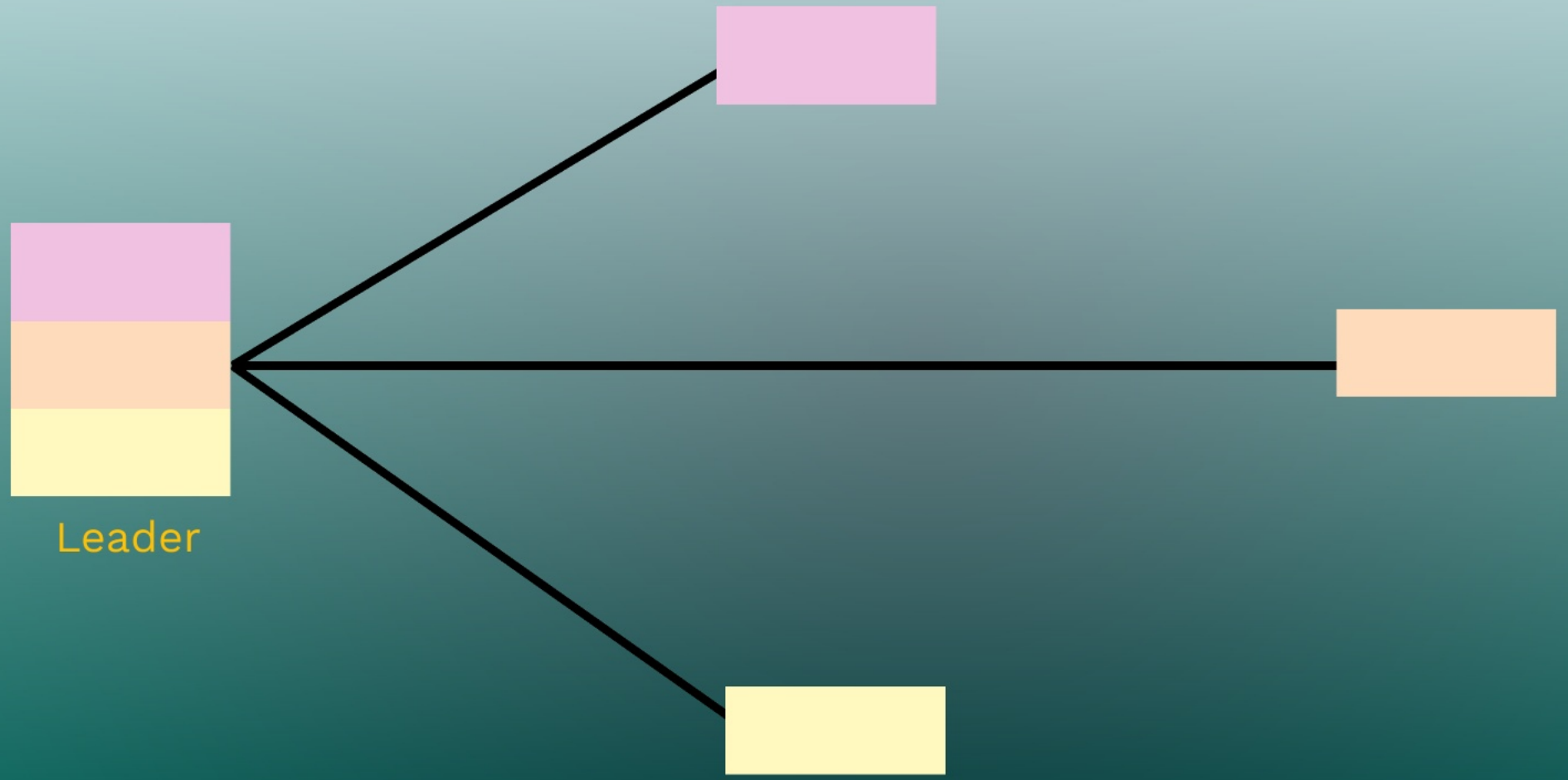
Significance of Erasure Coding



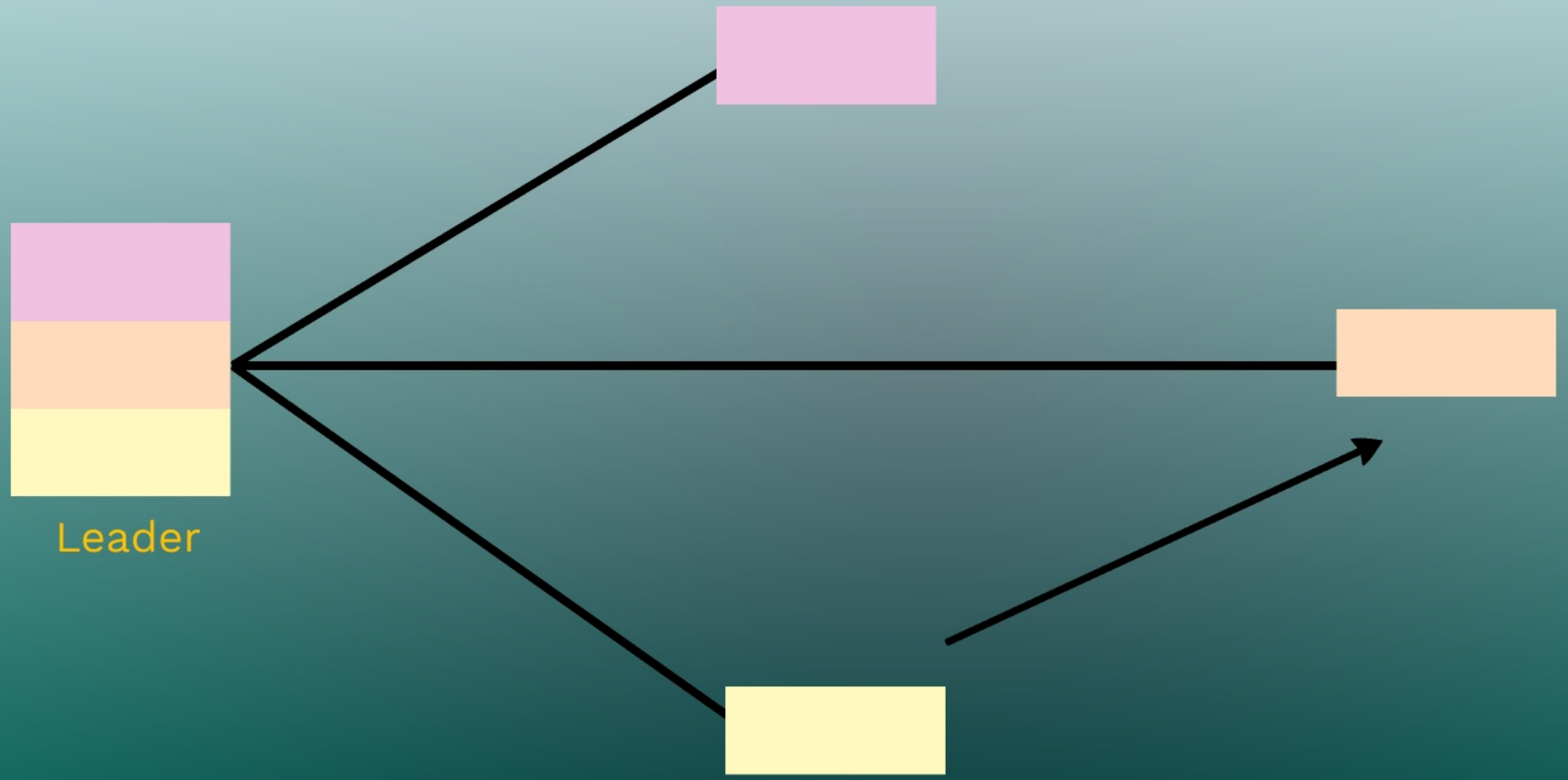
Significance of Erasure Coding



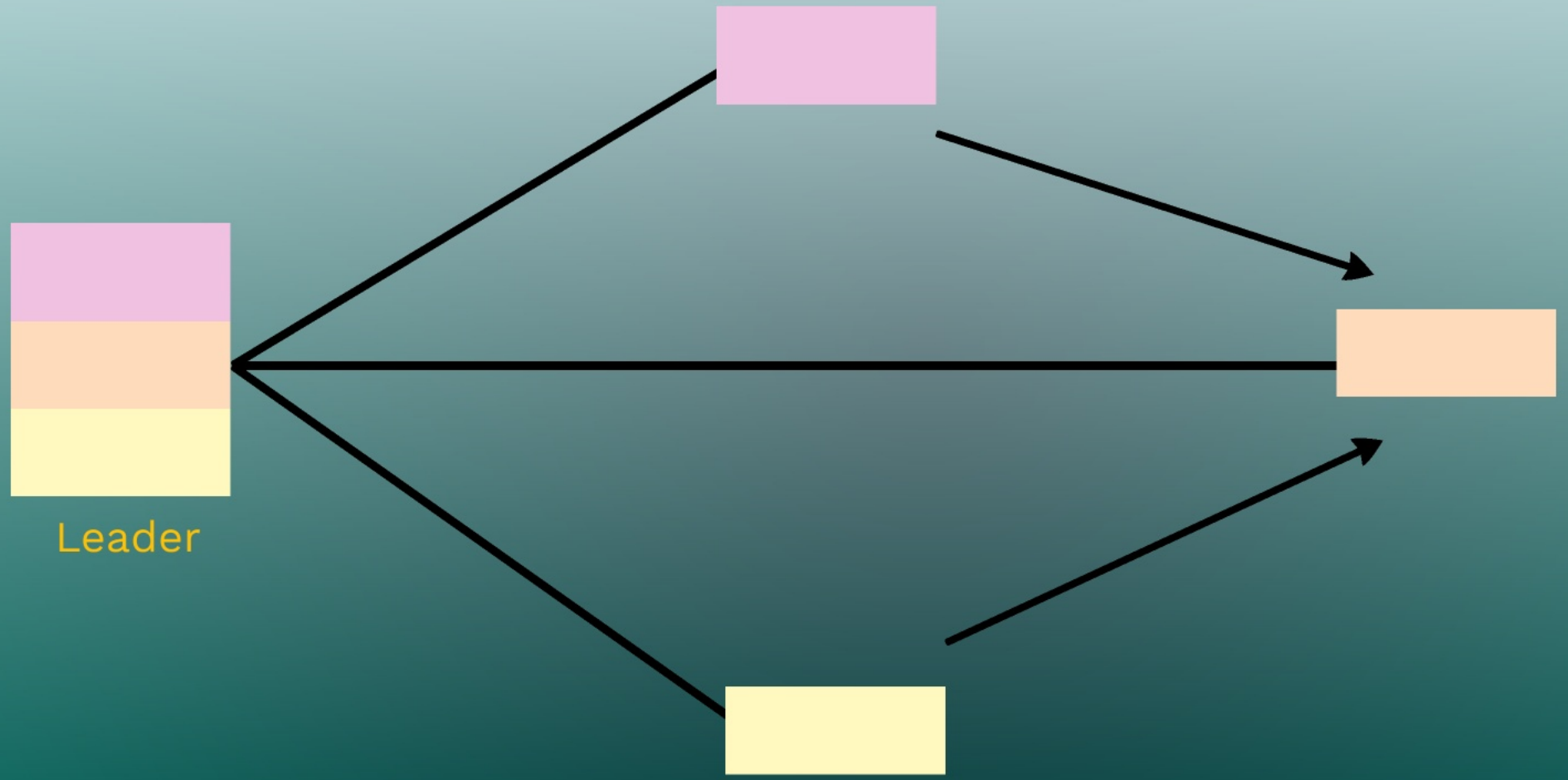
Significance of Erasure Coding



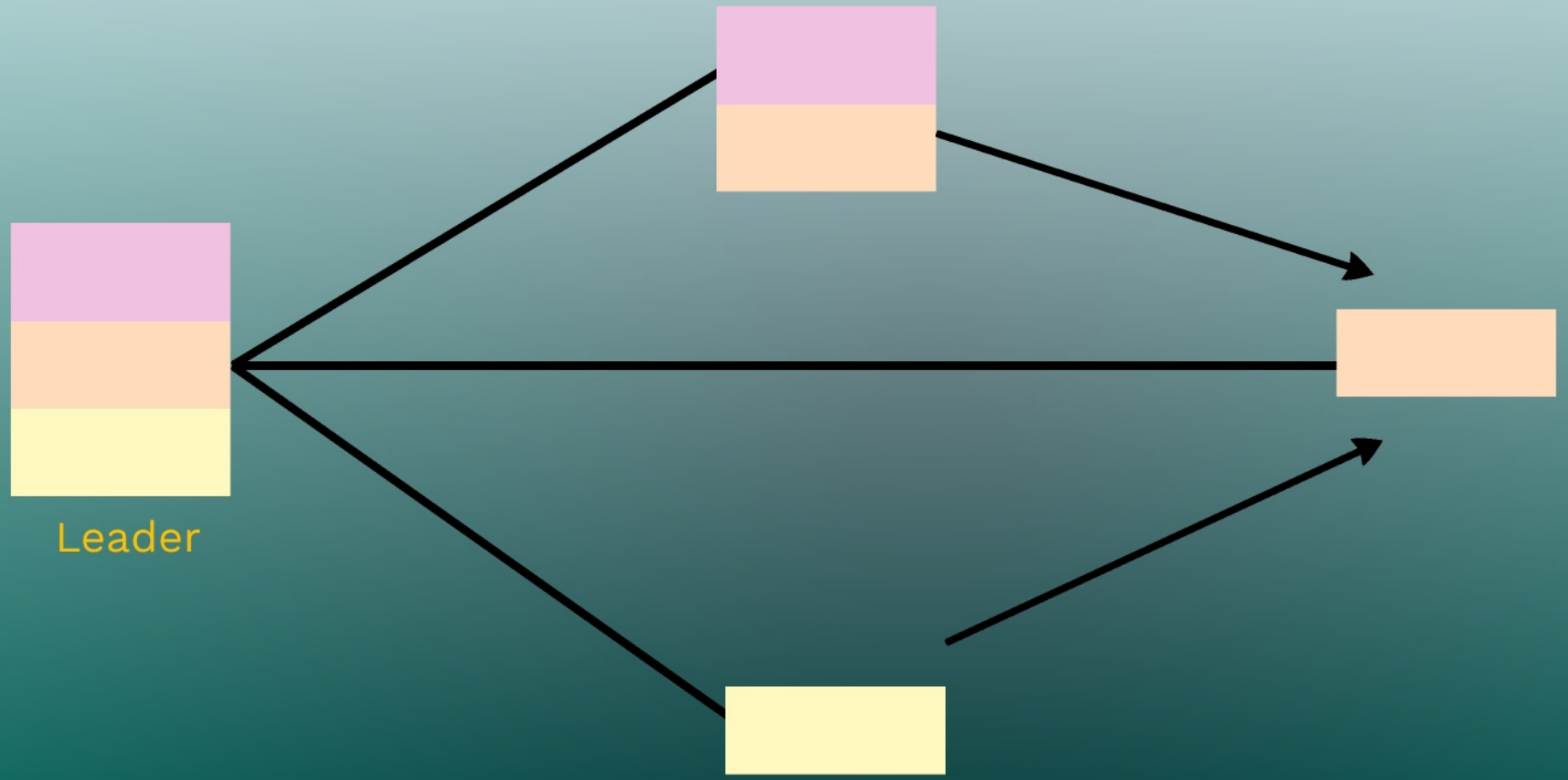
Significance of Erasure Coding



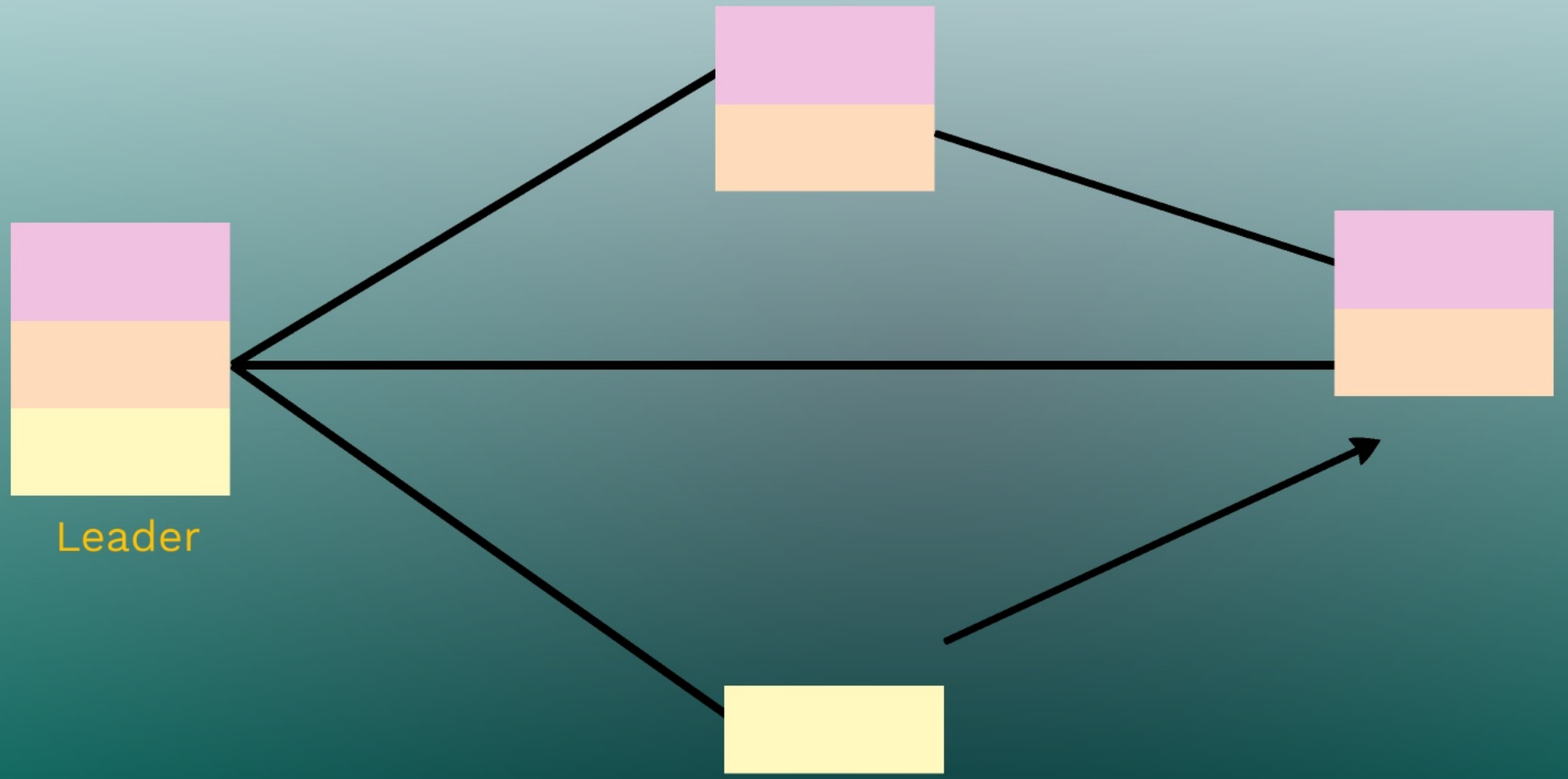
Significance of Erasure Coding



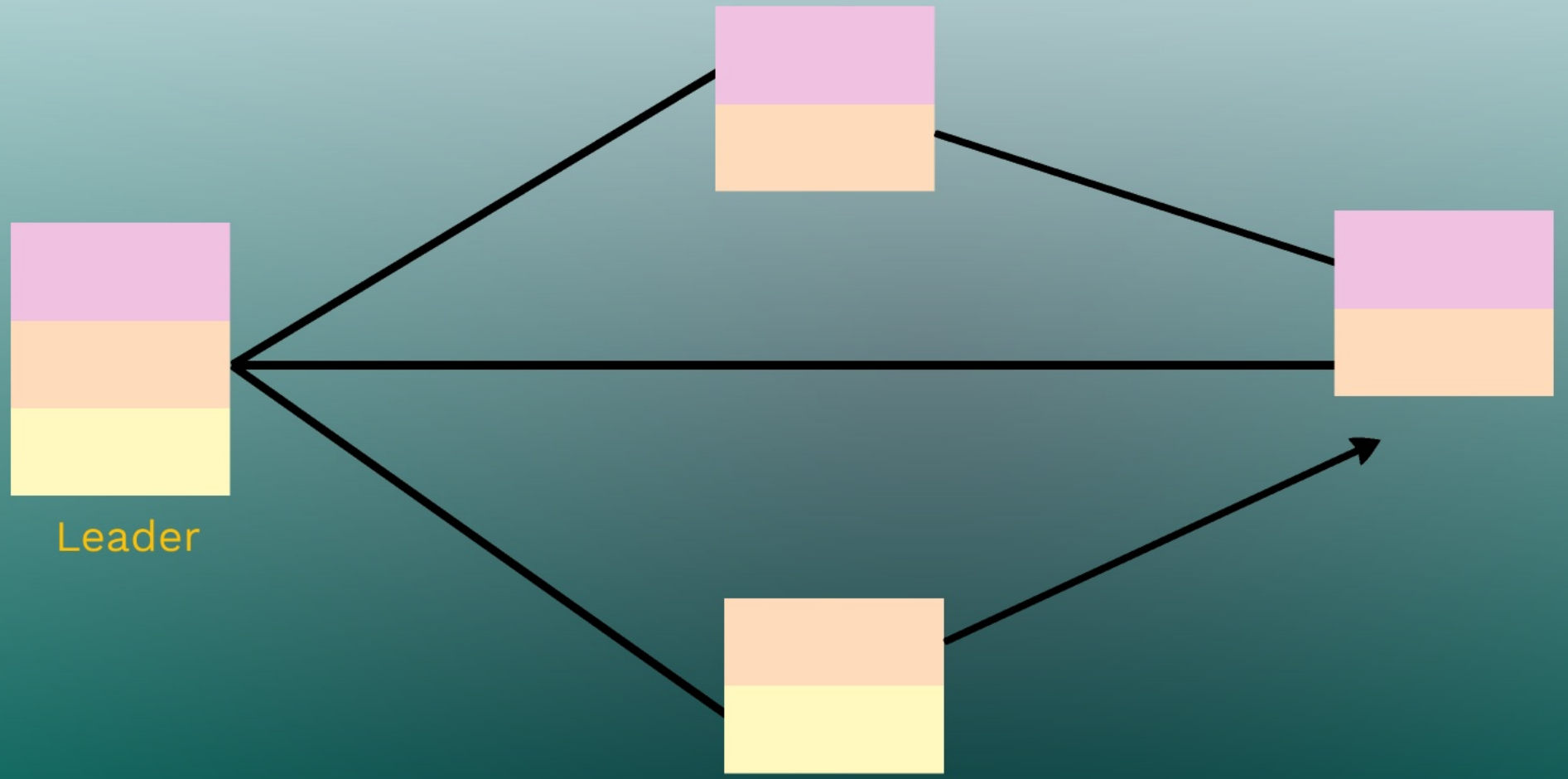
Significance of Erasure Coding



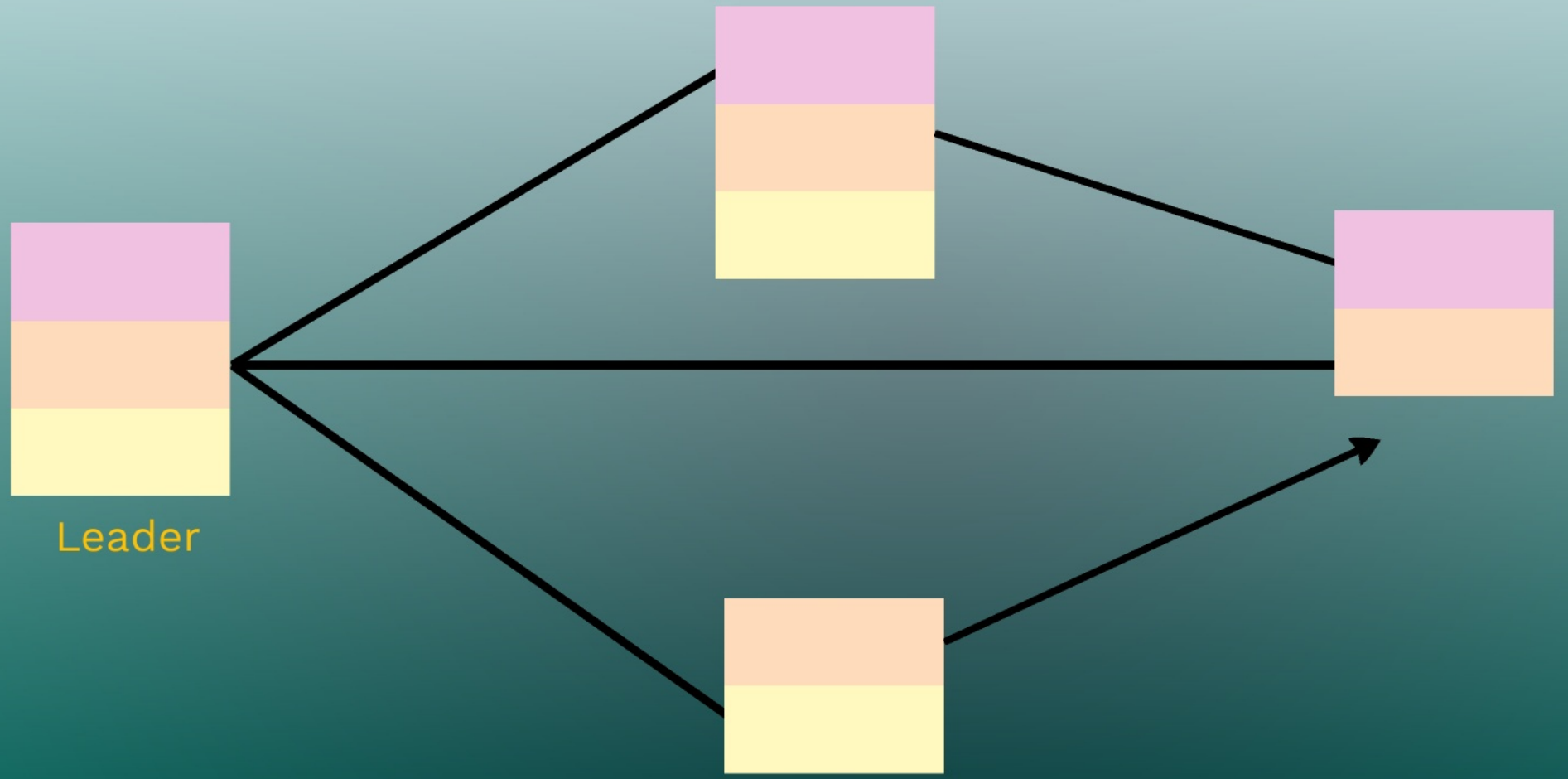
Significance of Erasure Coding



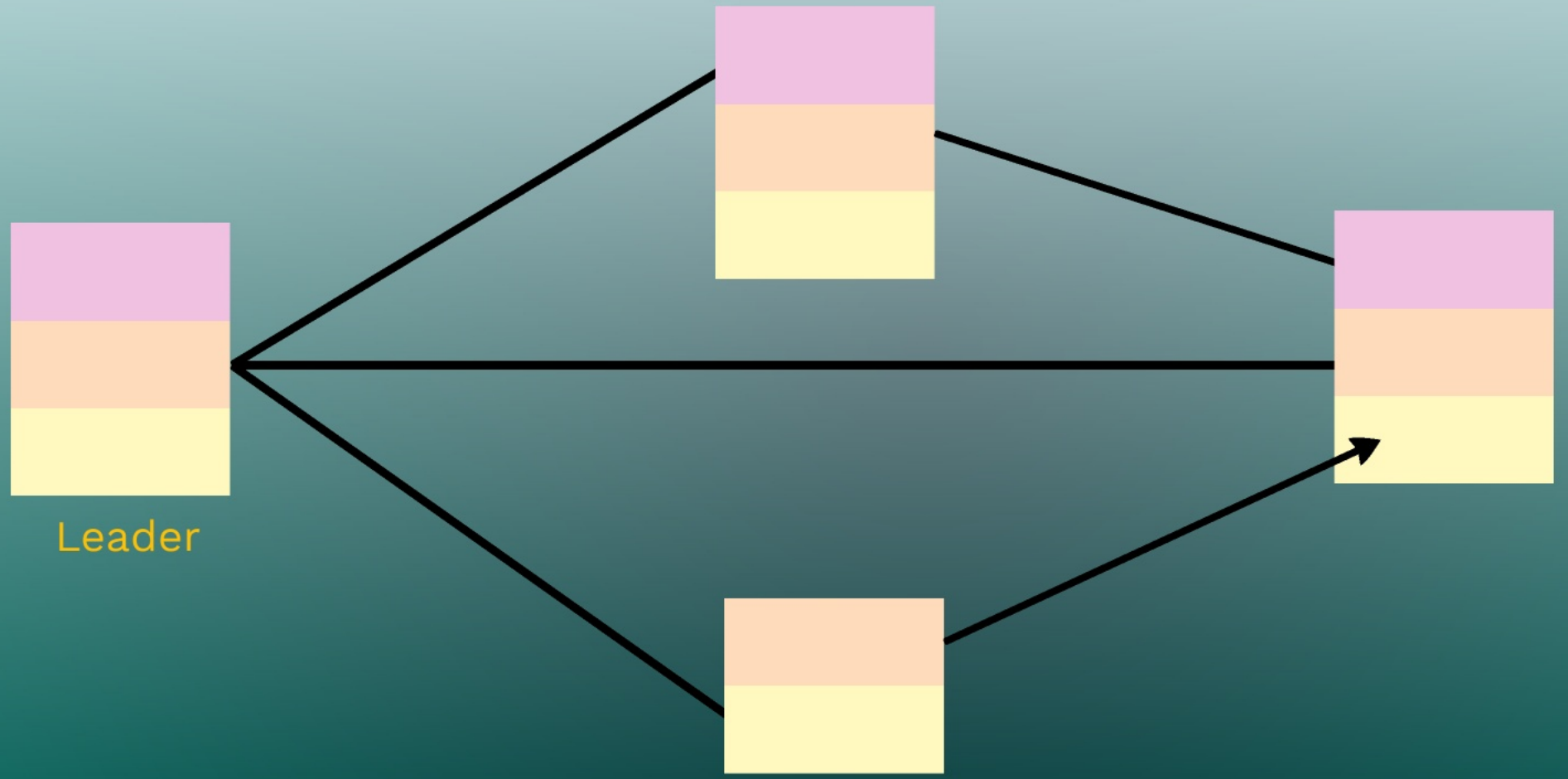
Significance of Erasure Coding



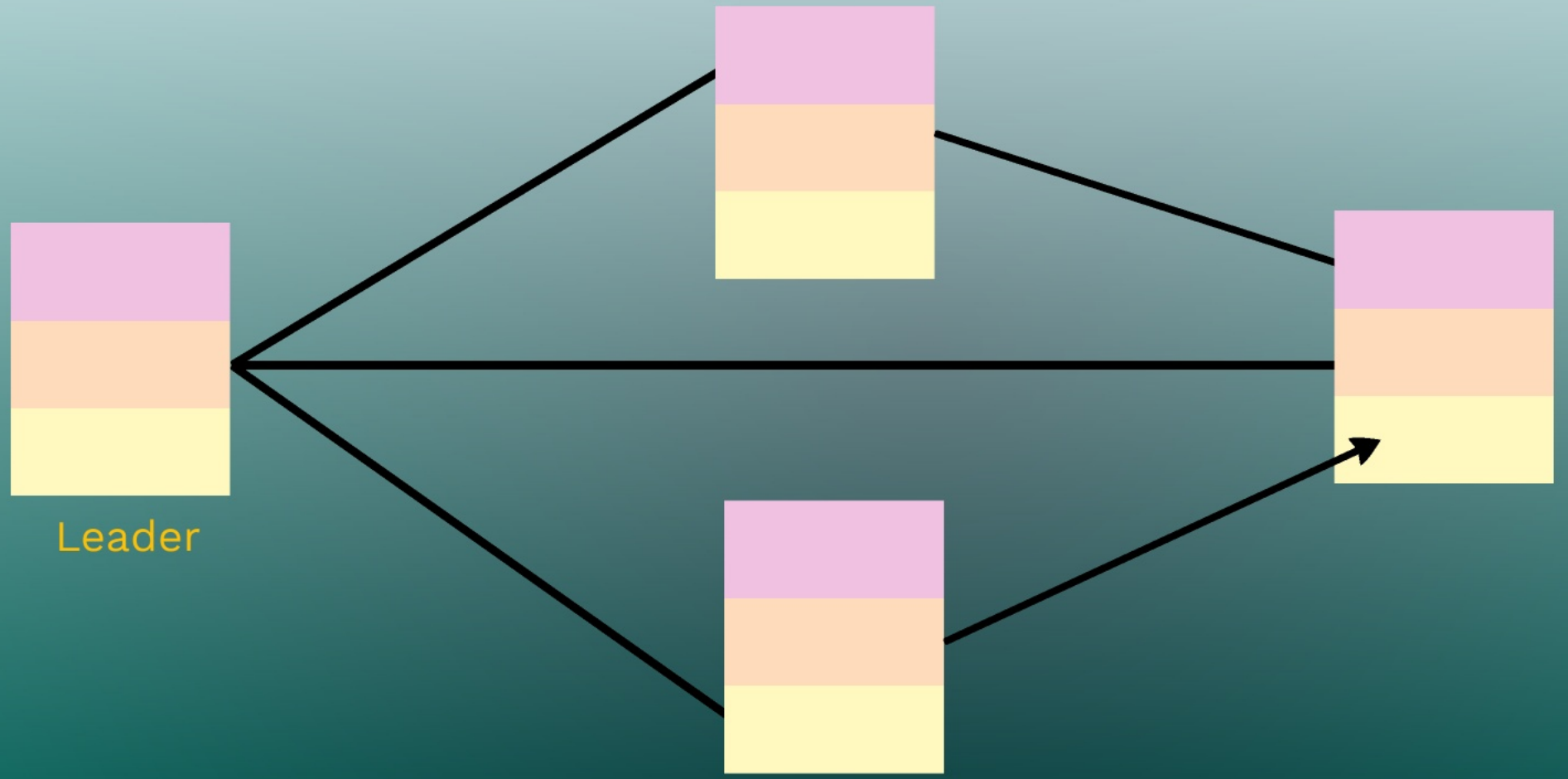
Significance of Erasure Coding

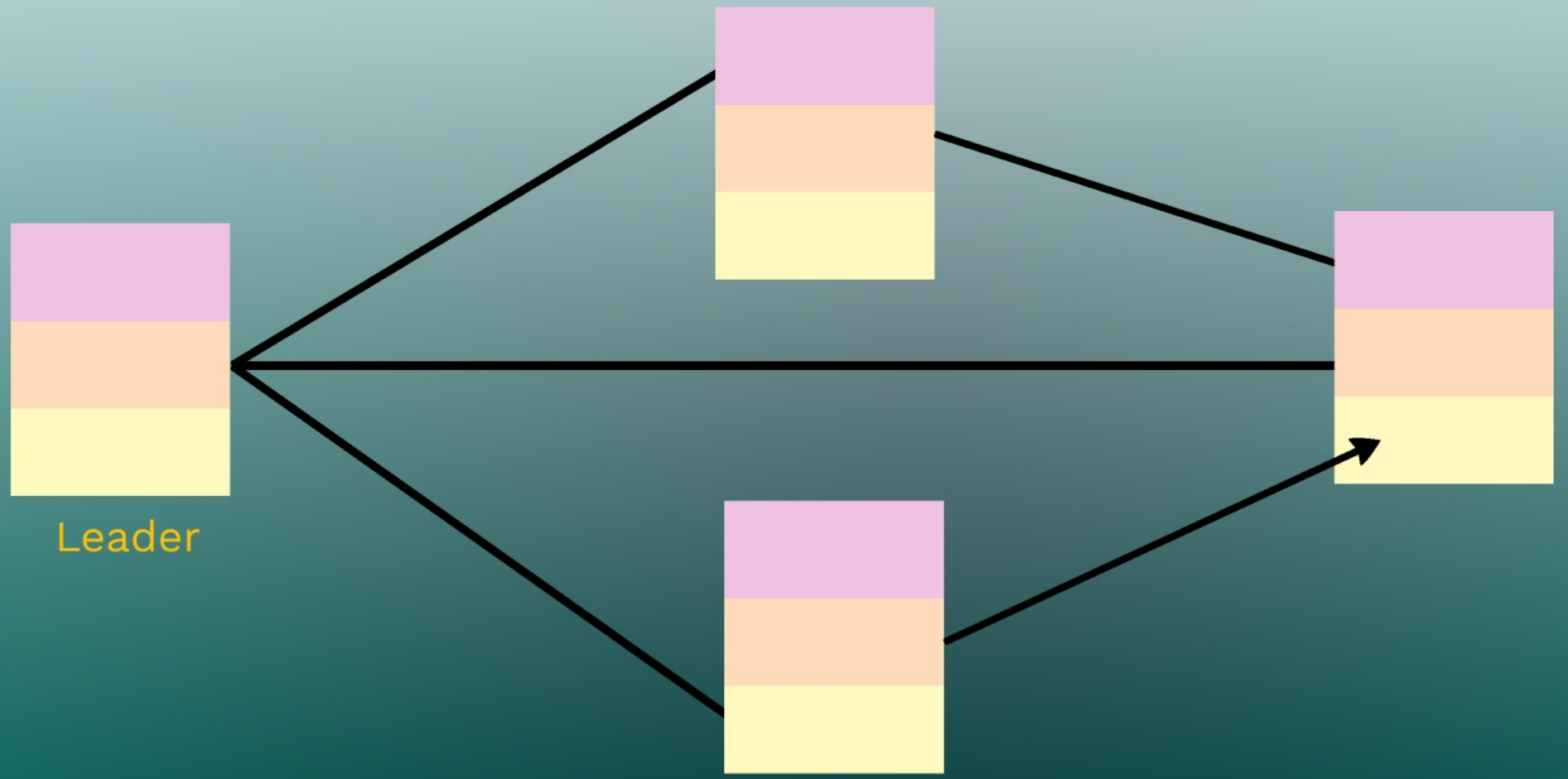


Significance of Erasure Coding

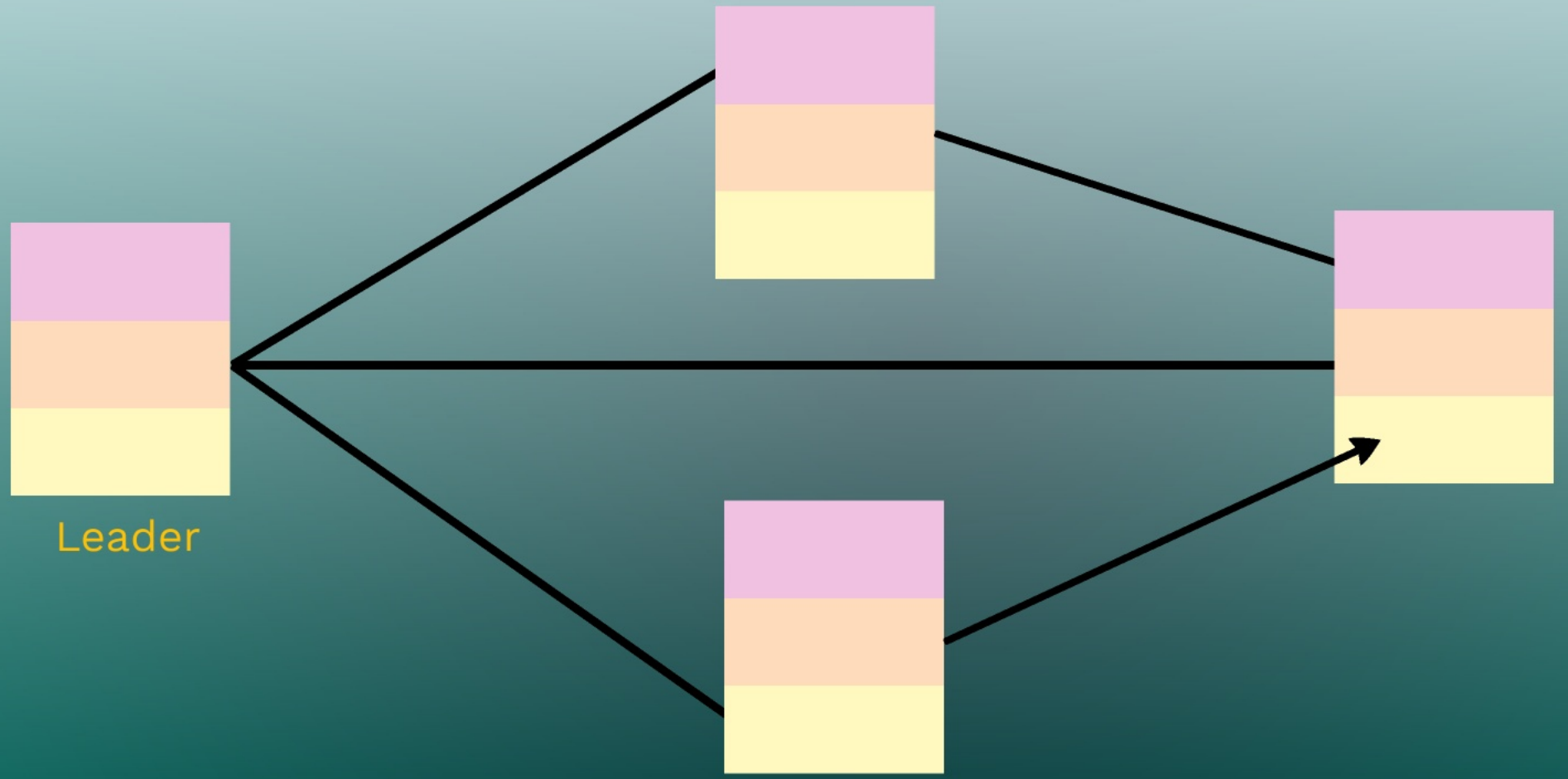


Significance of Erasure Coding

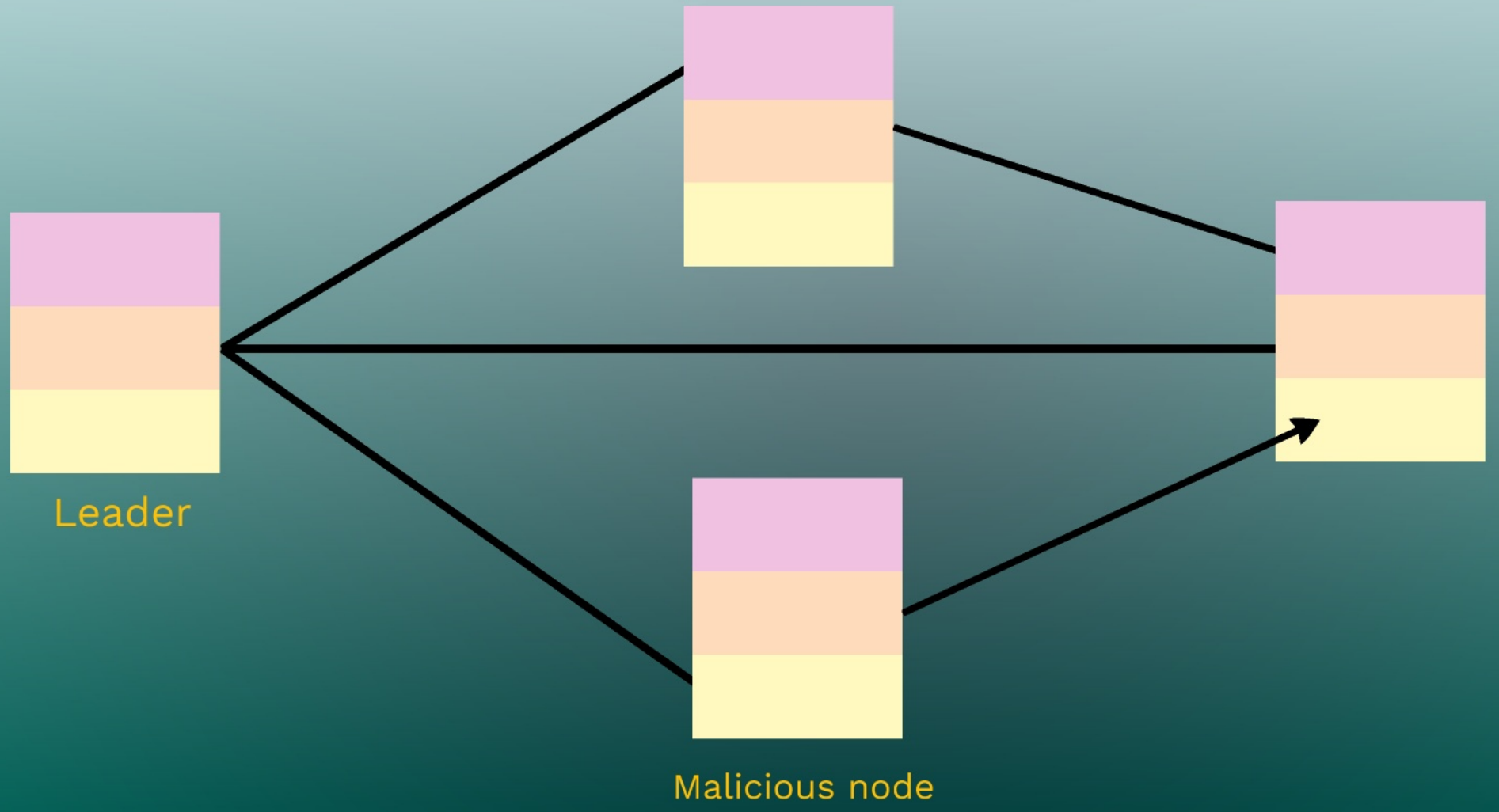



Leader

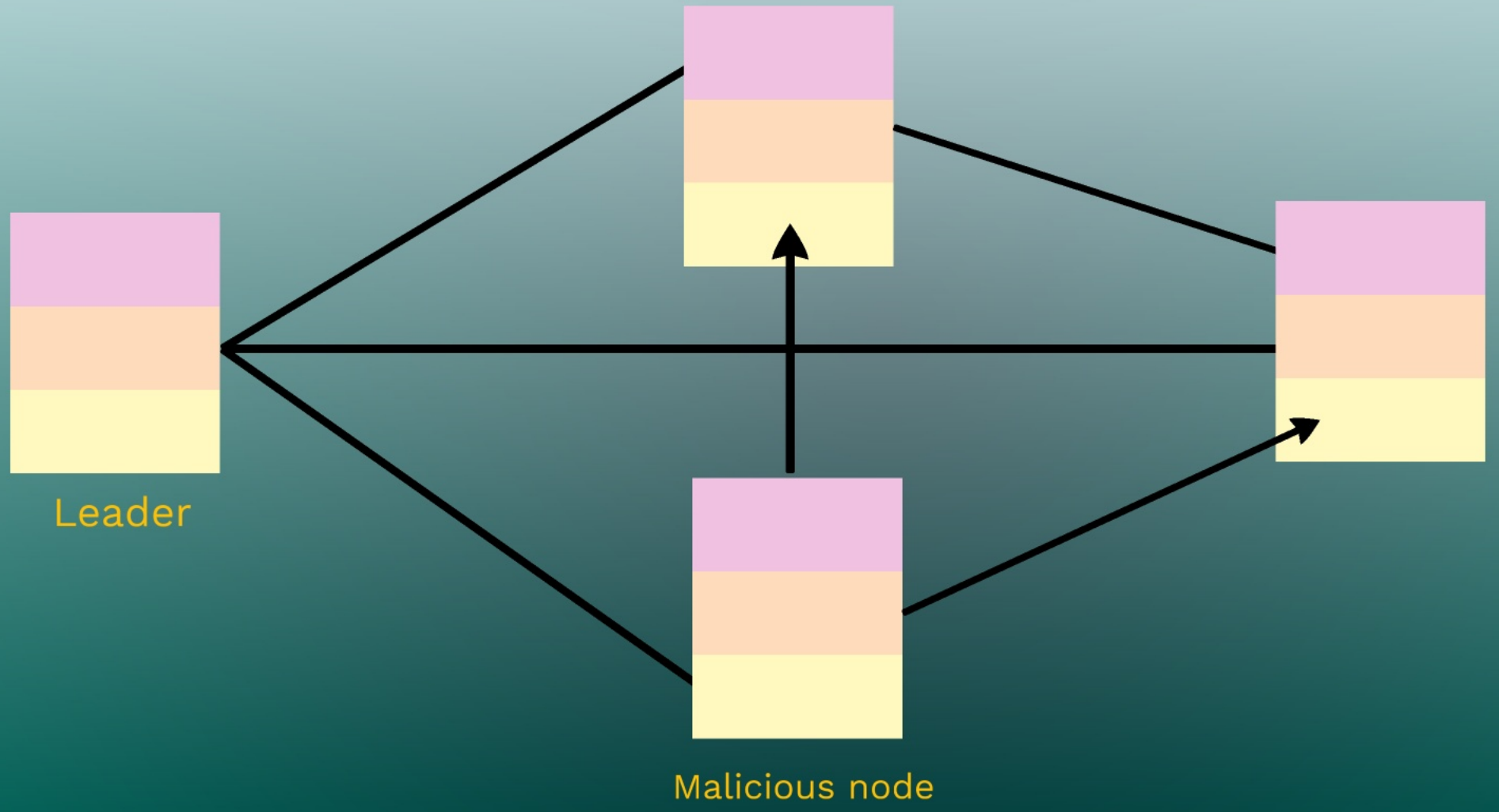
Significance of Merkel Tree



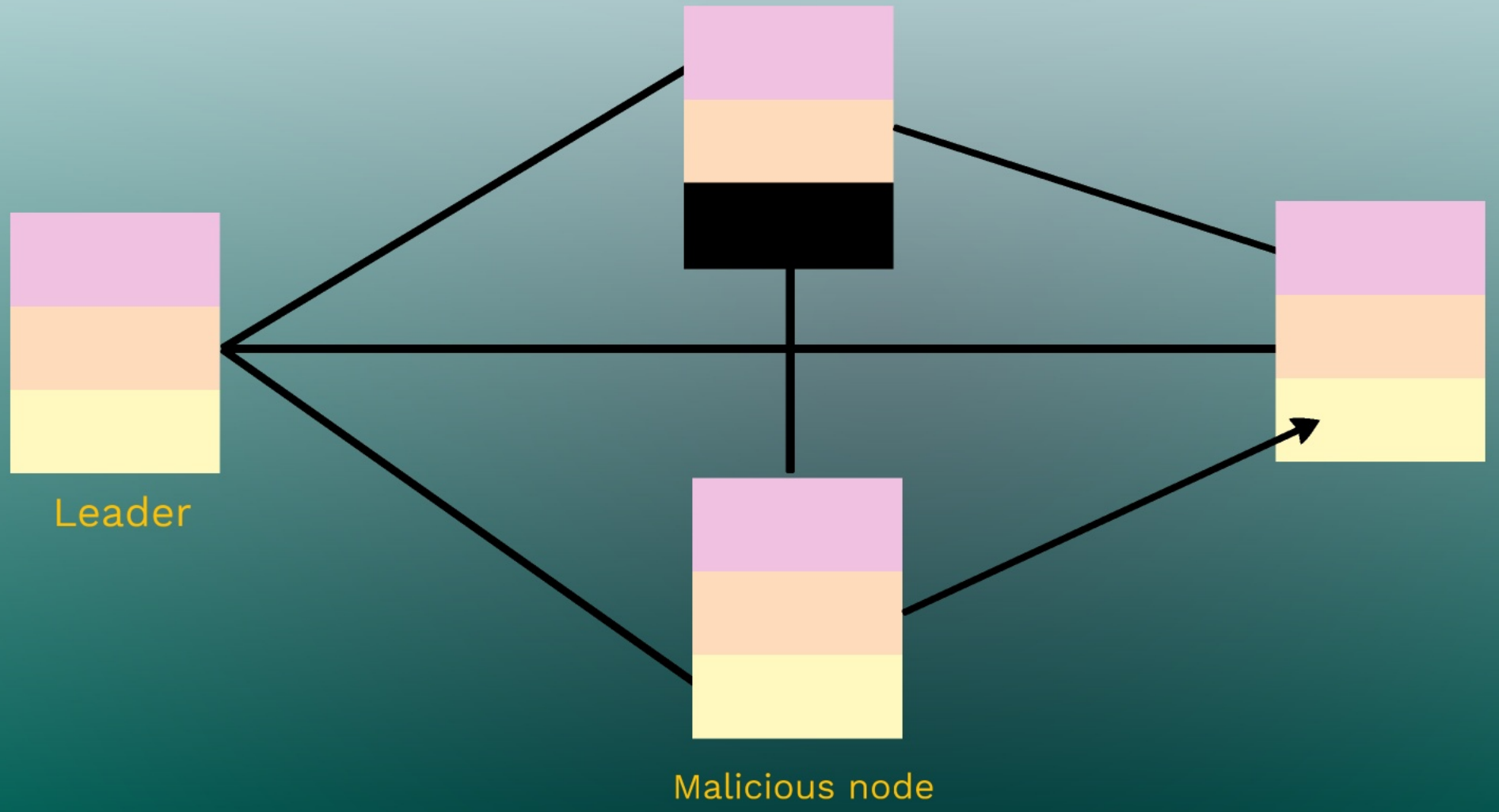
Significance of Merkel Tree



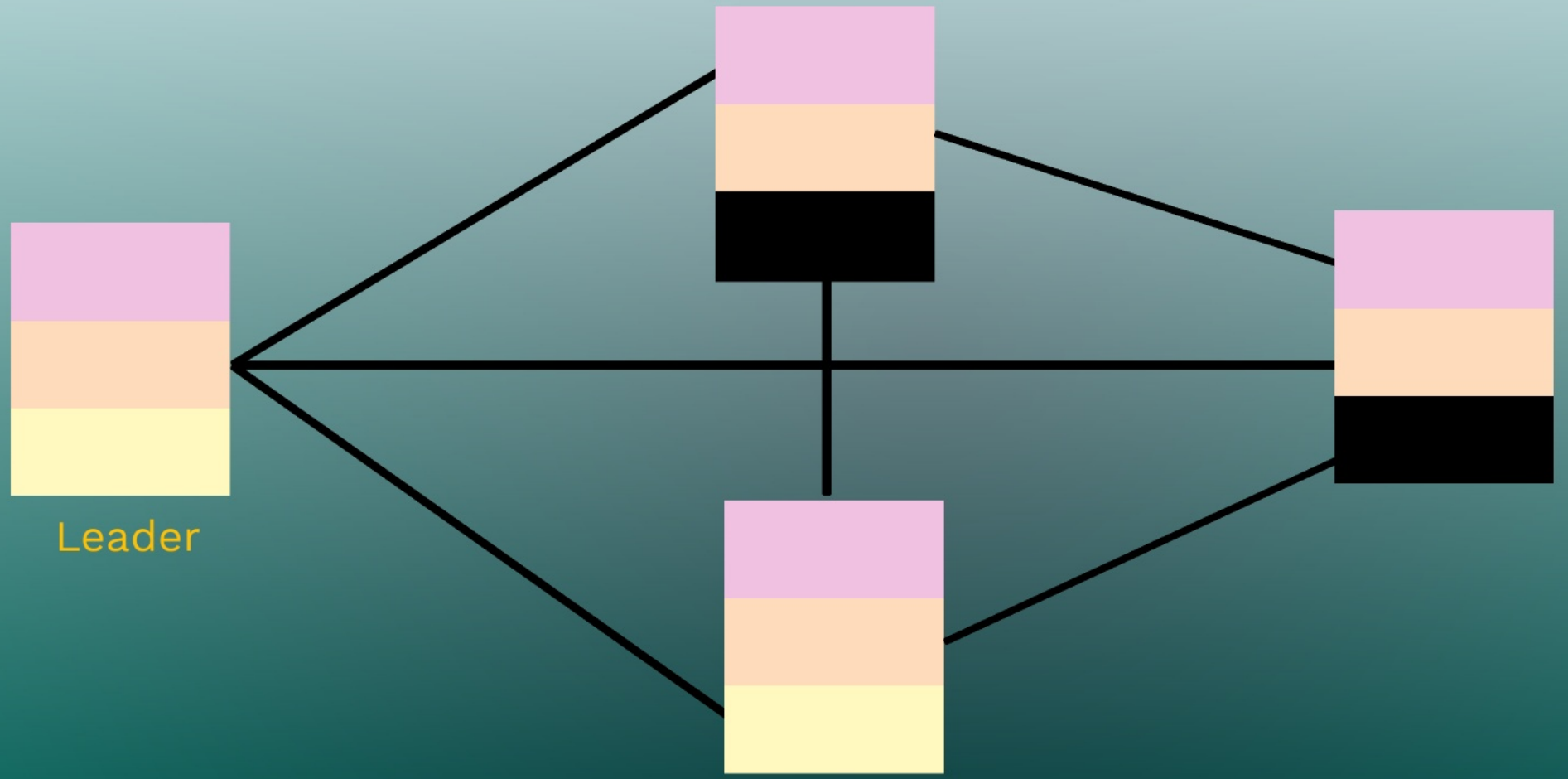
Significance of Merkel Tree



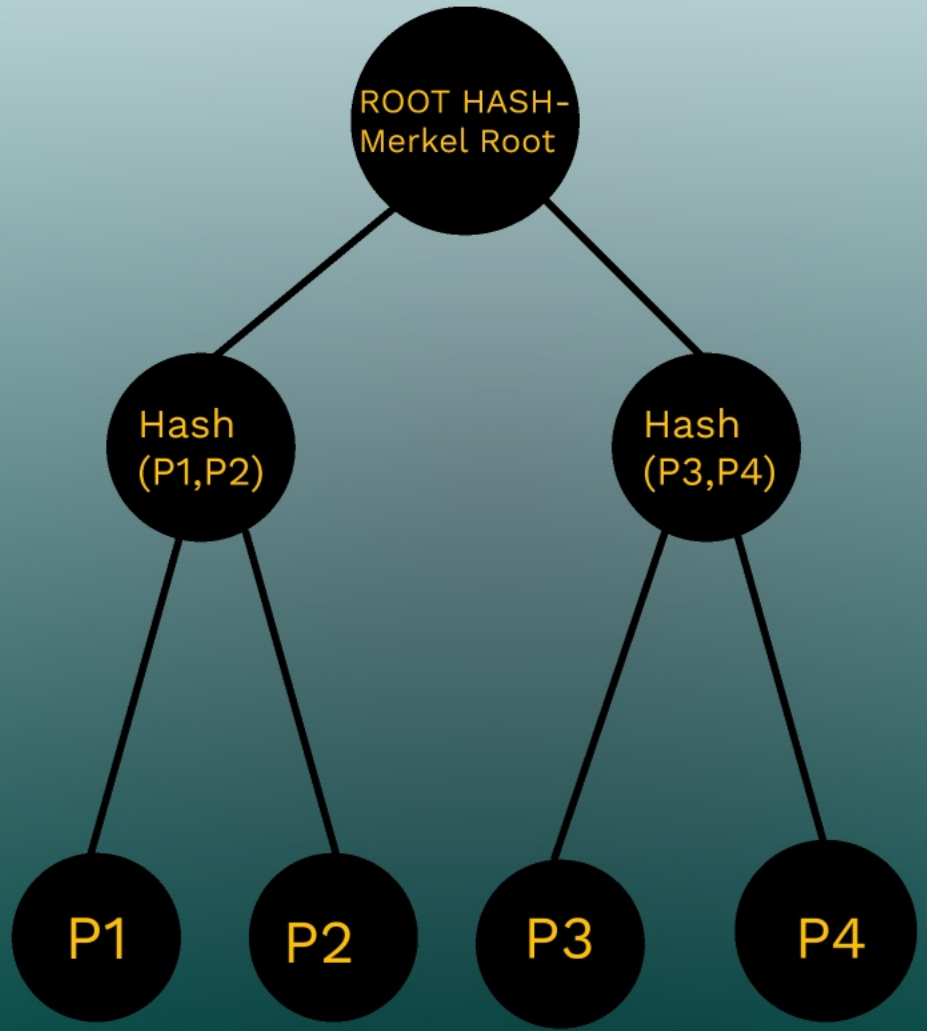
Significance of Merkel Tree



Significance of Merkel Tree



Merkel Tree



Broadcast Stage

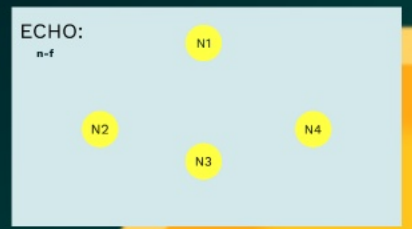
Reliable Broadcast Mechanism

P1: Encrypted Message

P1

ECHO:

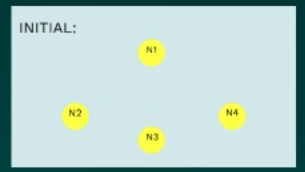
n-f



READY:

2f+1

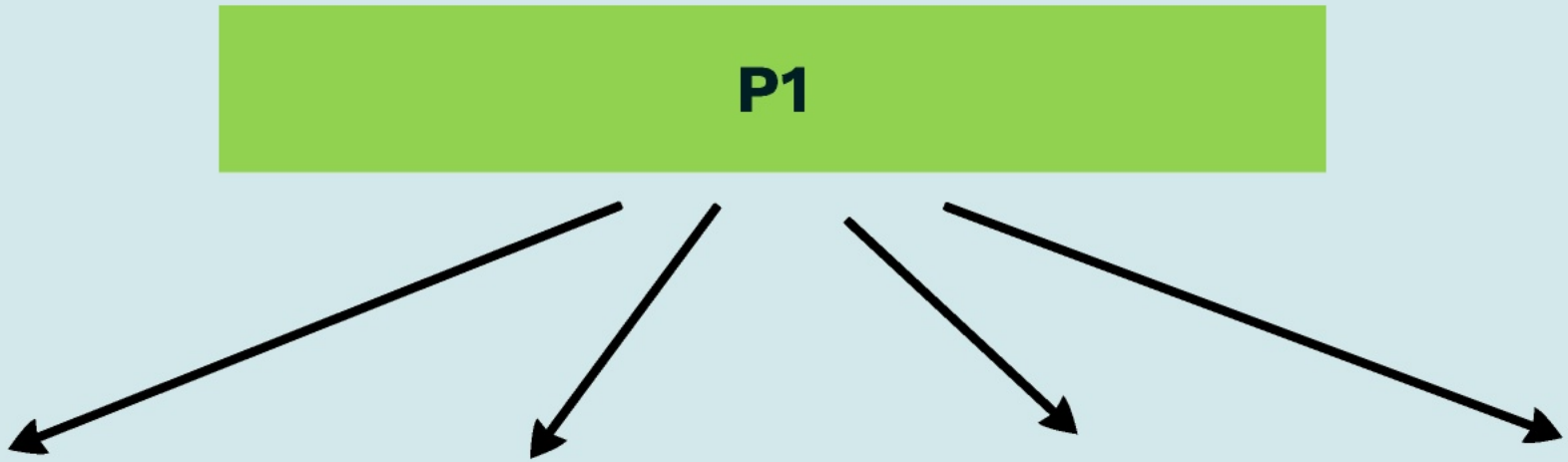
INITIAL:



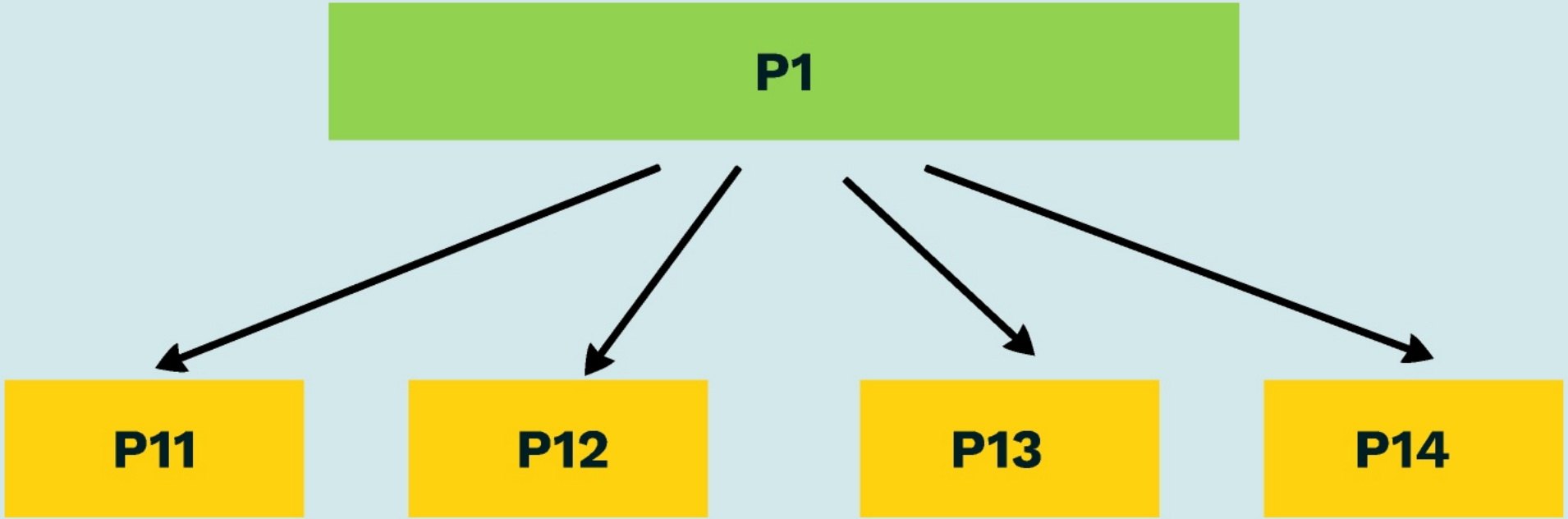
Reliable Broadcast Mechanism

P1: Encrypted Message

P1: Encrypted Message

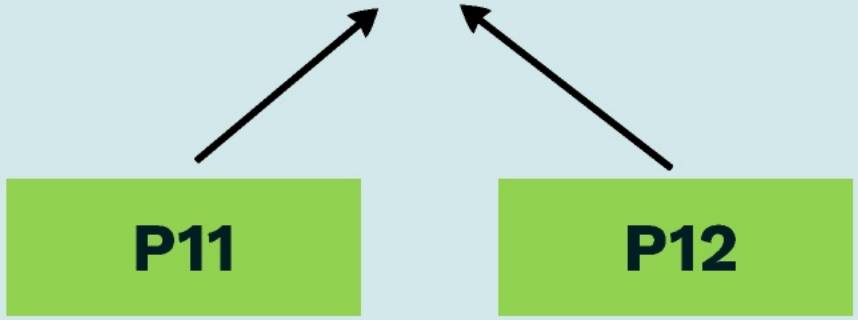


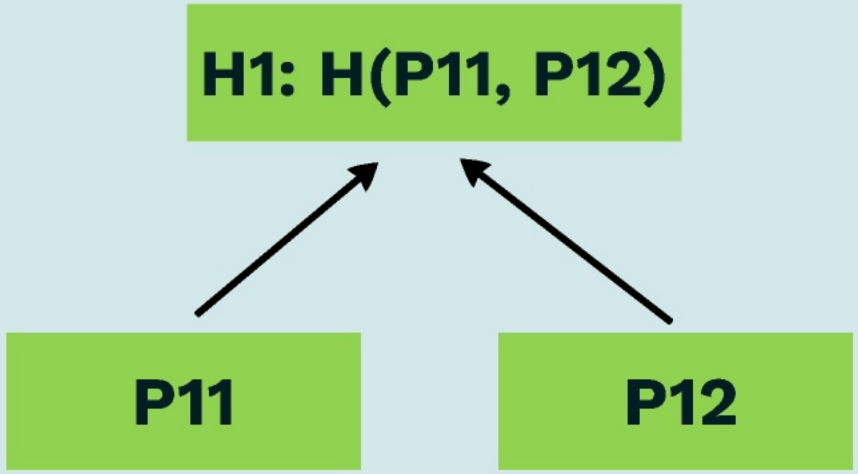
P1: Encrypted Message

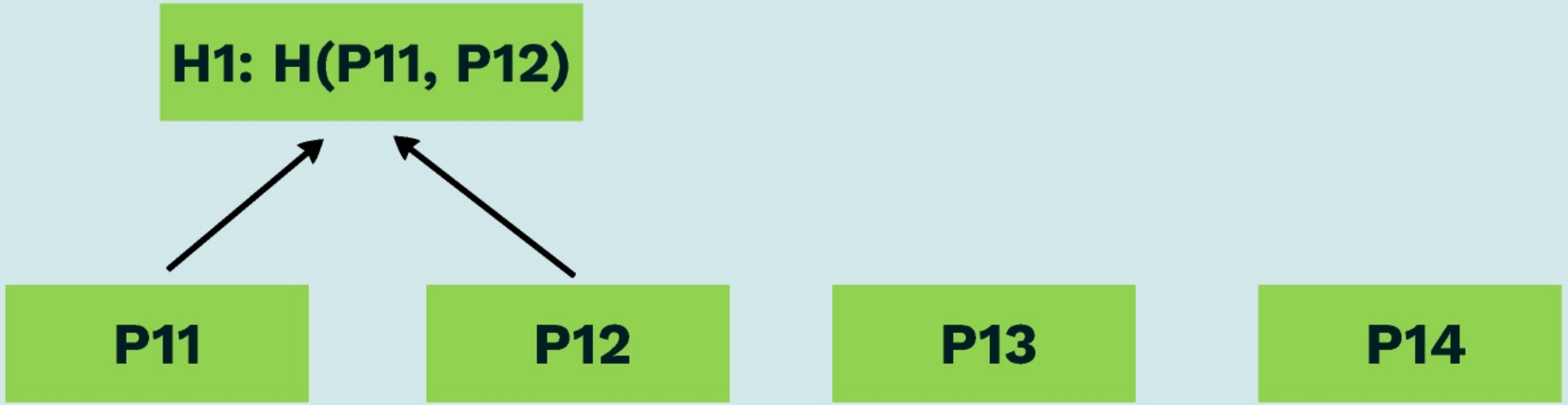


P11

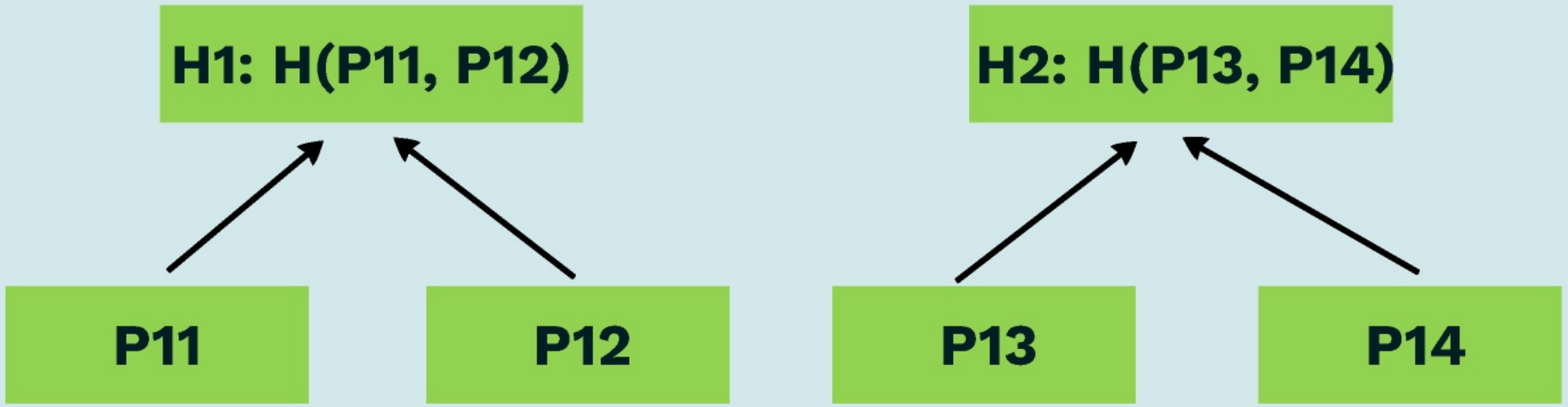
P12

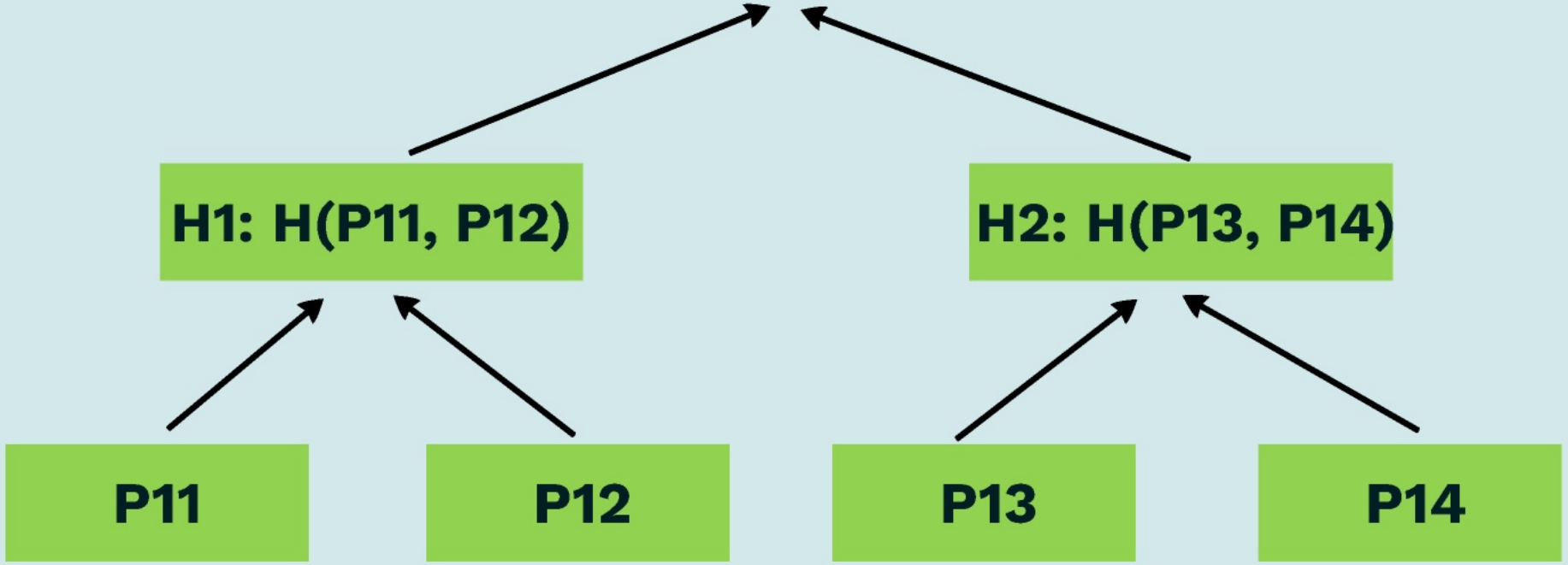

P11
P12

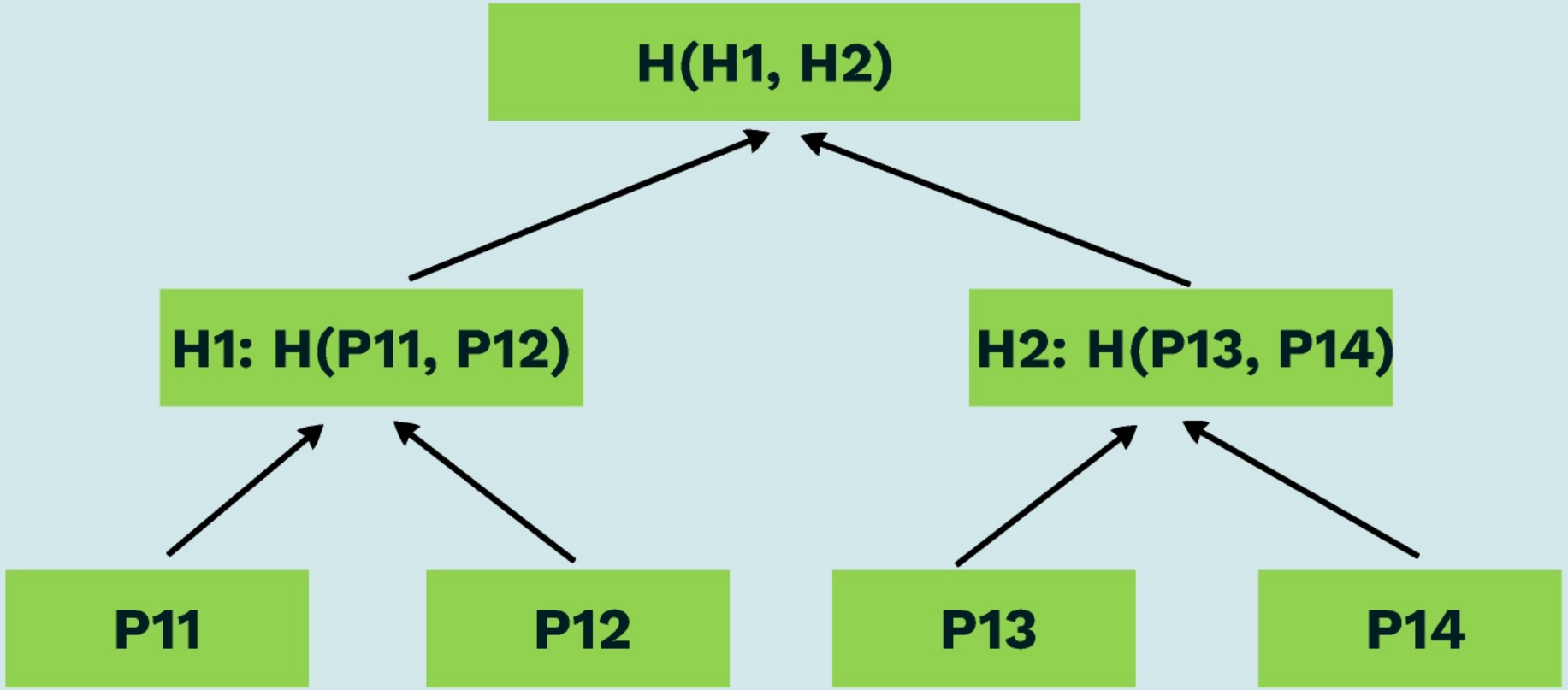

H1: H(P11, P12)
P11
P12


H1: H(P11, P12)
P11
P12
P13
P14

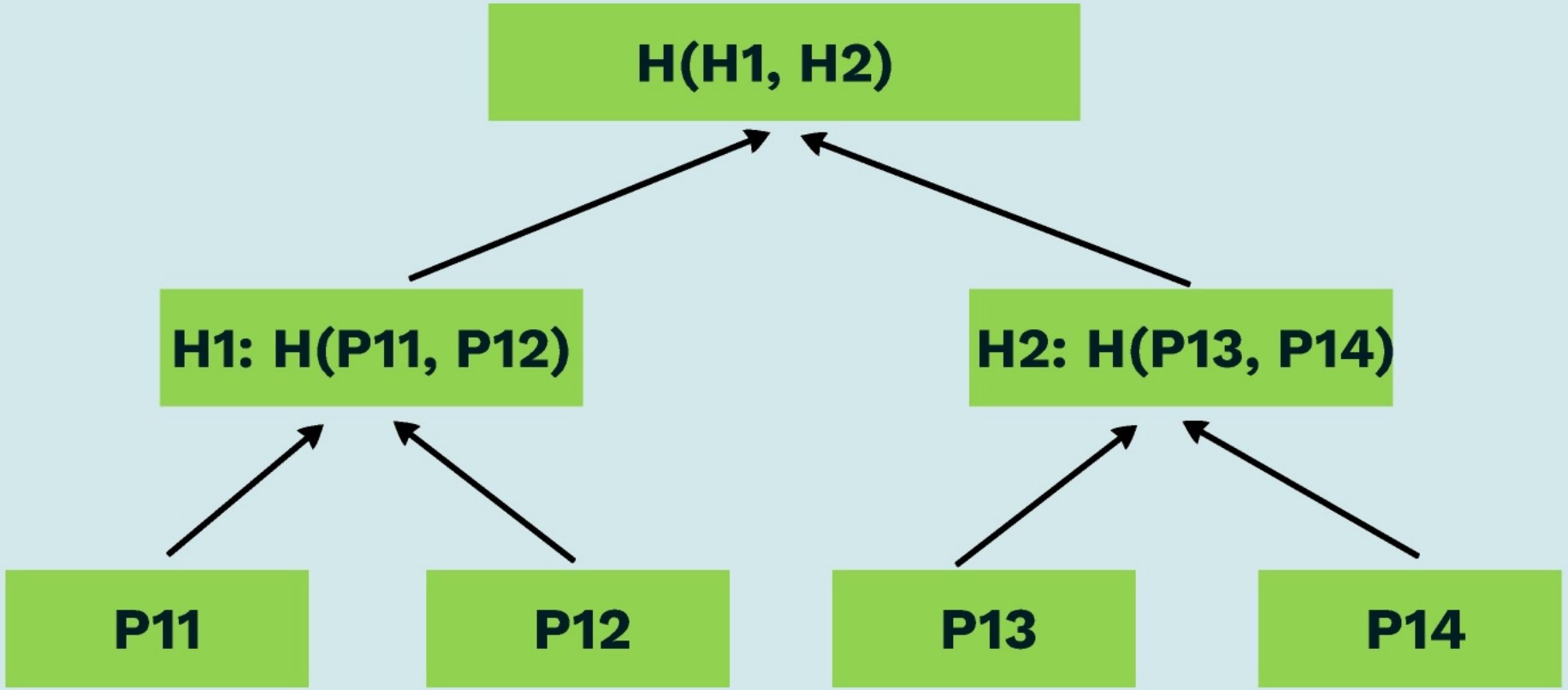

H1: H(P11, P12)
P11
P12
P13
P14


H1: H(P11, P12)
H2: H(P13, P14)
P11
P12
P13
P14

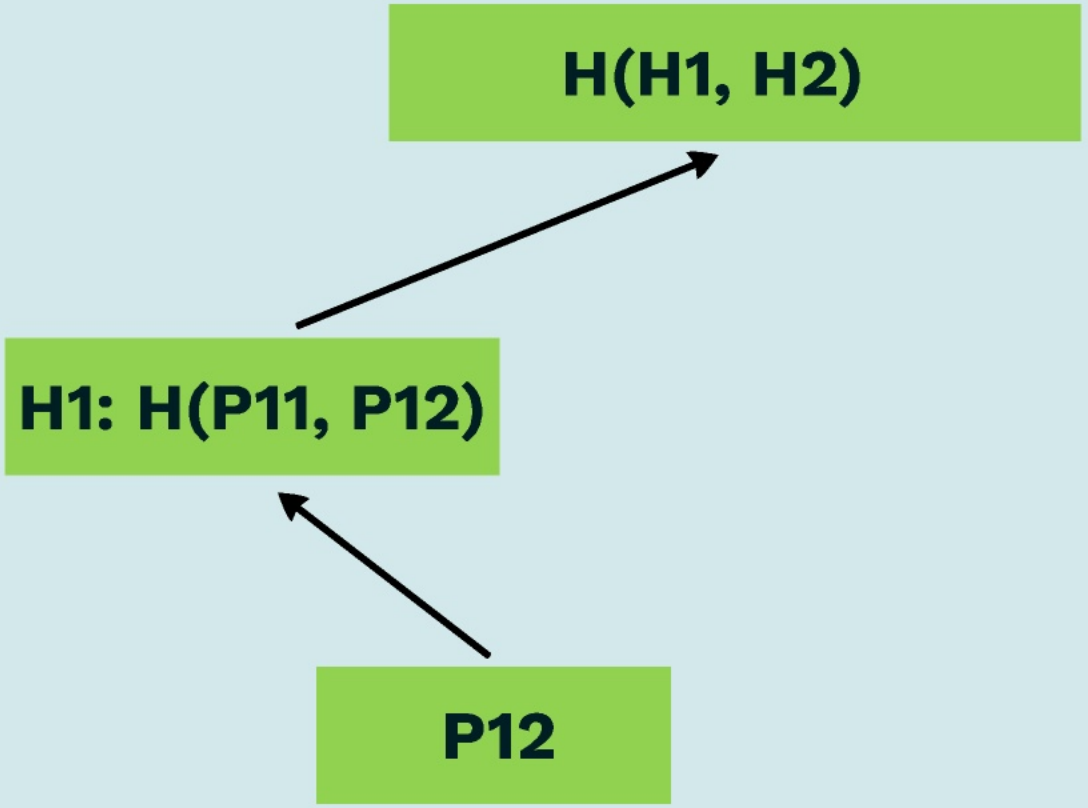

H1: H(P11, P12)
H2: H(P13, P14)
P11
P12
P13
P14


H(H1, H2)
H1: H(P11, P12)
H2: H(P13, P14)
P11
P12
P13
P14

FINALLY: H (Root)

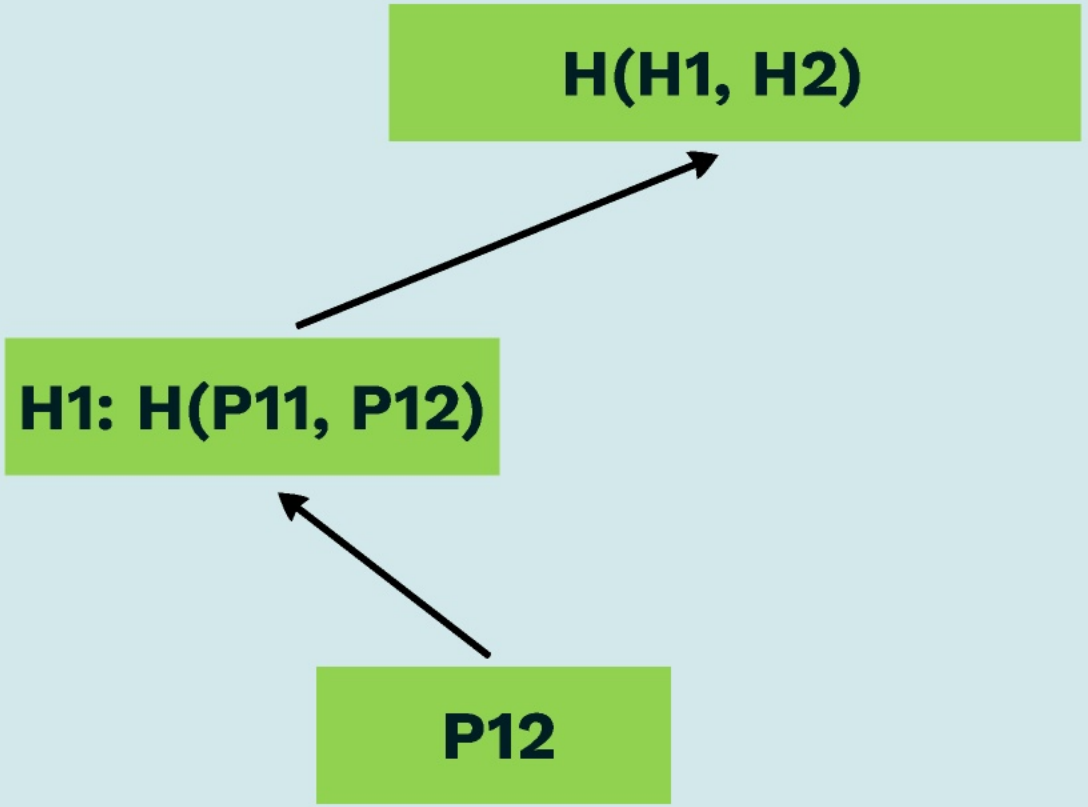


FINALLY: H (Root)

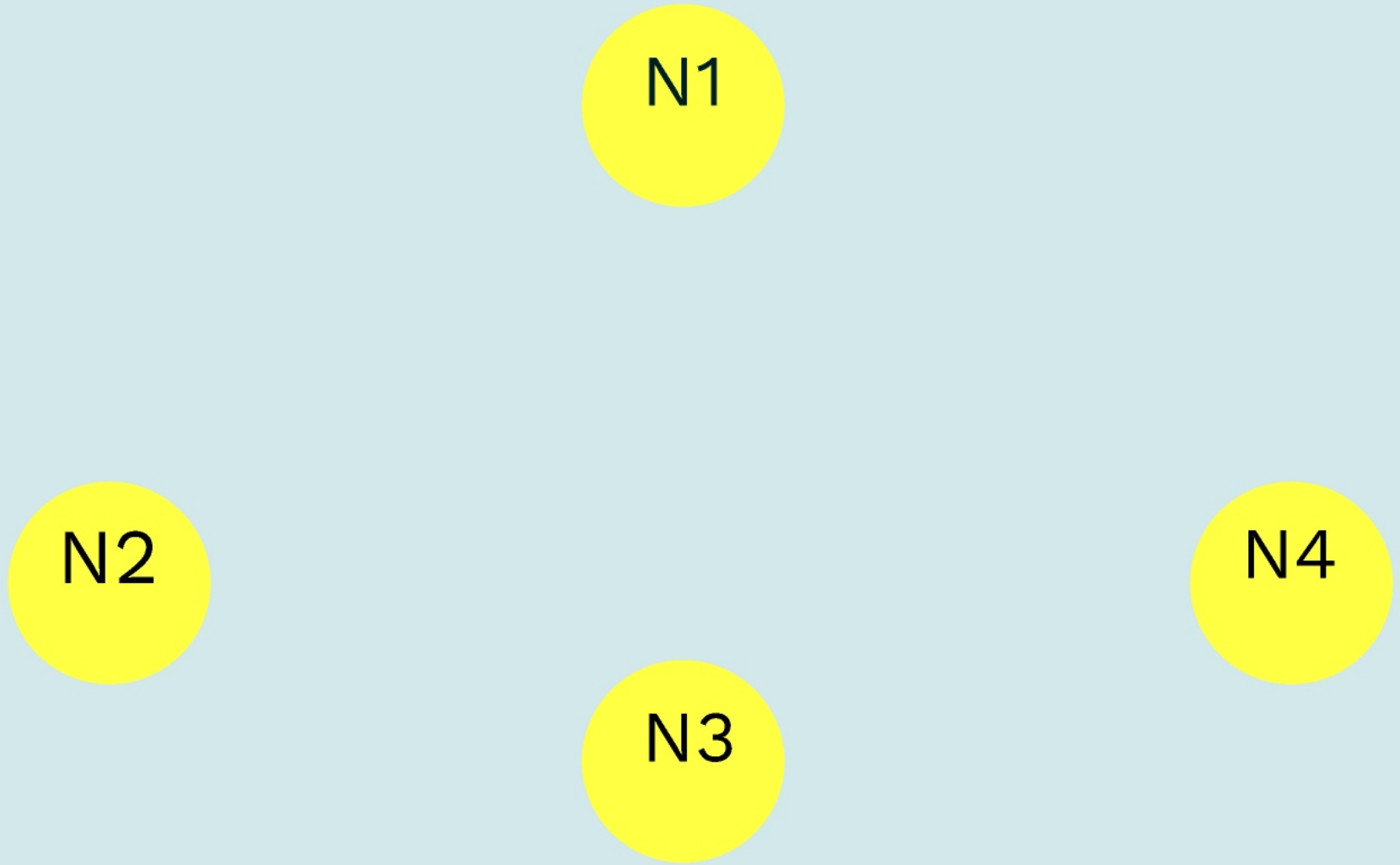


FINALLY: H (Root)

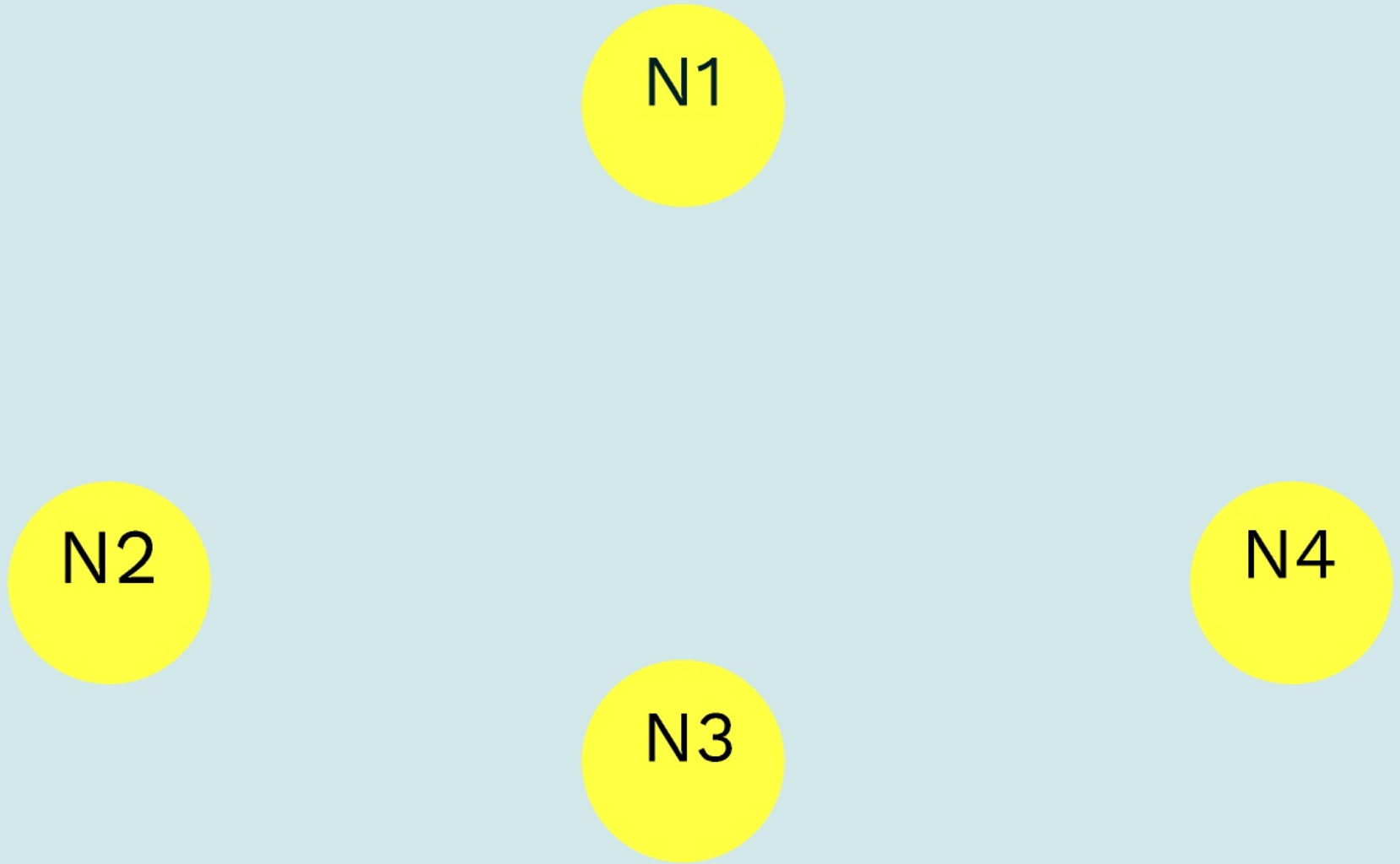
Branch: b2



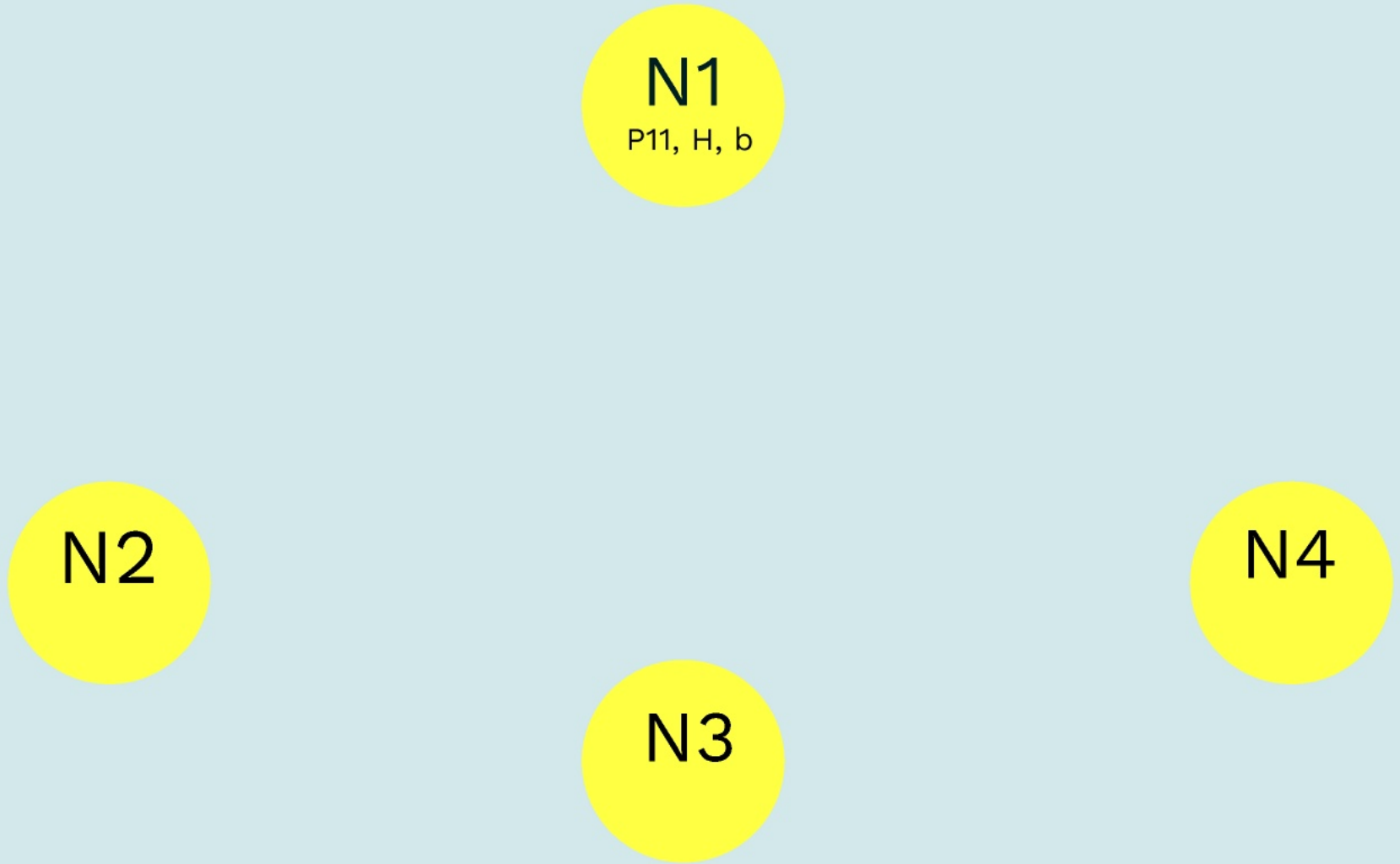
INITIAL:



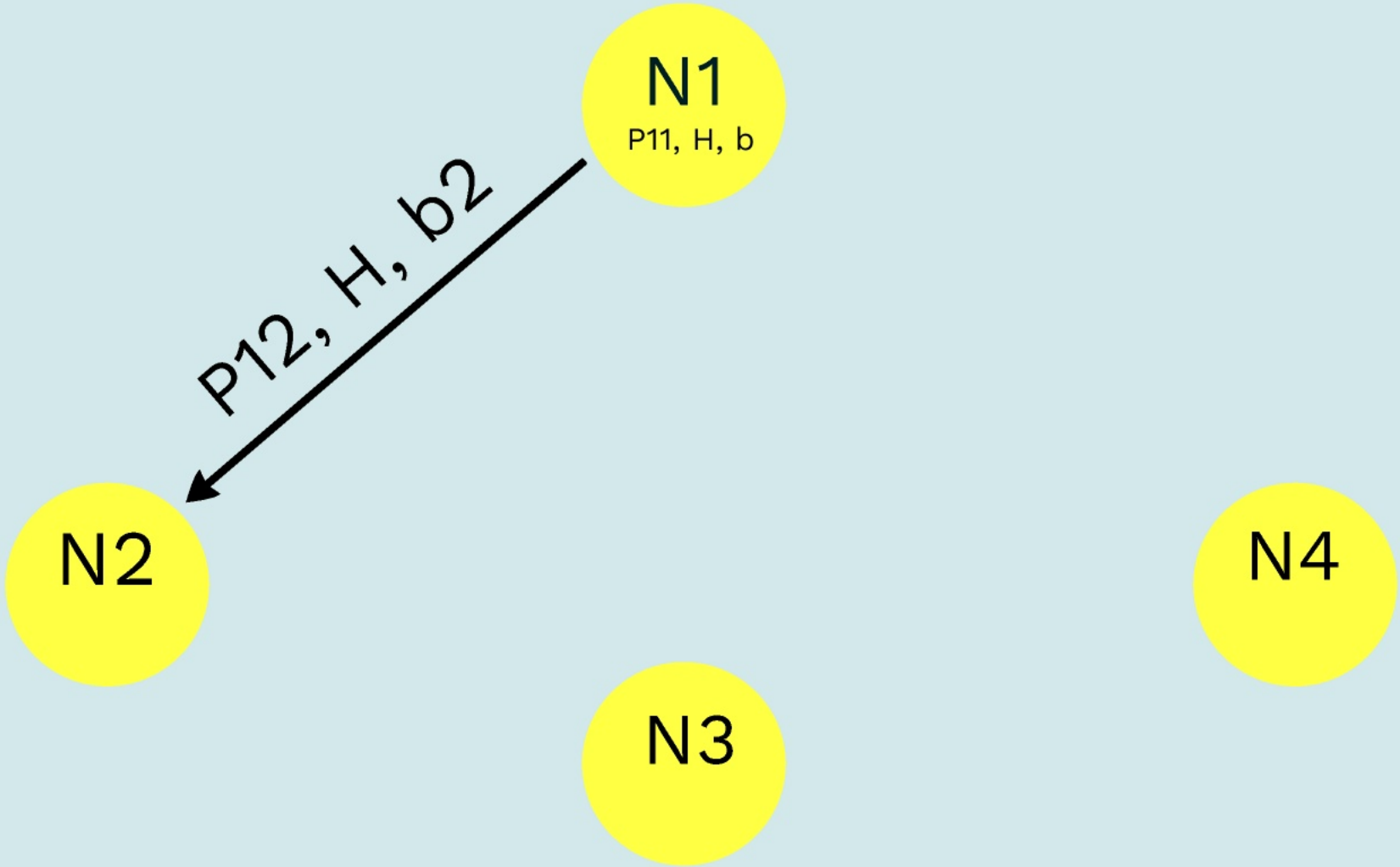
INITIAL: P11, P12, P13, P14, H, b



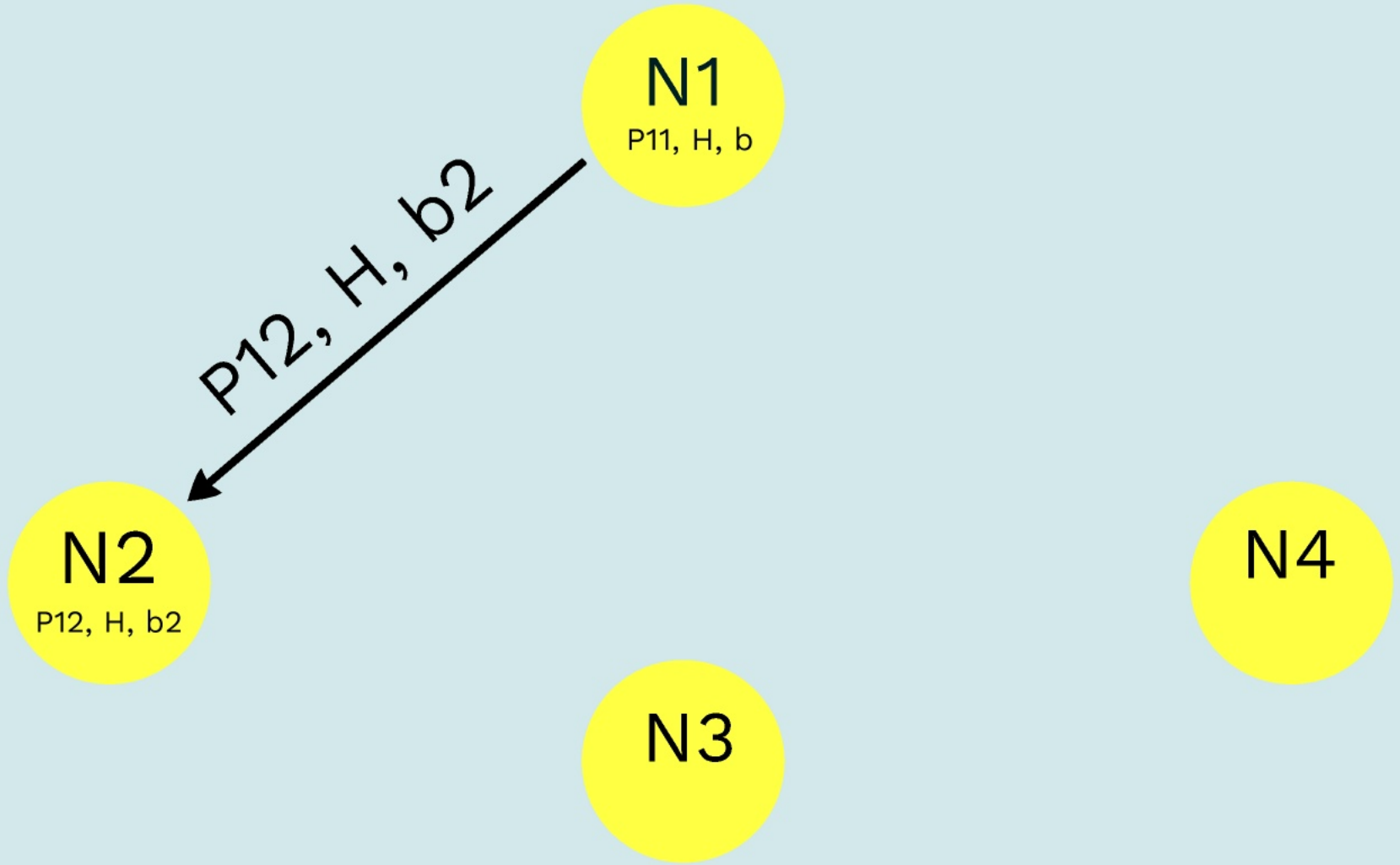
INITIAL: P11, P12, P13, P14, H, b



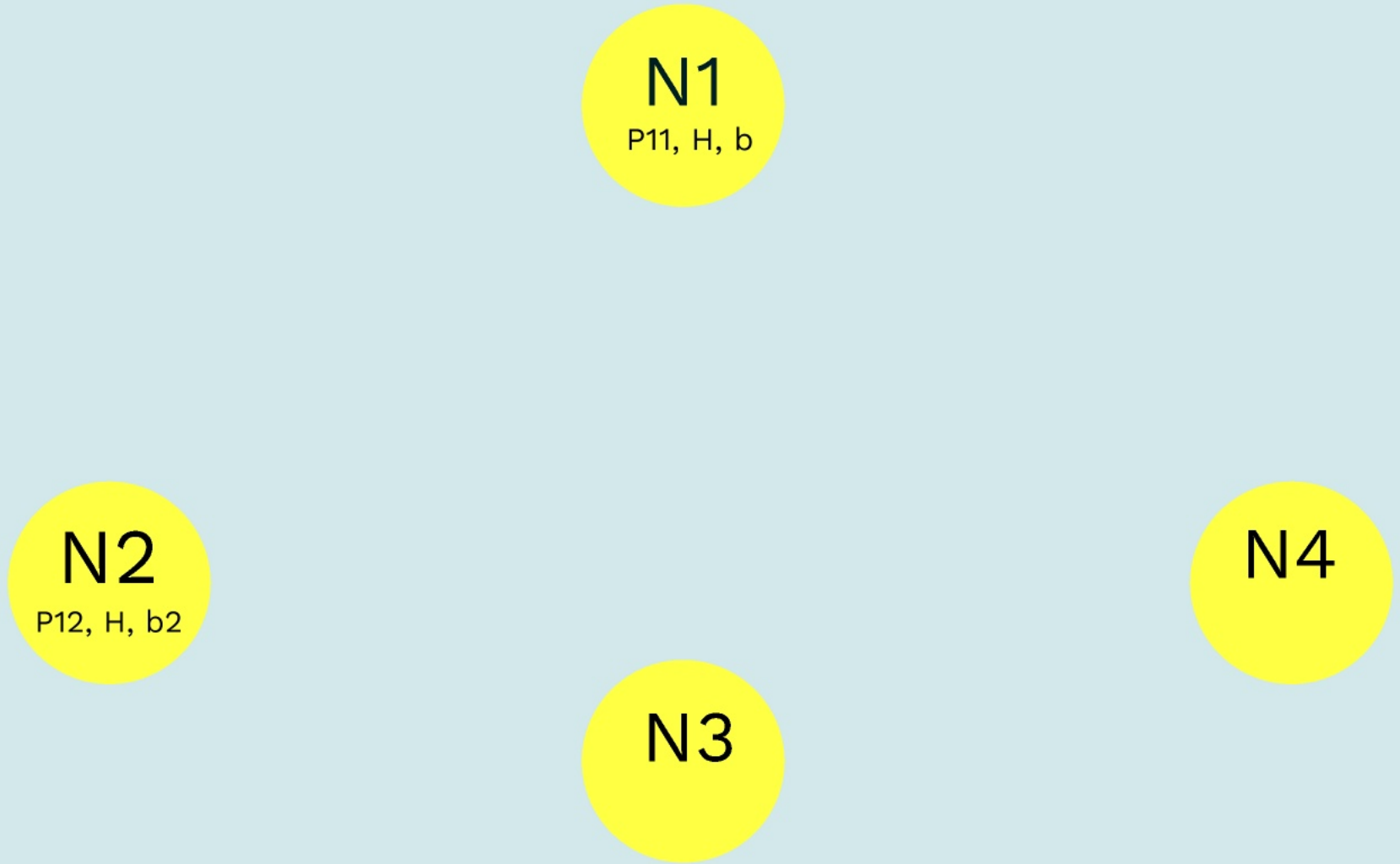
INITIAL: P11, P12, P13, P14, H, b



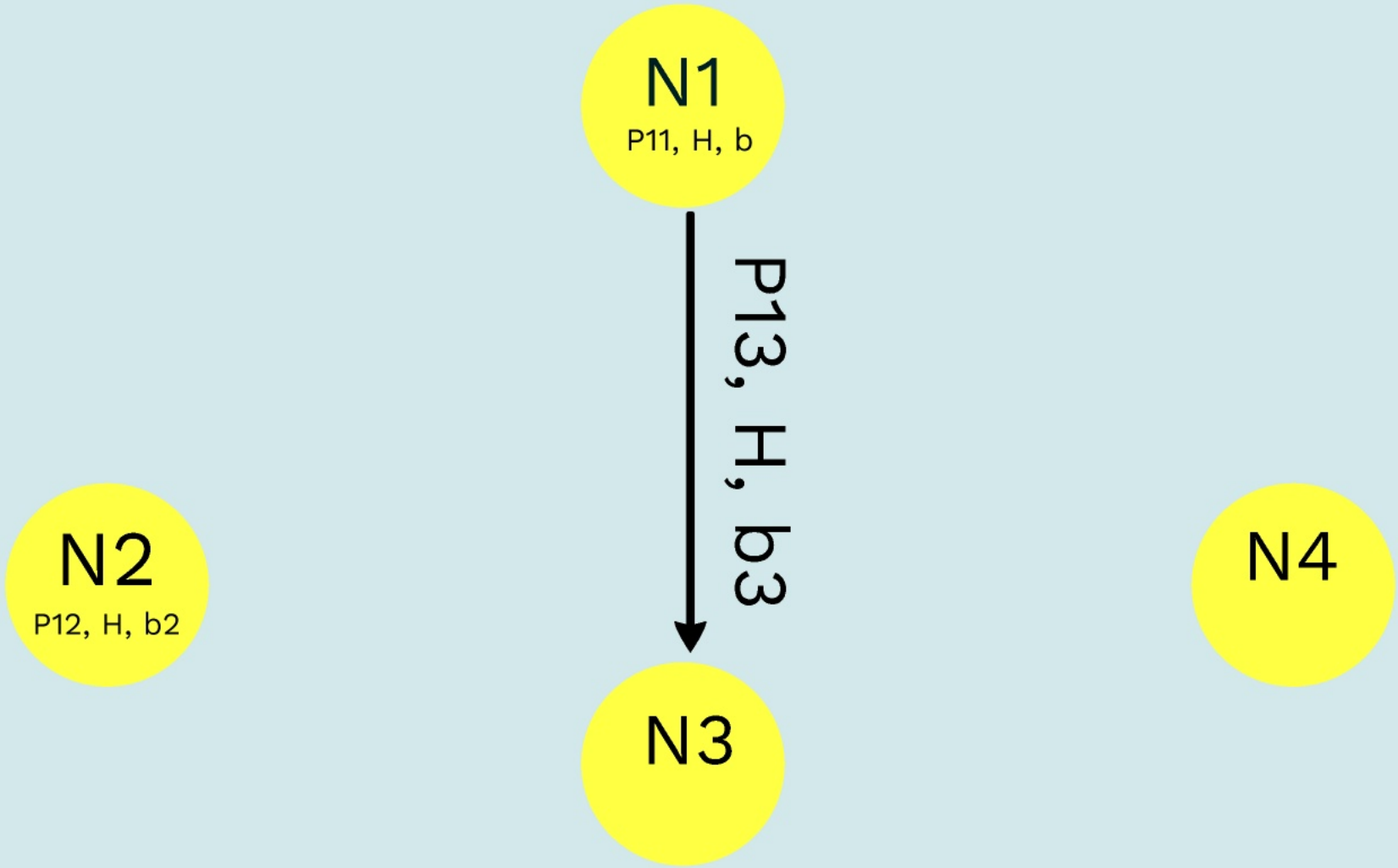
INITIAL: P11, P12, P13, P14, H, b



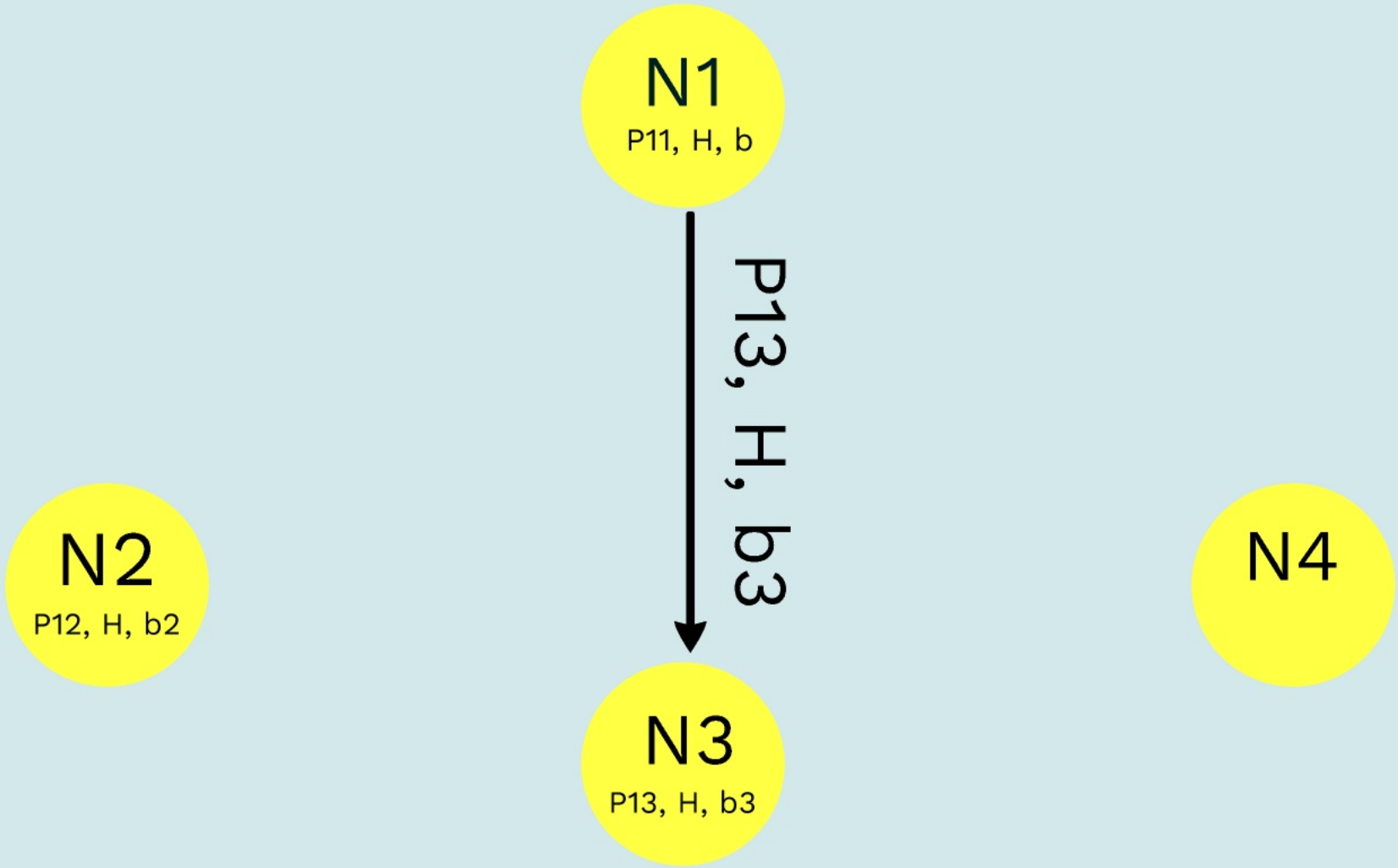
INITIAL: P11, P12, P13, P14, H, b



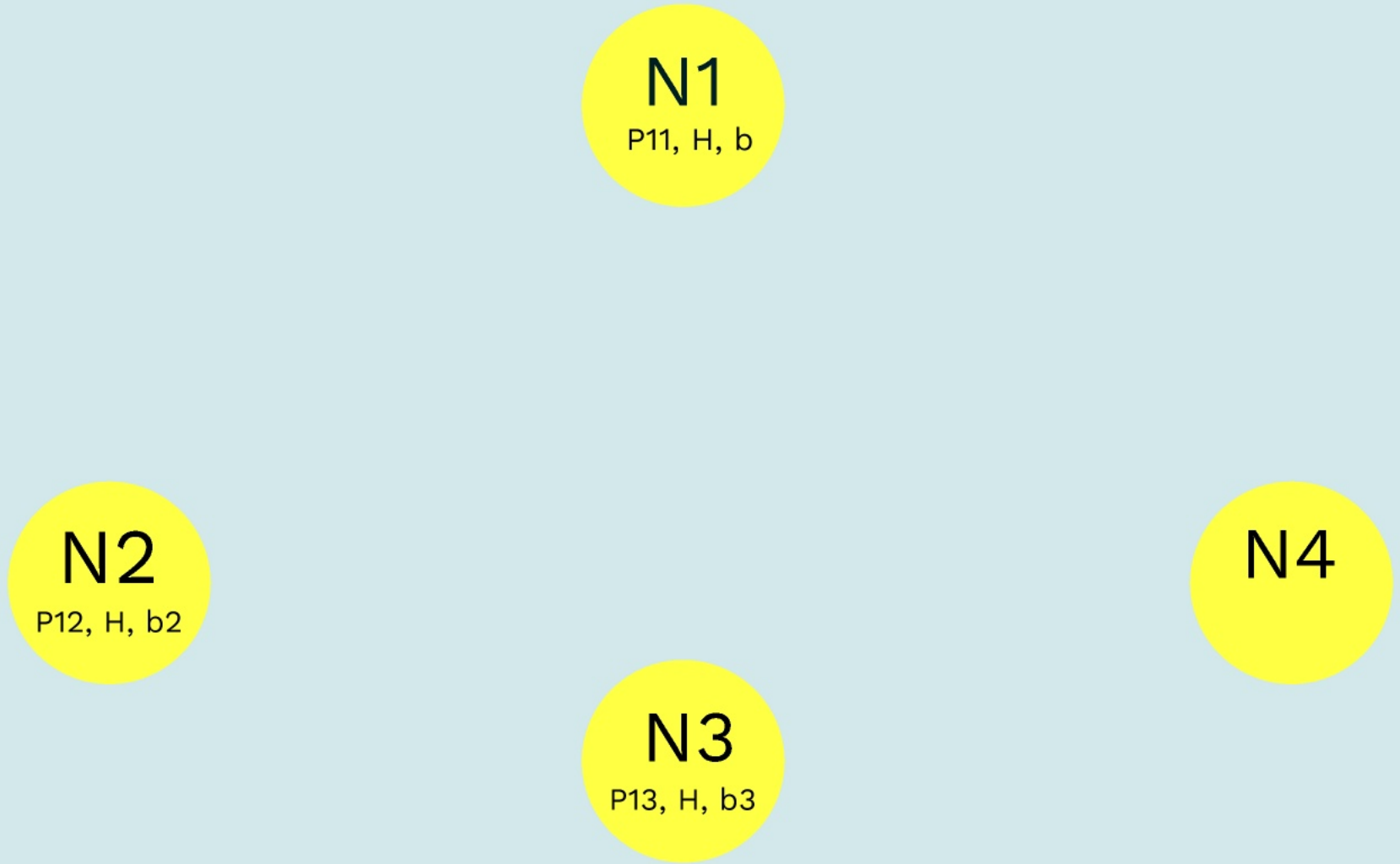
INITIAL: P11, P12, P13, P14, H, b



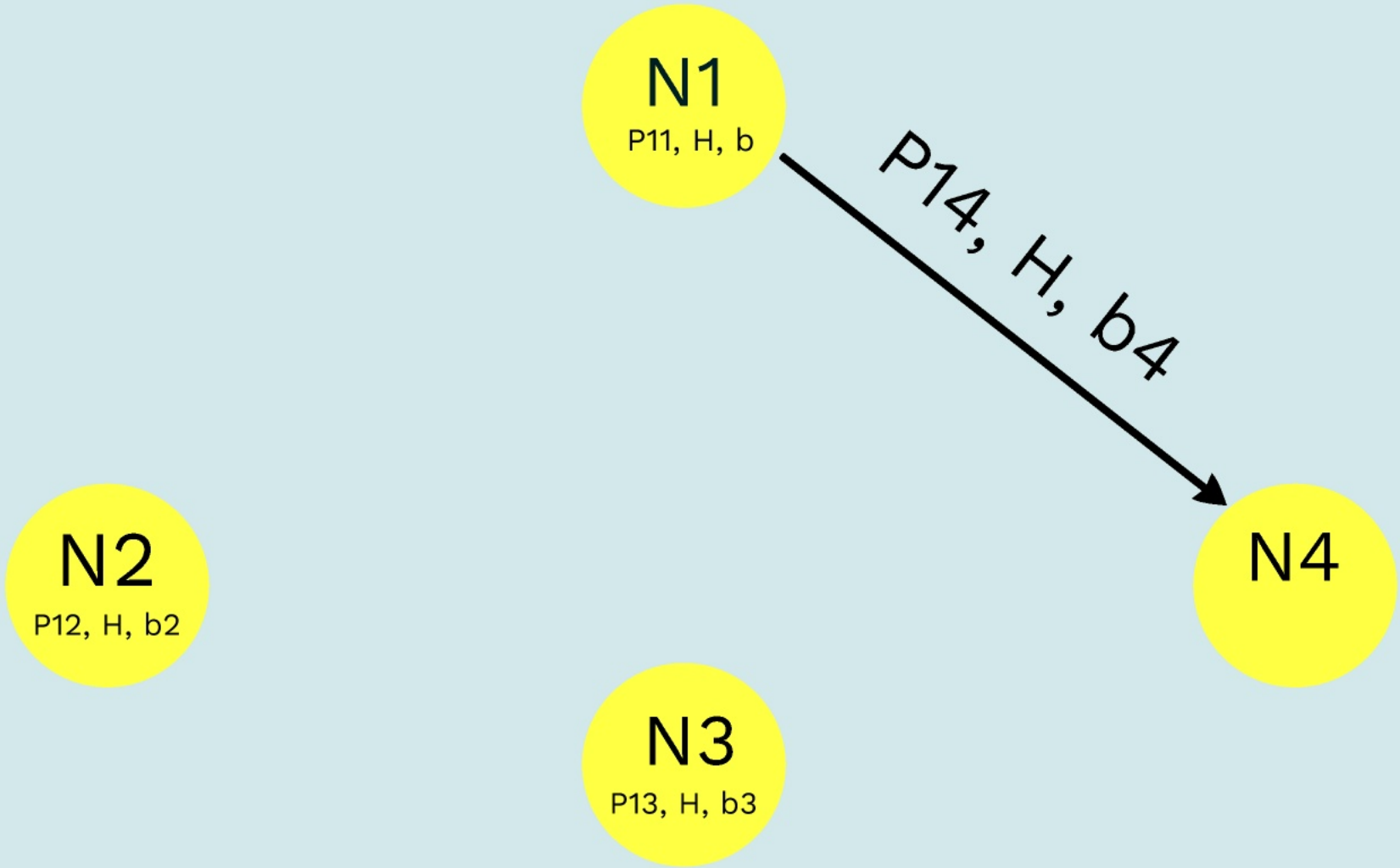
INITIAL: P11, P12, P13, P14, H, b



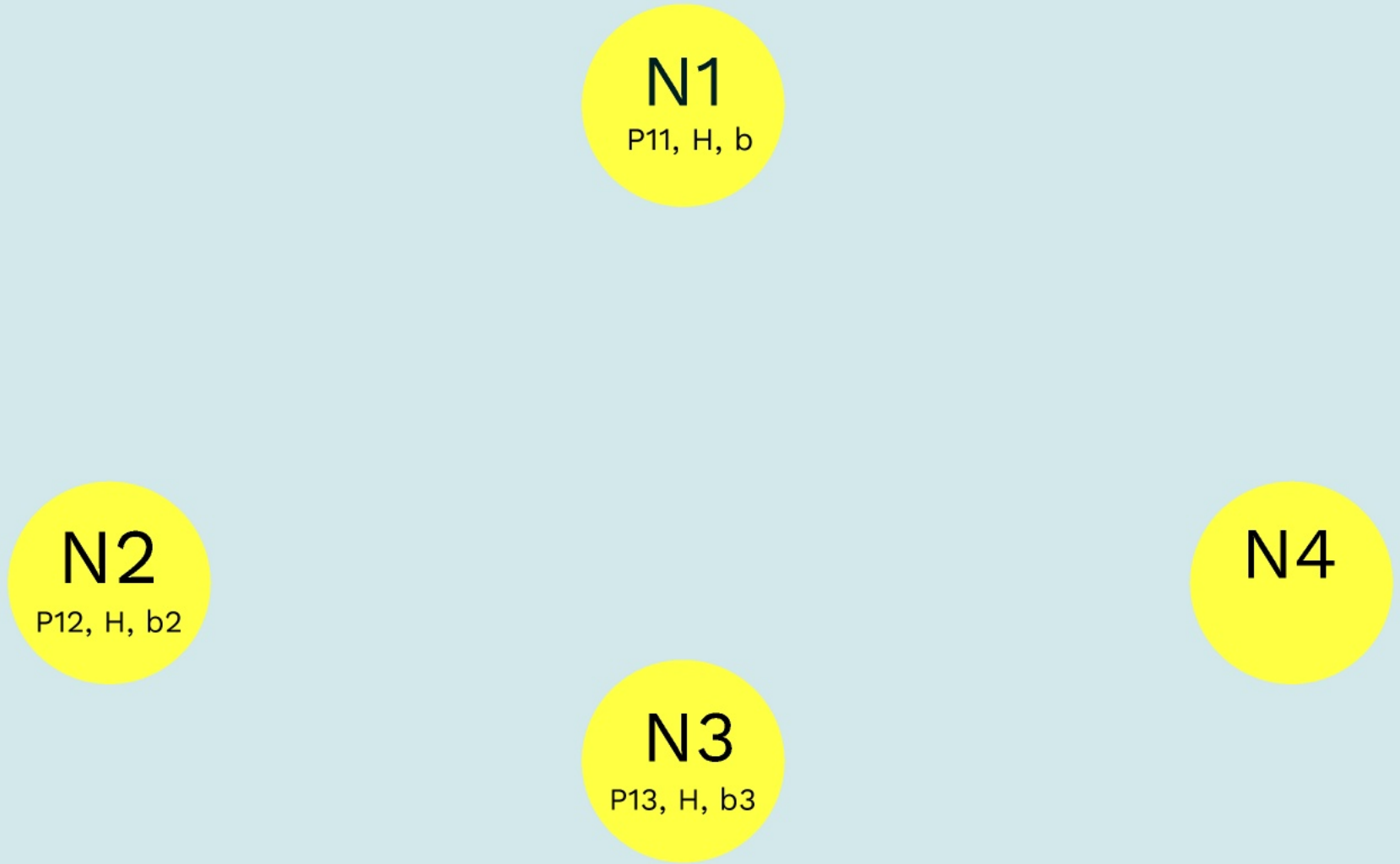
INITIAL: P11, P12, P13, P14, H, b



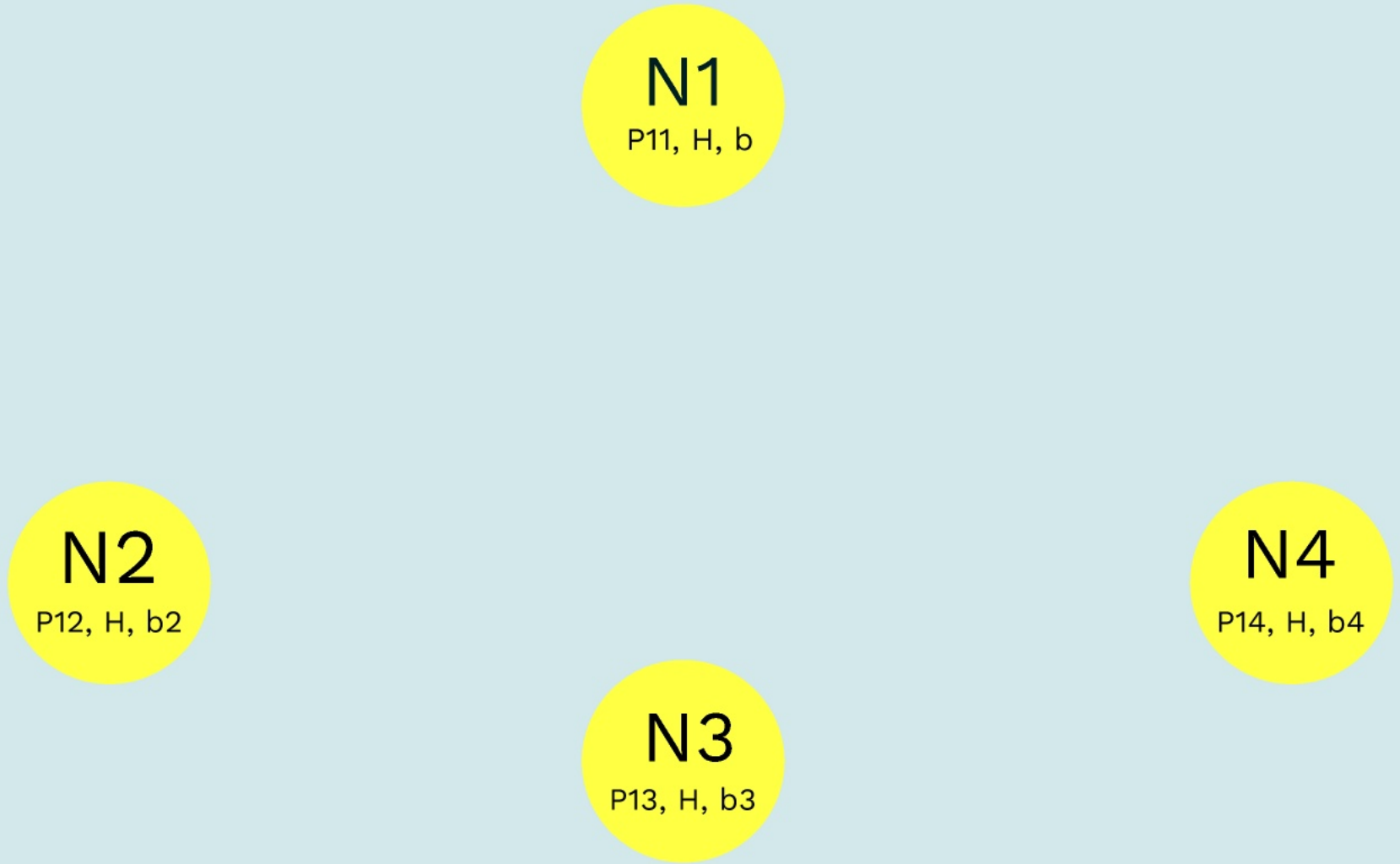
INITIAL: P11, P12, P13, P14, H, b



INITIAL: P11, P12, P13, P14, H, b

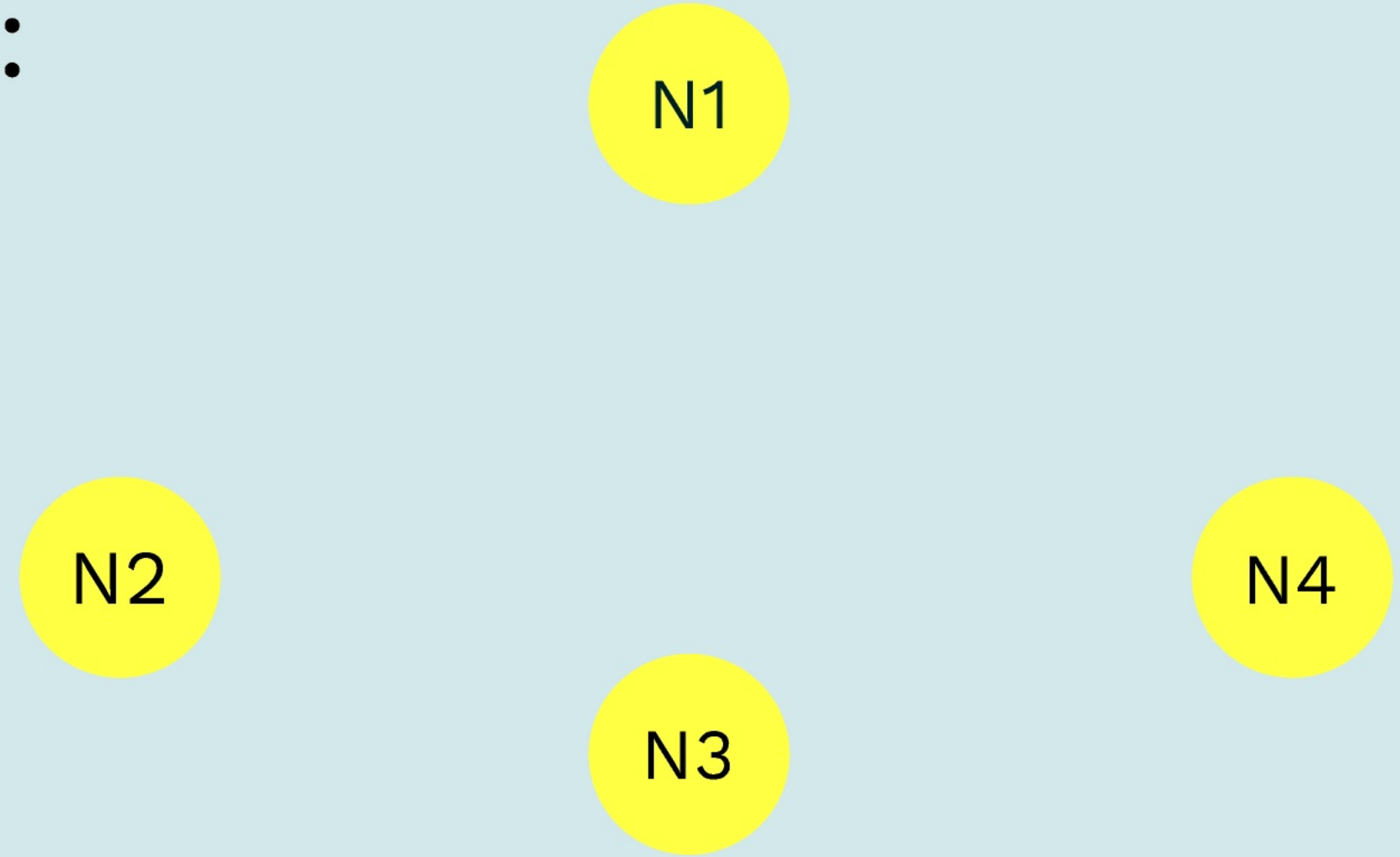


INITIAL: P11, P12, P13, P14, H, b



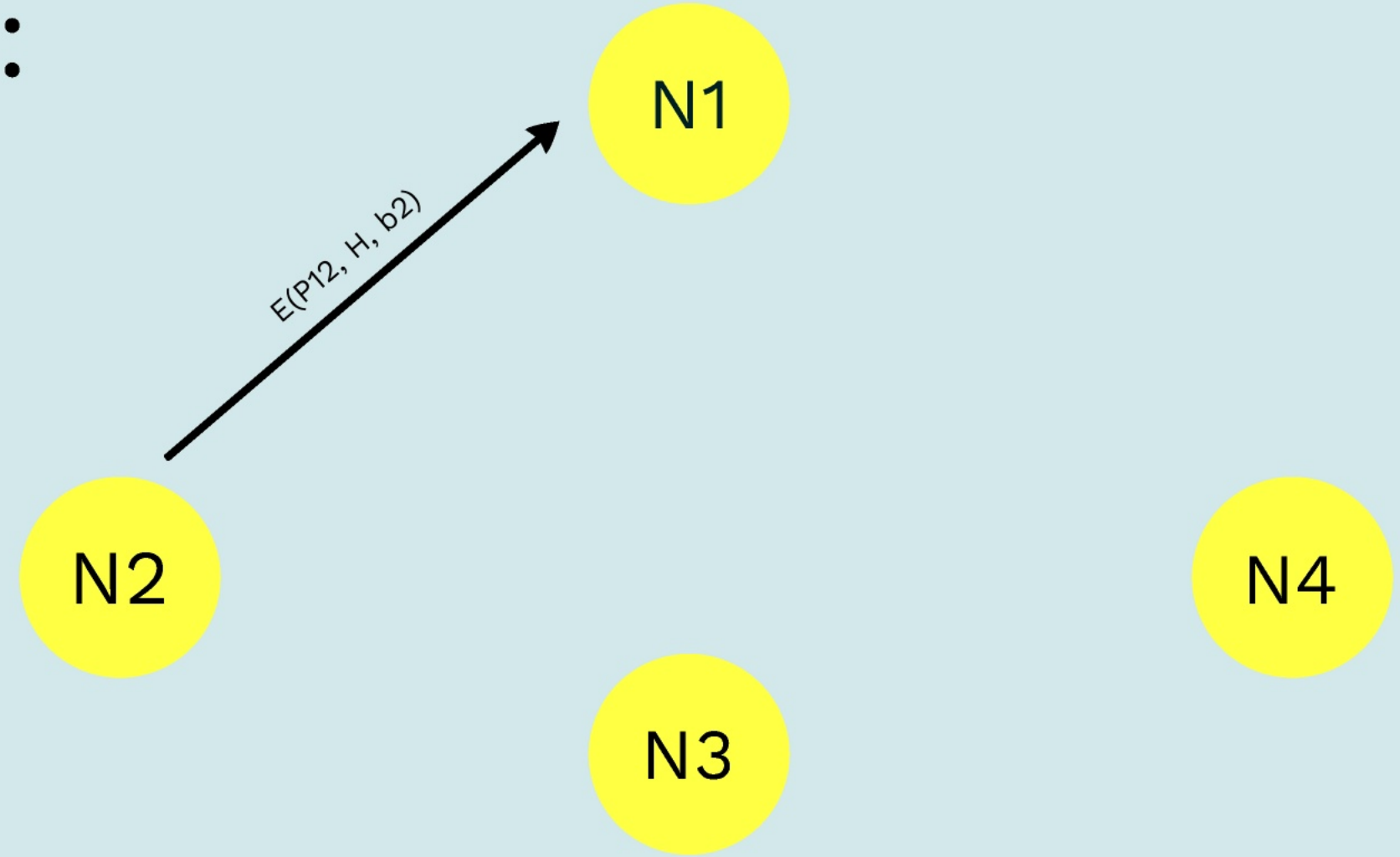
ECHO:

n-f



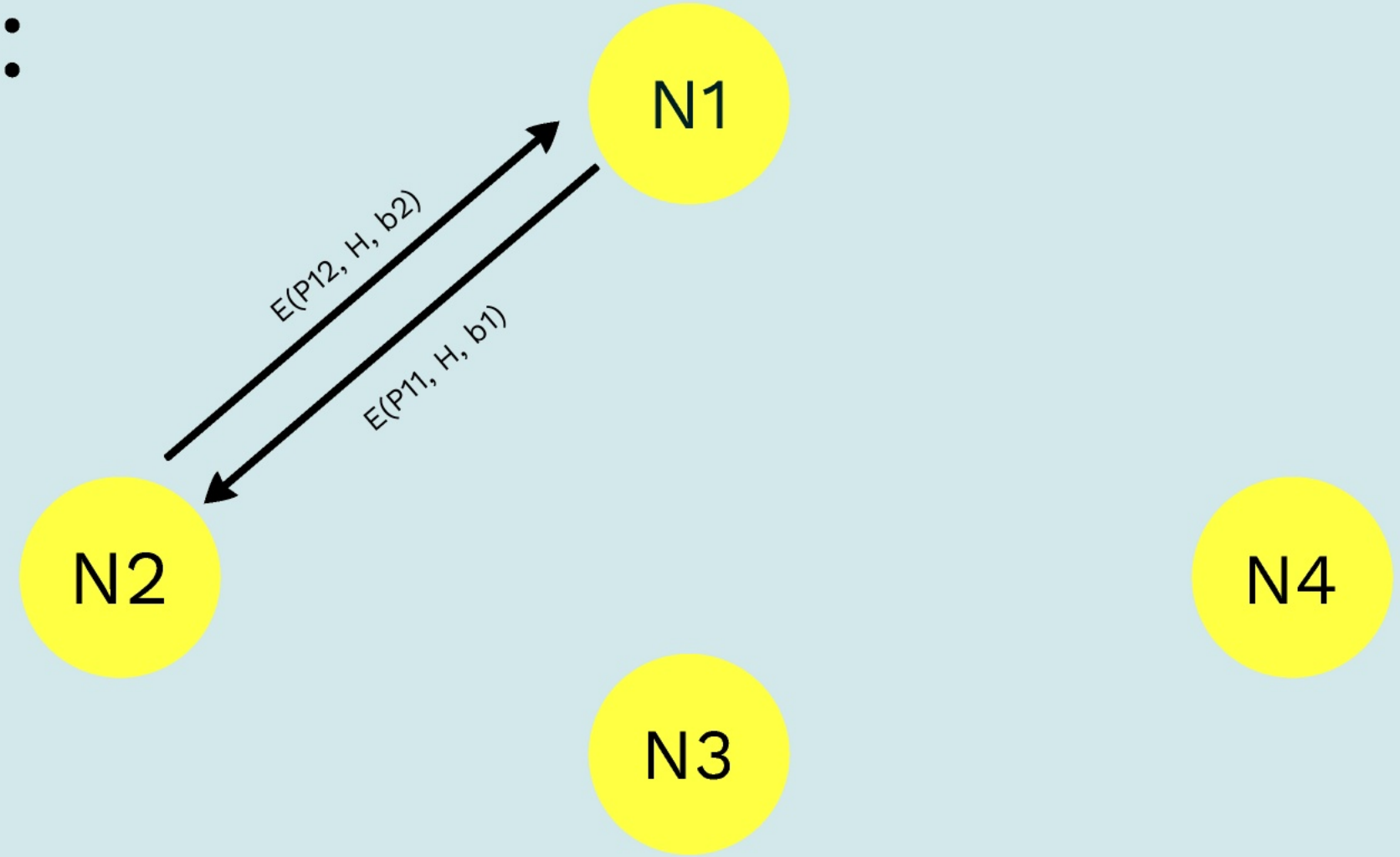
ECHO:

n-f



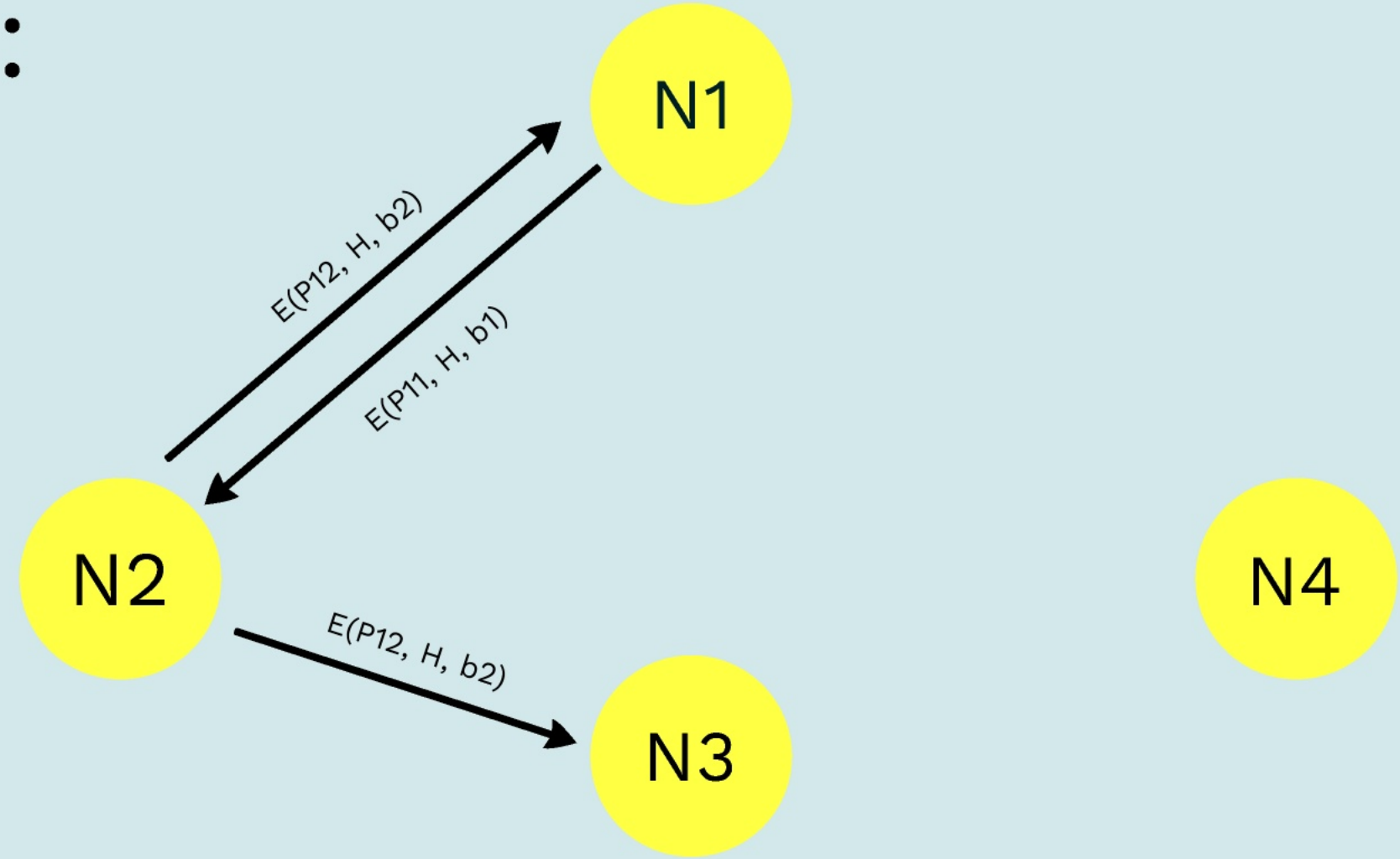
ECHO:

n-f



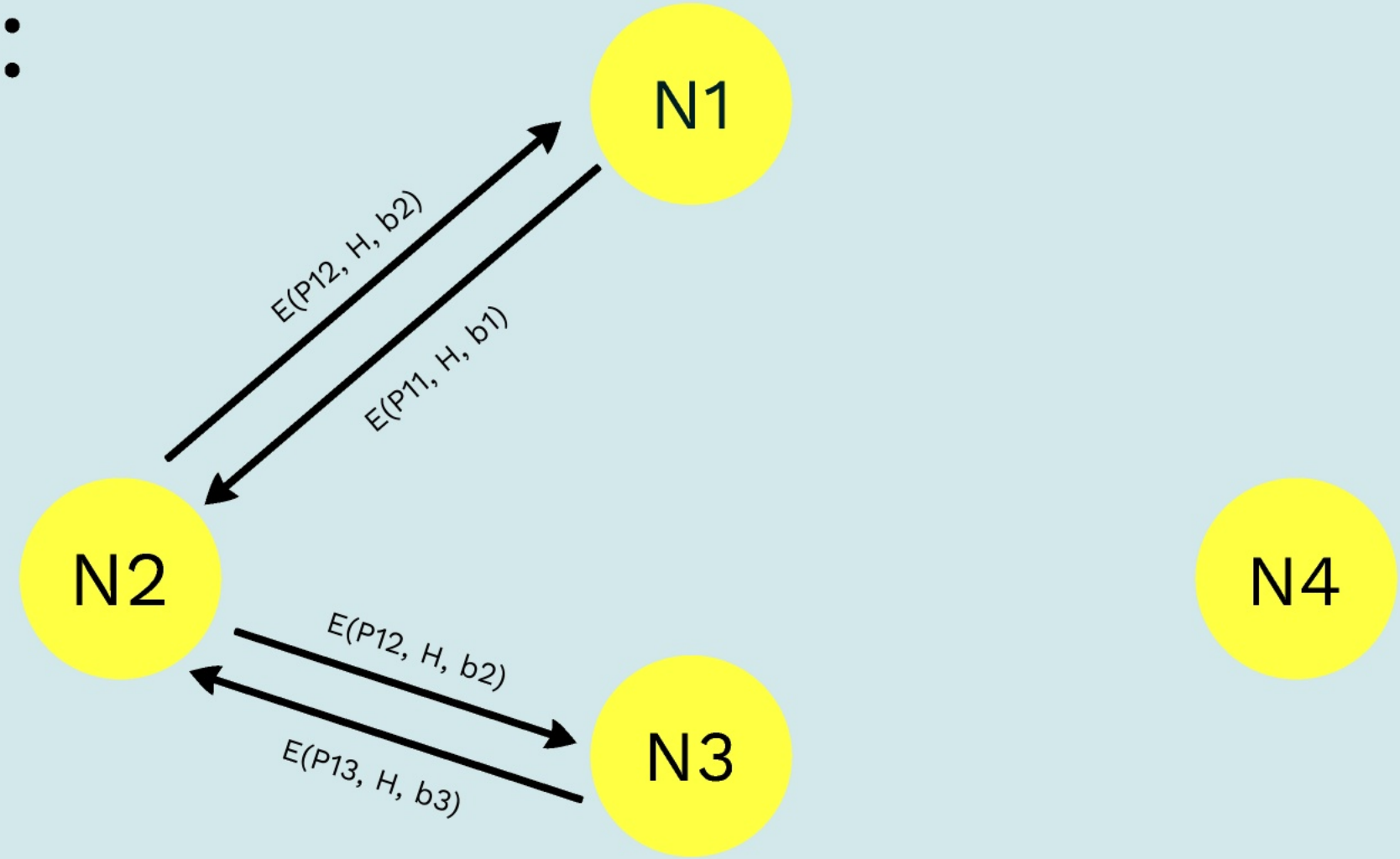
ECHO:

n-f



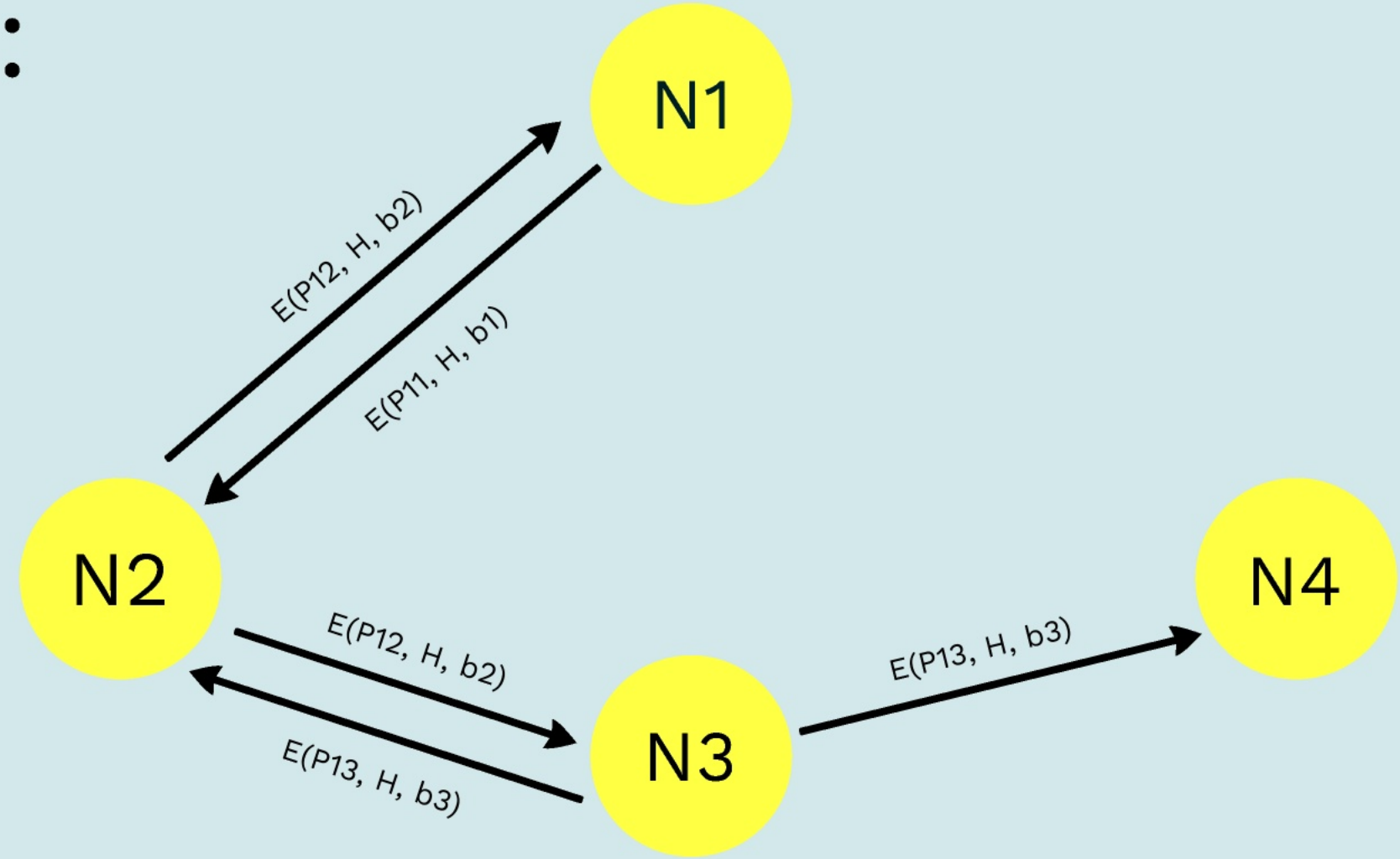
ECHO:

n-f



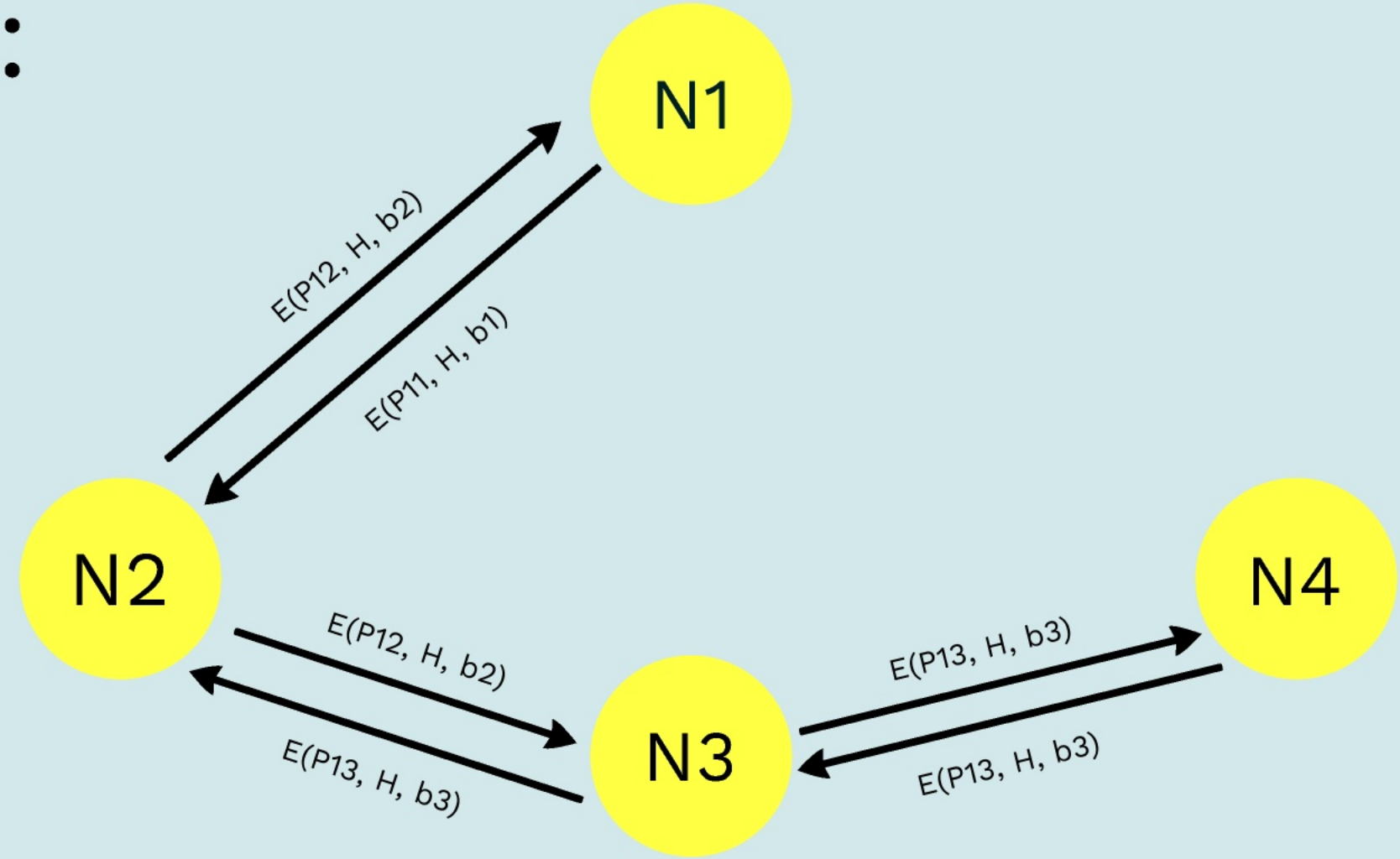
ECHO:

n-f



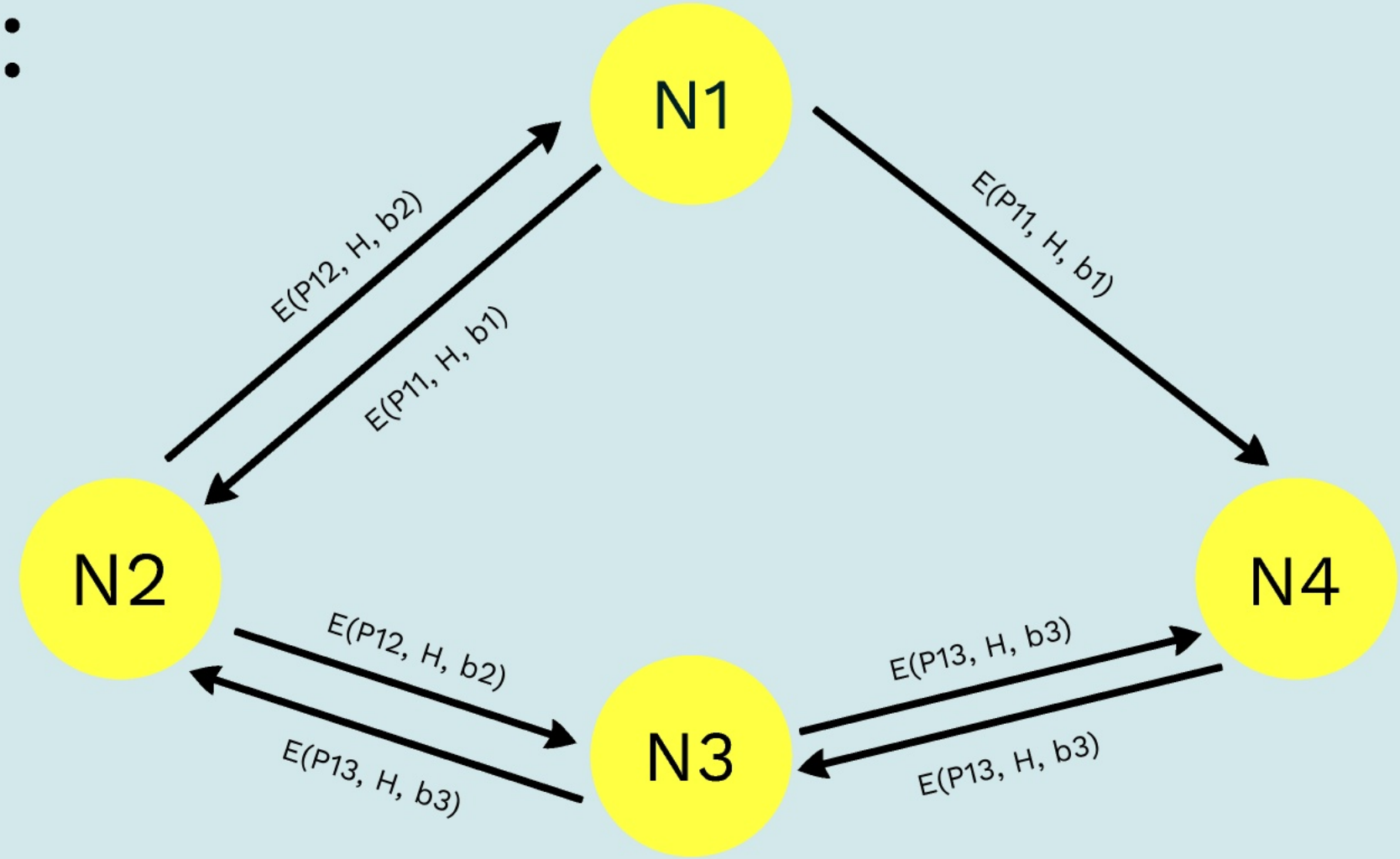
ECHO:

n-f


N1
N2
N3
N4
E(P12, H, b2)
E(P11, H, b1)
E(P12, H, b2)
E(P13, H, b3)
E(P13, H, b3)
E(P13, H, b3)

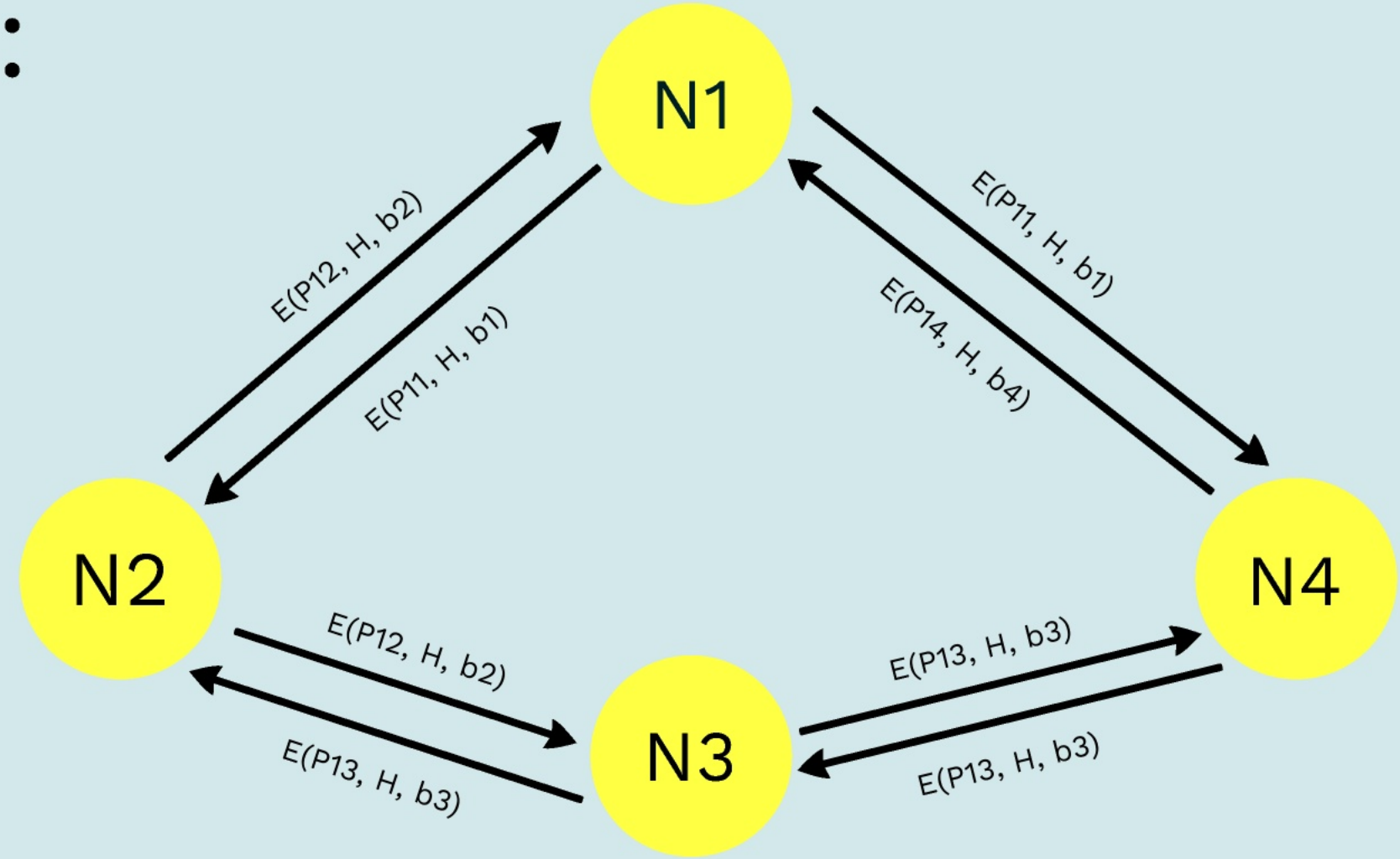
ECHO:

n-f


N1
N2
N3
N4
E(P12, H, b2)
E(P11, H, b1)
E(P11, H, b1)
E(P12, H, b2)
E(P13, H, b3)
E(P13, H, b3)
E(P13, H, b3)

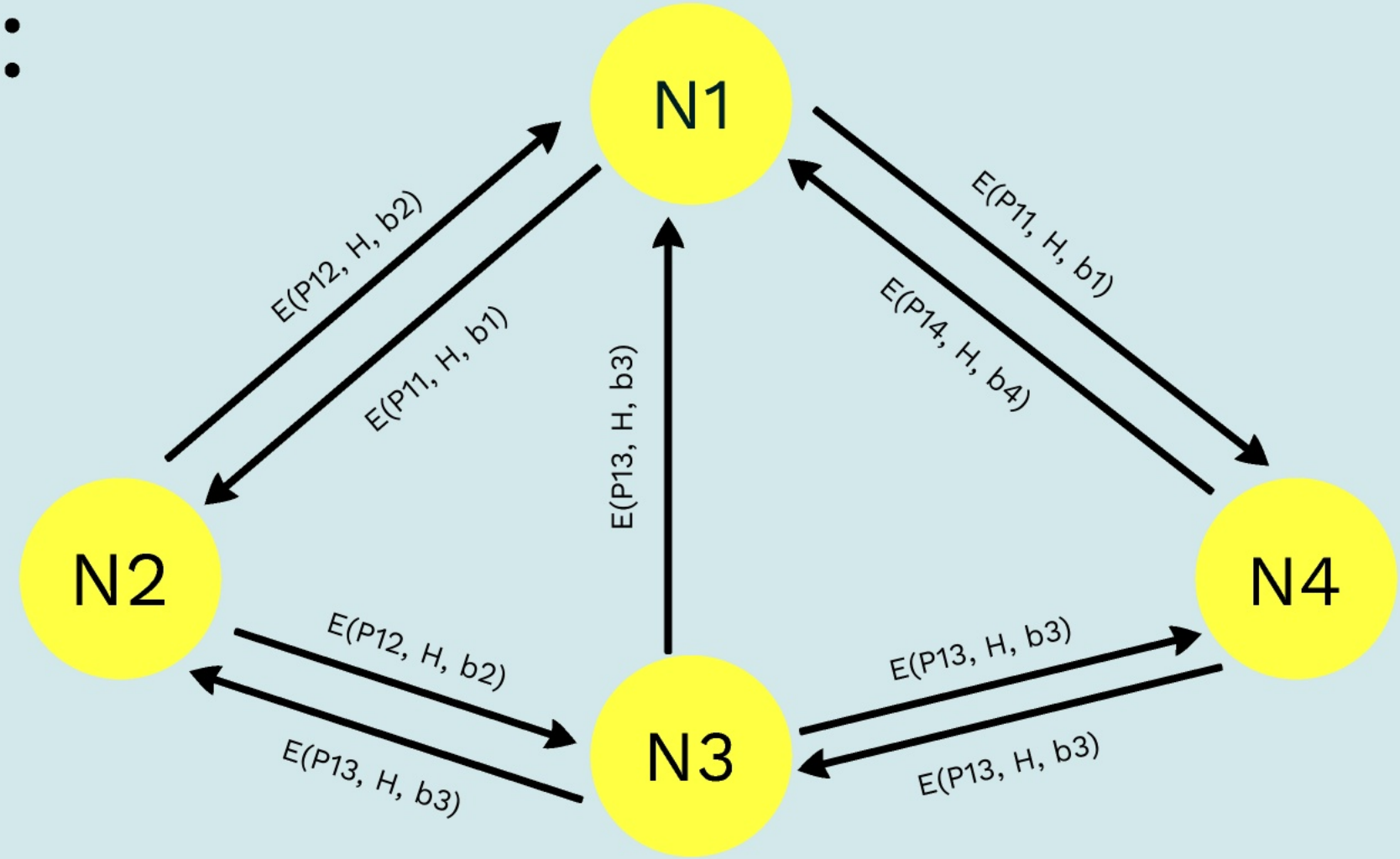
ECHO:

n-f


N1
N2
N3
N4
E(P12, H, b2)
E(P11, H, b1)
E(P11, H, b1)
E(P14, H, b4)
E(P12, H, b2)
E(P13, H, b3)
E(P13, H, b3)
E(P13, H, b3)

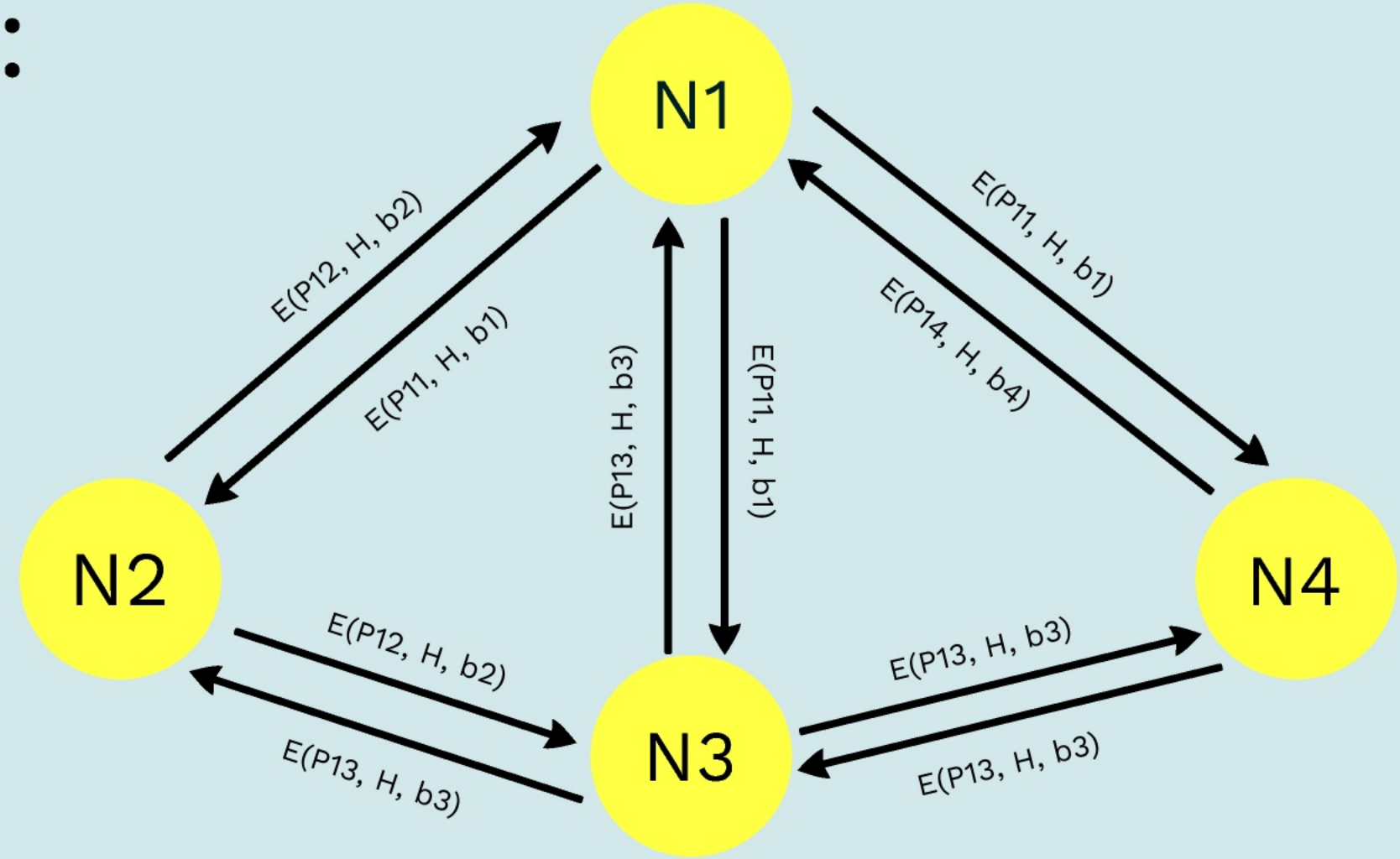
ECHO:

n-f


N1
N2
N3
N4
E(P12, H, b2)
E(P11, H, b1)
E(P11, H, b1)
E(P14, H, b4)
E(P13, H, b3)
E(P12, H, b2)
E(P13, H, b3)
E(P13, H, b3)
E(P13, H, b3)

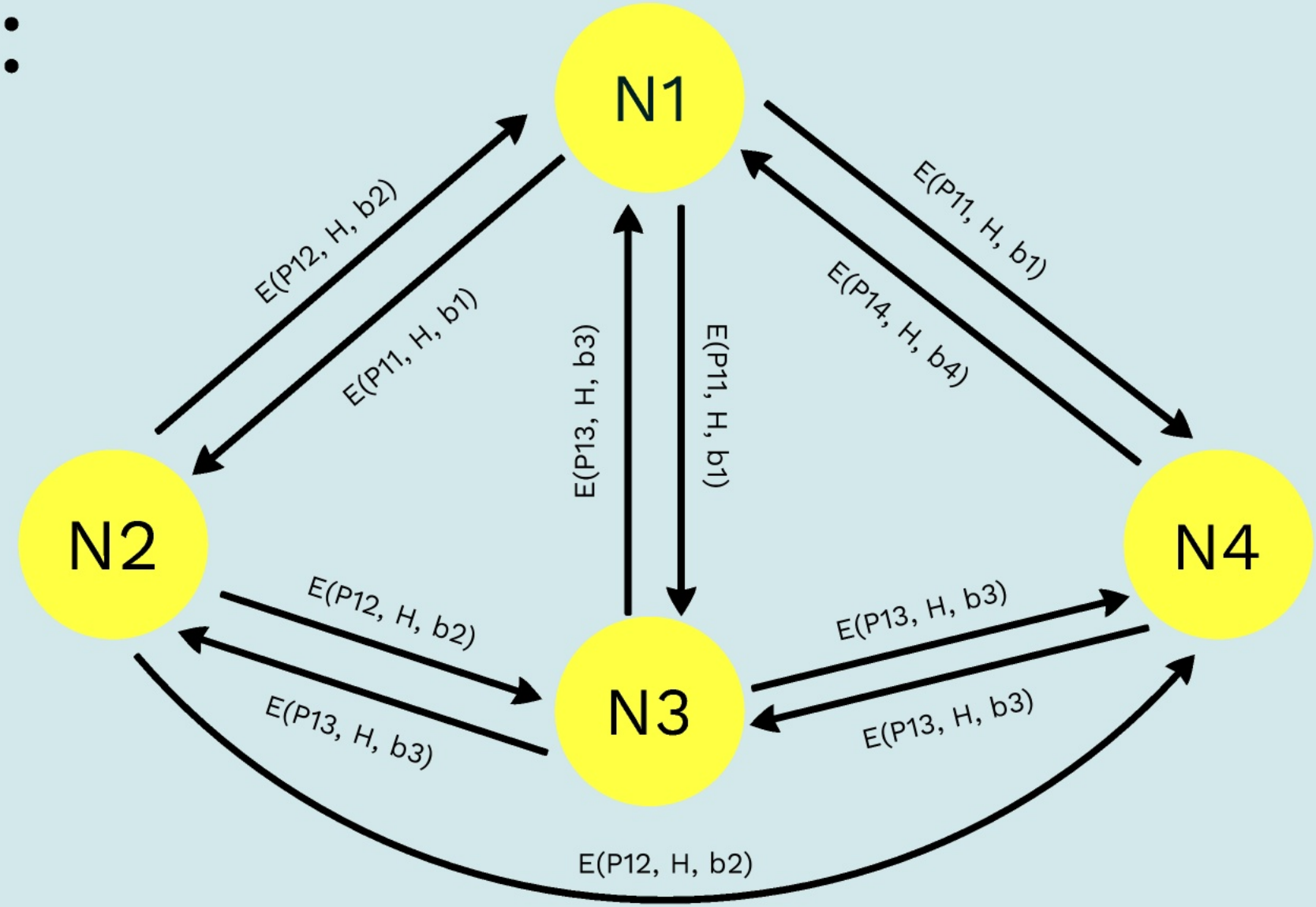
ECHO:

n-f


N1
N2
N3
N4
E(P12, H, b2)
E(P11, H, b1)
E(P11, H, b1)
E(P14, H, b4)
E(P13, H, b3)
E(P11, H, b1)
E(P12, H, b2)
E(P13, H, b3)
E(P13, H, b3)
E(P13, H, b3)

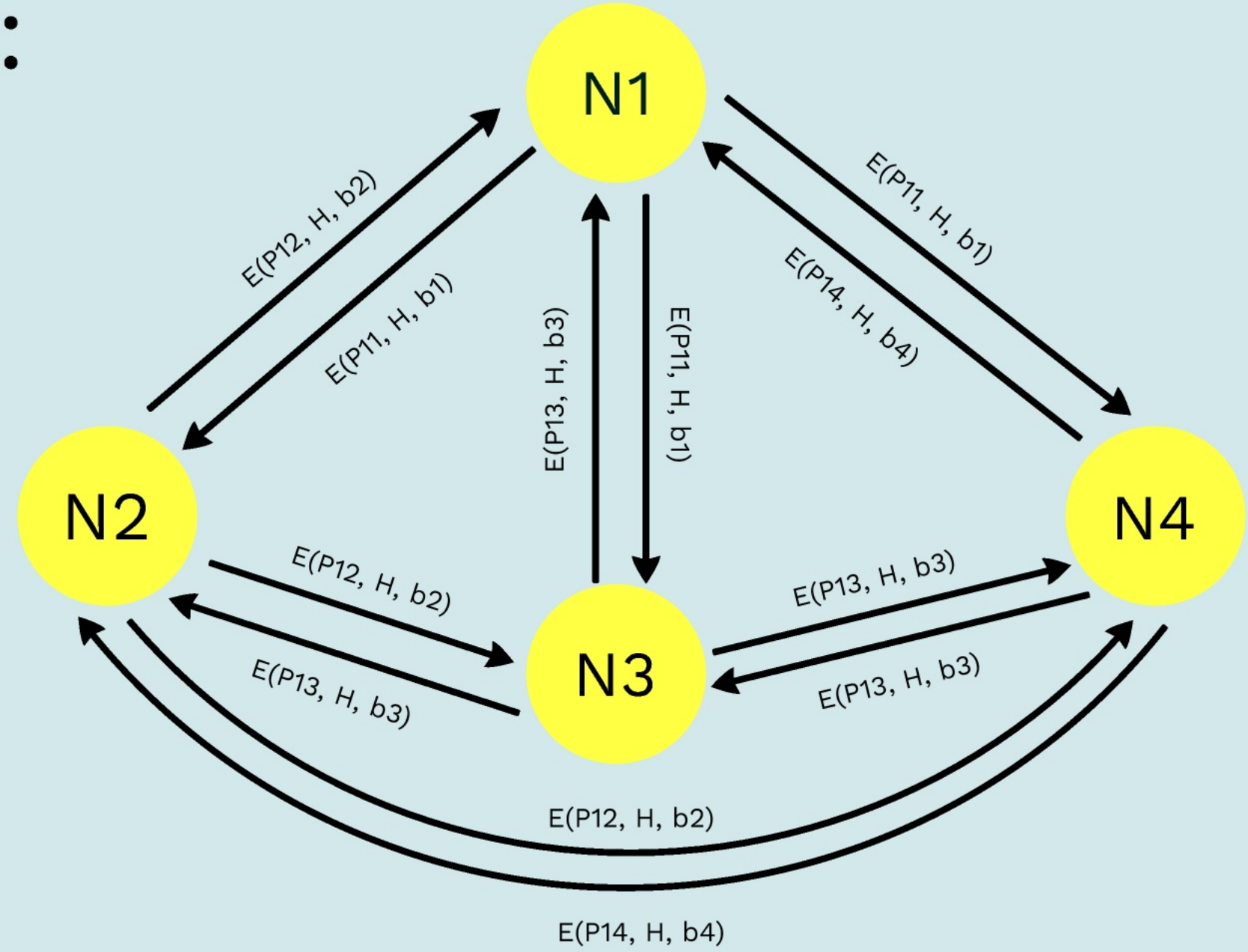
ECHO:

n-f


N1
N2
N3
N4
E(P12, H, b2)
E(P11, H, b1)
E(P13, H, b3)
E(P11, H, b1)
E(P11, H, b1)
E(P14, H, b4)
E(P12, H, b2)
E(P13, H, b3)
E(P13, H, b3)
E(P13, H, b3)
E(P12, H, b2)

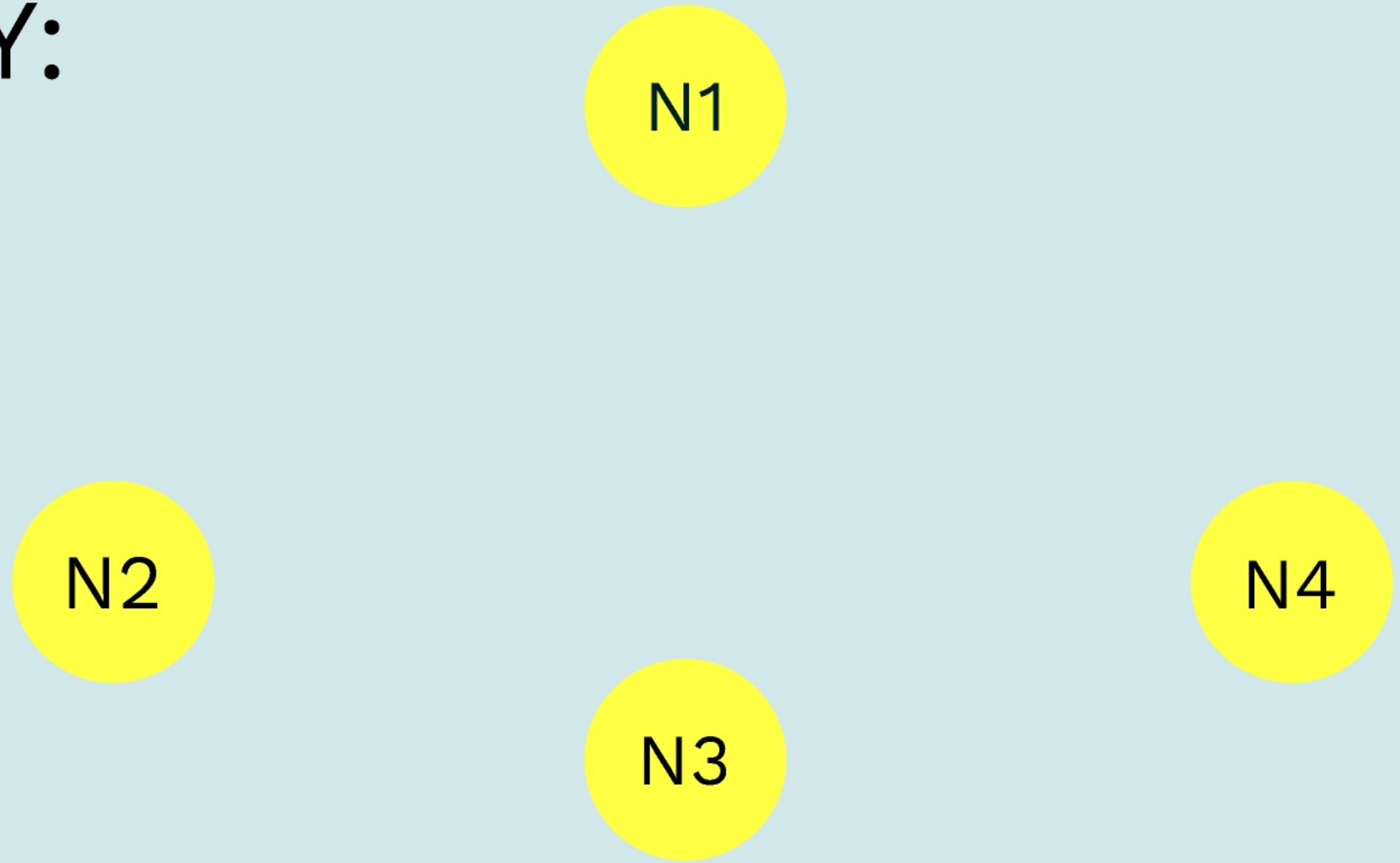
ECHO:

n-f


N1
N2
N3
N4
E(P12, H, b2)
E(P11, H, b1)
E(P13, H, b3)
E(P11, H, b1)
E(P11, H, b1)
E(P14, H, b4)
E(P12, H, b2)
E(P13, H, b3)
E(P13, H, b3)
E(P13, H, b3)
E(P12, H, b2)
E(P14, H, b4)

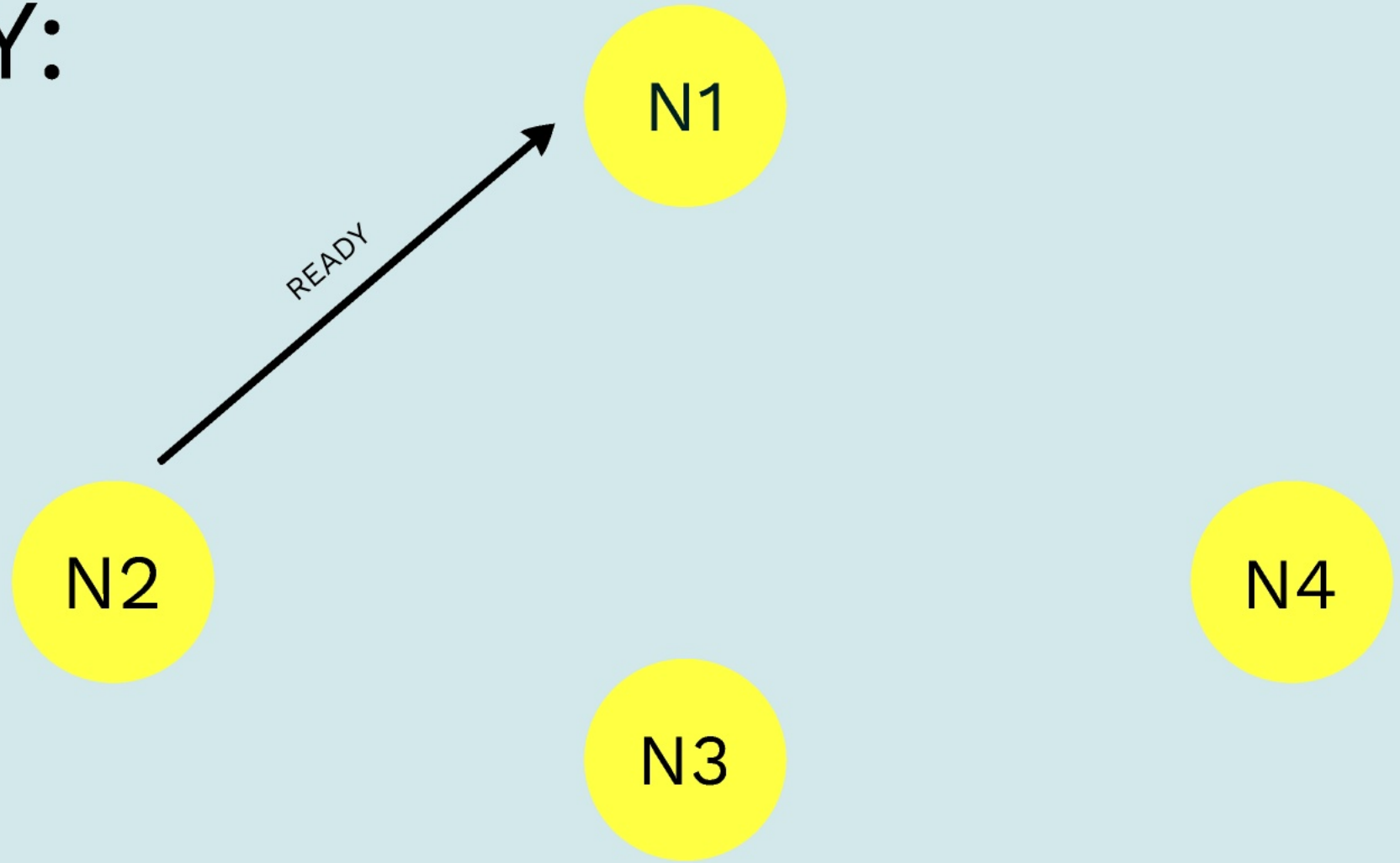
READY:

2f+1



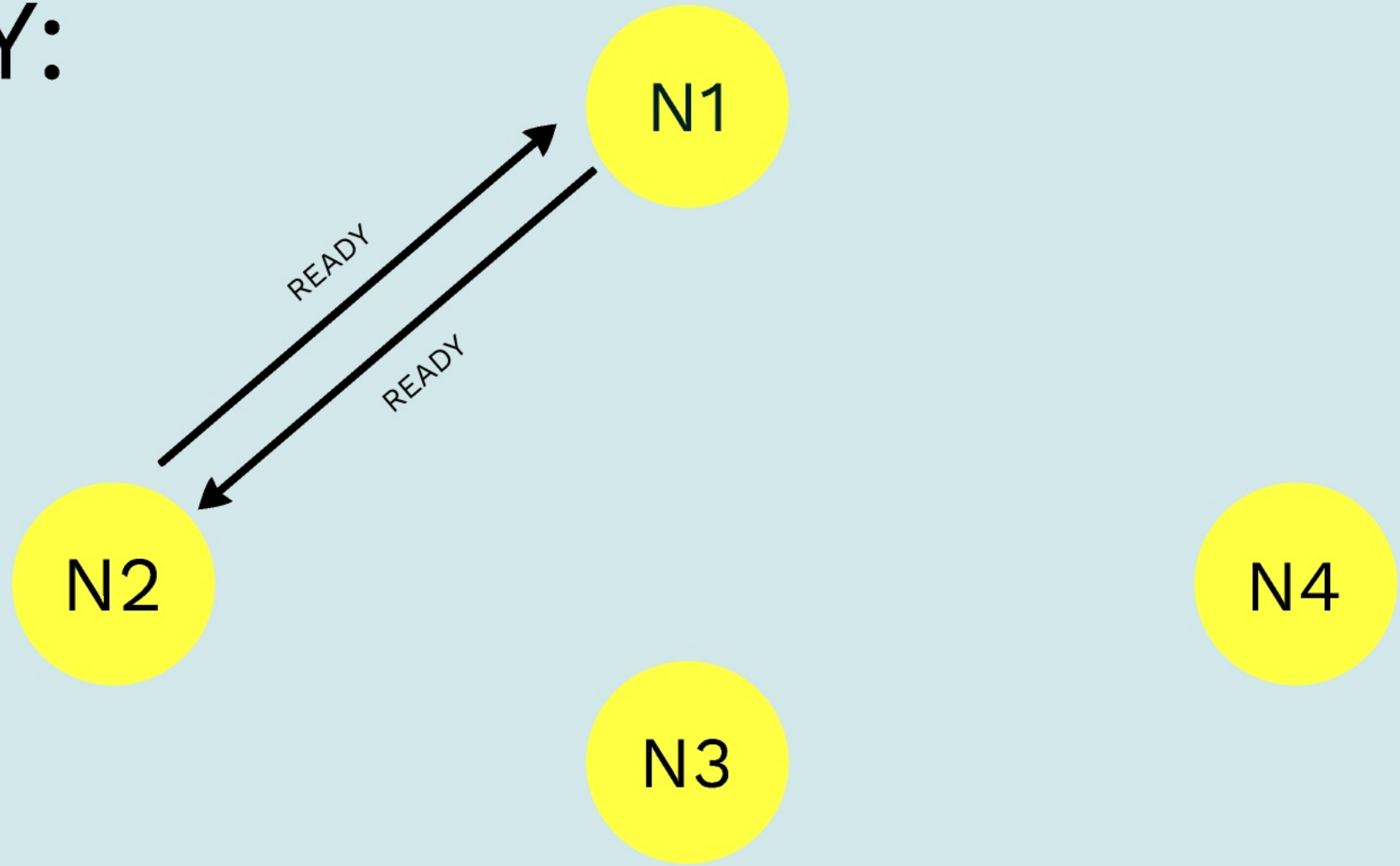
READY:

2f+1



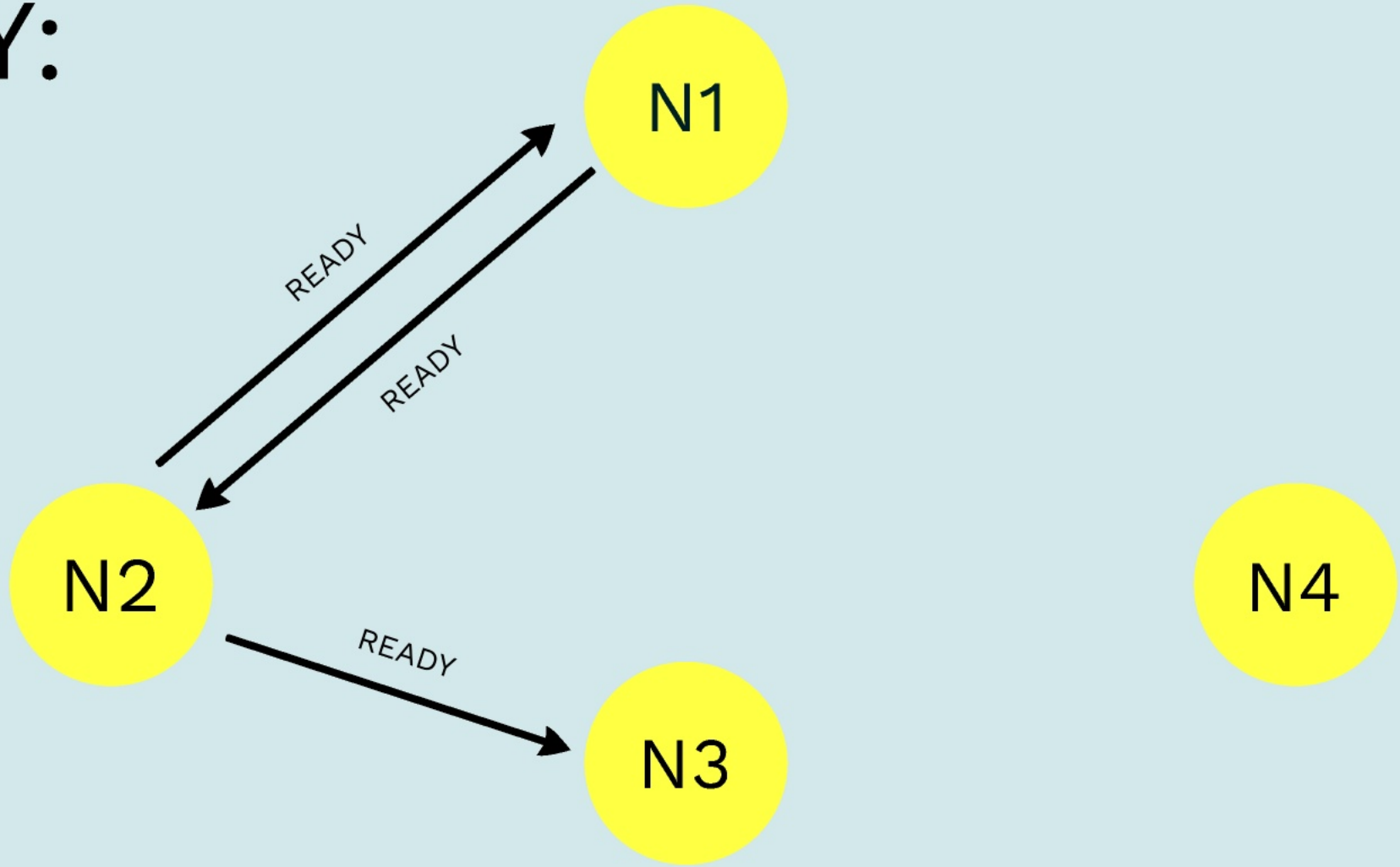
READY:

2f+1



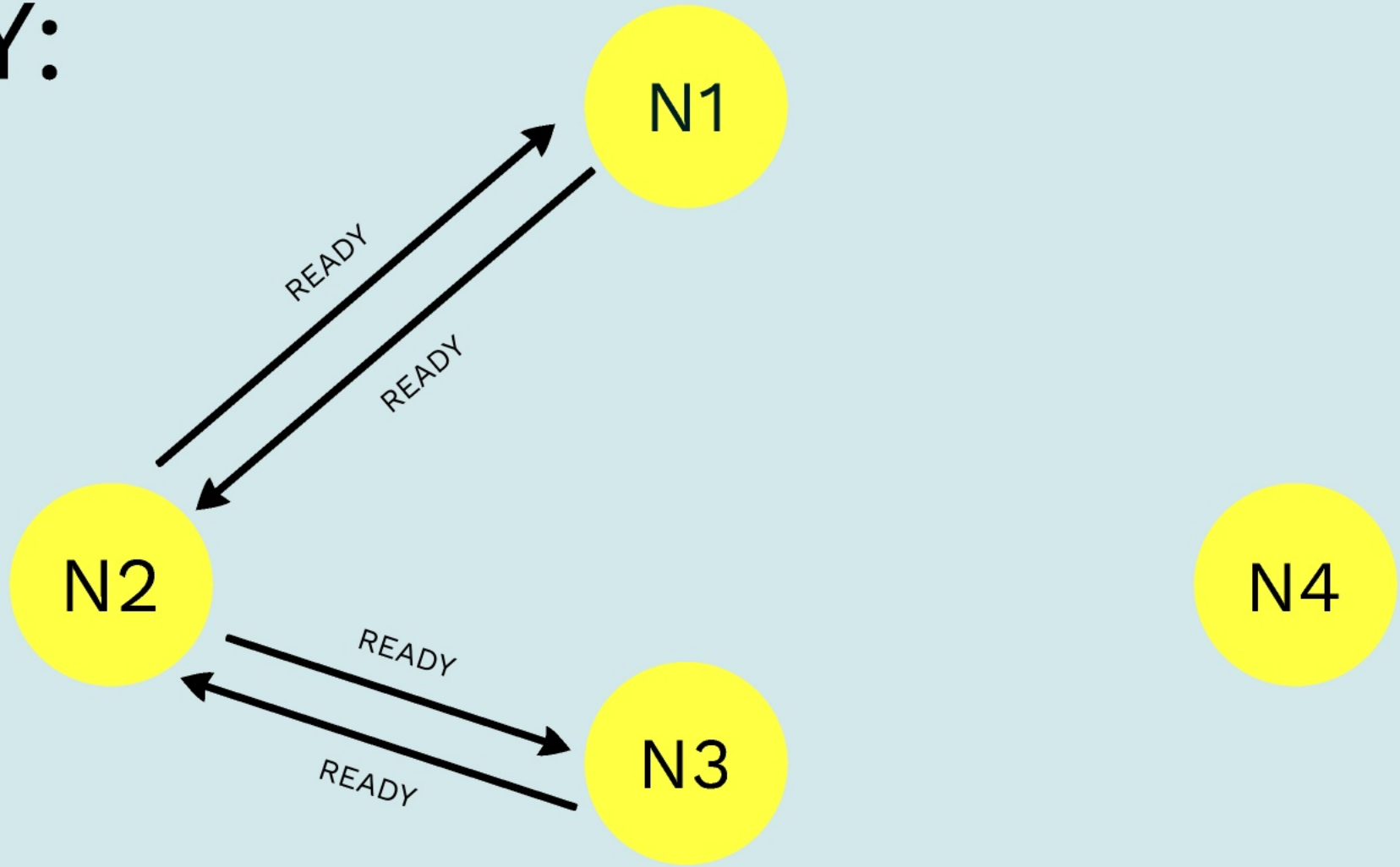
READY:

2f+1



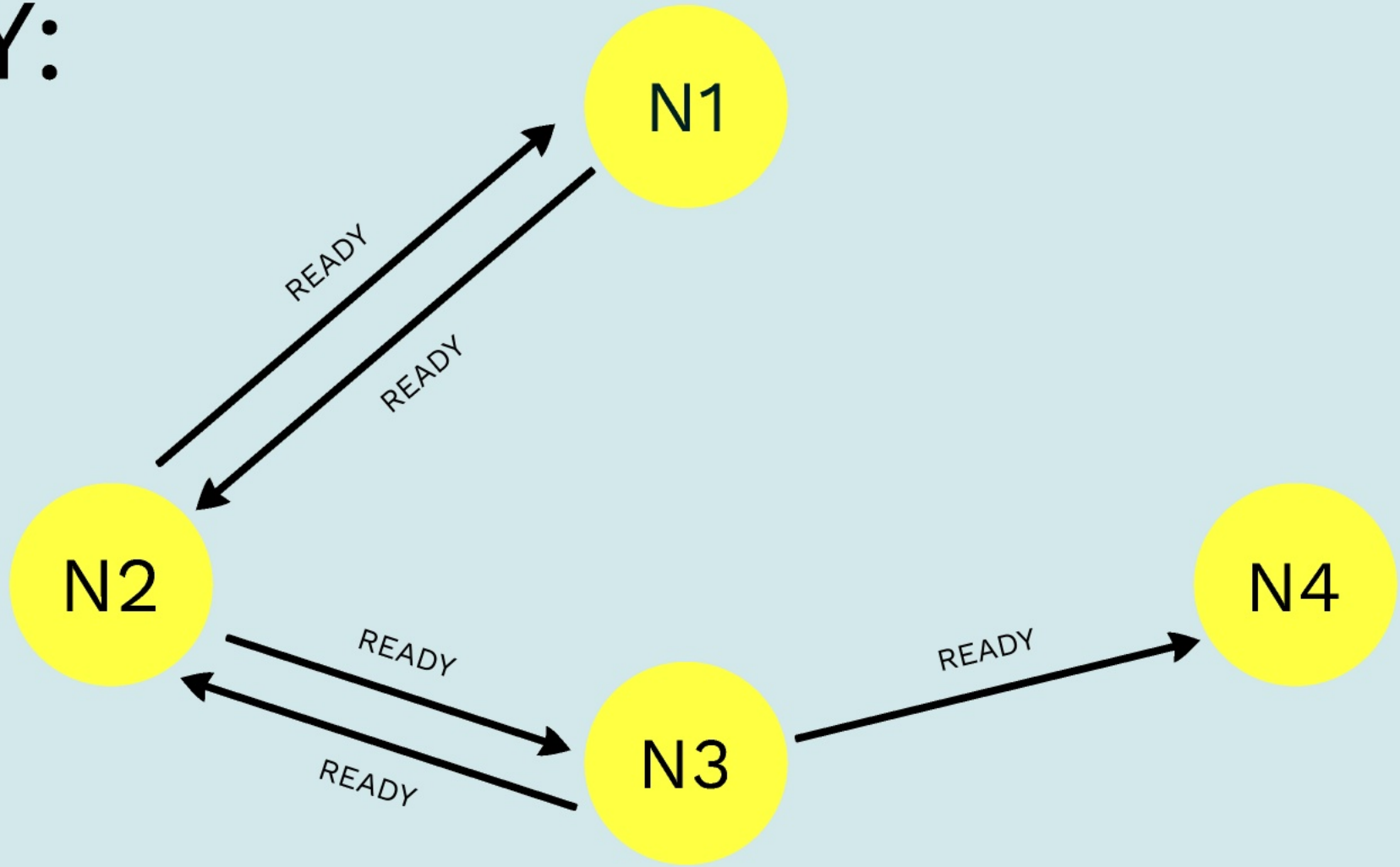
READY:

2f+1



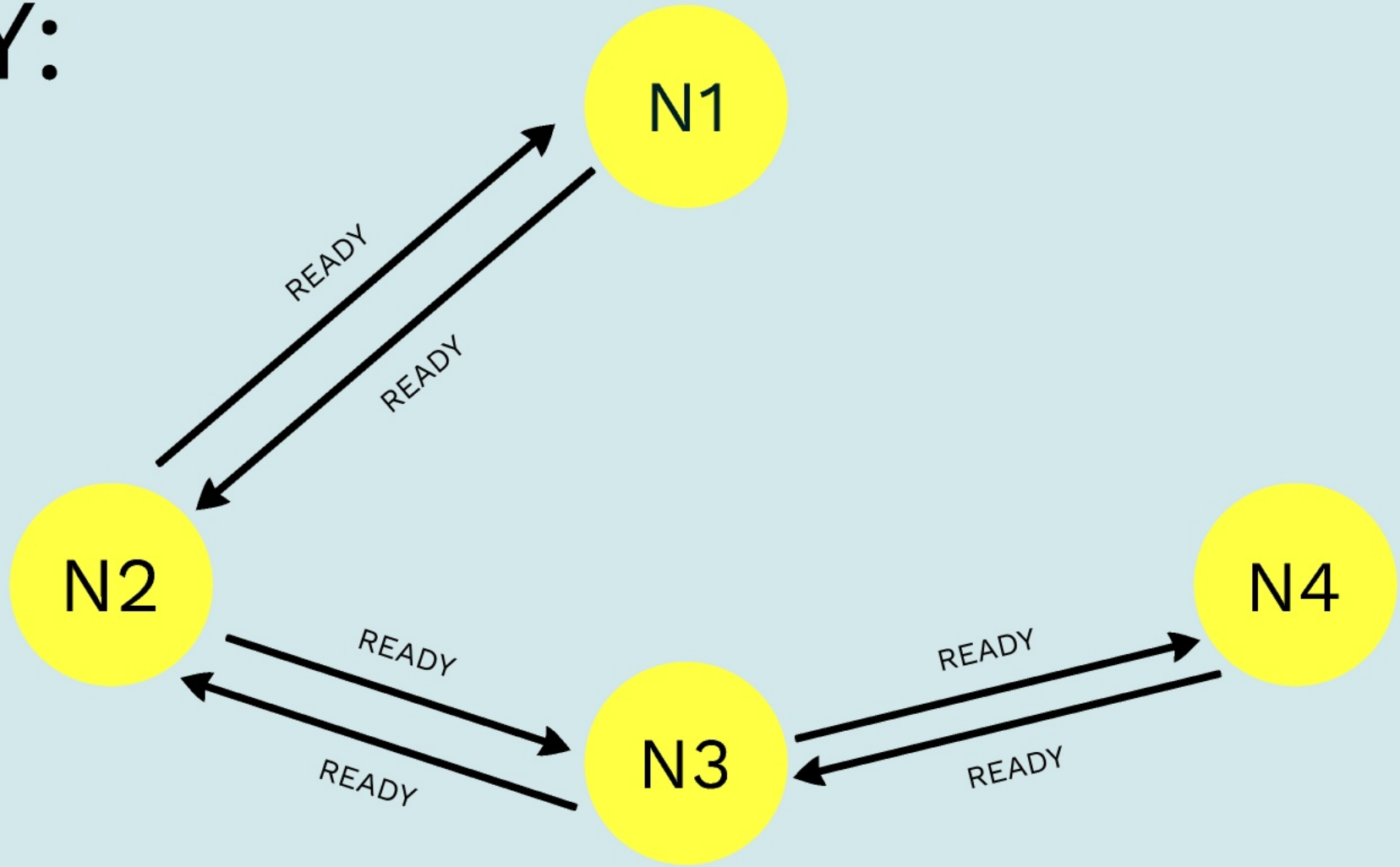
READY:

2f+1



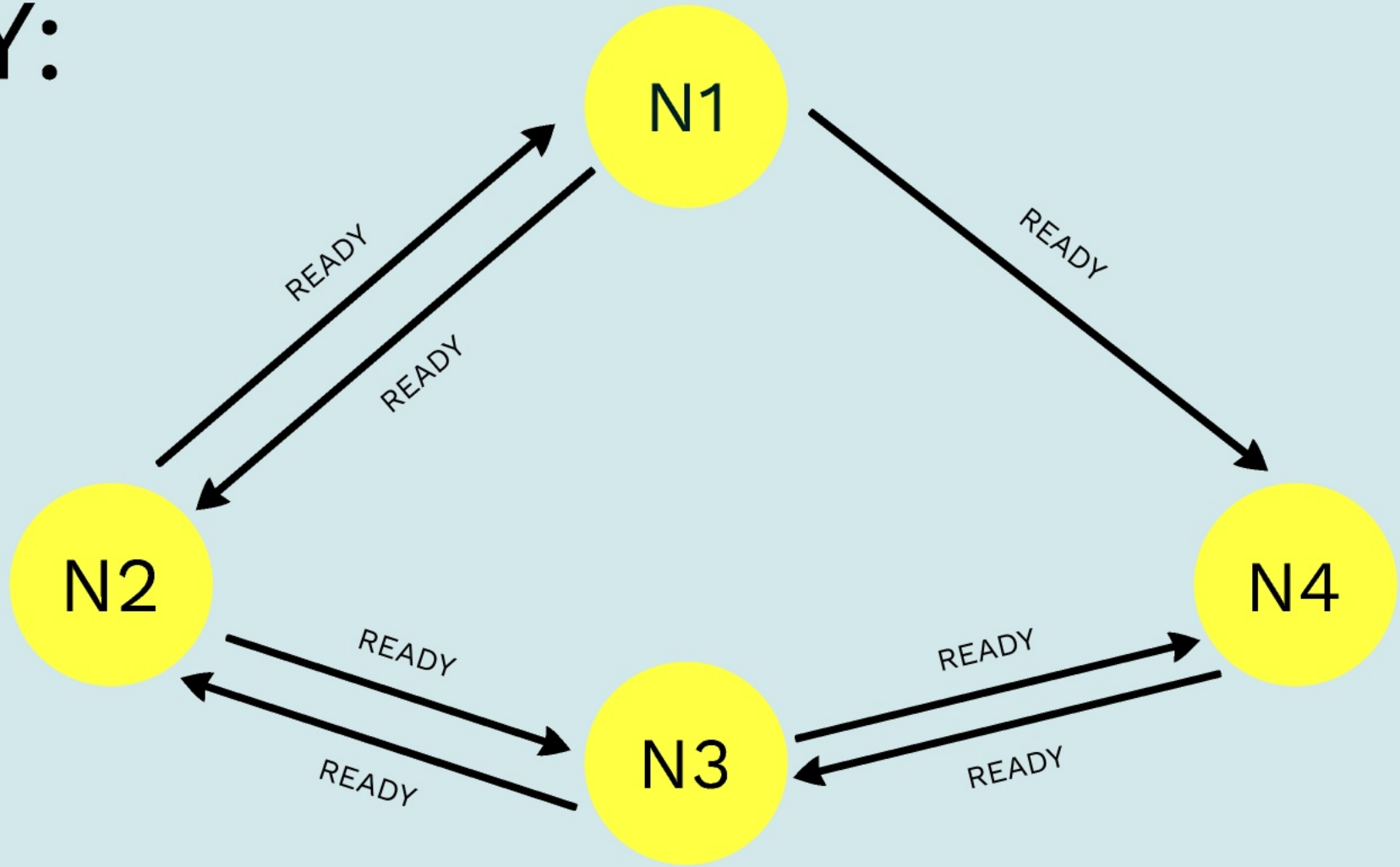
READY:

2f+1



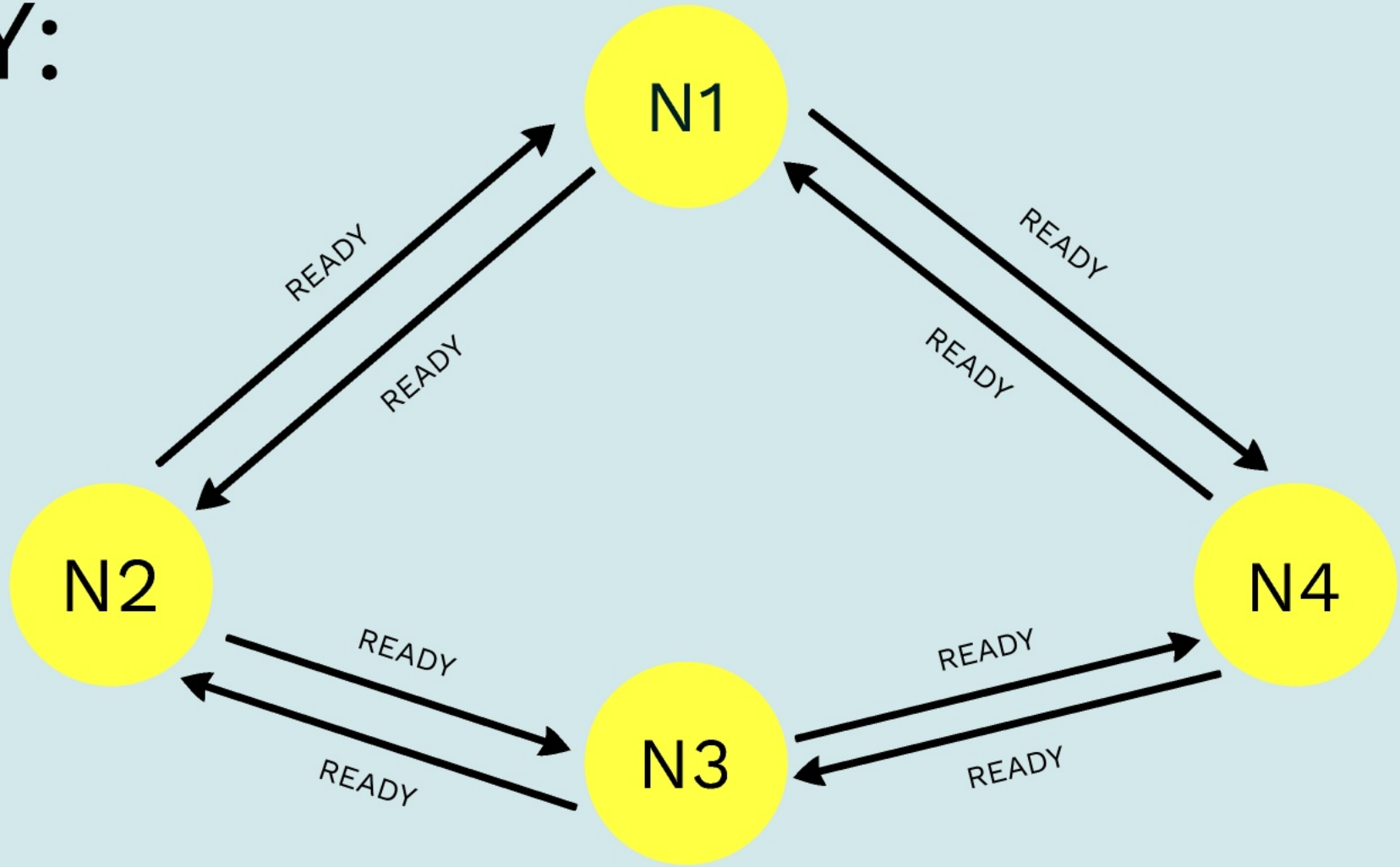
READY:

2f+1



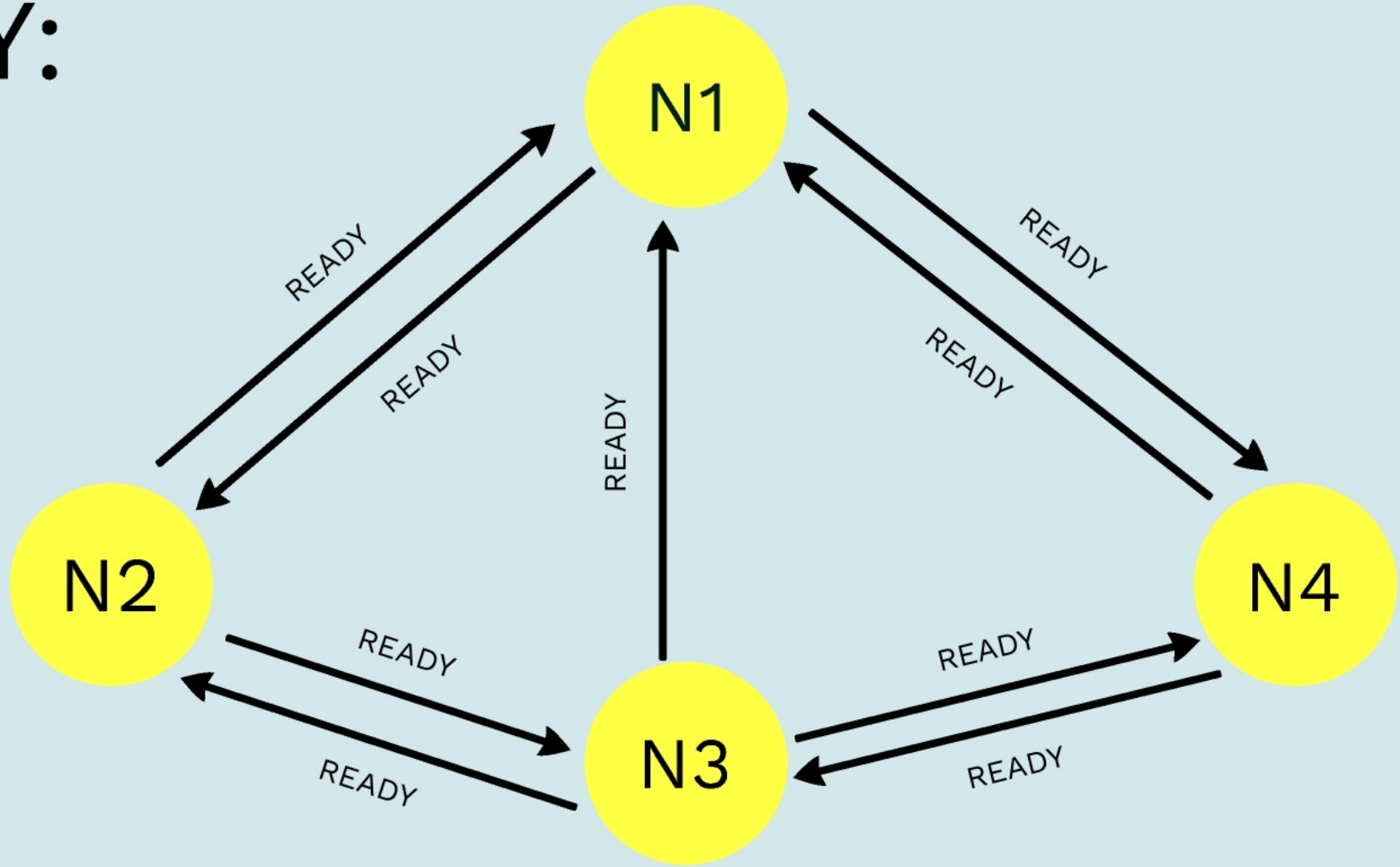
READY:

2f+1



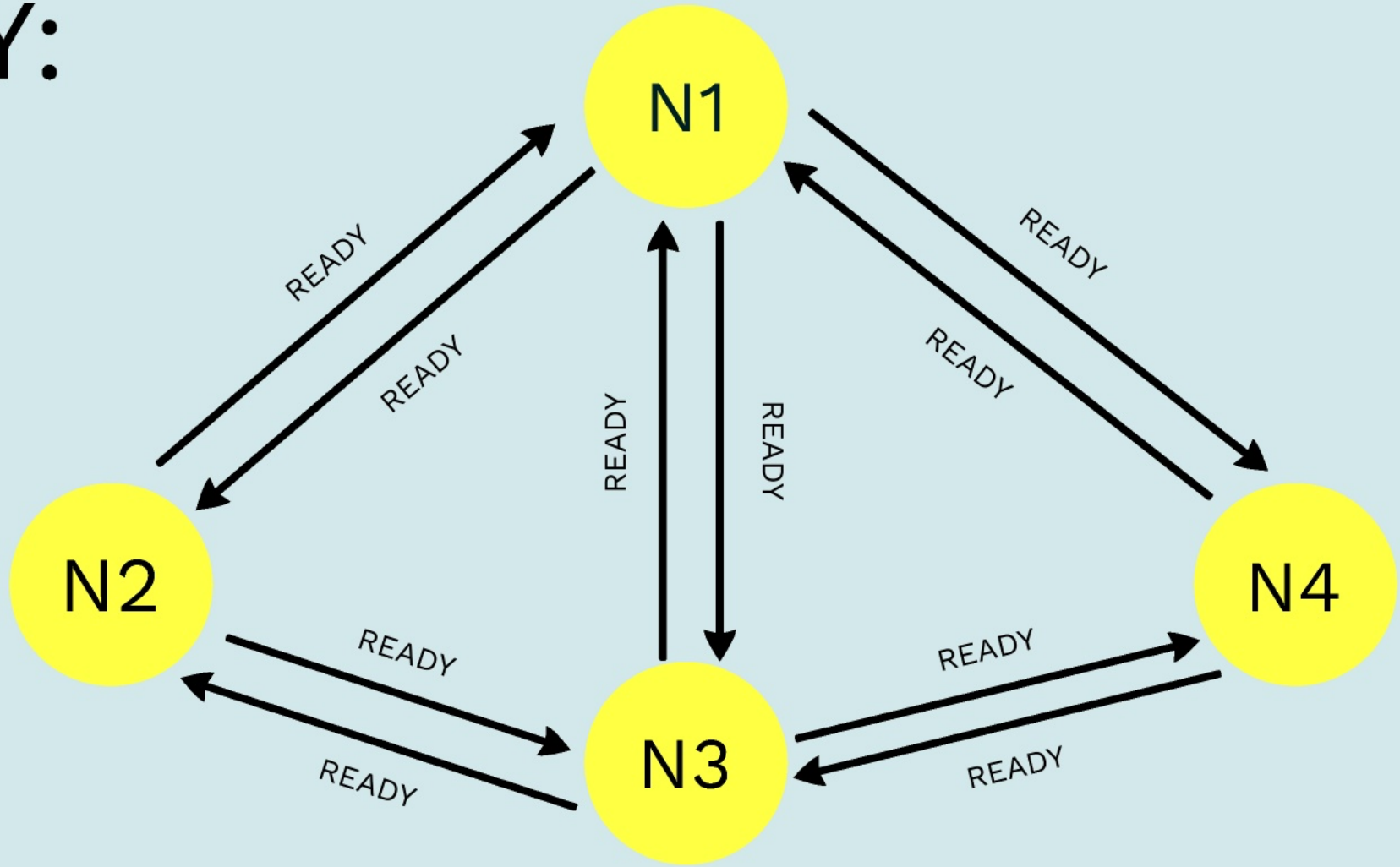
READY:

2f+1


N1
N2
N3
N4
READY
READY
READY
READY
READY
READY
READY
READY
READY
READY

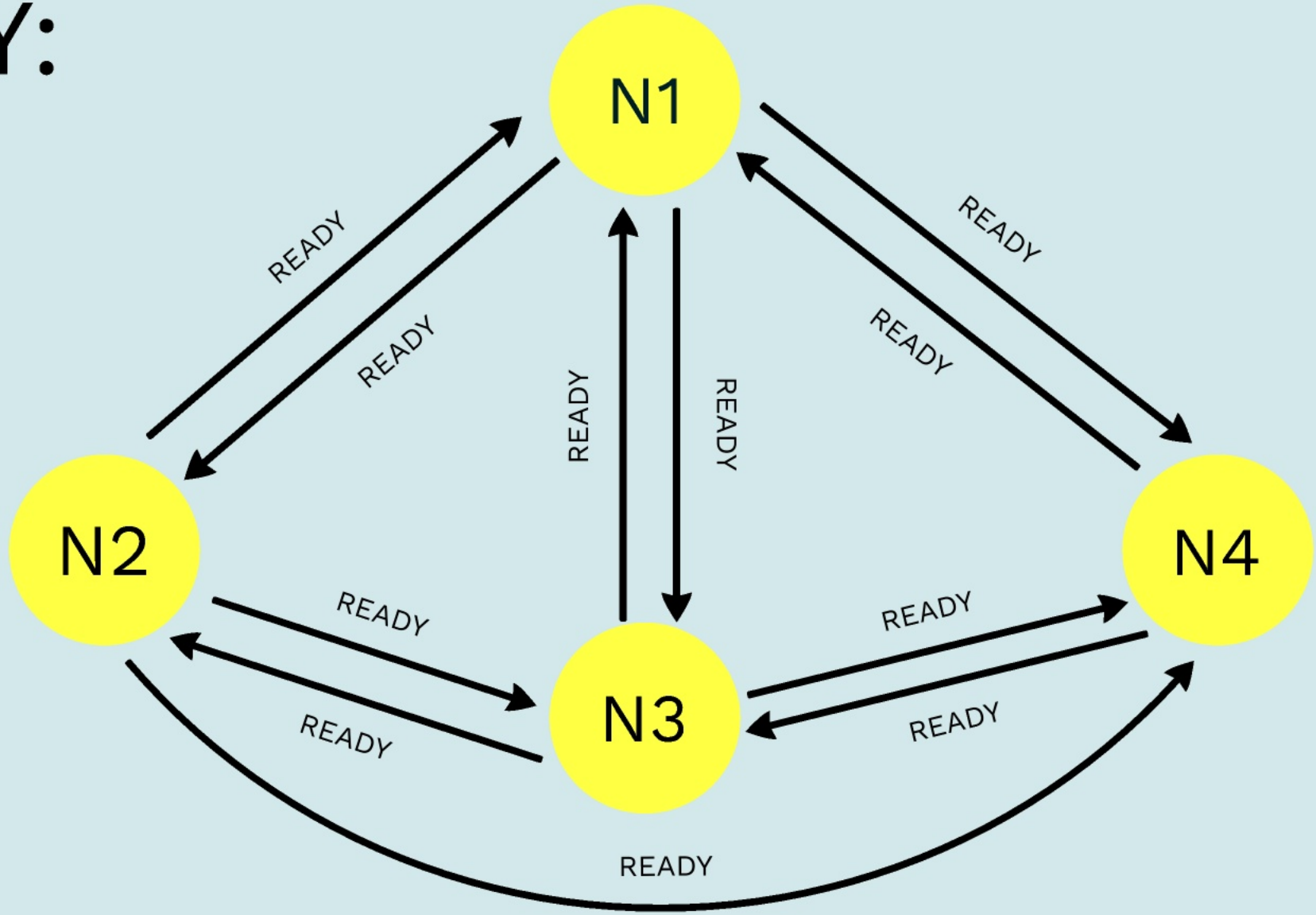
READY:

2f+1


N1
N2
N3
N4
READY
READY
READY
READY
READY
READY
READY
READY
READY
READY

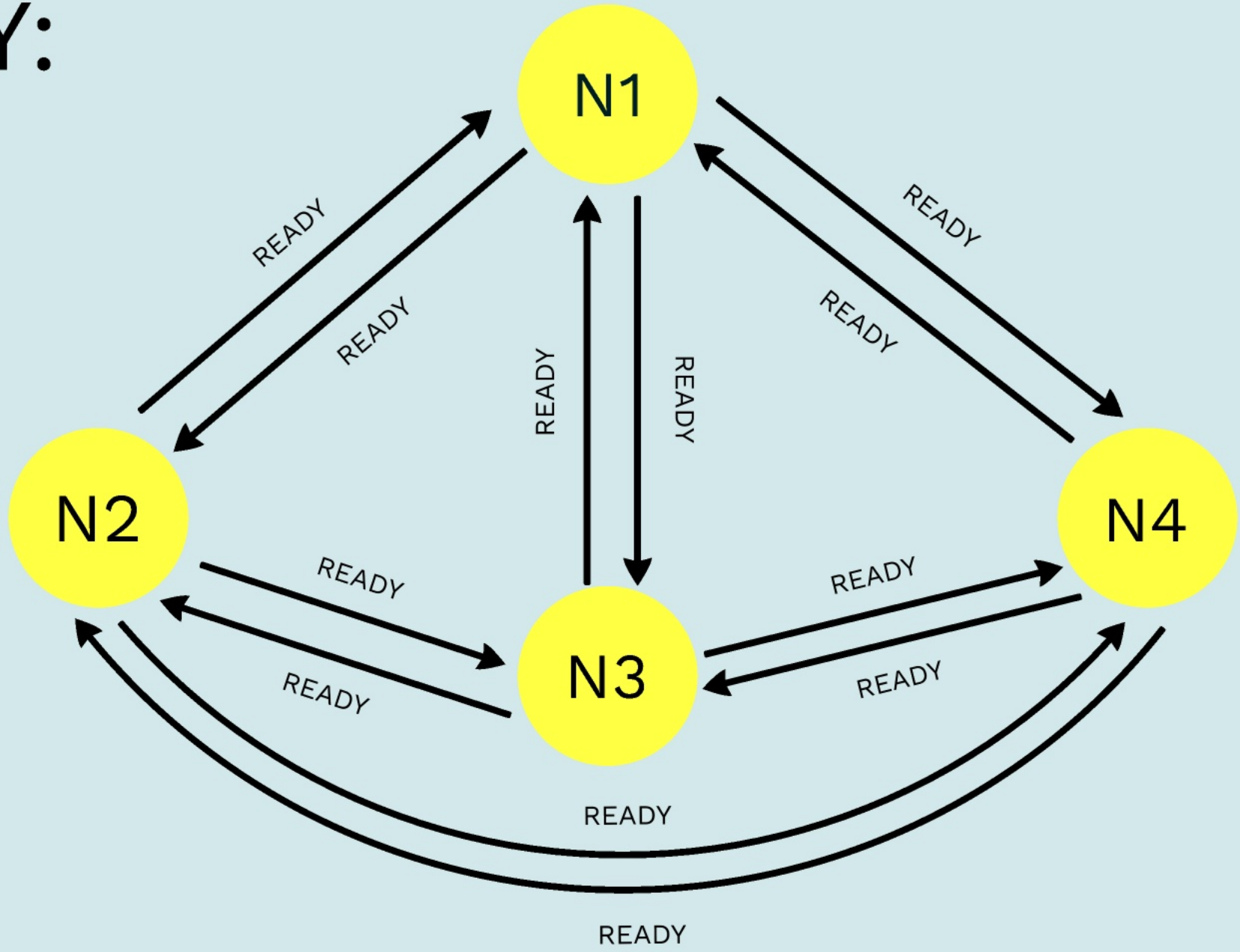
READY:

2f+1


N1
N2
N3
N4
READY
READY
READY
READY
READY
READY
READY
READY
READY
READY
READY

READY:

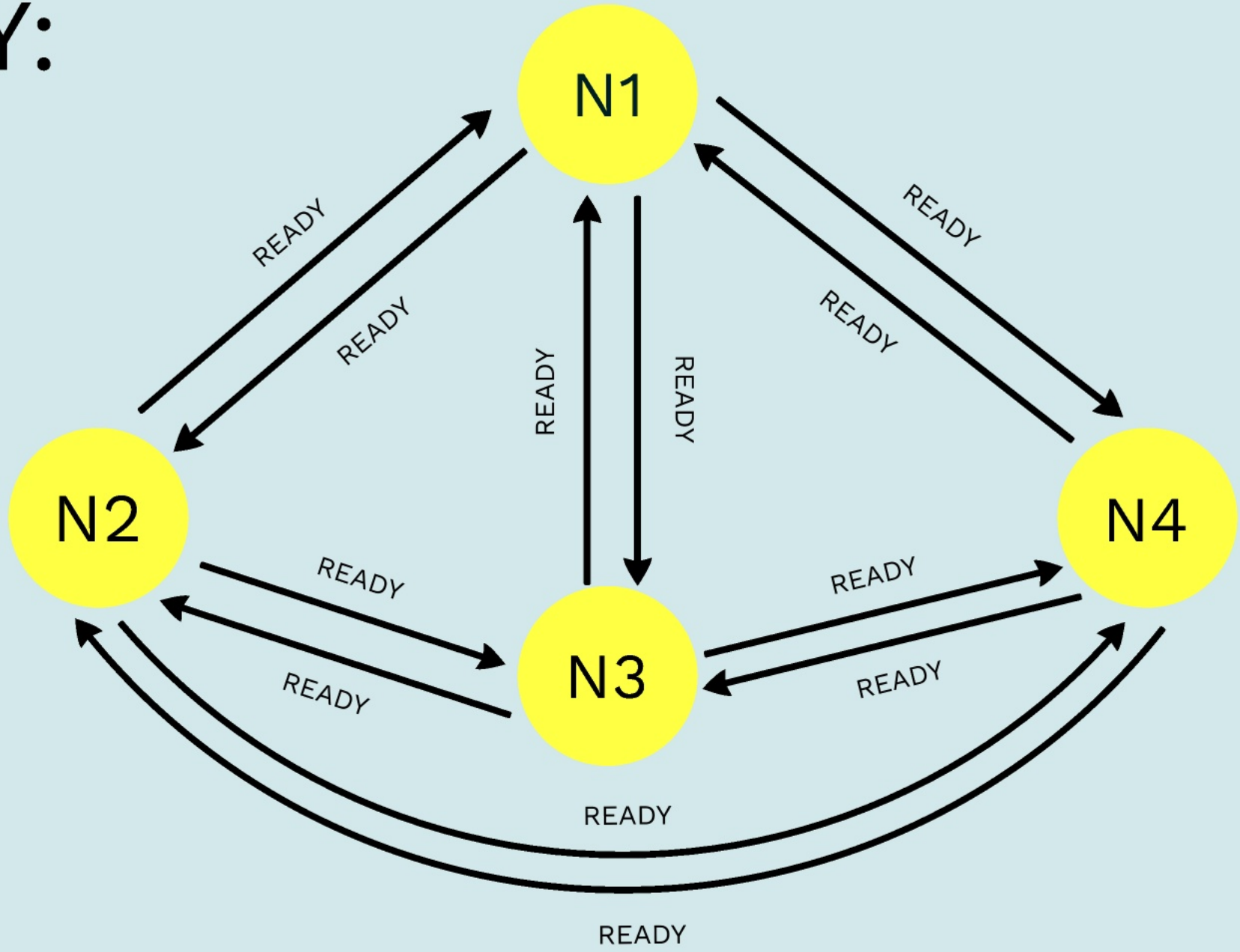
2f+1


N1
N2
N3
N4
READY
READY
READY
READY
READY
READY
READY
READY
READY
READY
READY
READY
READY

READY:

2f+1

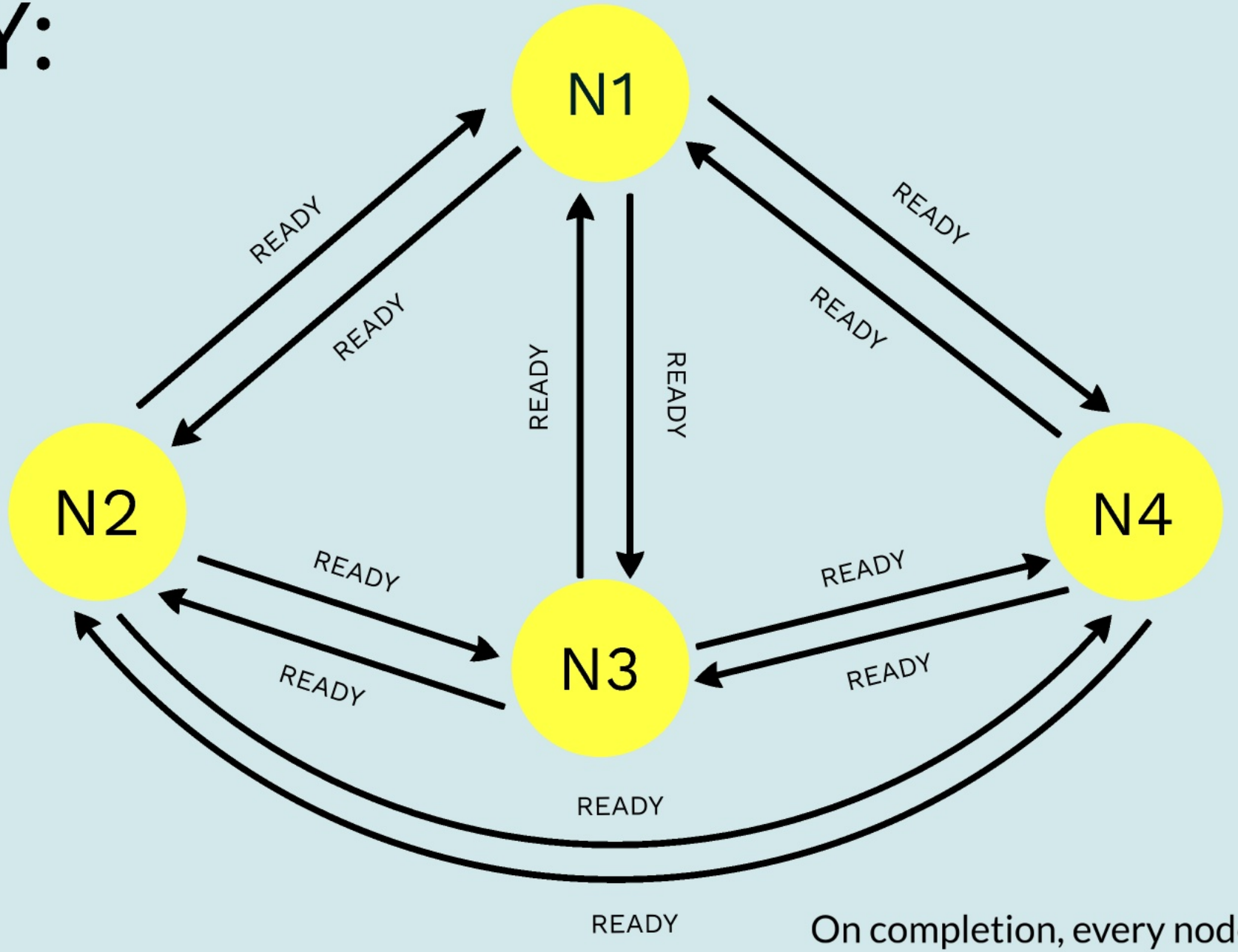
Ready nodes reconstruct P1 with **n-2f** shares



READY:

2f+1

Ready nodes reconstruct P1 with **n-2f** shares



On completion, every node has **0 or 1** for P1

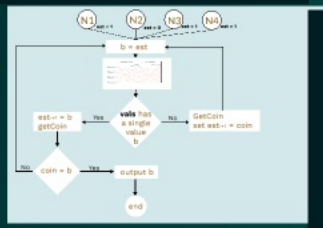
Binary Byzantine Agreement

Binary Byzantine Agreement

Nodes agree on which of the proposals are chosen to be included in the final set of transactions to be commited in this epoch

BA 1

BA for P1 (Proposal from Node 1)



Outputs

After completion of all the BA instances.
Every node outputs a bit vector consisting 0, 1 - (1, 1, 0, 1)
If b = 1, that node's proposal is selected to be included in final set

Questions

1. What is the role of common coin in the Binary Byzantine Agreement protocol if the majority is reached in the step after exchanging BVAL messages?

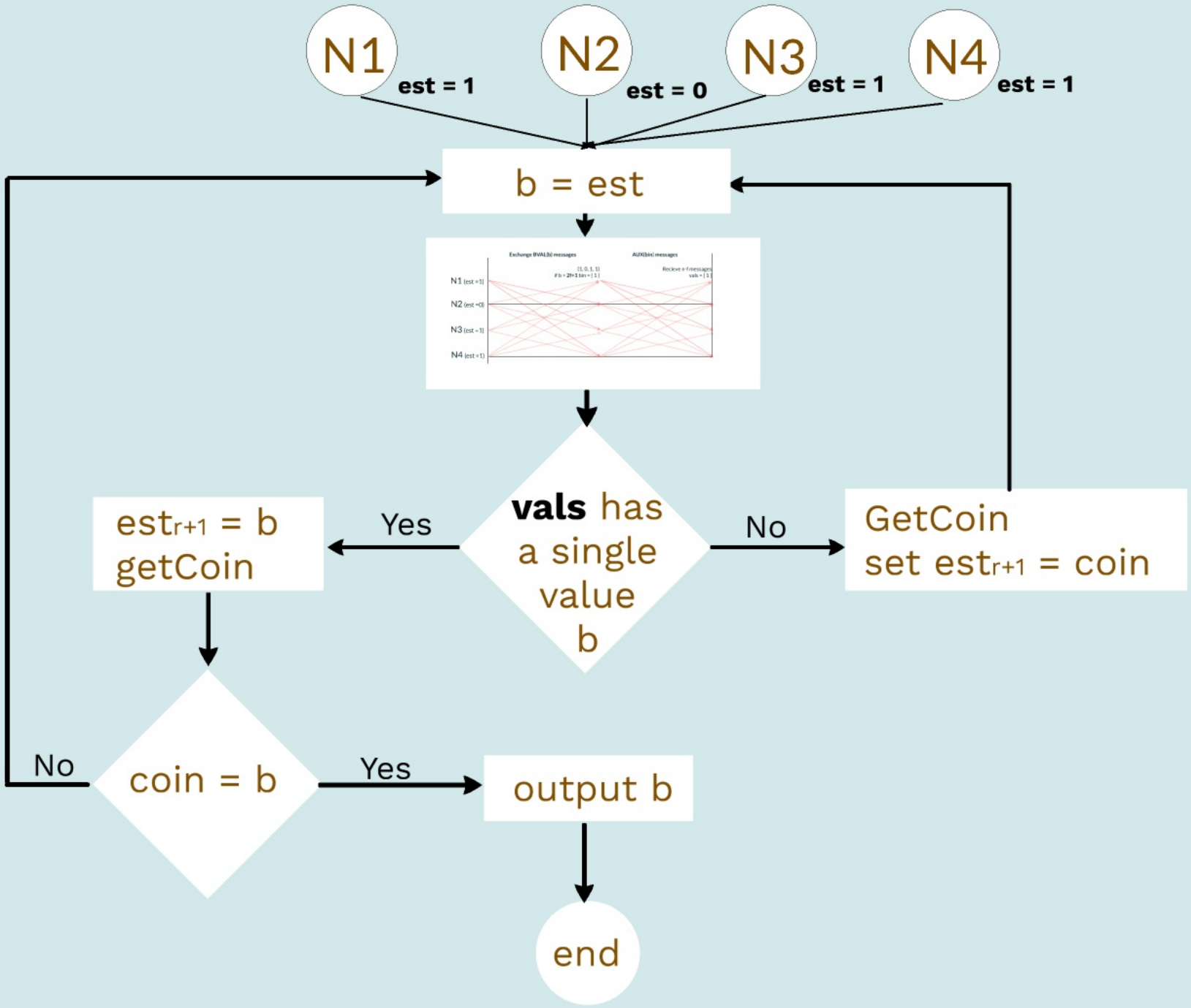
THANK YOU

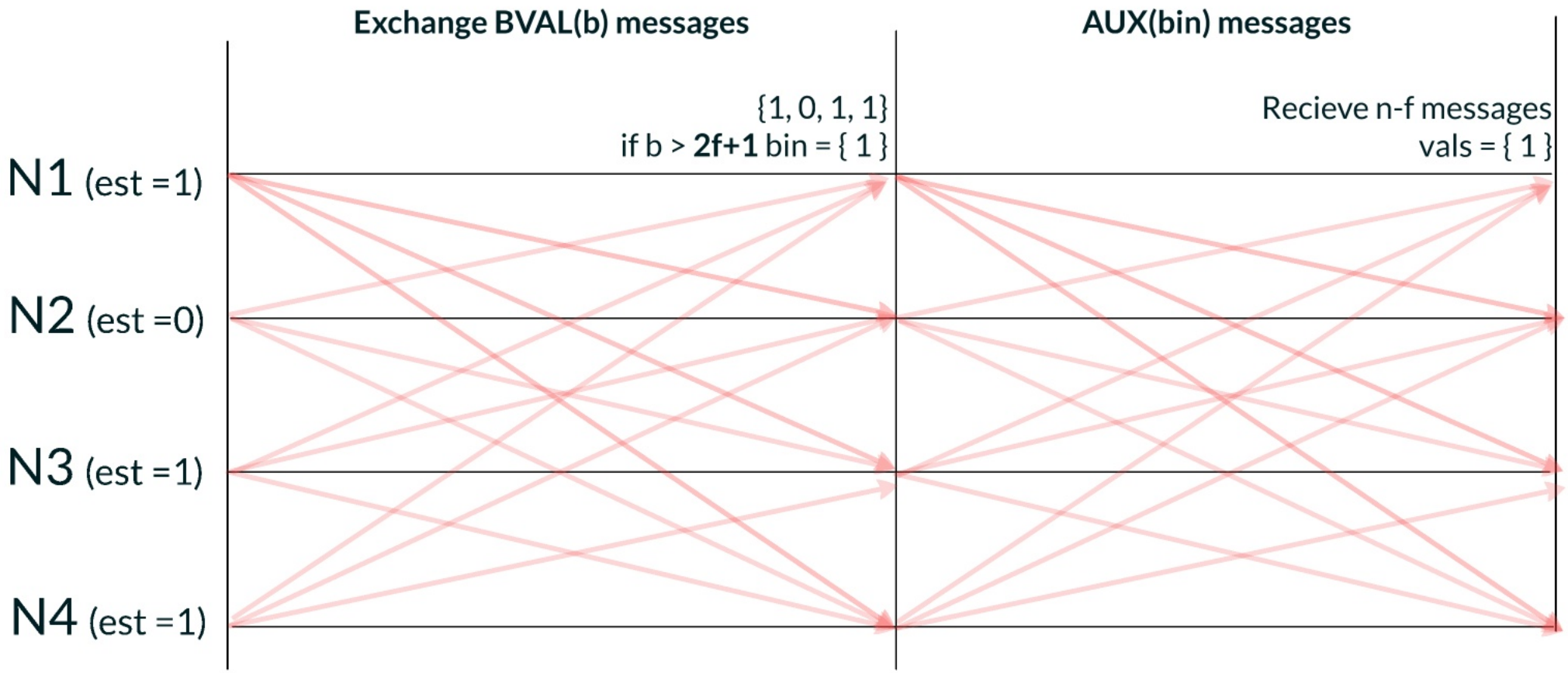
Binary Byzantine Agreement

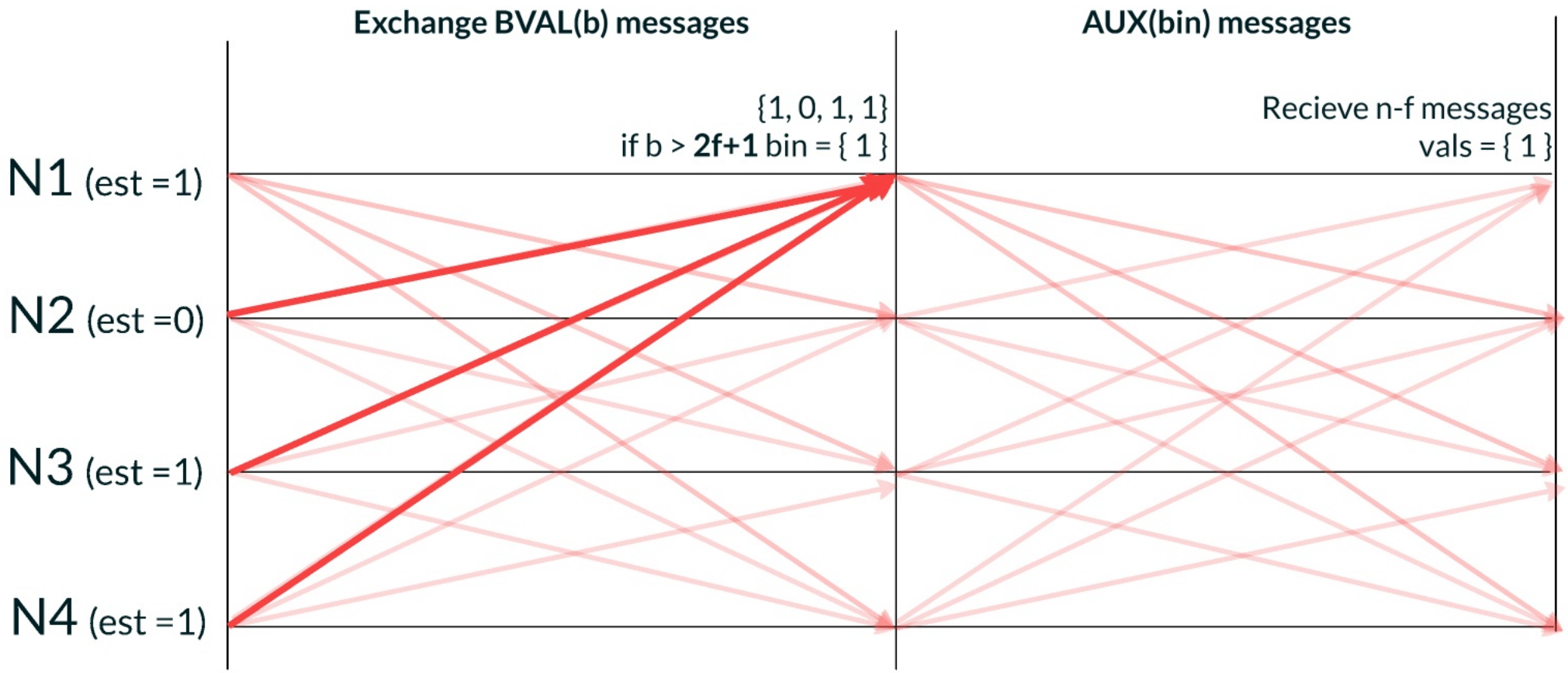
Nodes agree on which of the proposals are chosen to be included in the final set of transactions to be commited in this epoch

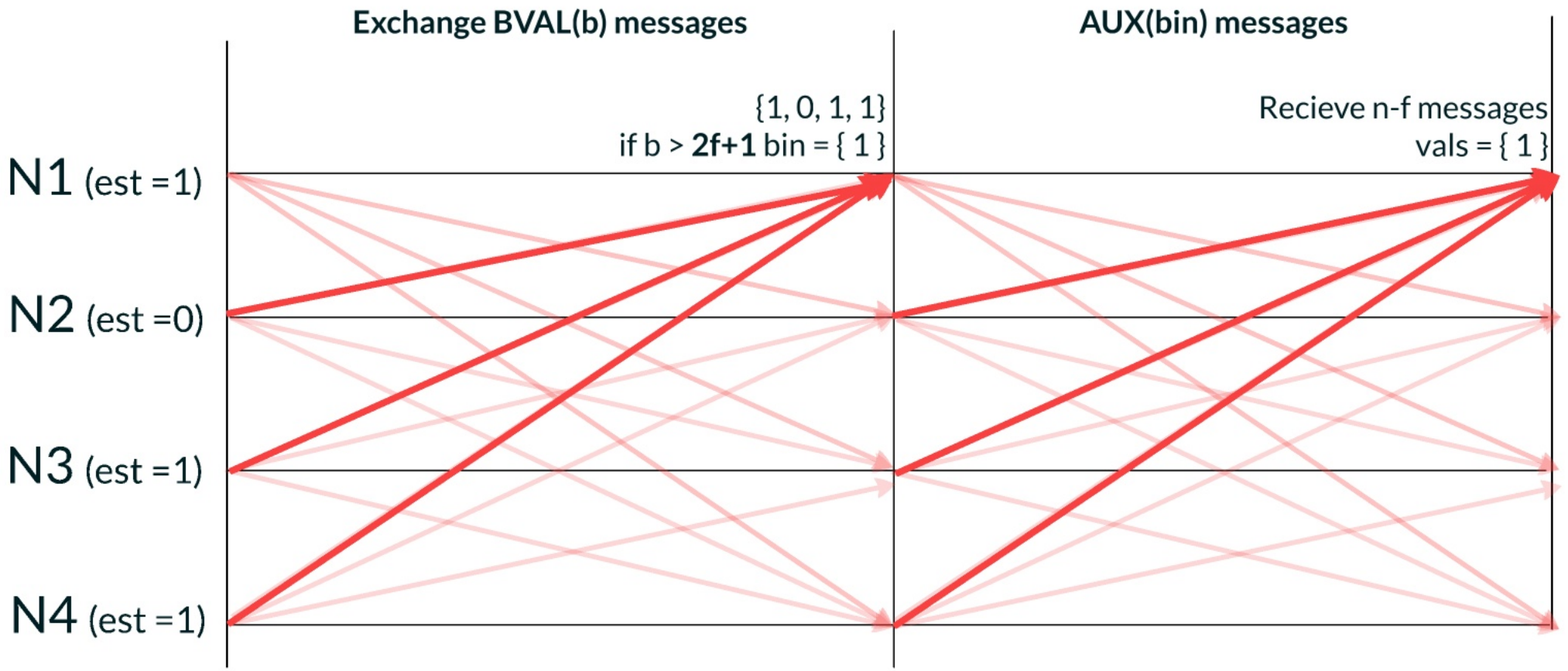
BA 1

BA for P1 (Proposal from Node 1)



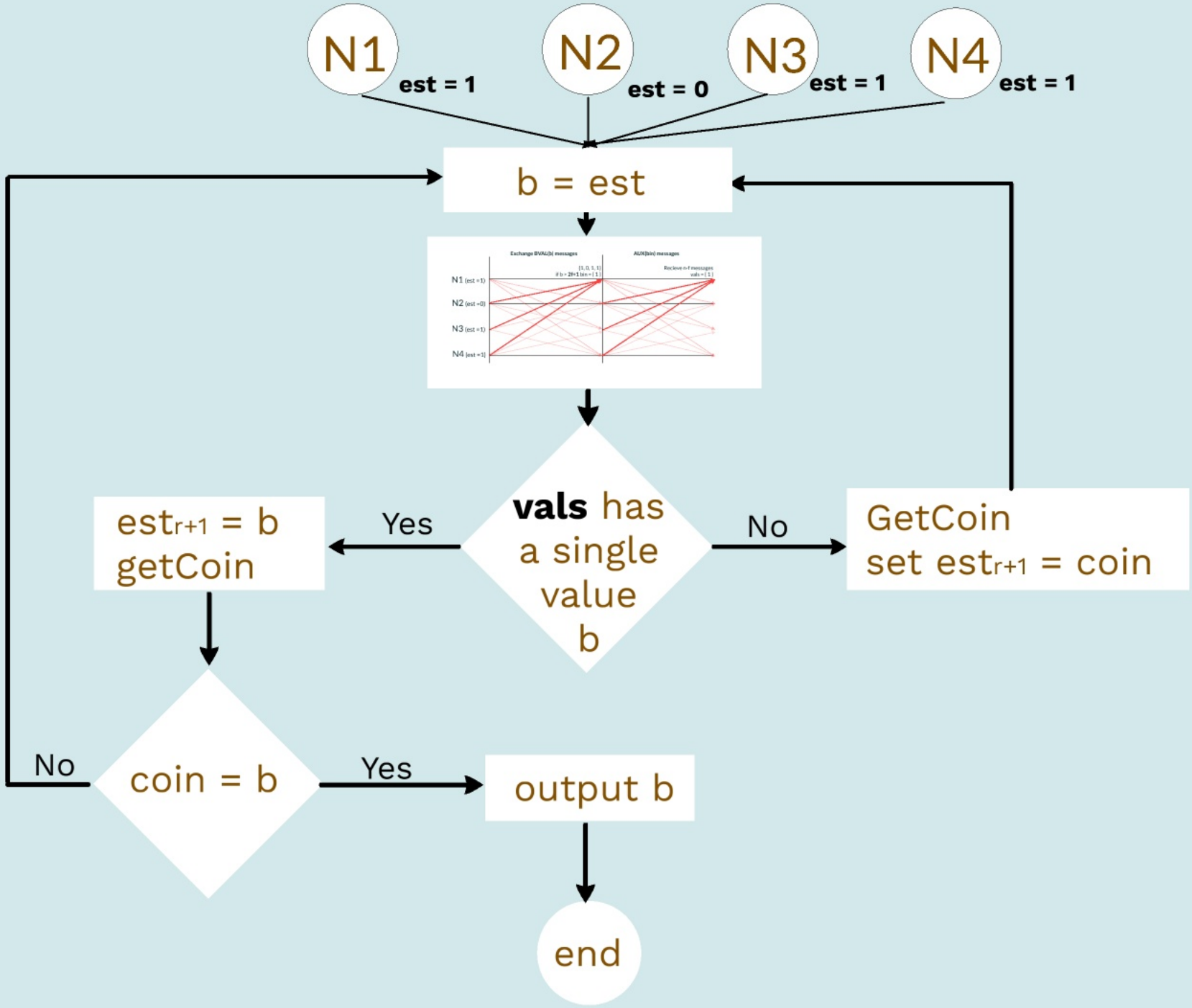

Exchange BVAL(b) messages
AUX(bin) messages
{1, 0, 1, 1}
if b > 2f+1 bin = { 1 }
Recieve n-f messages
vals = { 1 }
N1 (est =1)
N2 (est =0)
N3 (est =1)
N4 (est =1)


Exchange BVAL(b) messages
AUX(bin) messages
{1, 0, 1, 1}
if b > 2f+1 bin = { 1 }
Recieve n-f messages
vals = { 1 }
N1 (est =1)
N2 (est =0)
N3 (est =1)
N4 (est =1)


Exchange BVAL(b) messages
AUX(bin) messages
{1, 0, 1, 1}
if b > 2f+1 bin = { 1 }
Recieve n-f messages
vals = { 1 }
N1 (est =1)
N2 (est =0)
N3 (est =1)
N4 (est =1)

1

or P1
sal from
le 1)



Outputs

After completion of all the BA instances,
Every node outputs a bit vector consisting 0, 1 - {1, 1, 0, 1}
If b = 1, that node's proposal is selected to be included in final set

Questions

1. What is the role of common coin in the Binary Byzantine Agreement protocol if the majority is reached in the step after exchanging BVAL messages?

THANK YOU