Mysticeti

The new core of the Sui blockchain

Alberto Sonnino

Tailoring the Talk

Do you know:

1. How blockchains work (roughly)?
2. What Byzantine Fault Tolerance (BFT) means?
3. What DAG-based consensus are?
4. How Narwhal / Bullshark work (roughly)?

Byzantine Fault Tolerance

> 2/3

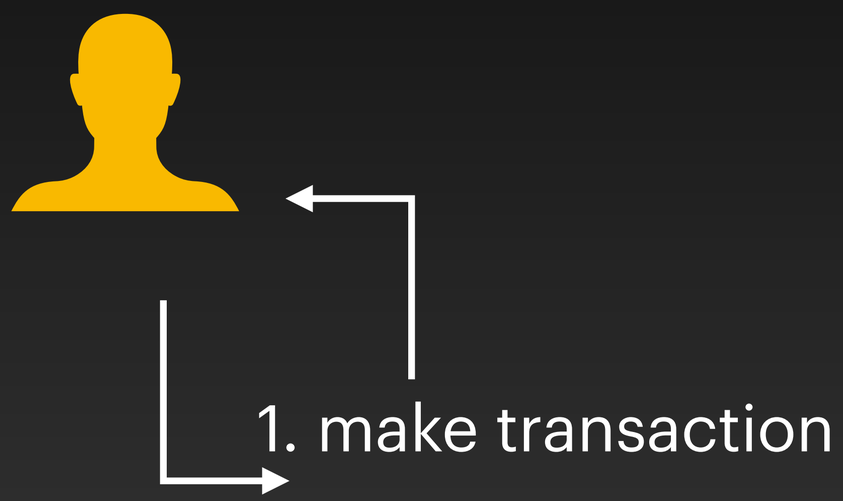
Byzantine Fault Tolerance

>= 2f+1

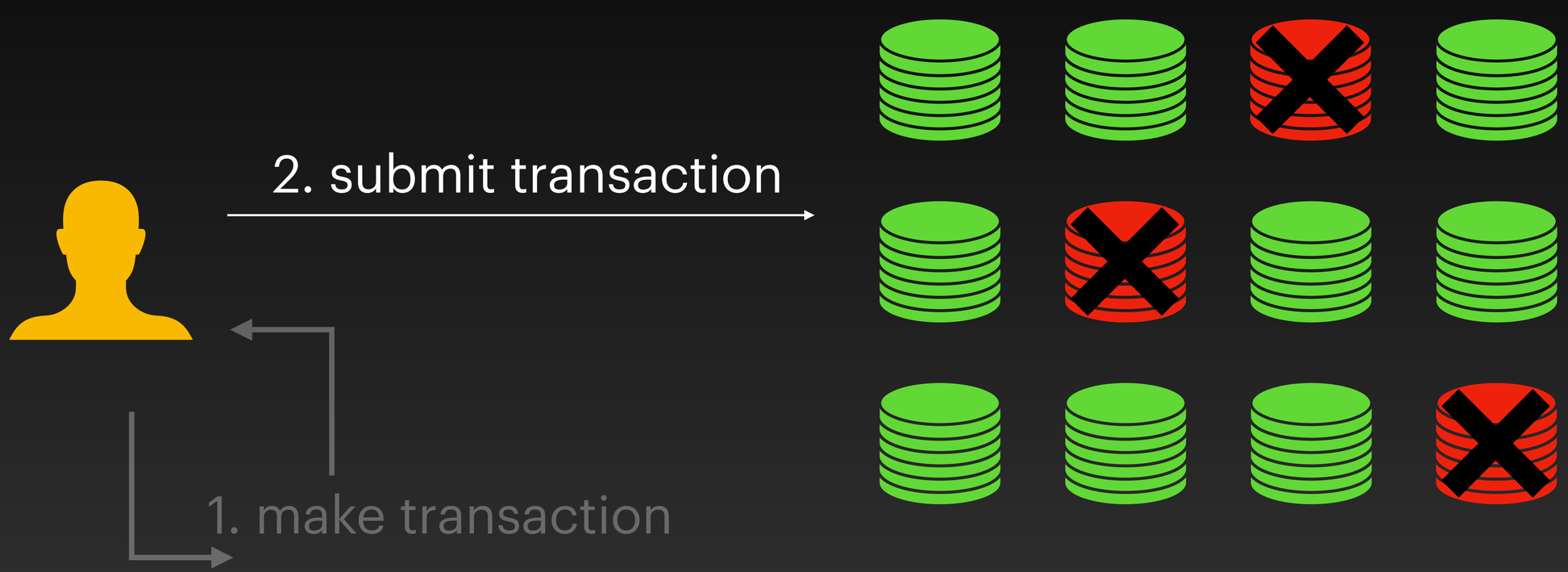
3f+1

Partial Synchrony

Blockchains

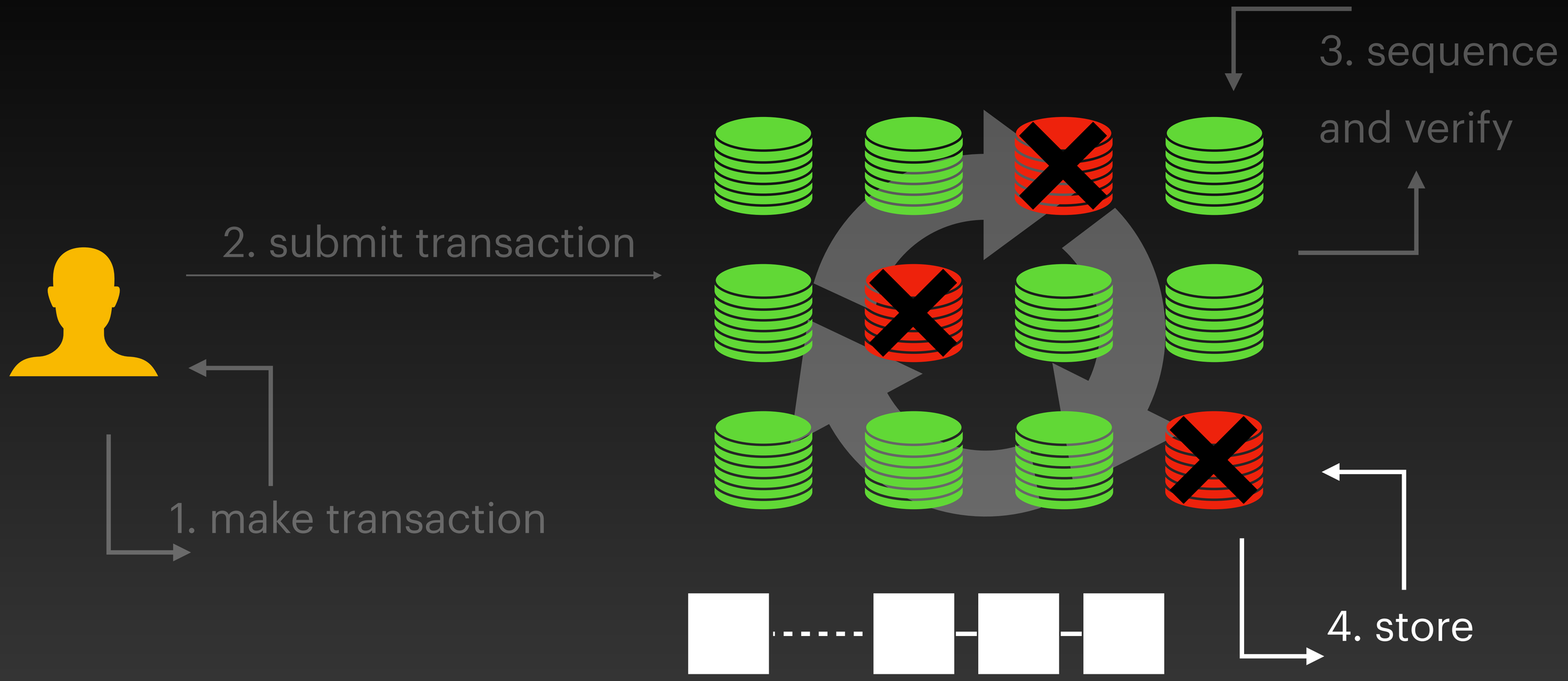


Blockchains



Blockchains

Blockchains



Keeping the Talk Short

In scope

- Ordering (quorum-based)

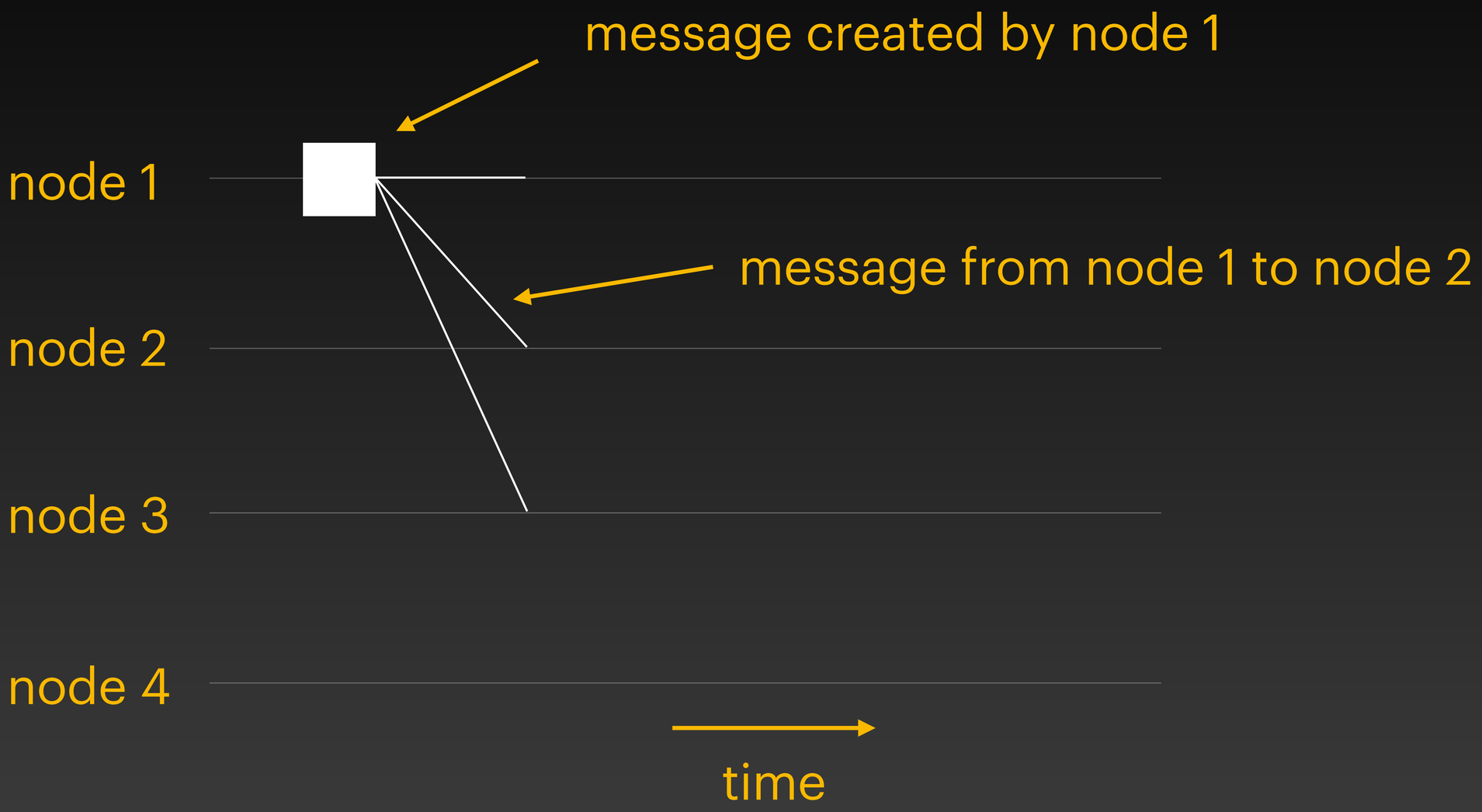
Not in scope

- Nodes selection?
- Committee reconfiguration?
- Transactions execution?
- Transactions language?
- Financial incentives?
- etc

Mysticeti

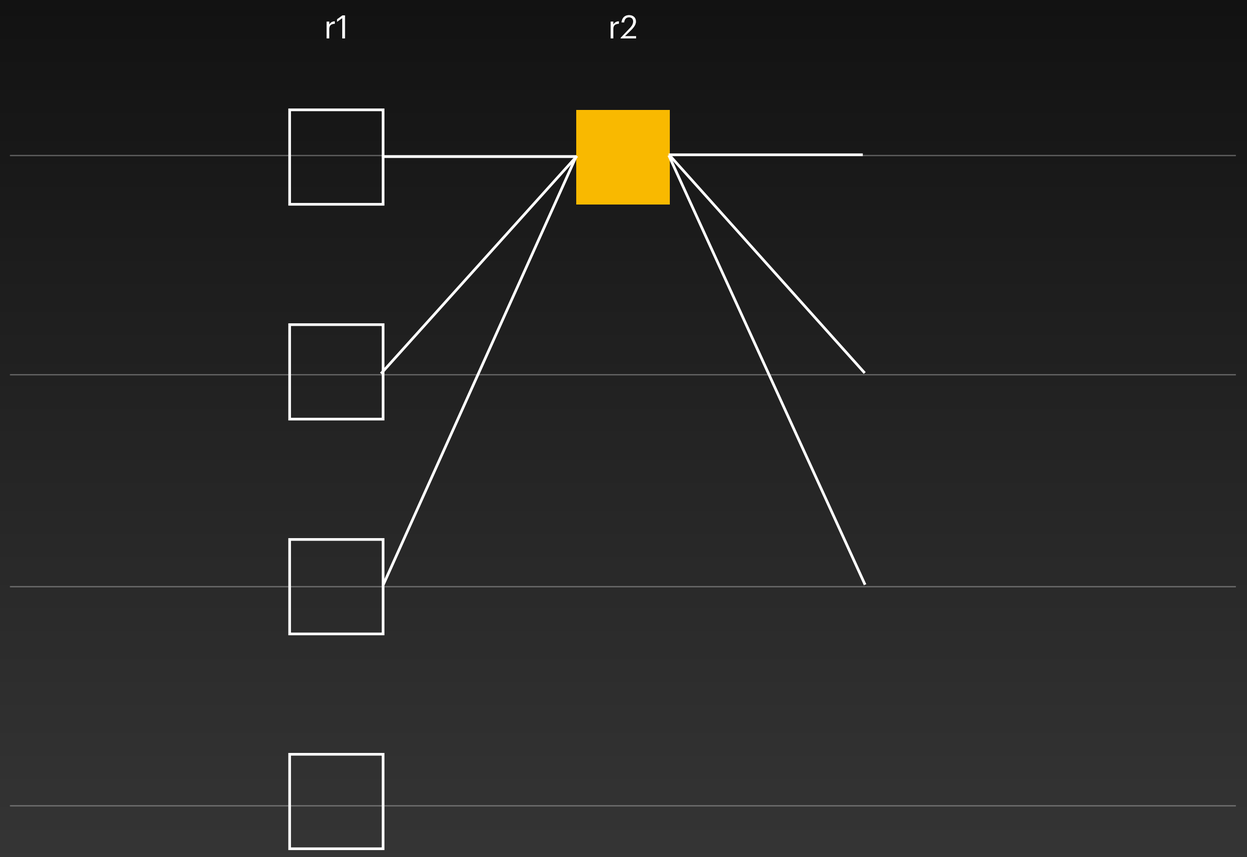
Low-latency DAG consensus with fast commit path

Lamport Diagram



The Mysticeti DAG

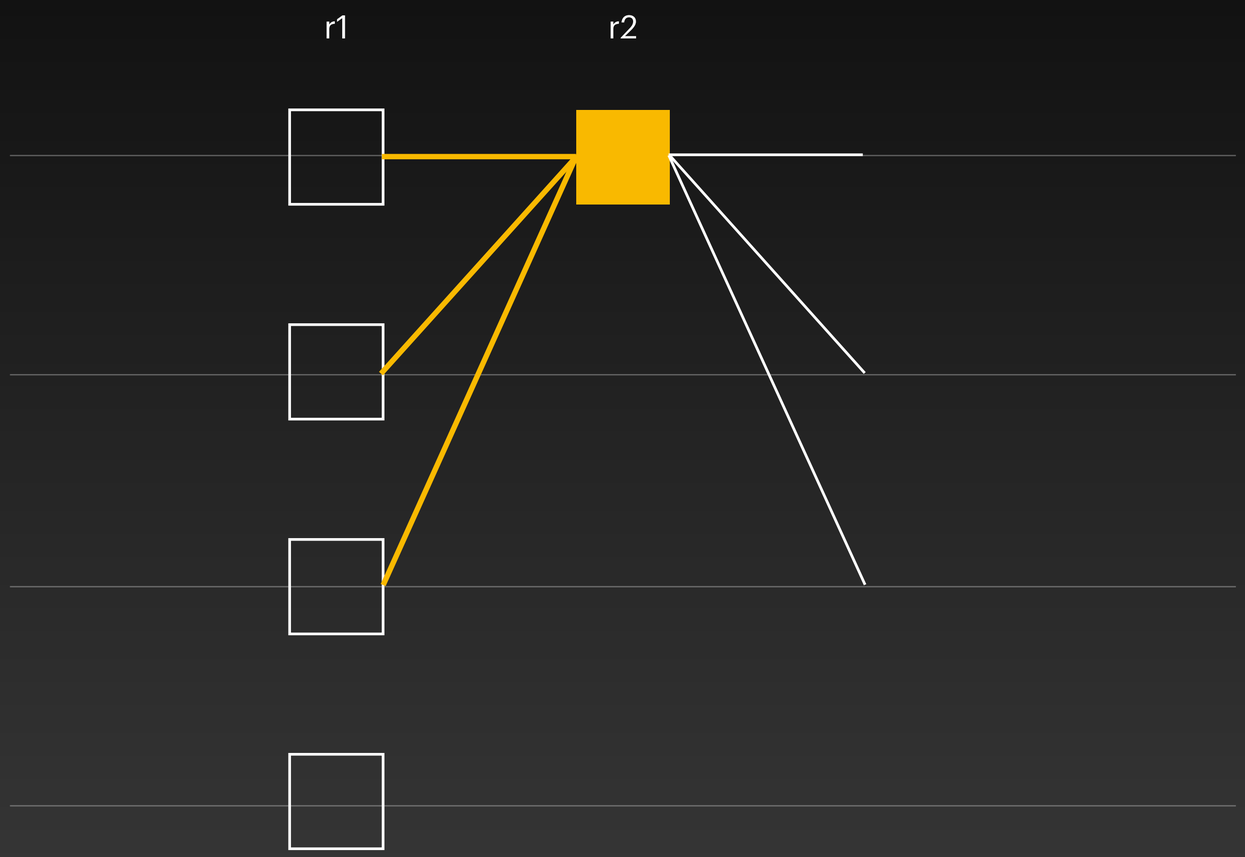
Block Creation



- Round number
- Author
- Payload (transactions)
- Signature

The Mysticeti DAG

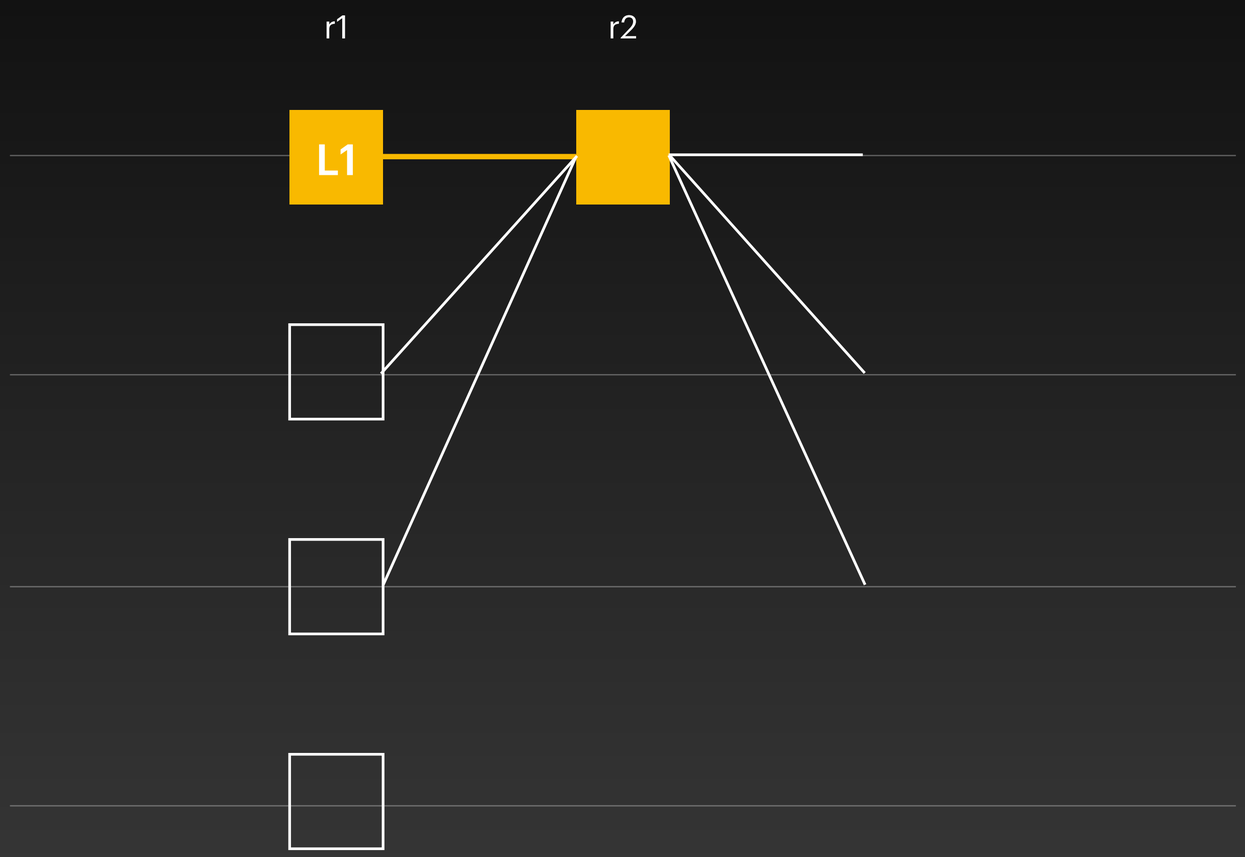
Rule 1: Link to 2f+1 parents



- Total nodes: **3f+1 = 4**
- Quorum: **2f+1 = 3**

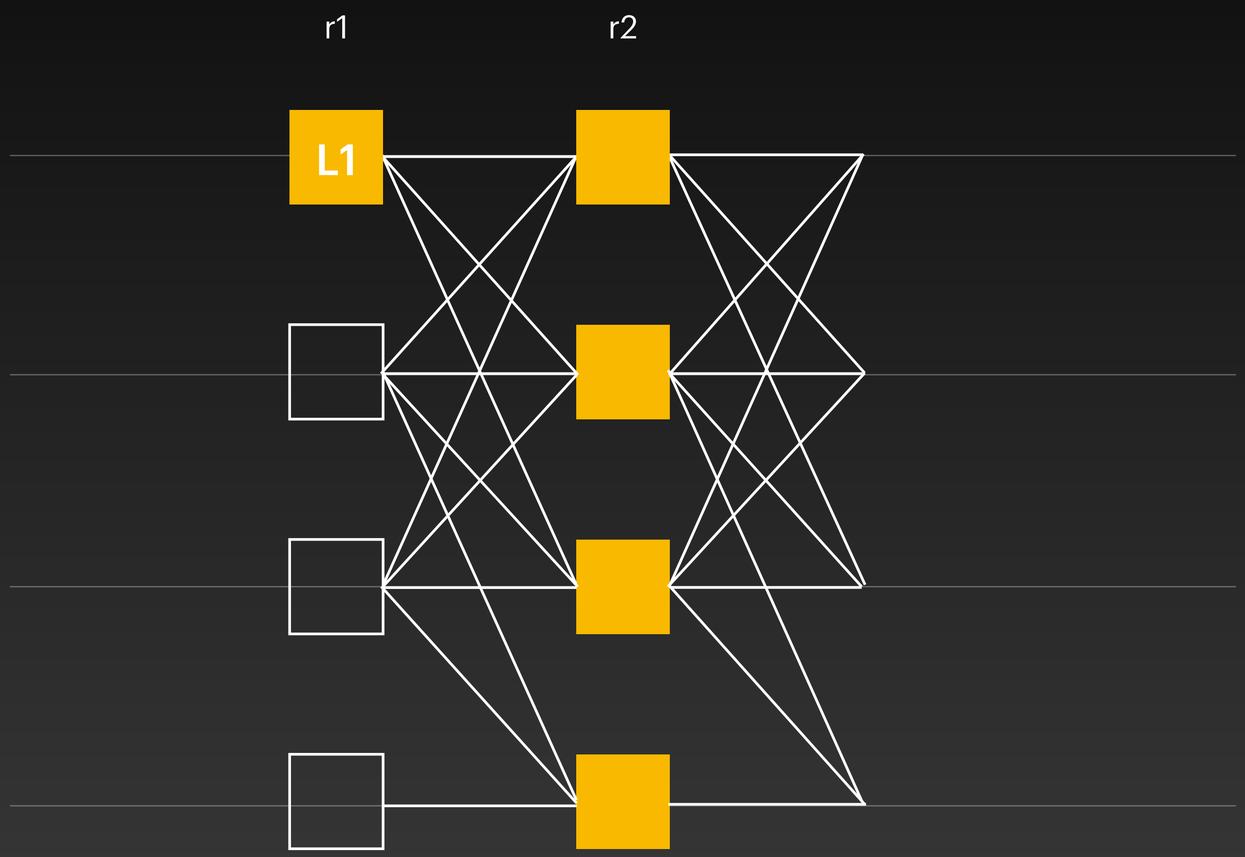
The Mysticeti DAG

Rule 2: Every node waits and links to leaders

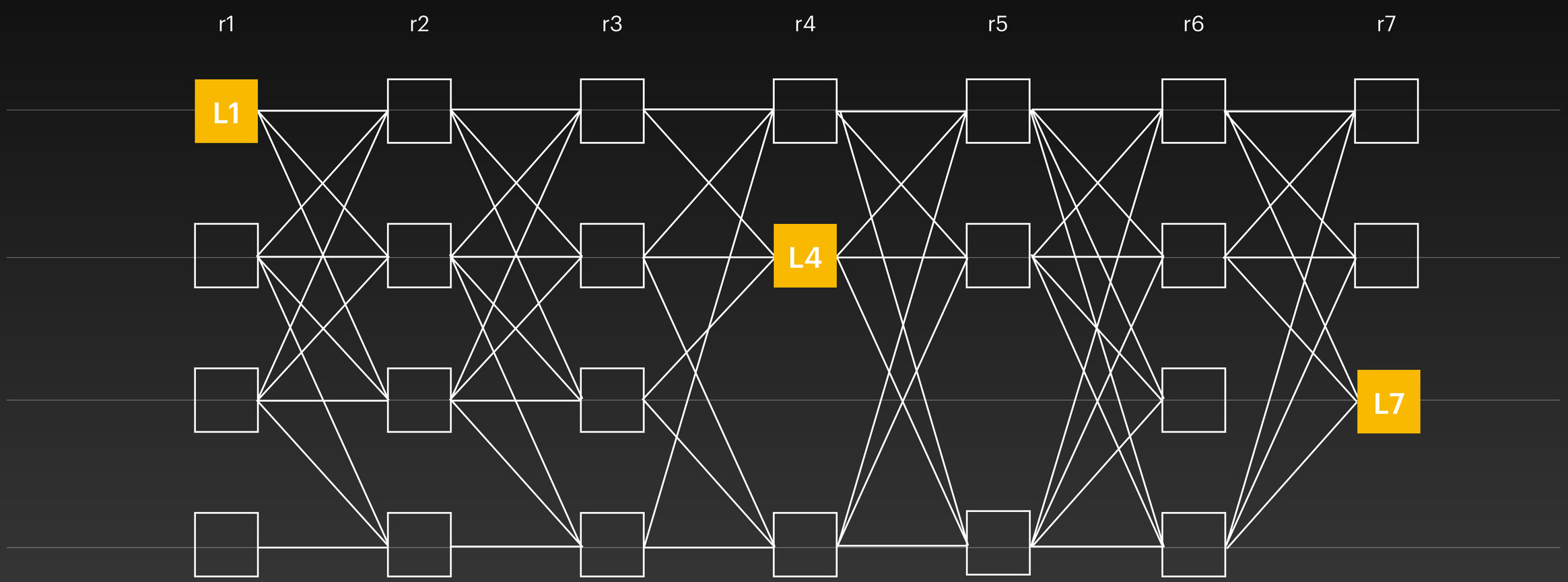


The Mysticeti DAG

Rule 3: All node run in parallel



The Mysticeti DAG

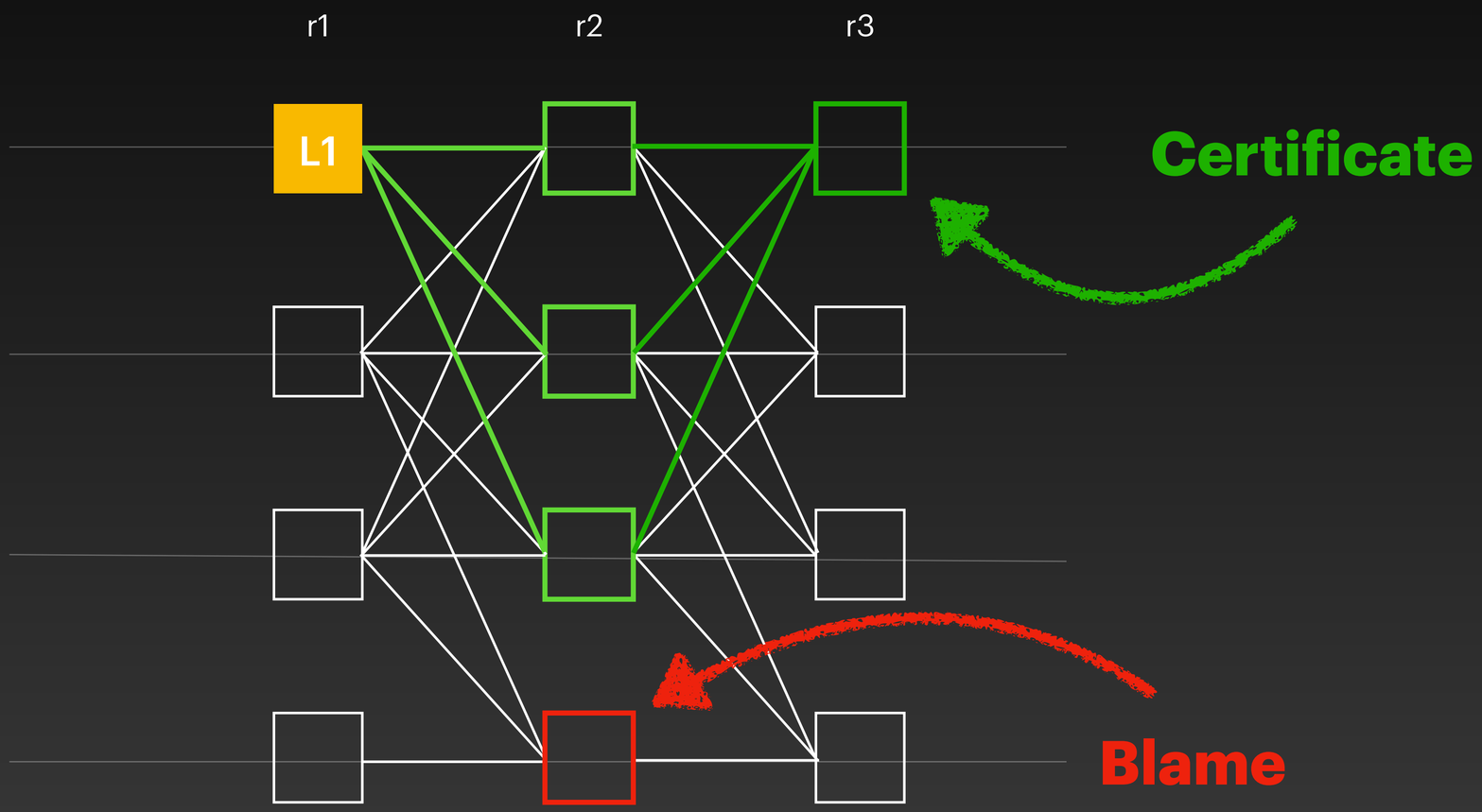


Main Ingredient:

All messages embedded in the DAG

- Fewer signatures
- Simpler synchronisation
- Define interpretable patterns on the DAG
- Run multiple protocols on the same DAG

Interpreting DAG Patterns



Two Protocols, One DAG

Mysticeti-C Consensus

- No rounds without leader
- Multiple leaders per round

Mysticeti-FPC Adding Fast Finality

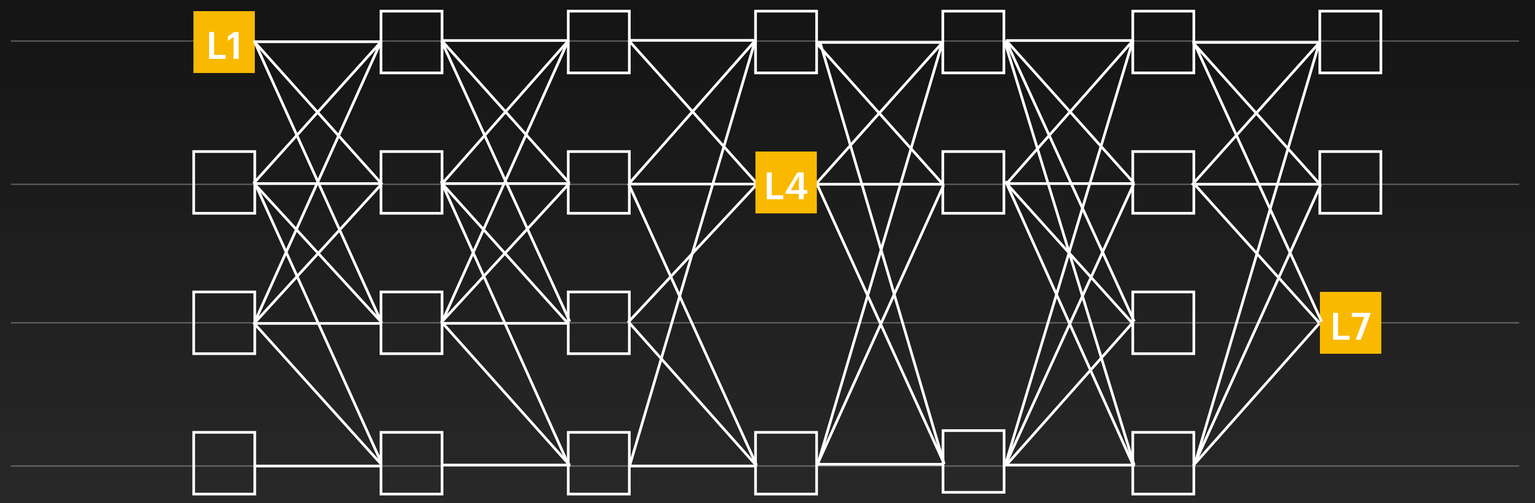
- Interpret BCB on DAG

Mysticeti-C

The consensus protocol

End Goal

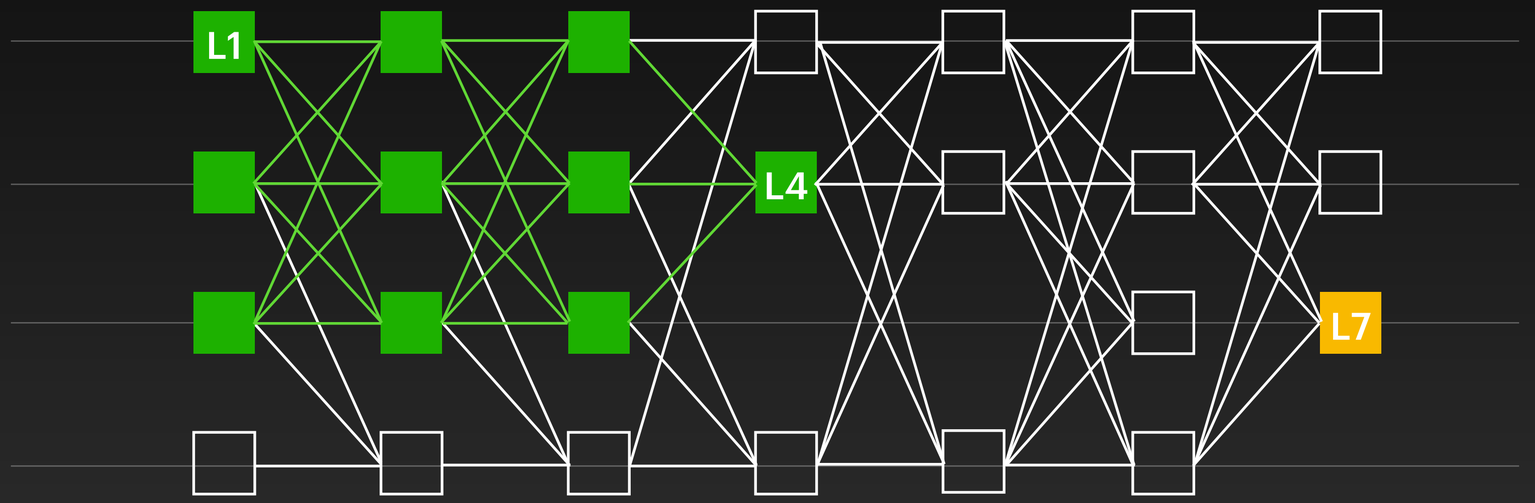
Ordering leaders



- We focus on ordering leaders: L1 L4 L7

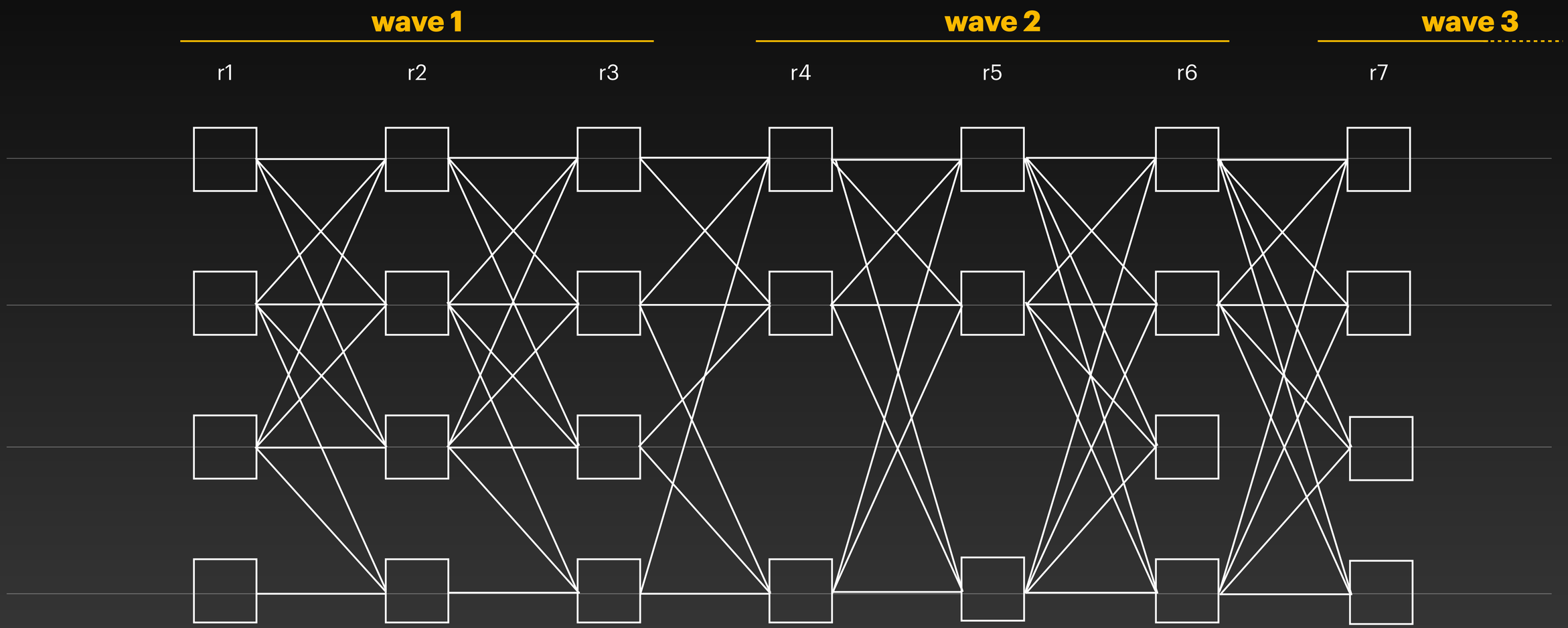
End Goal

Ordering leaders

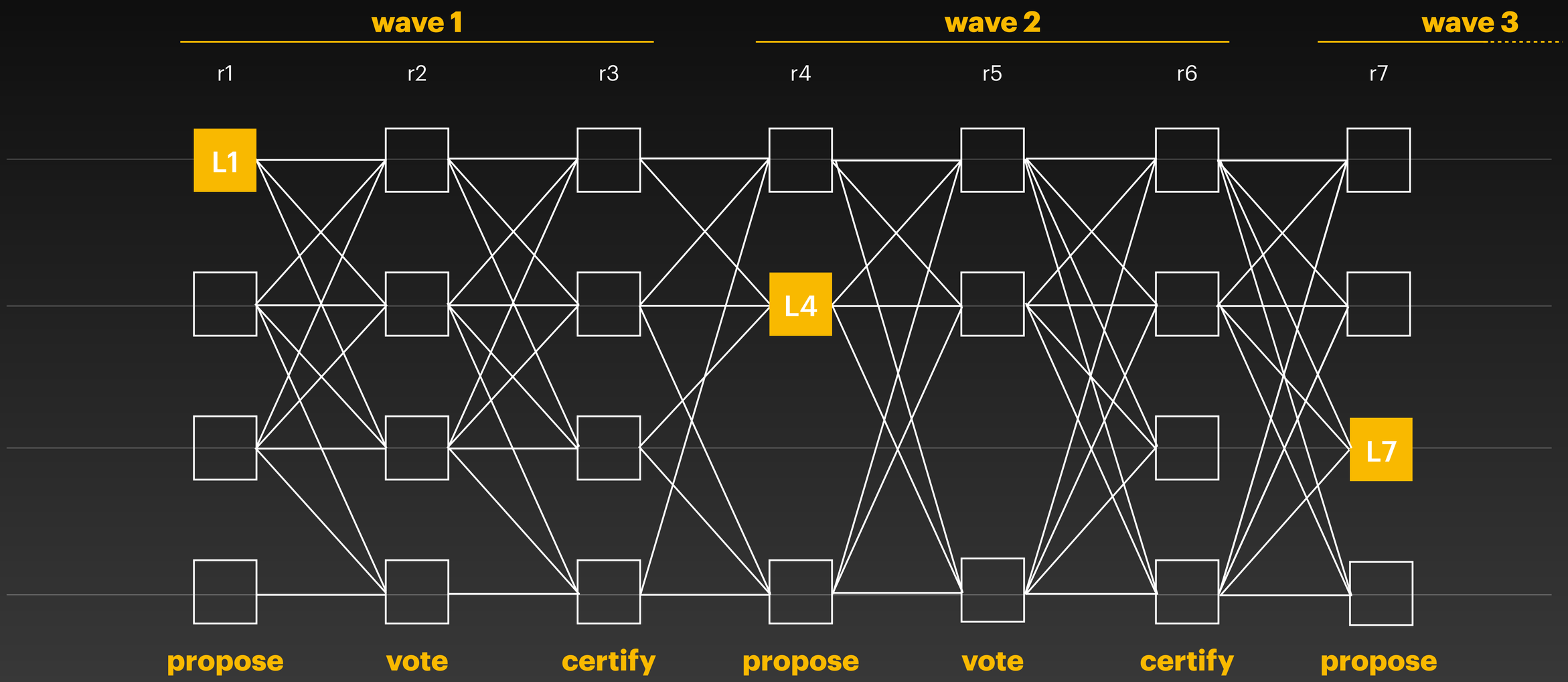


- We focus on ordering leaders: L1 L4 L7
- Linearising the sub-DAG is simple

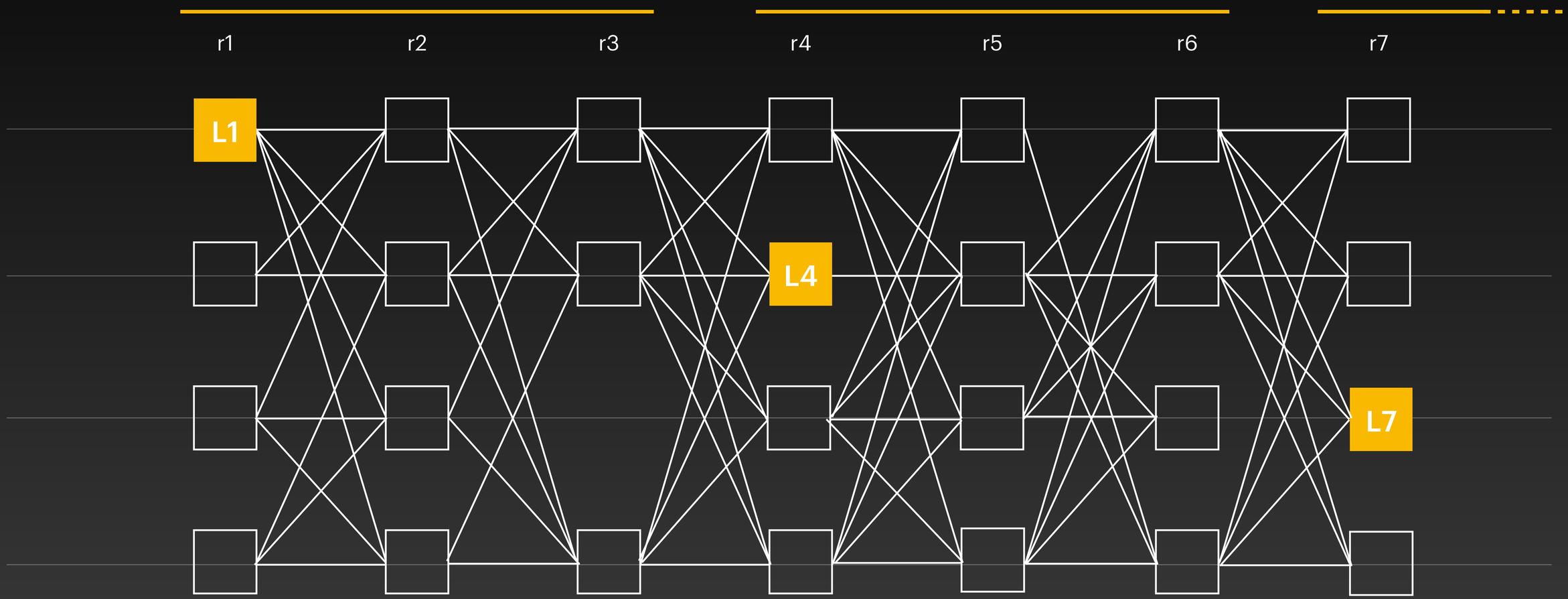
DAG Structure



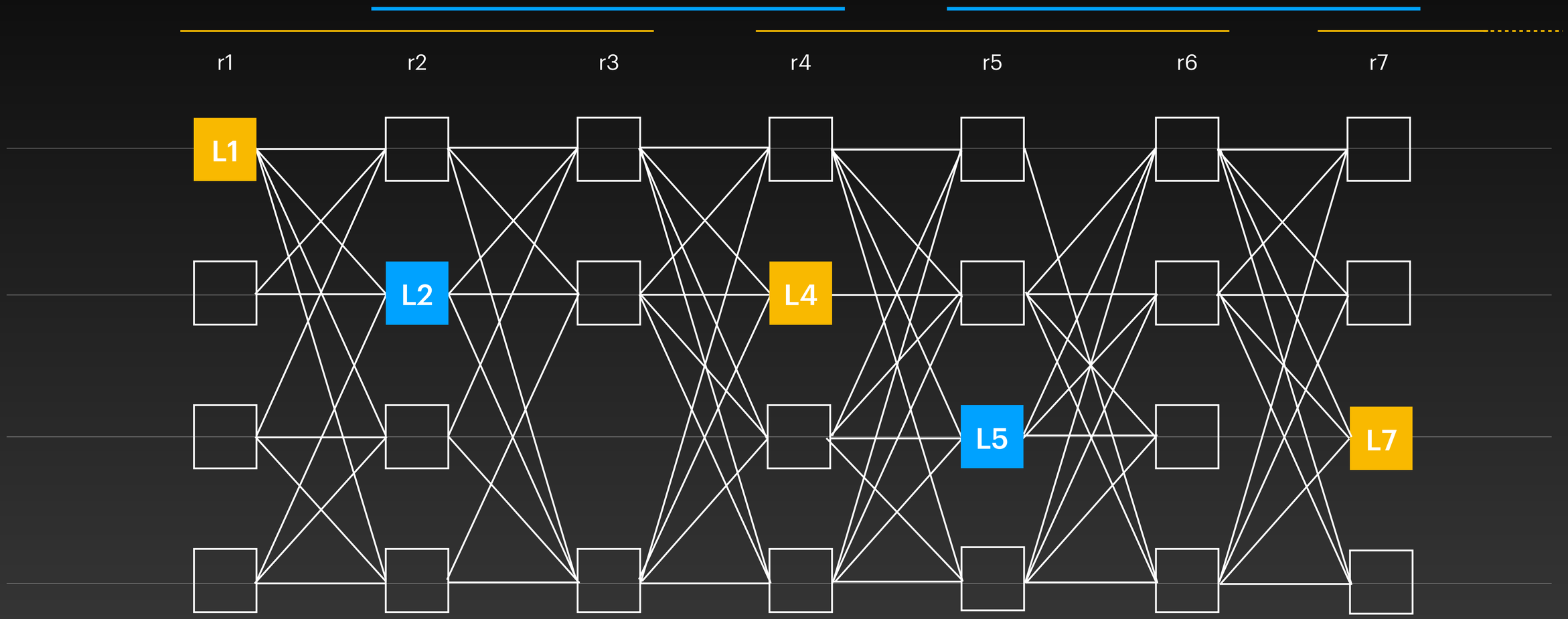
DAG Structure



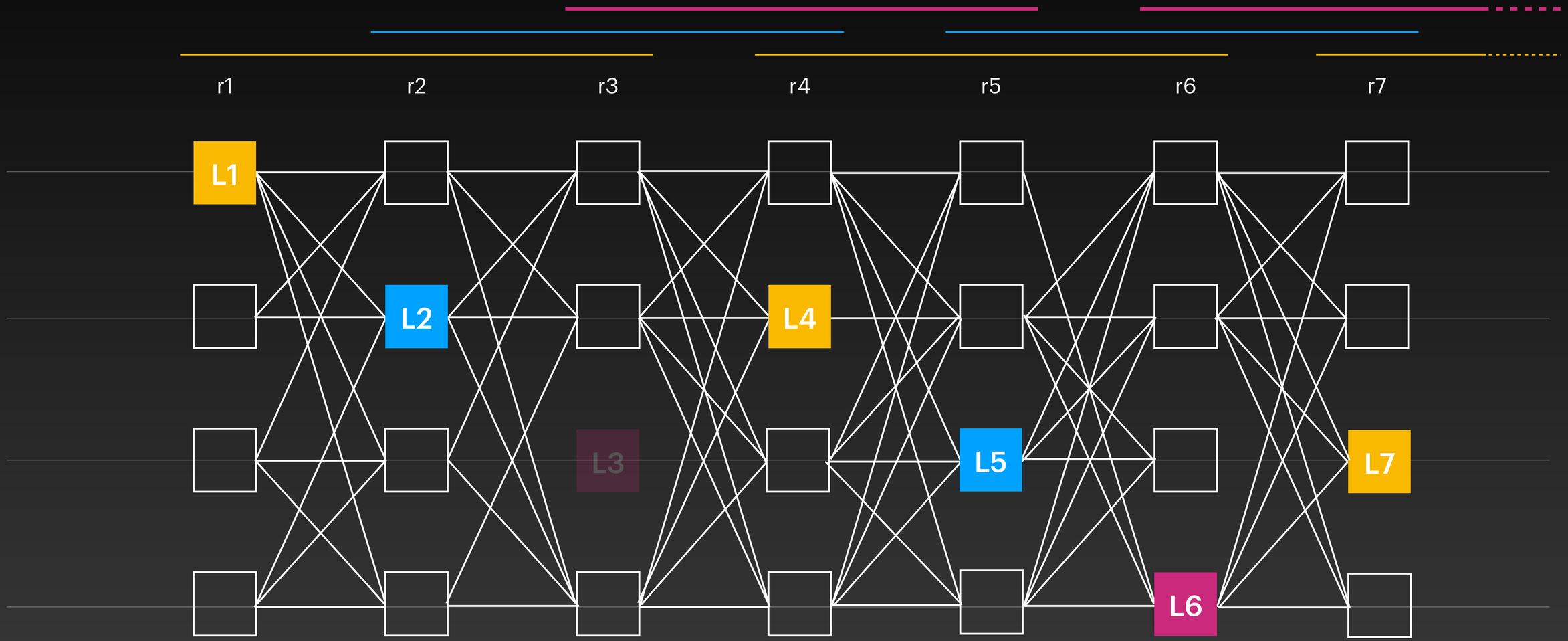
DAG Structure



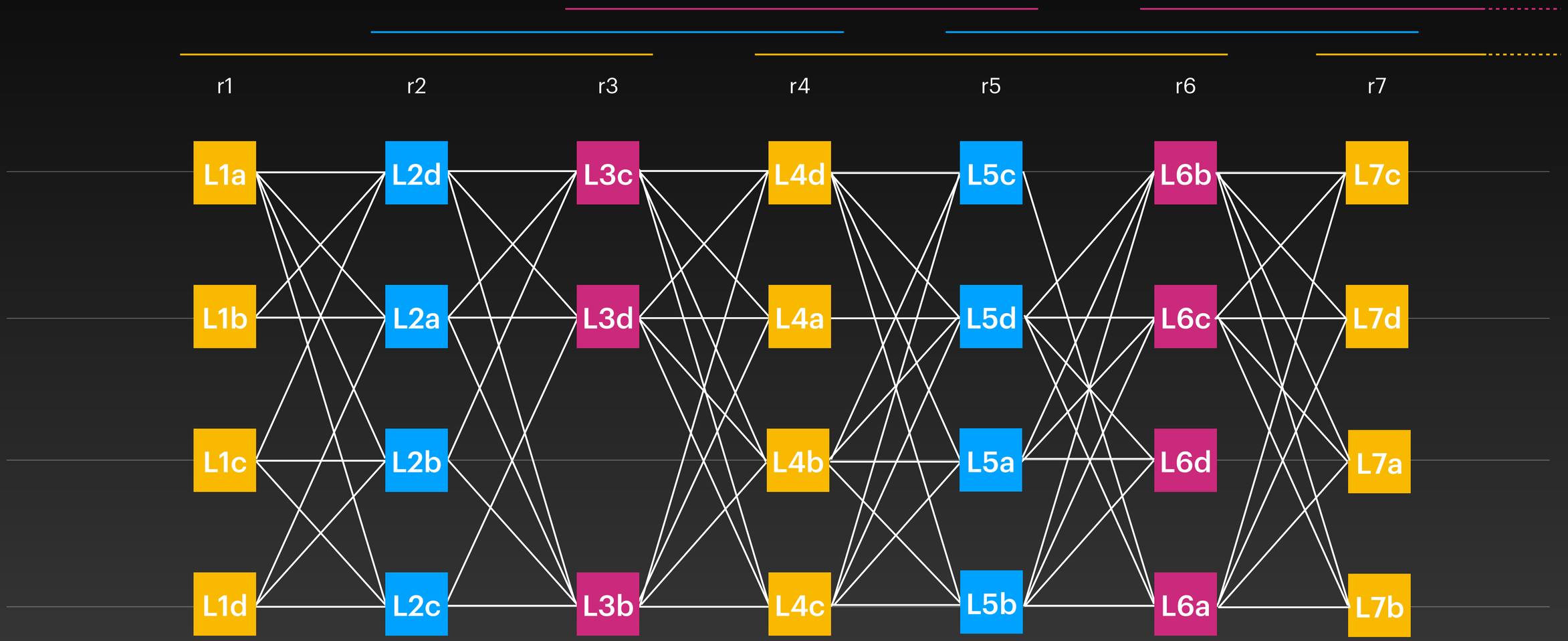
DAG Structure



DAG Structure

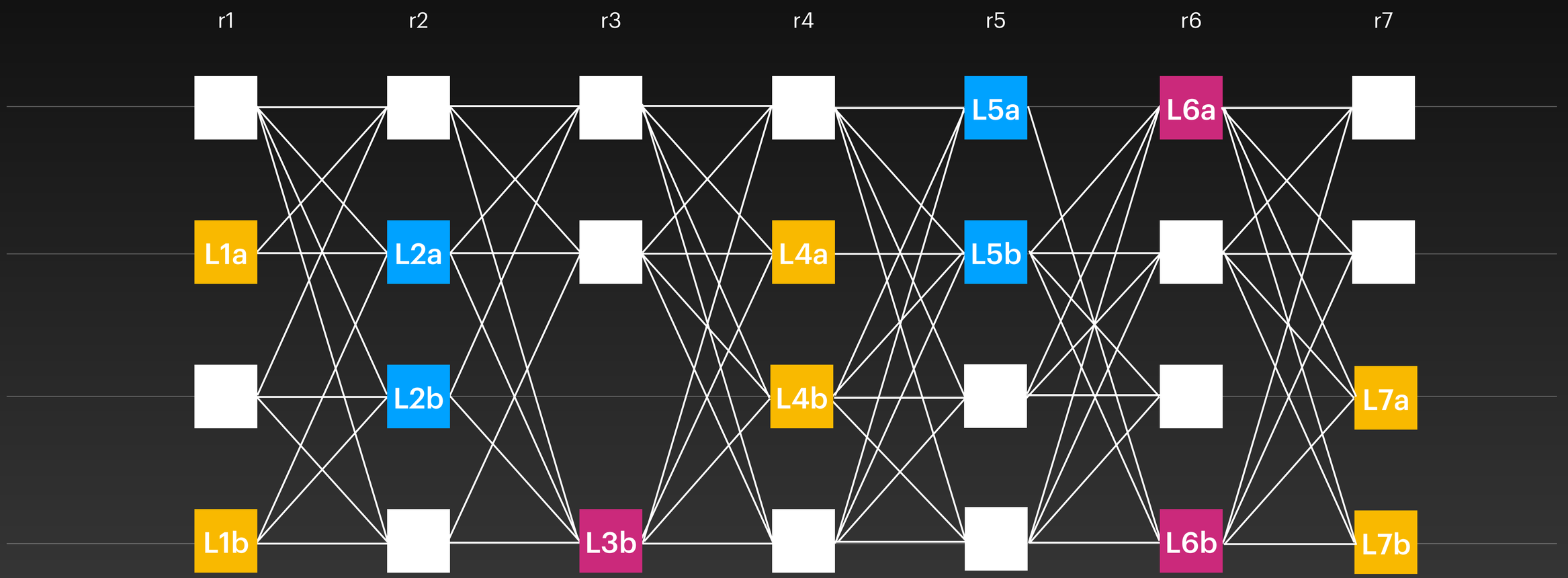


DAG Structure



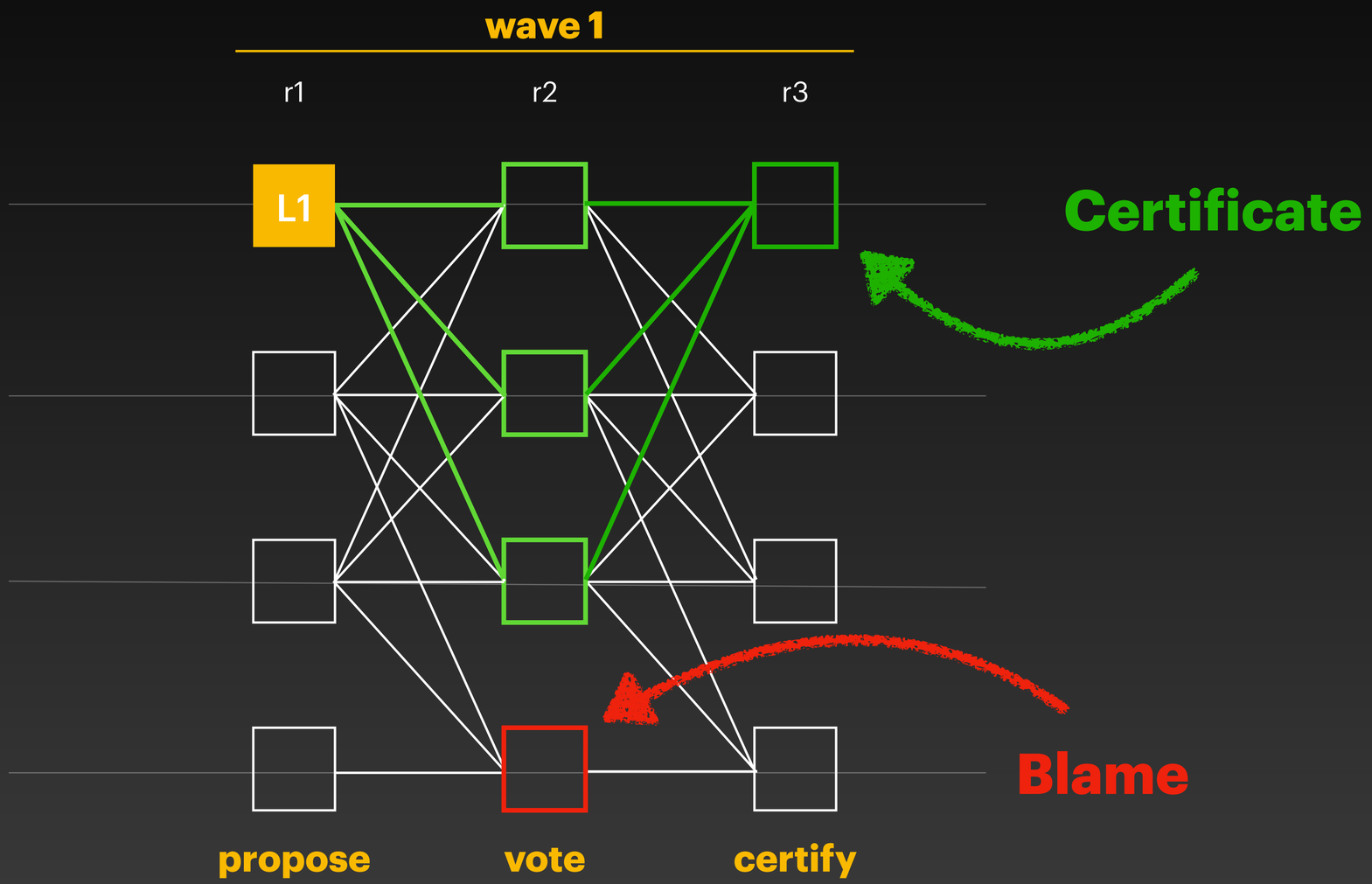
Practical Implementation

Select only 2 leaders per round



Interpreting DAG Patterns

Reminder



Direct Decision Rule

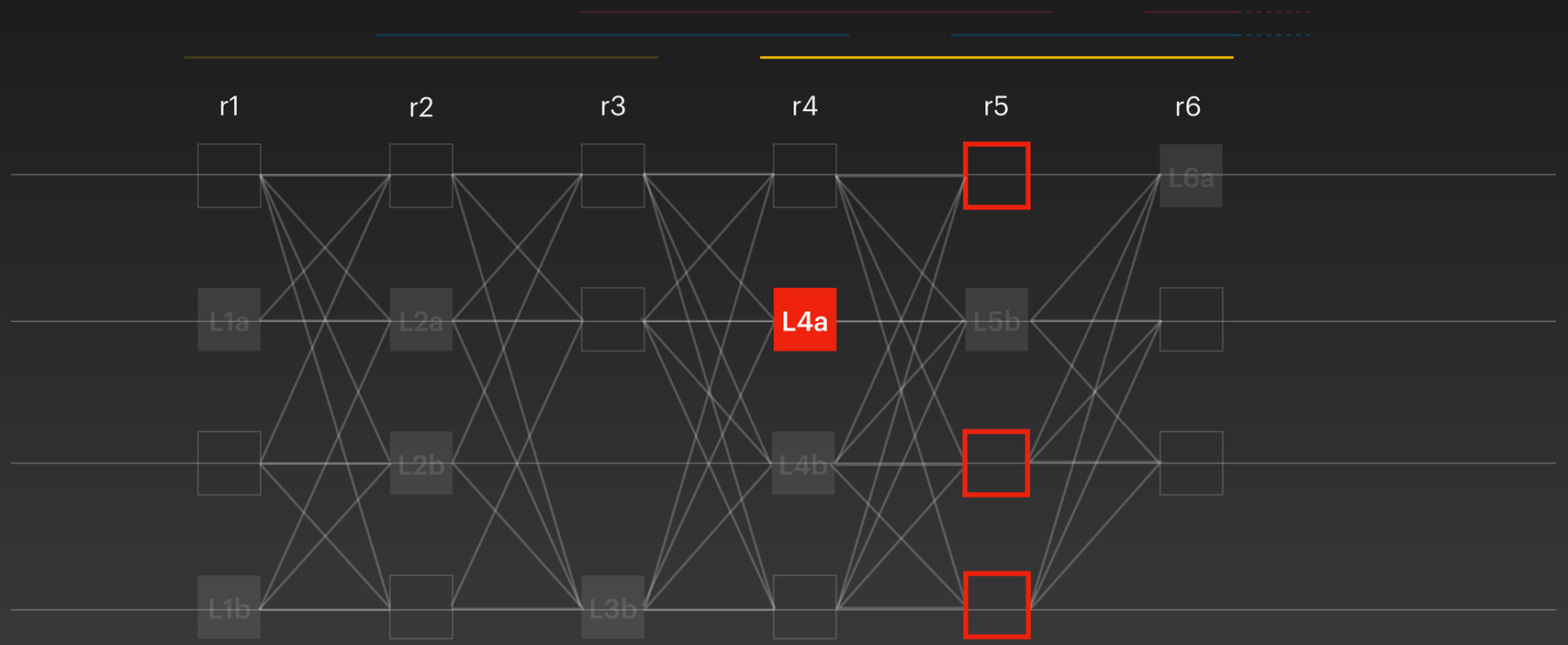
On each leader starting from highest round:

- **Skip** if 2f+1 blames
- **Commit** if 2f+1 certificates
- **Undecided** otherwise

Direct Decision Rule

On each leader starting from highest round:

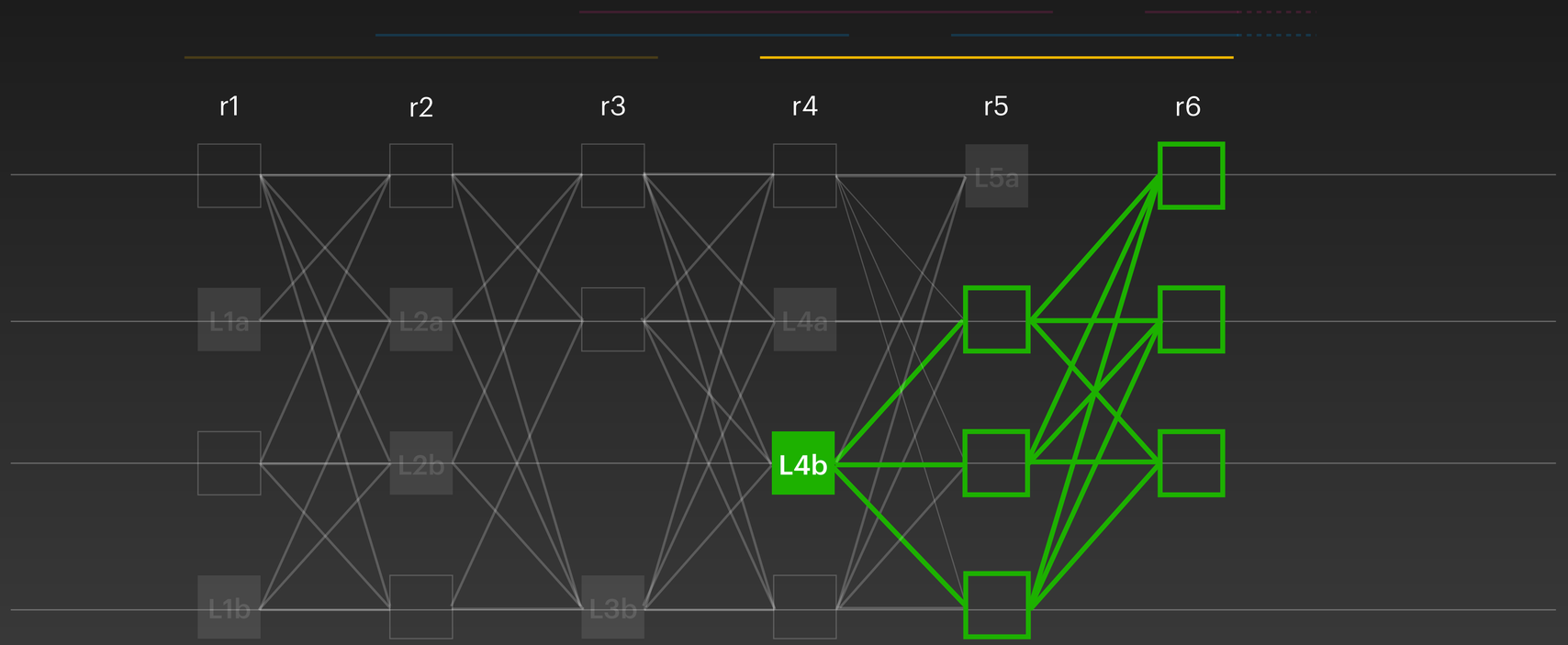
- **Skip** if 2f+1 blames
- **Commit** if 2f+1 certificates
- **Undecided** otherwise



Direct Decision Rule

On each leader starting from highest round:

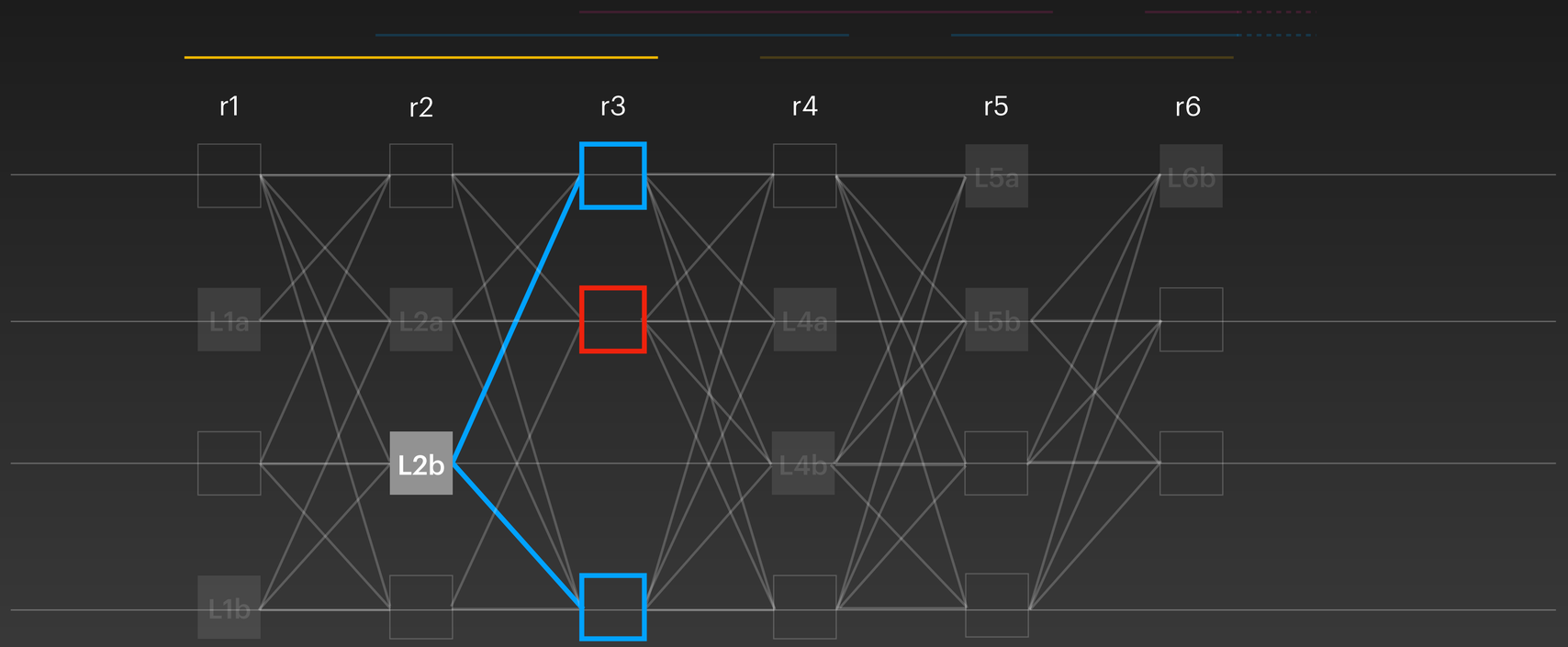
- **Skip** if 2f+1 blames
- **Commit** if 2f+1 certificates
- **Undecided** otherwise



Direct Decision Rule

On each leader starting from highest round:

- **Skip** if 2f+1 blames
- **Commit** if 2f+1 certificates
- **Undecided** otherwise

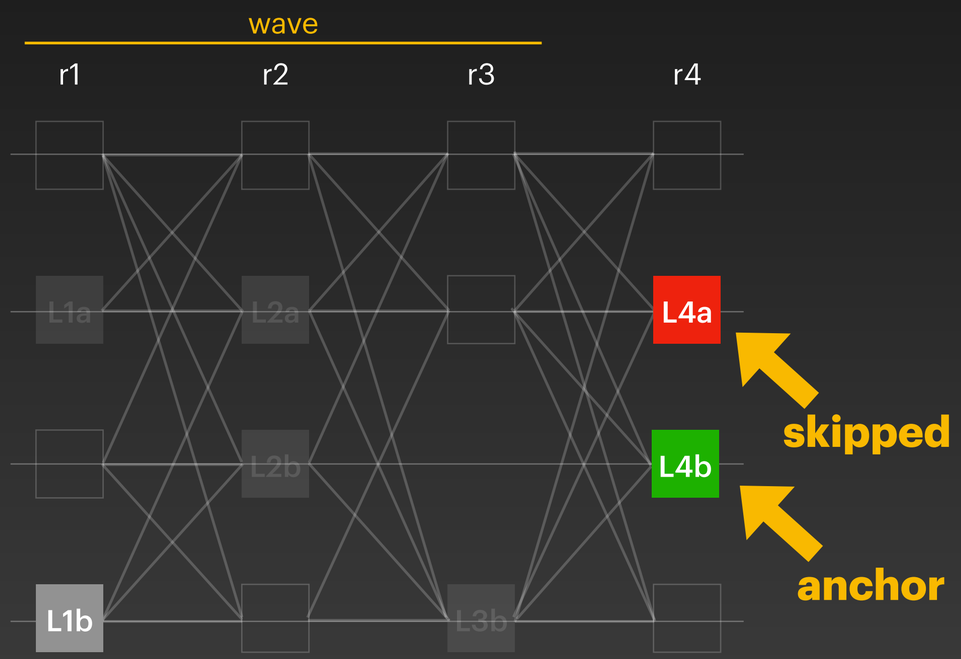


Indirect Decision Rule

Indirect Decision Rule

1. Find Anchor

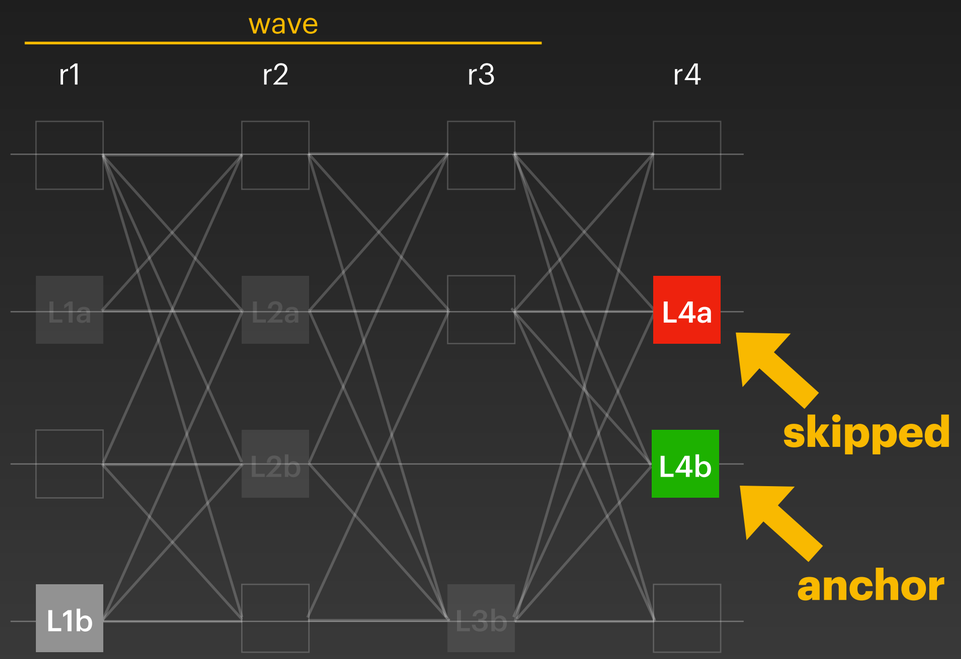
- First block with round > r+2 that is **Commit** or **Undecided**



Indirect Decision Rule

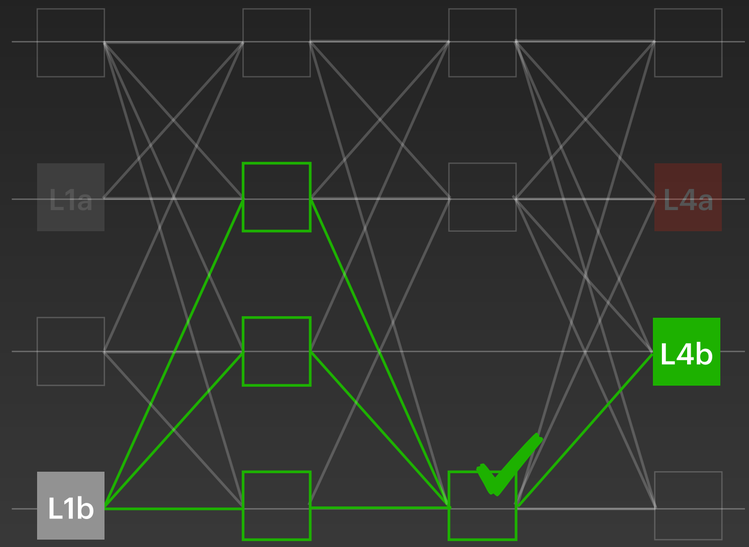
1. Find Anchor

- First block with round > r+2 that is **Commit** or **Undecided**

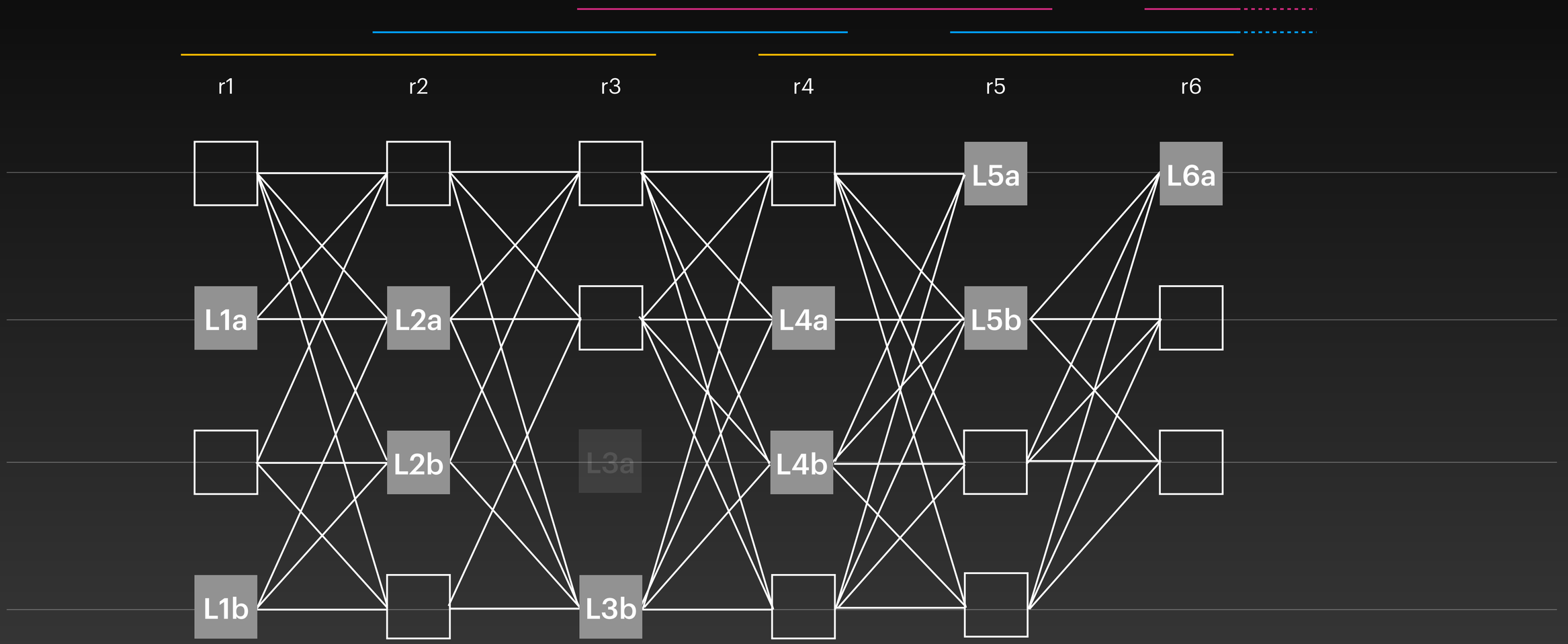


2. Certified link

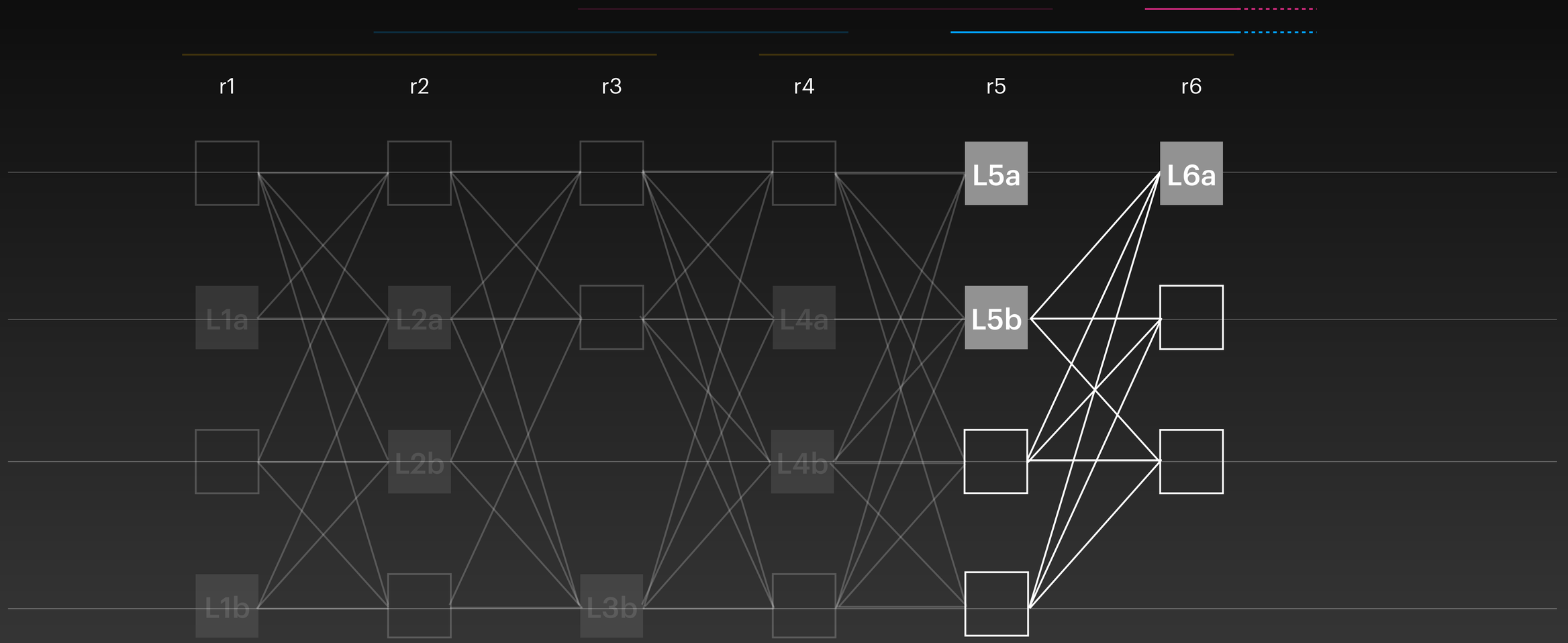
- **Commit** if `B <-> certified link <-> A` otherwise **Skip**



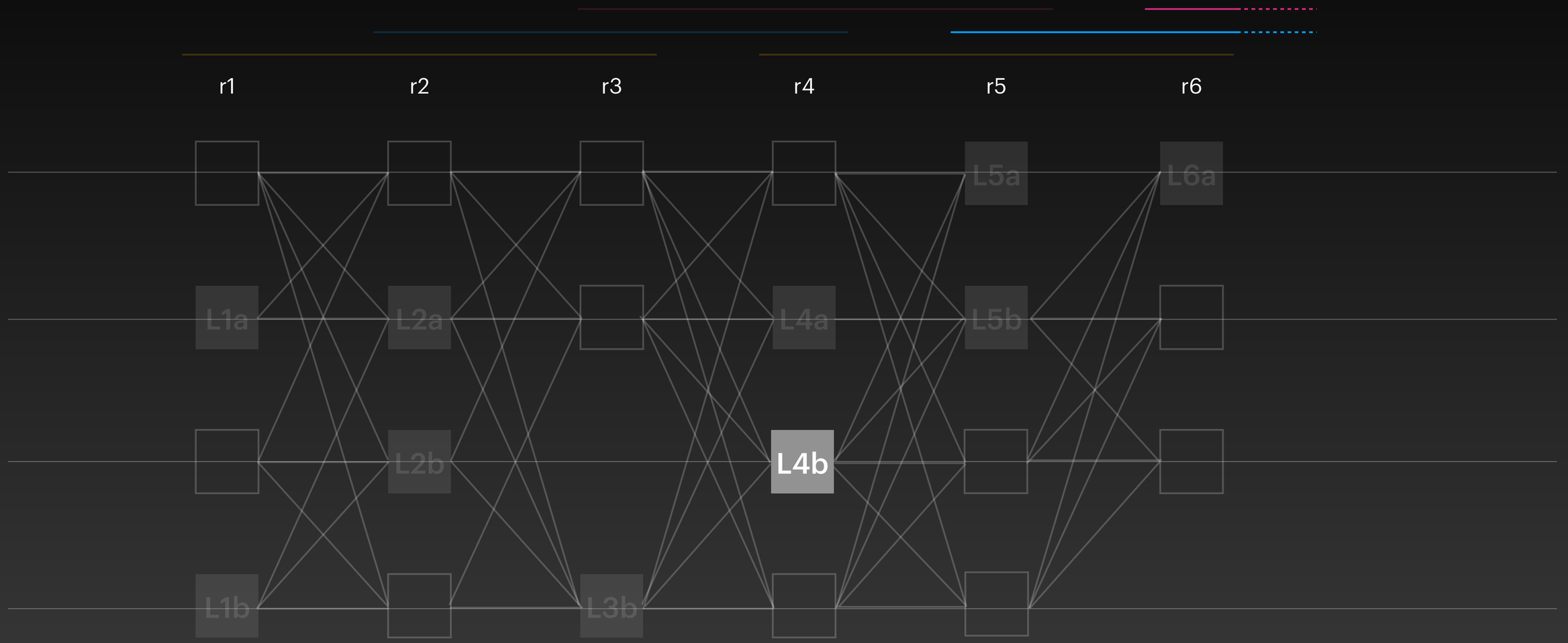
All Start at Undecided



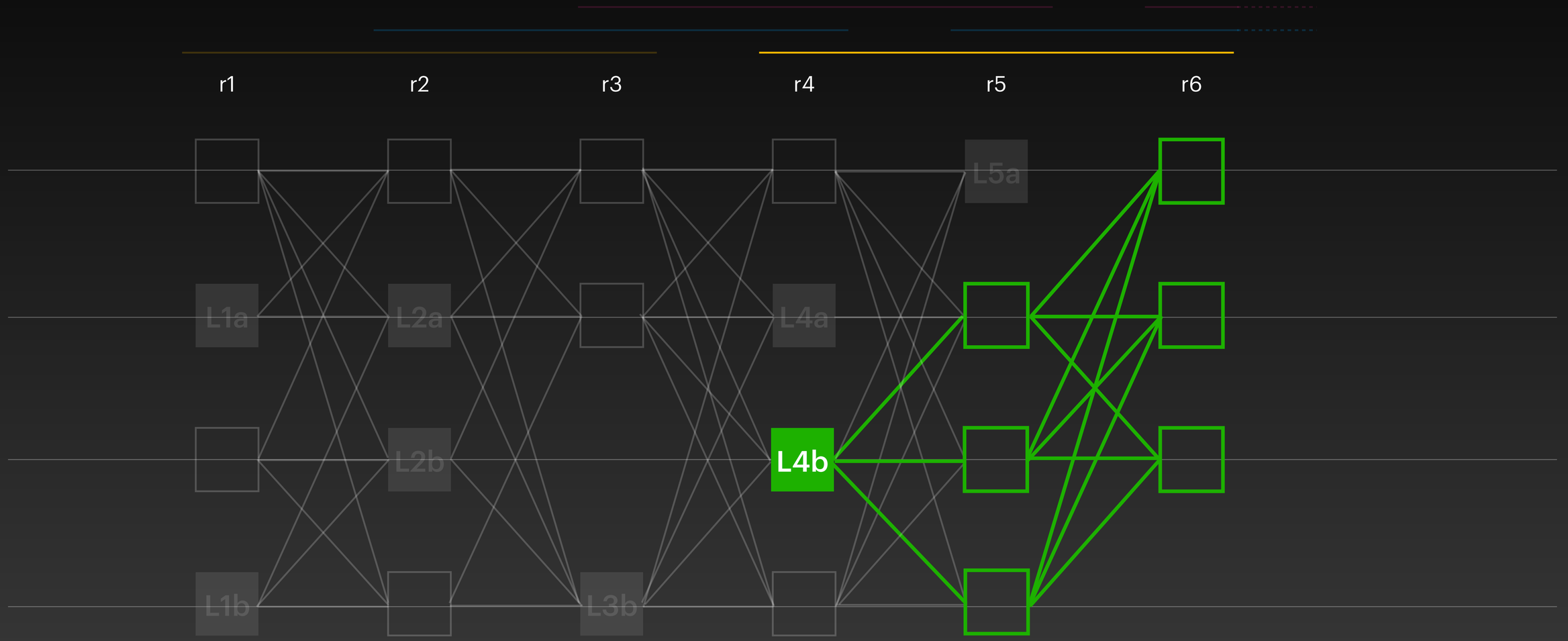
Ignore Incomplete Waves



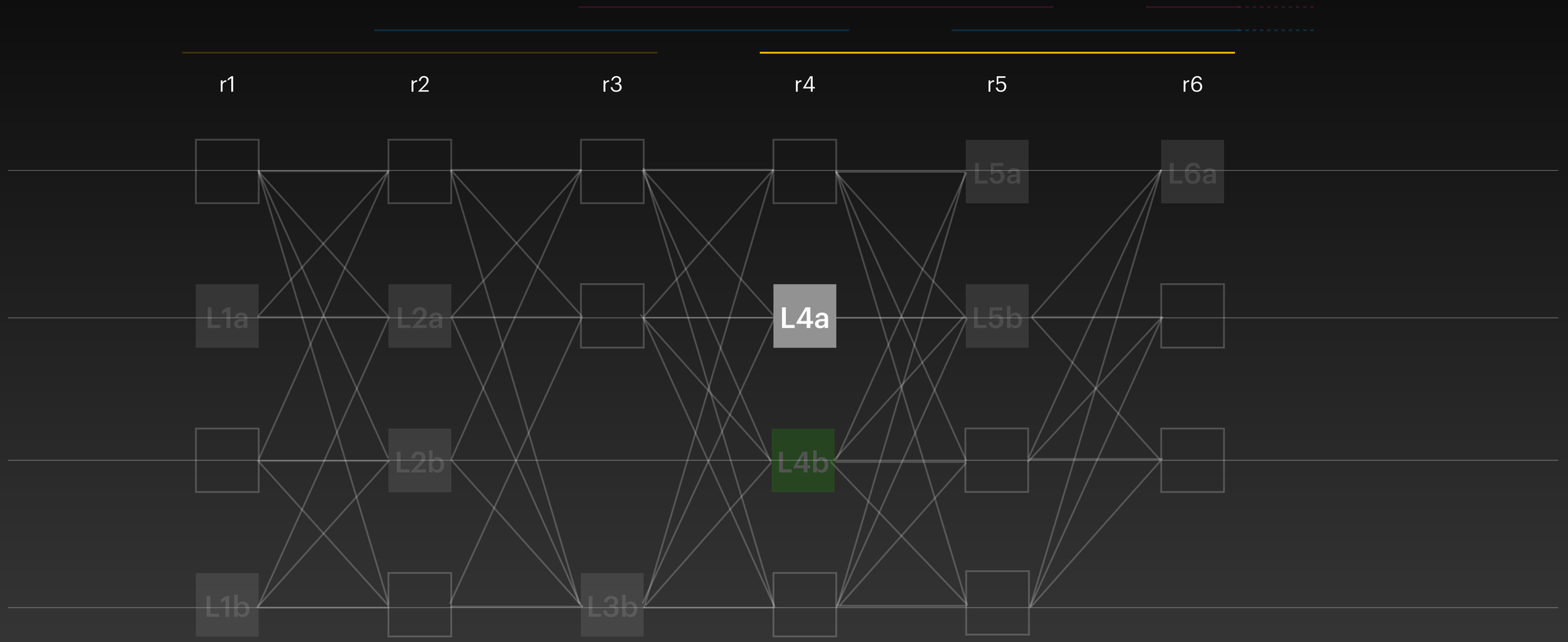
Apply Direct Rule



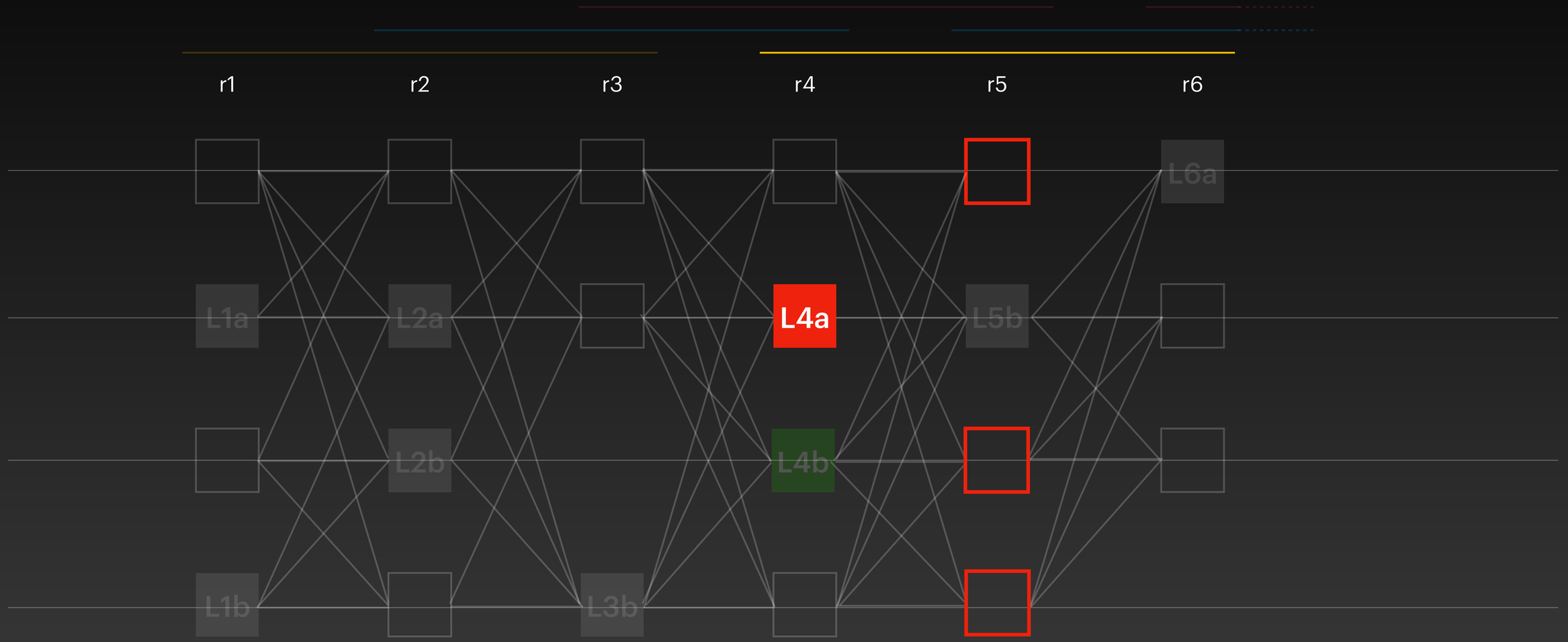
Apply Direct Rule



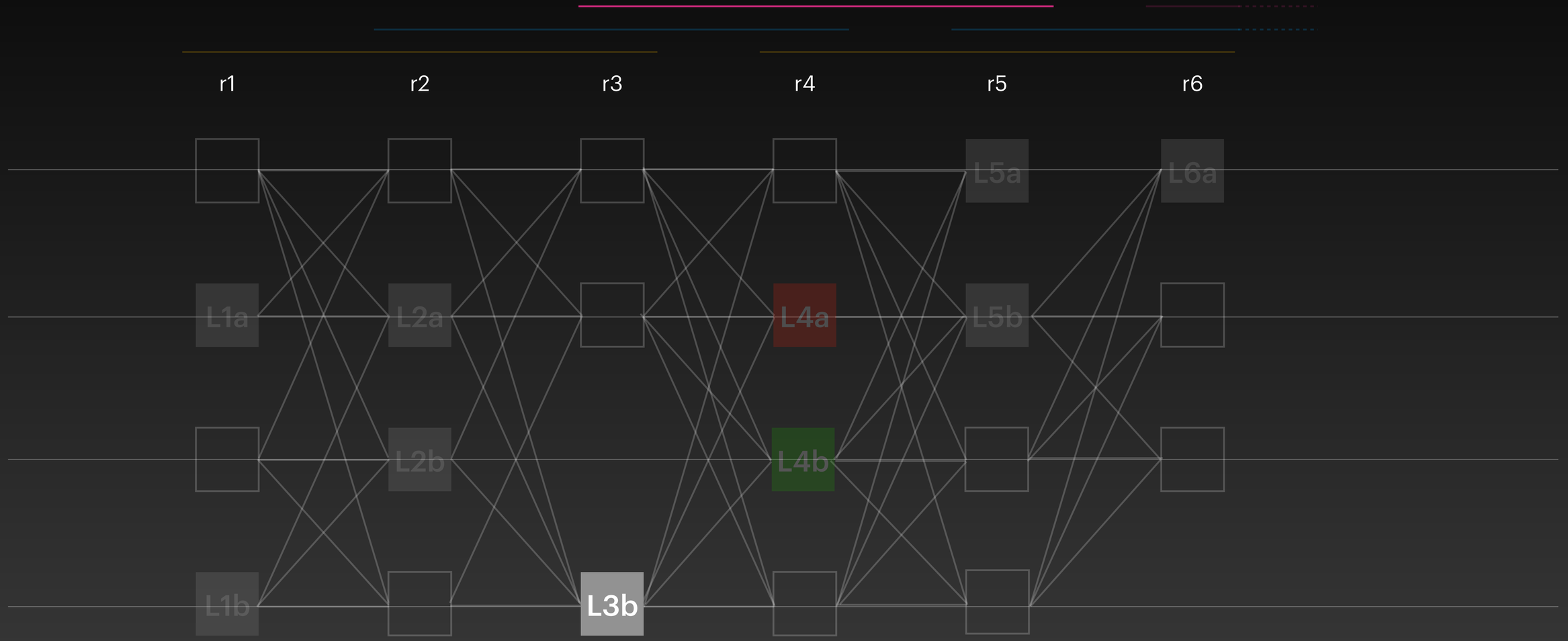
Apply Direct Rule



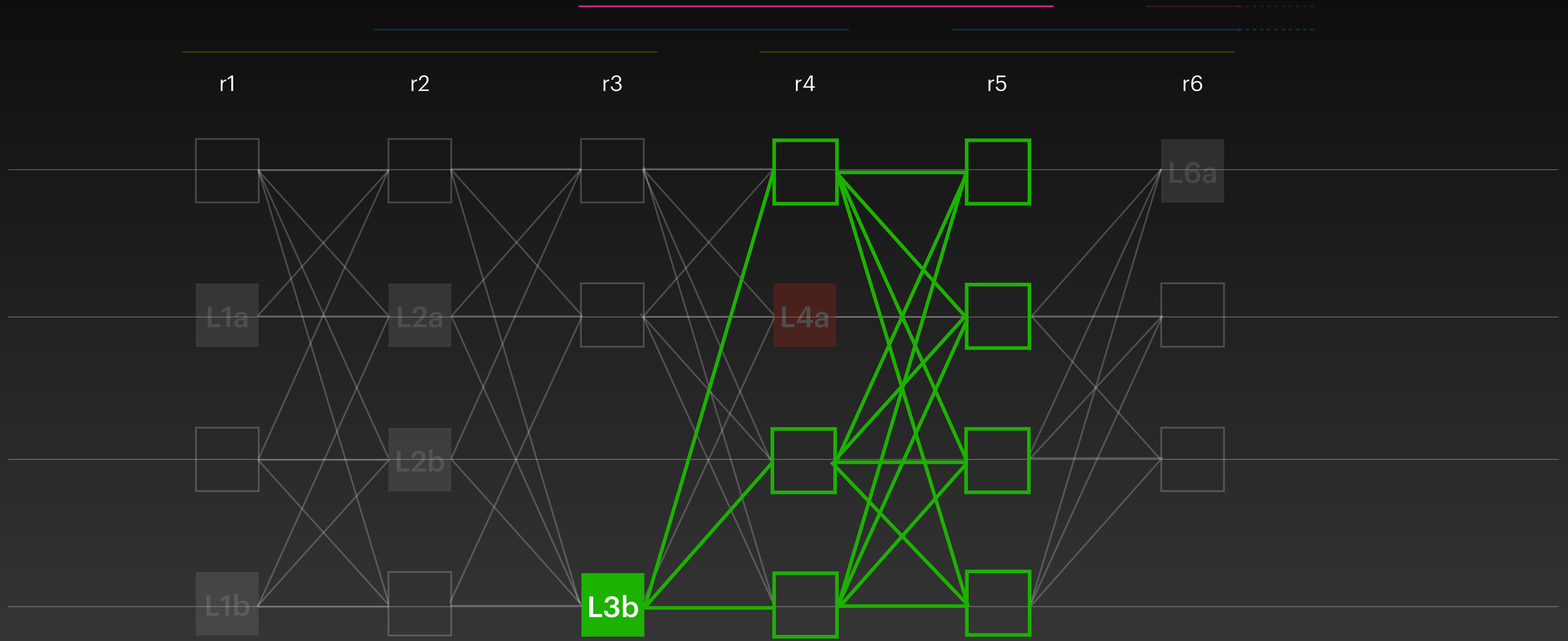
Apply Direct Rule



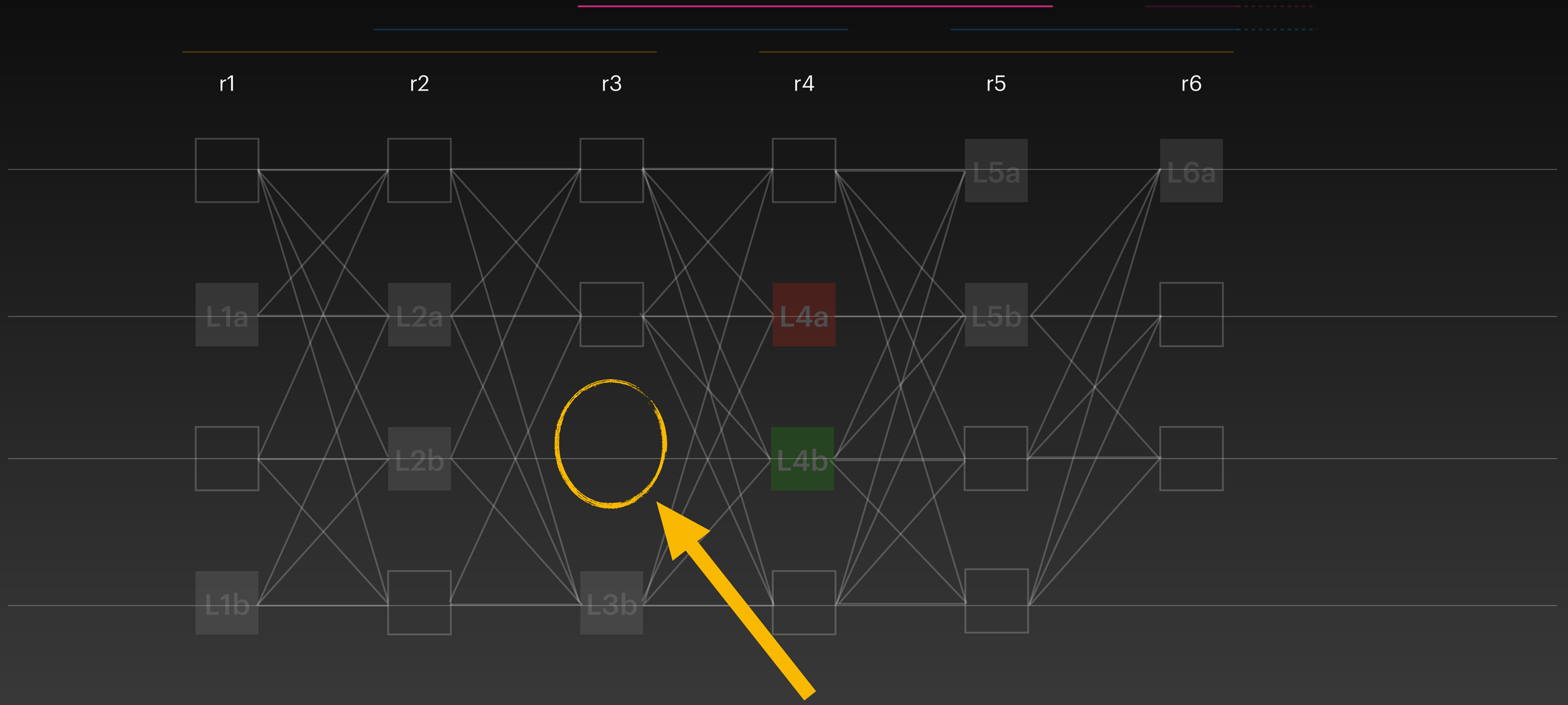
Apply Direct Rule



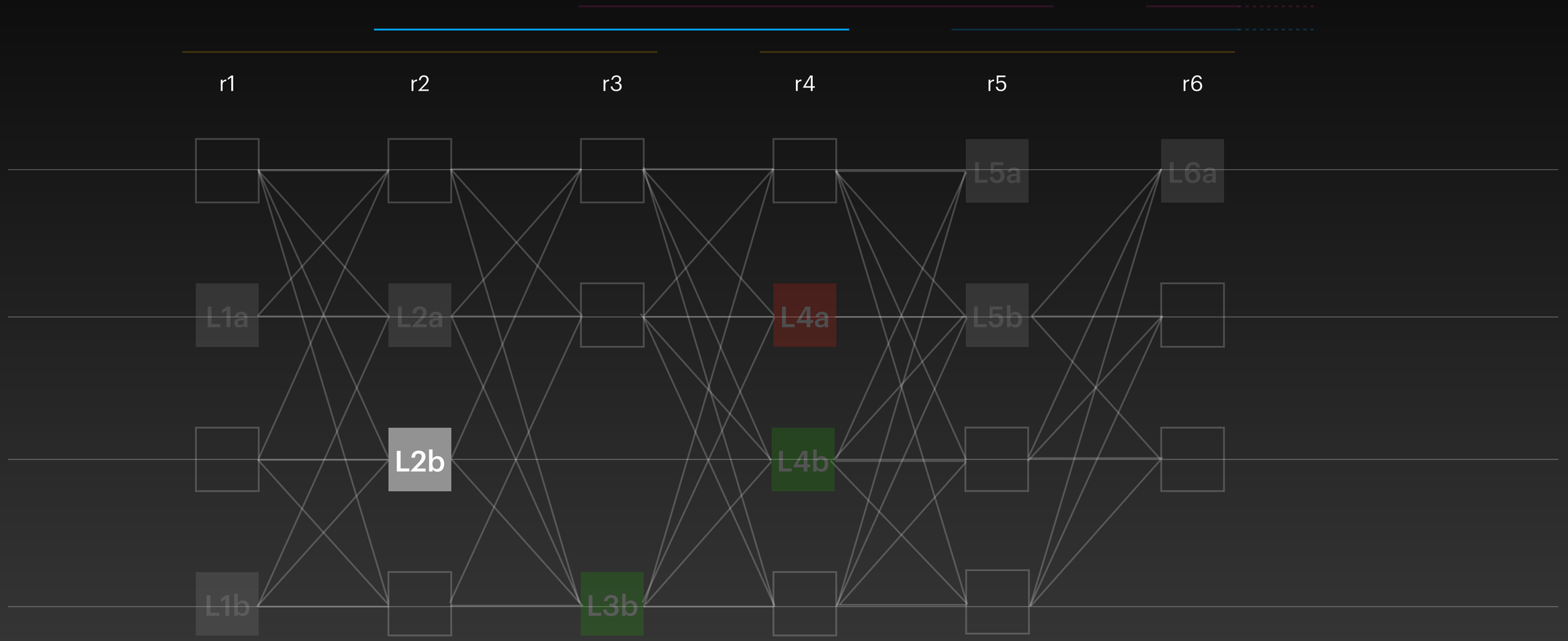
Apply Direct Rule



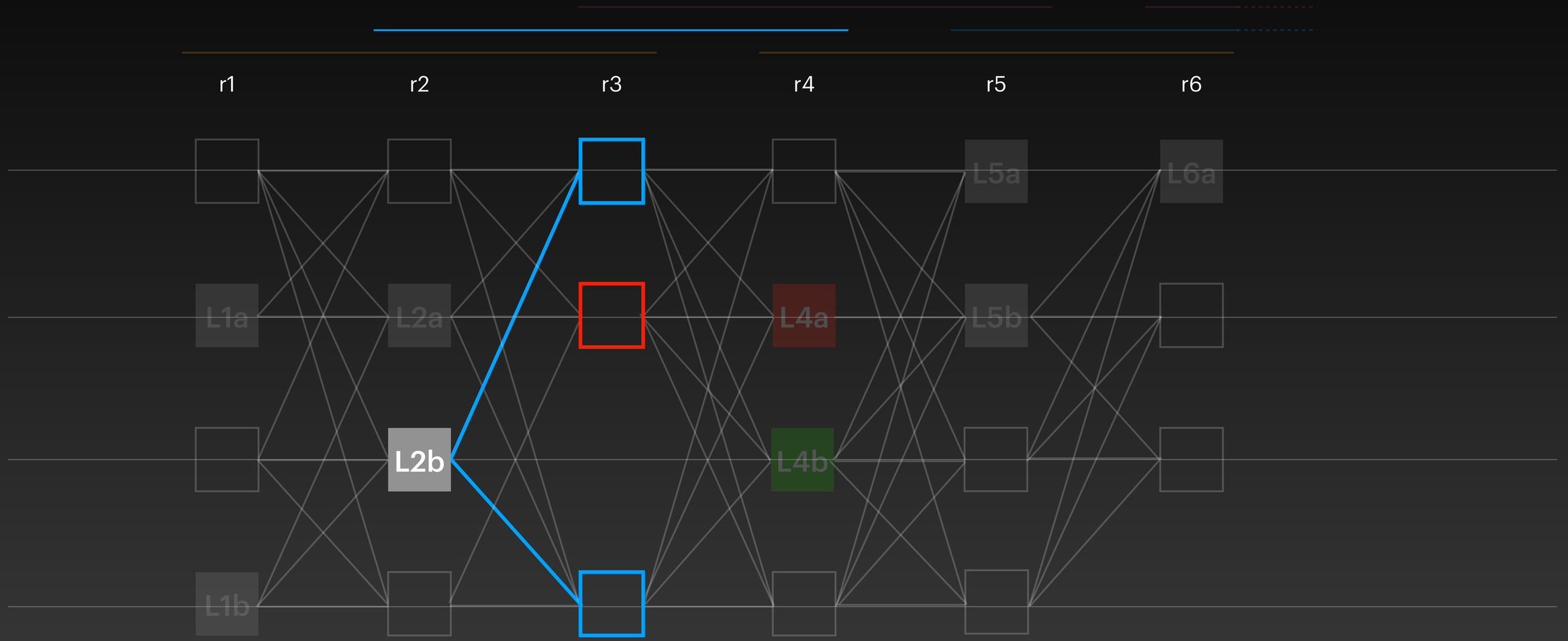
Ignore Missing Leader



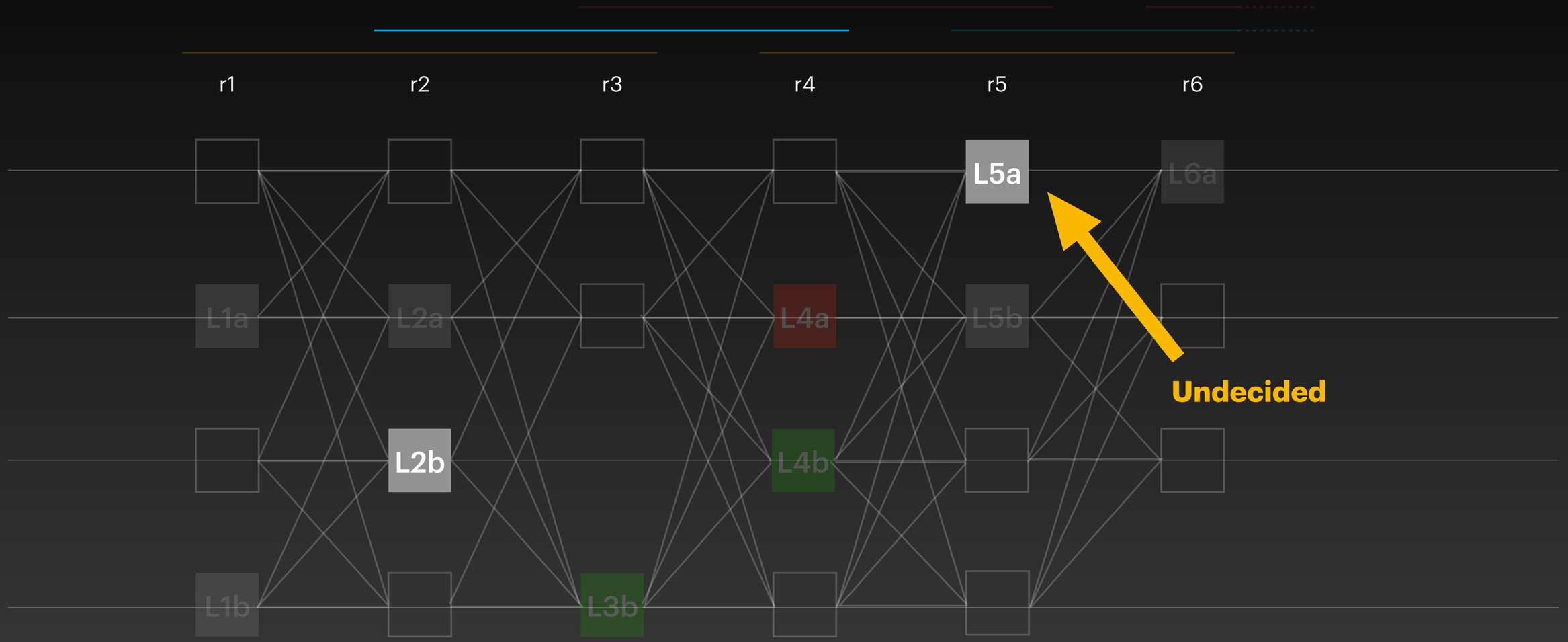
Apply Direct Rule



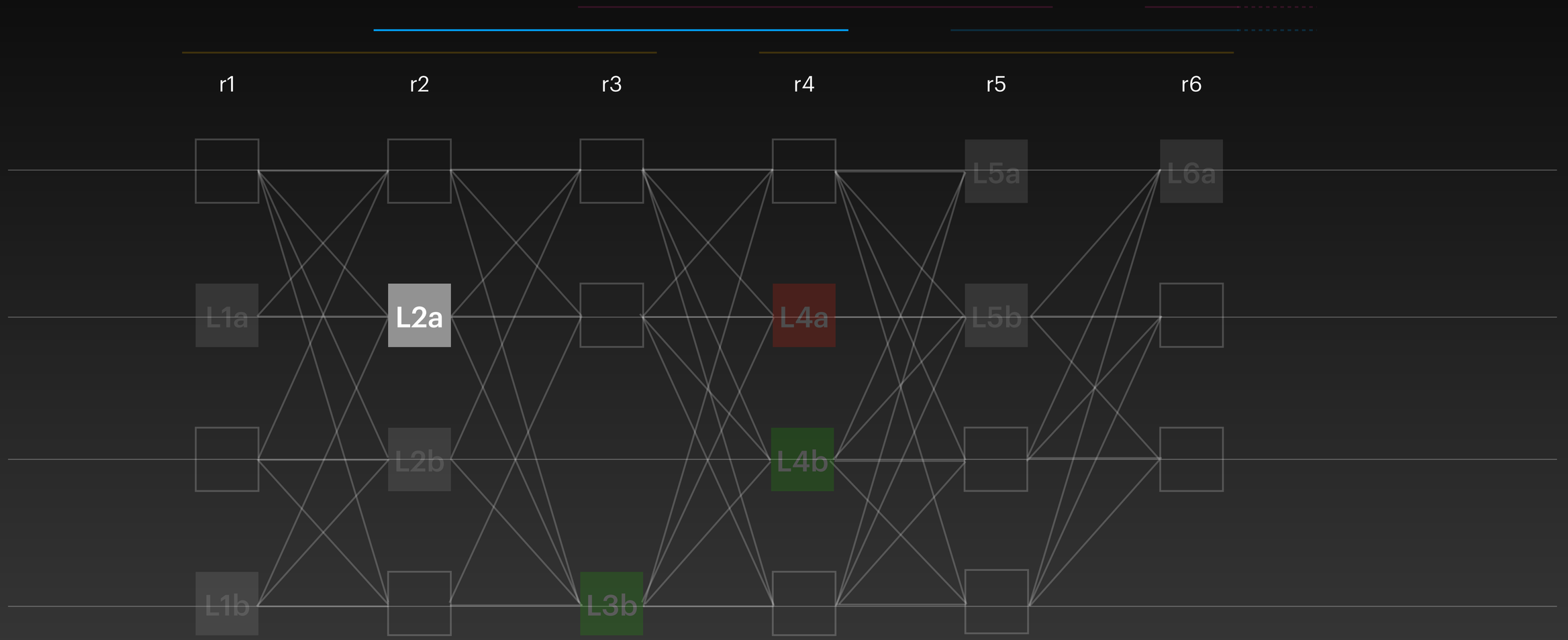
Apply Direct Rule



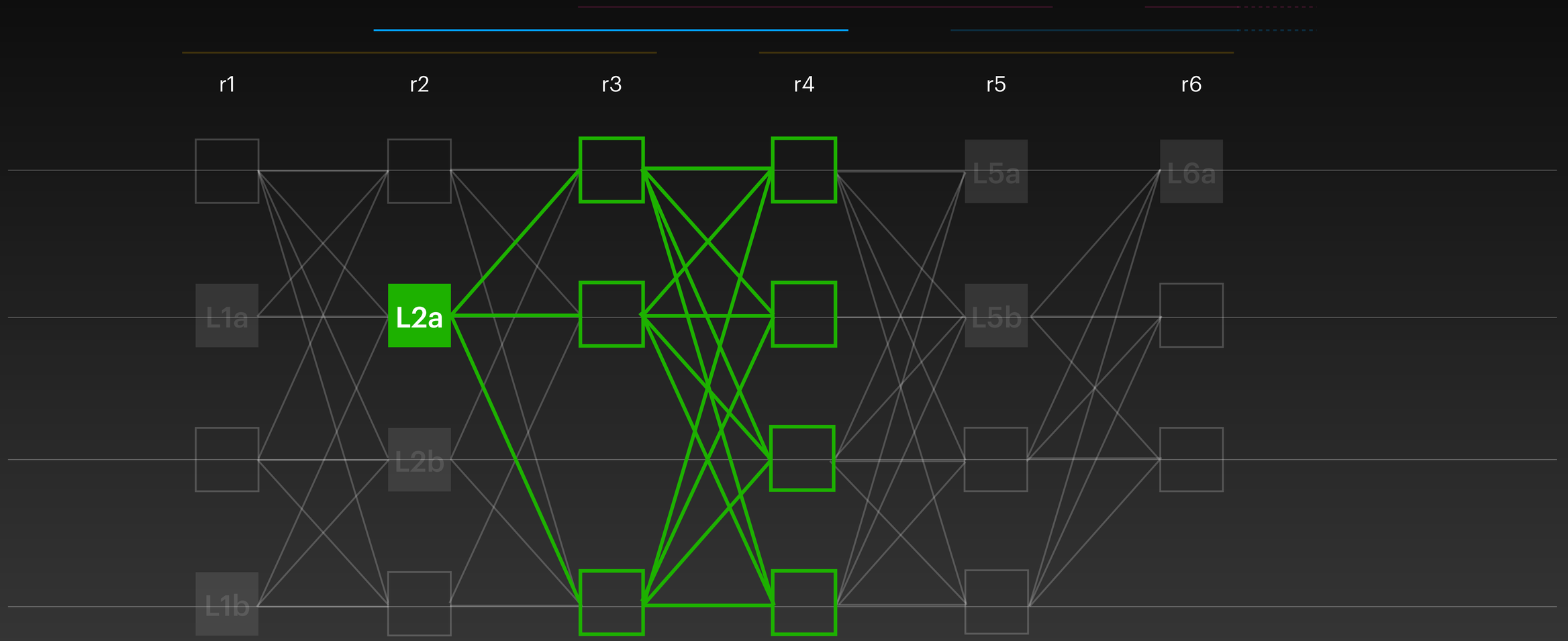
Apply Indirect Rule



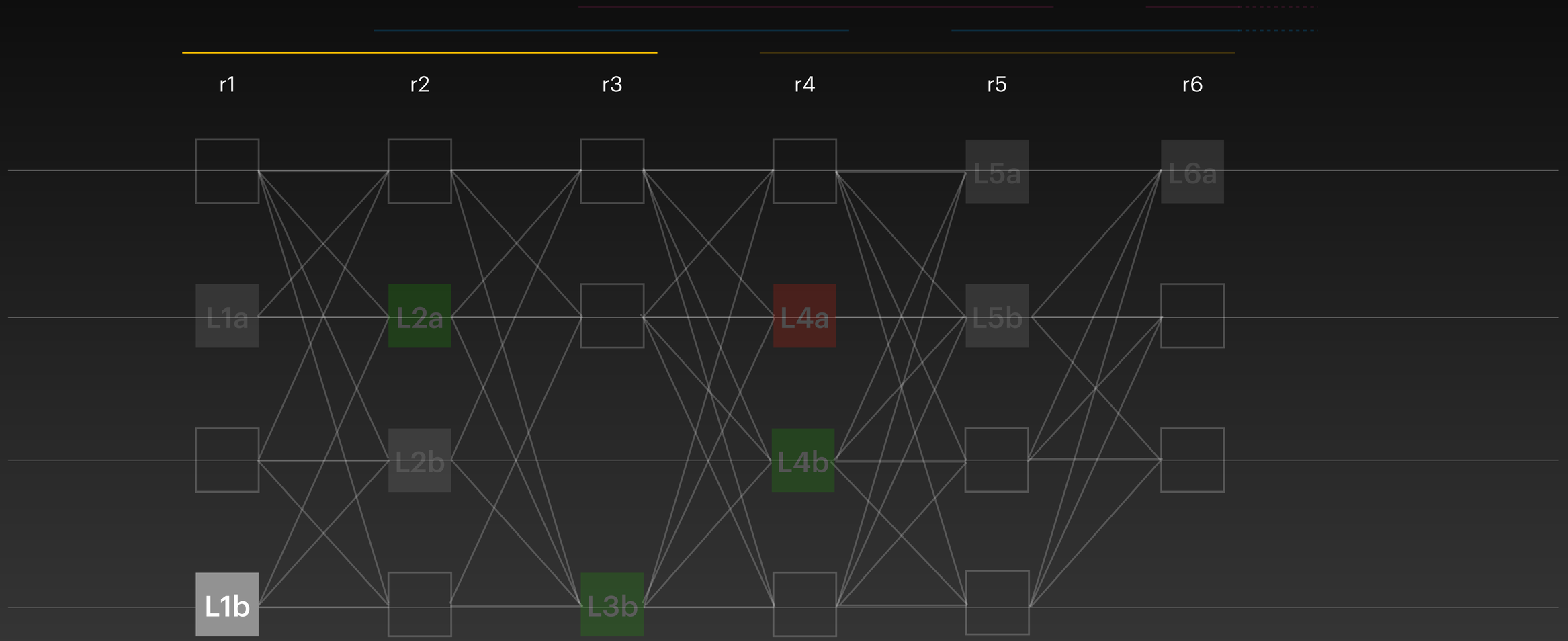
Apply Direct Rule



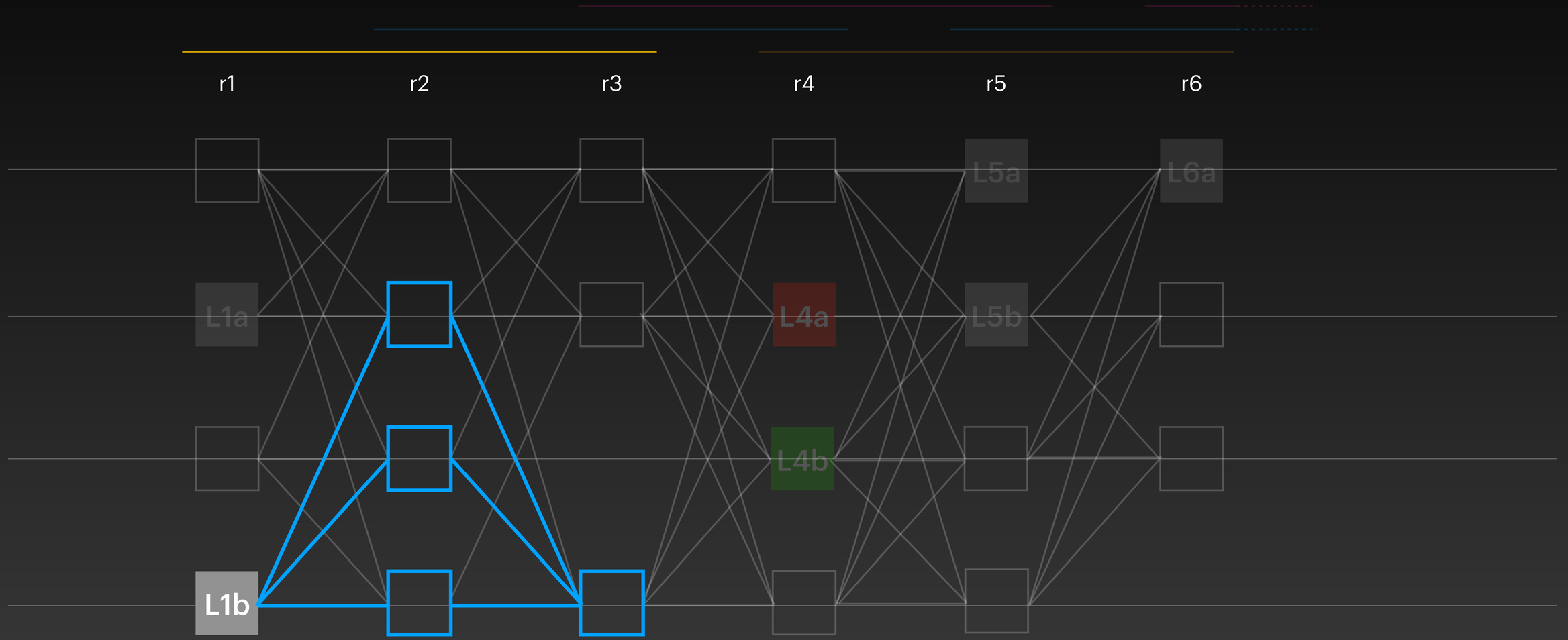
Apply Direct Rule



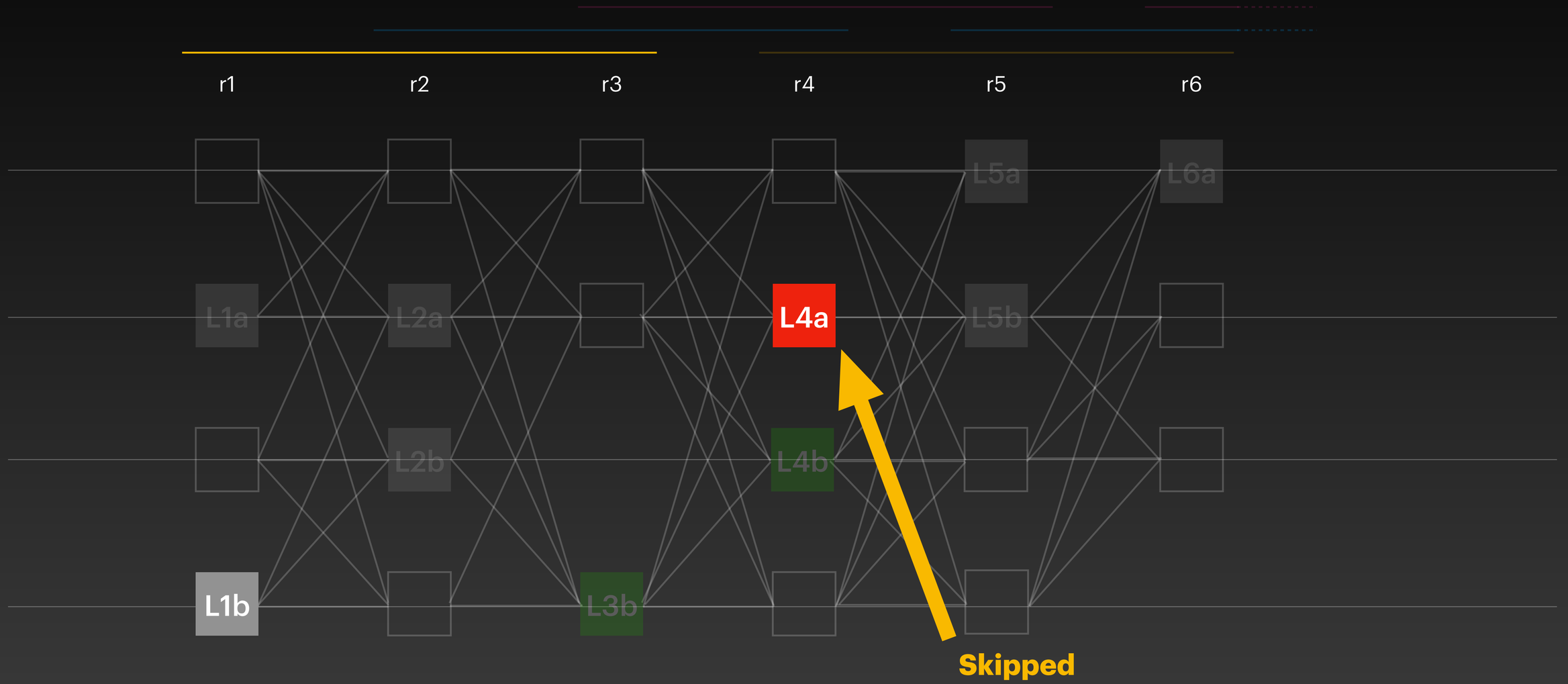
Apply Direct Rule



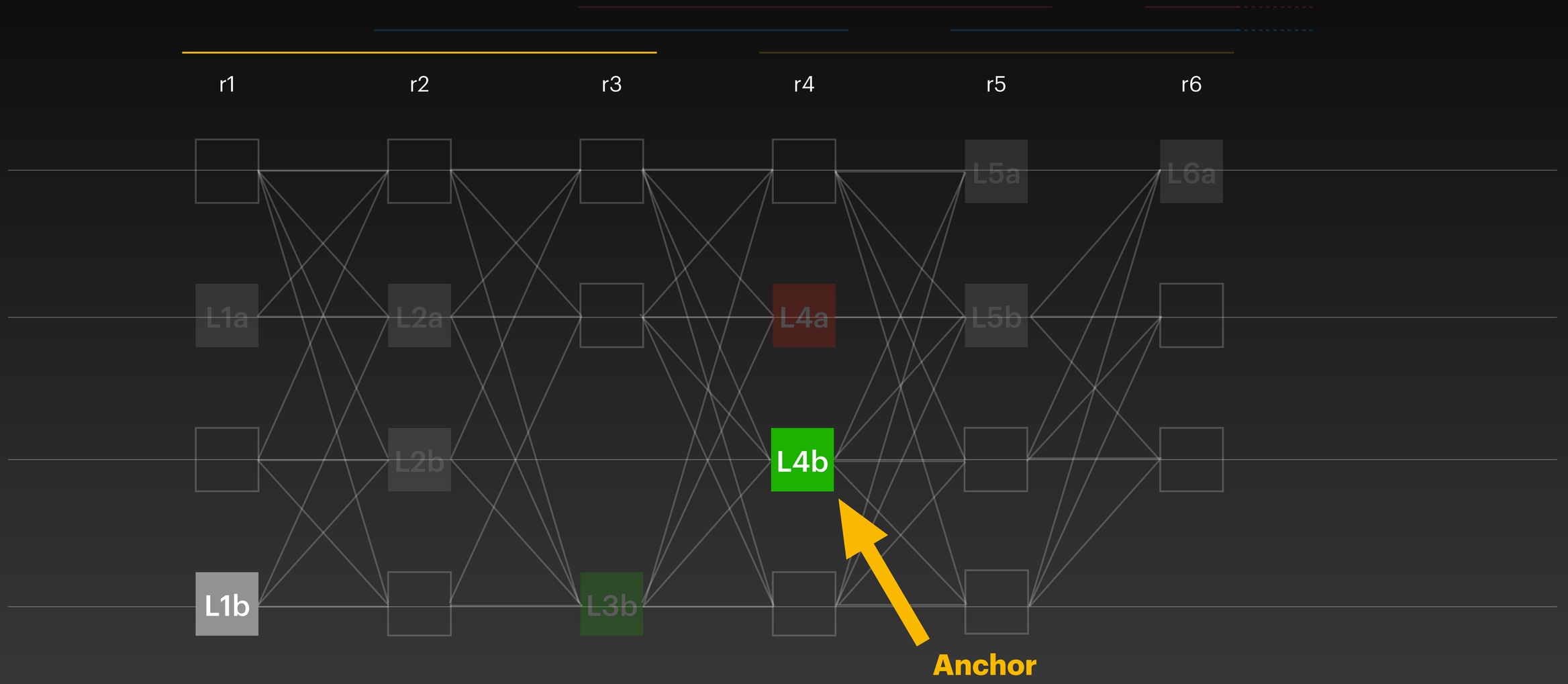
Apply Direct Rule



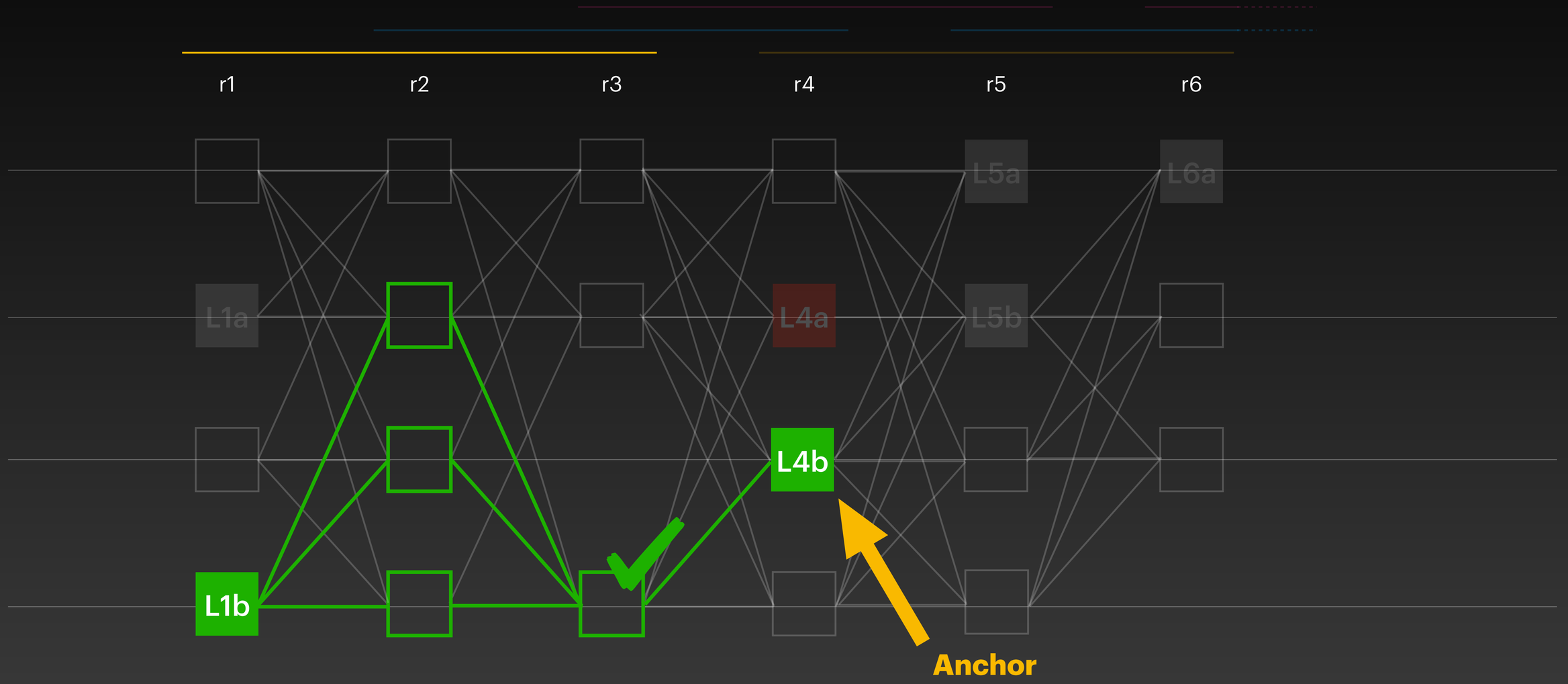
Apply Indirect Rule



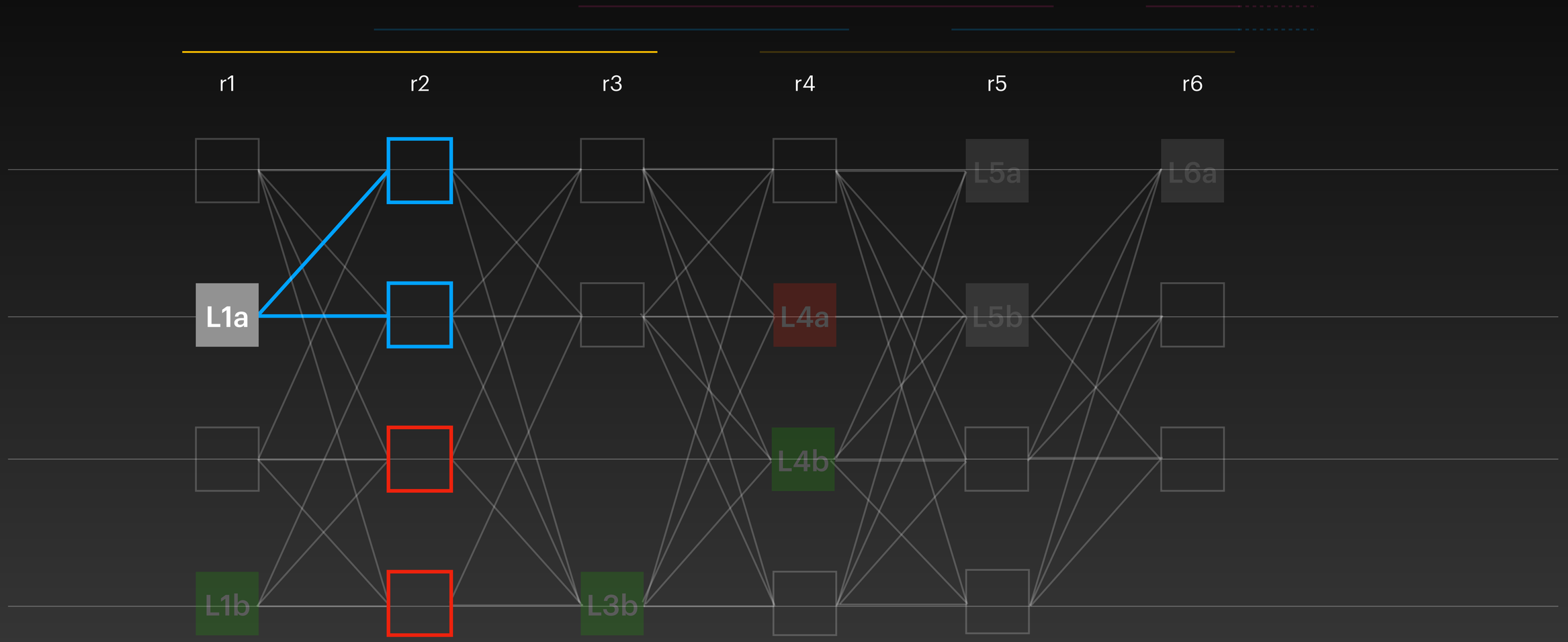
Apply Indirect Rule



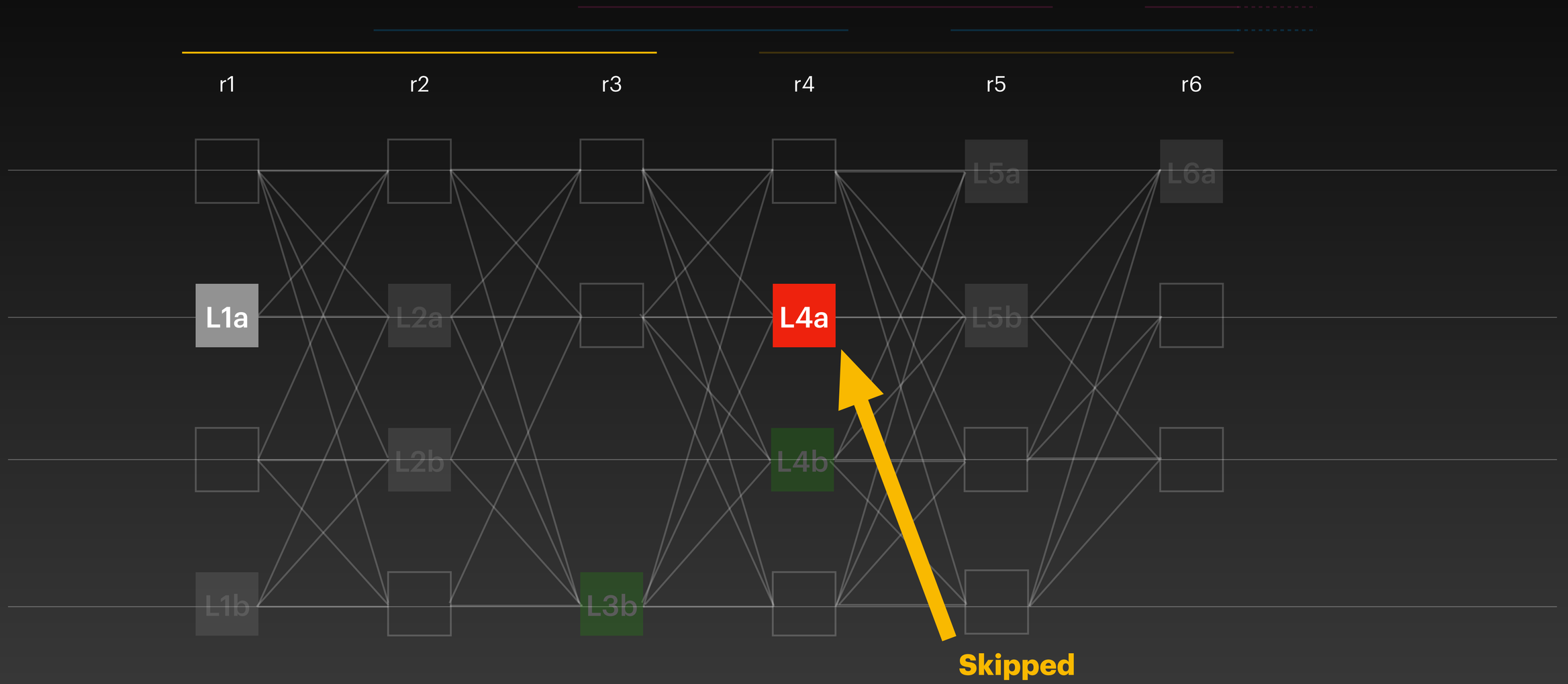
Apply Indirect Rule



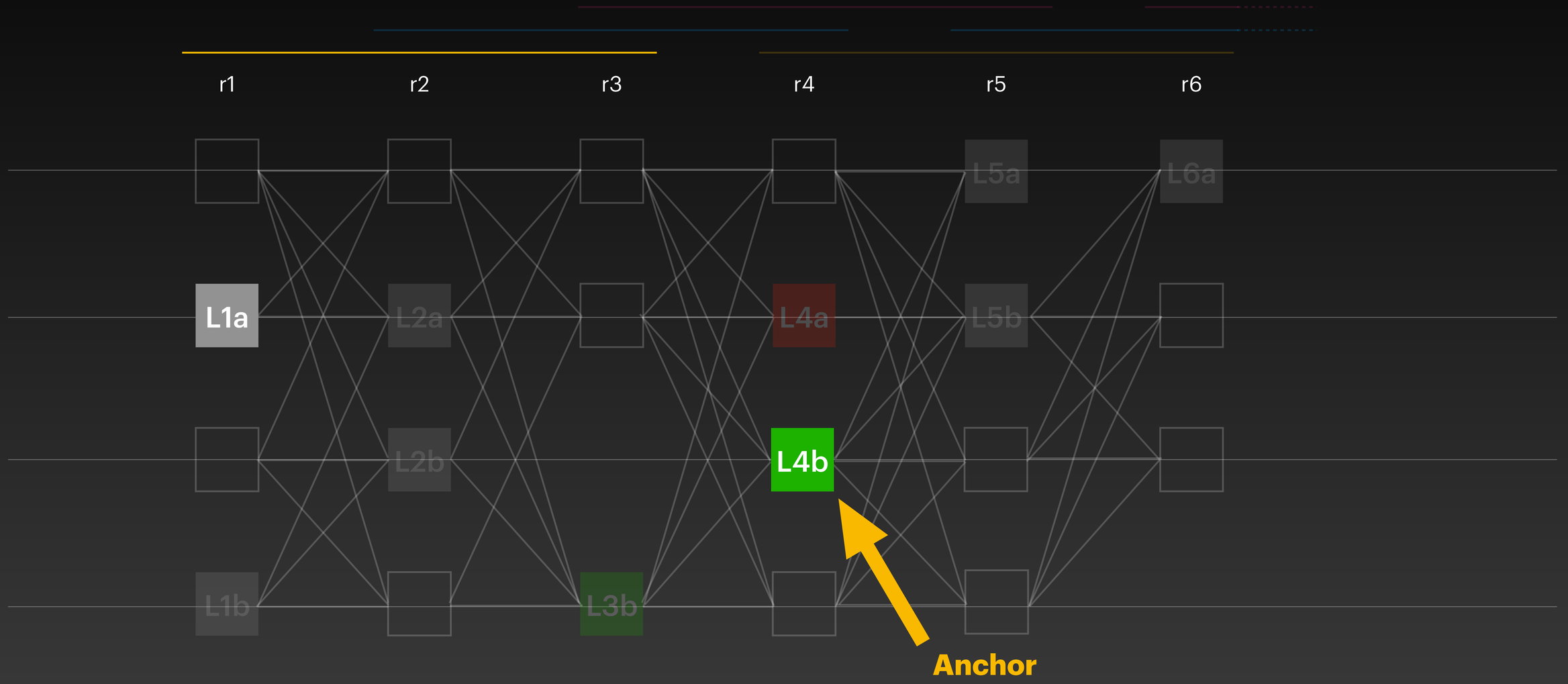
Apply Direct Rule



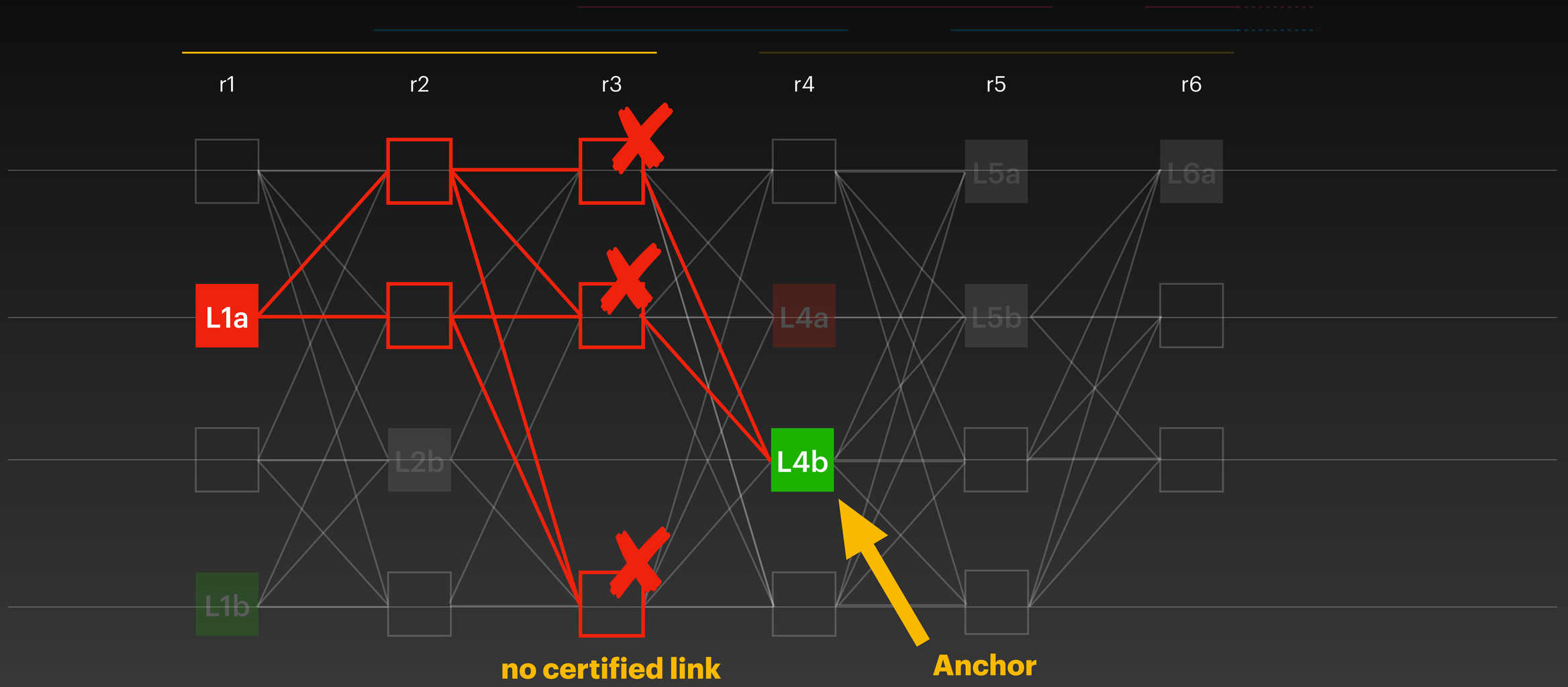
Apply Indirect Rule



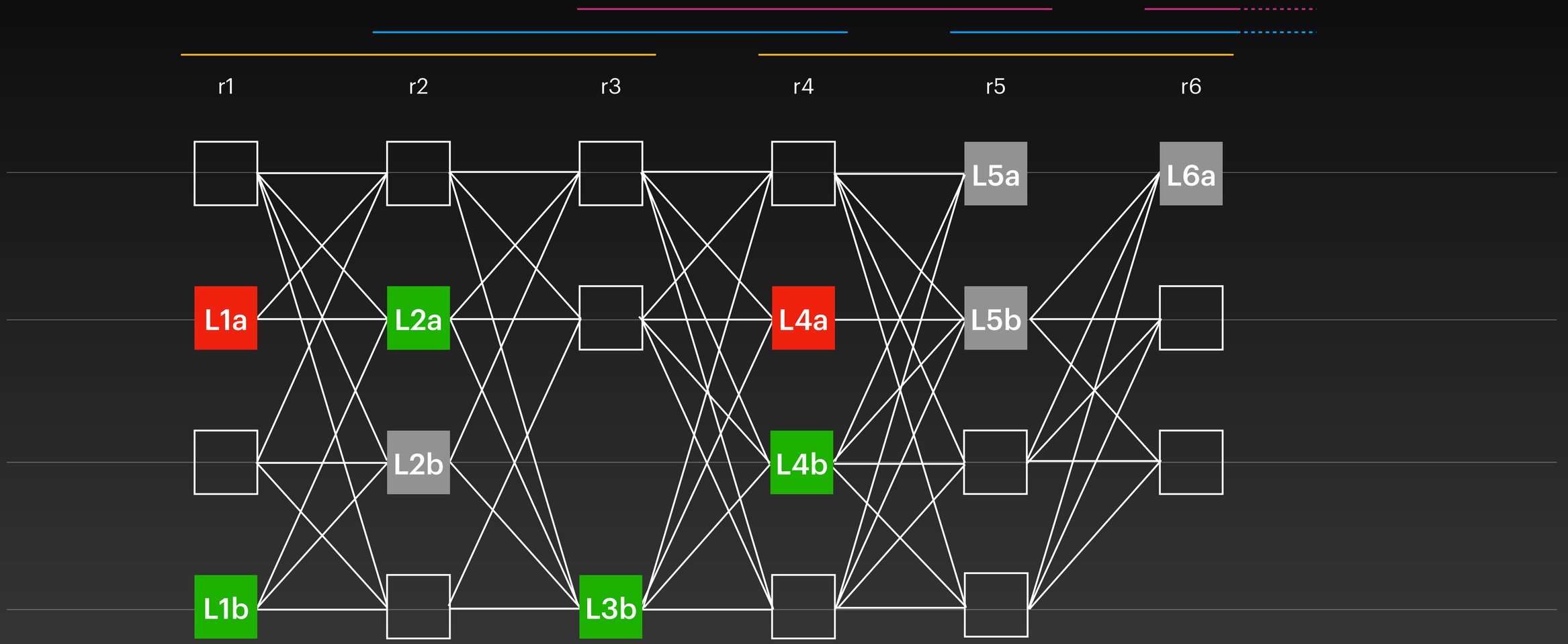
Apply Indirect Rule



Apply Indirect Rule

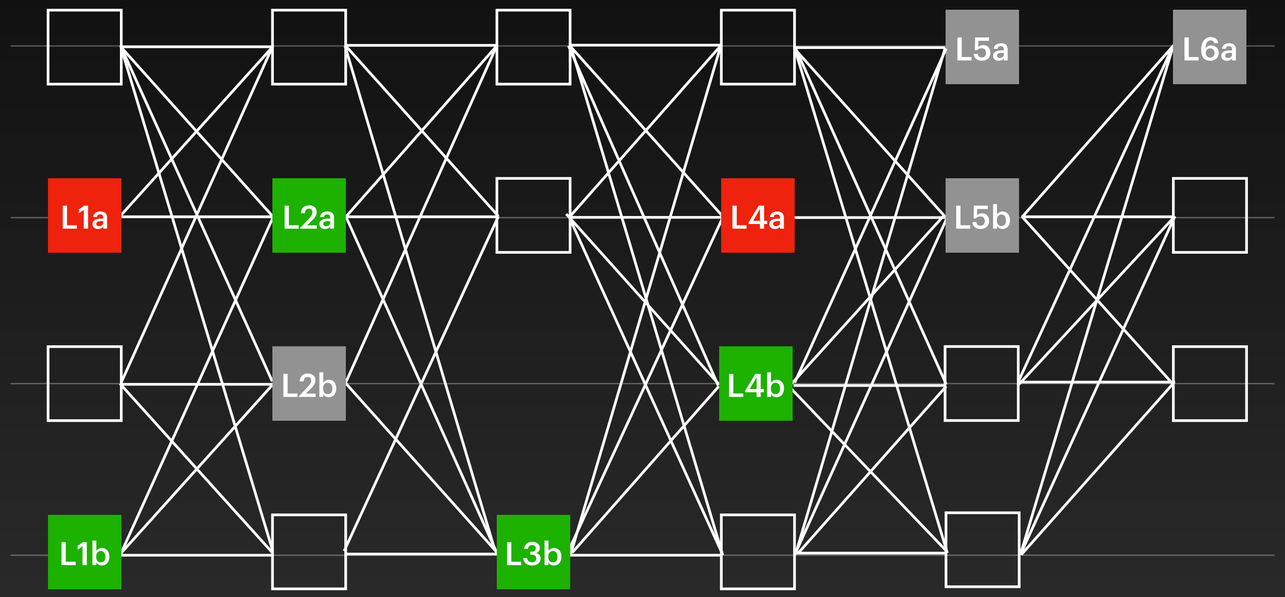


Current Status



Commit Sequence

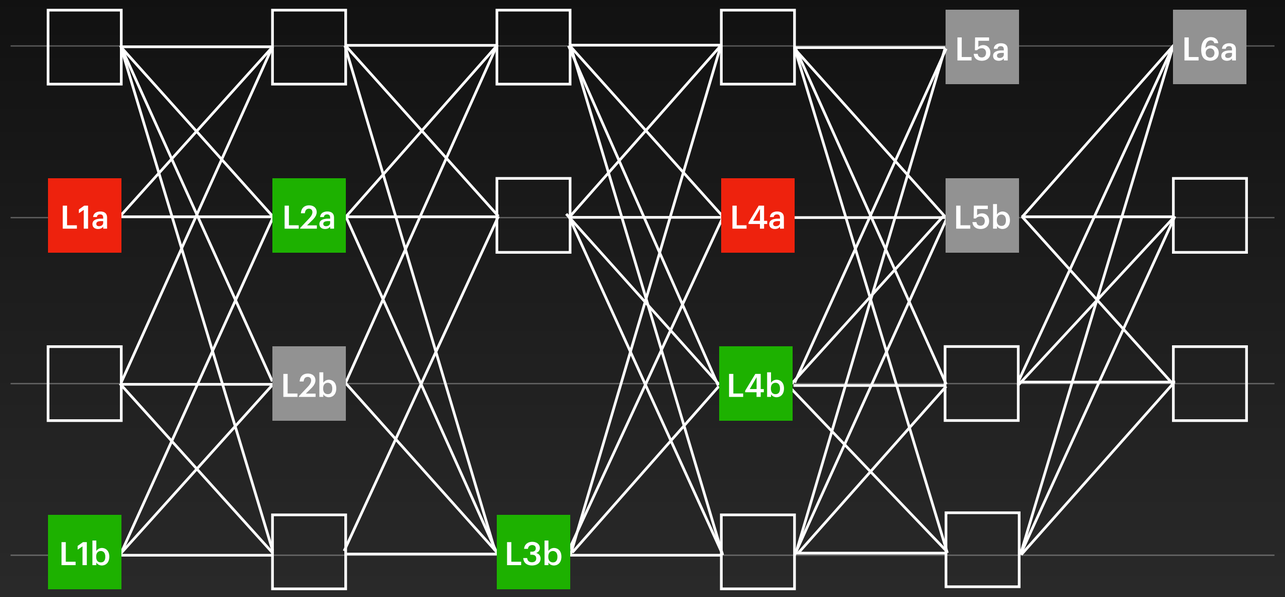
Take all leaders in order



leaders sequence: L1a L1b L2a L2b L3a L3b L4a L4b

Commit Sequence

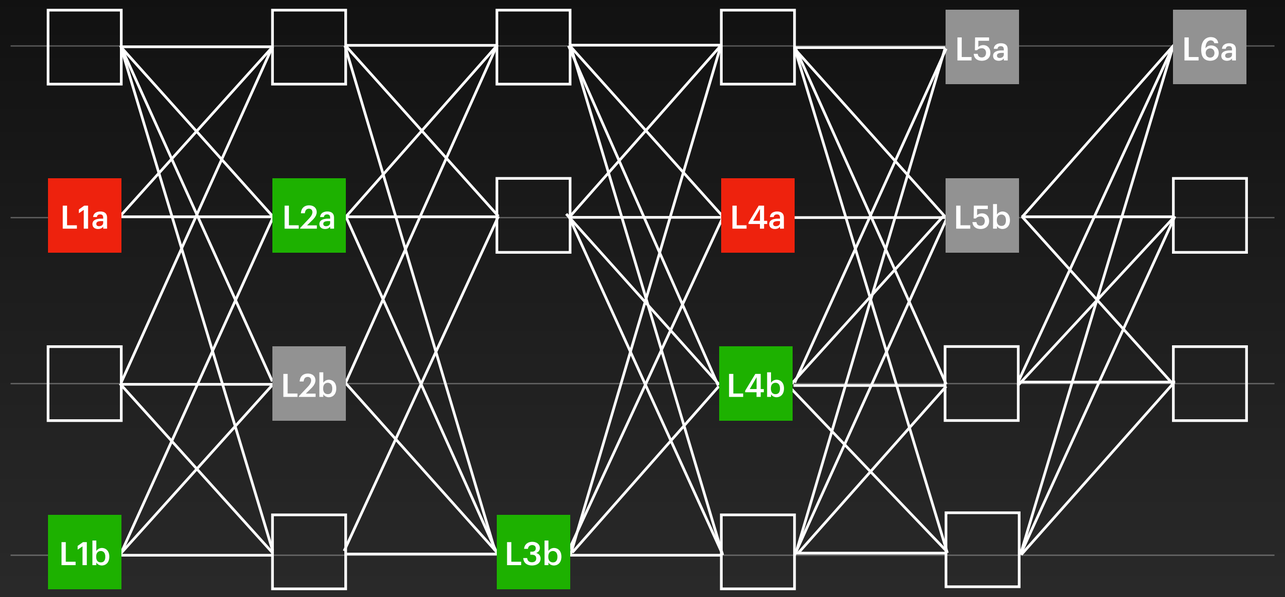
Stop at the first Undecided leader



leaders sequence: L1a L1b L2a L2b L3a L3b L4a L4b

Commit Sequence

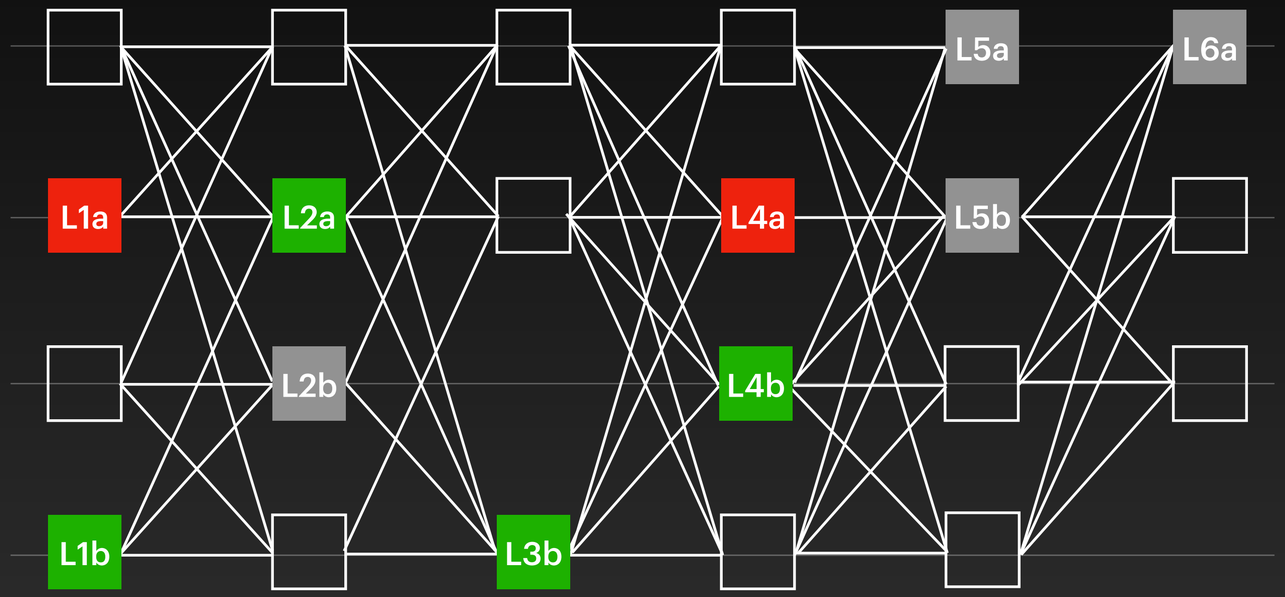
Remove skipped leaders



leaders sequence: L1a L1b L2a L2b L3a L3b L4a L4b

Commit Sequence

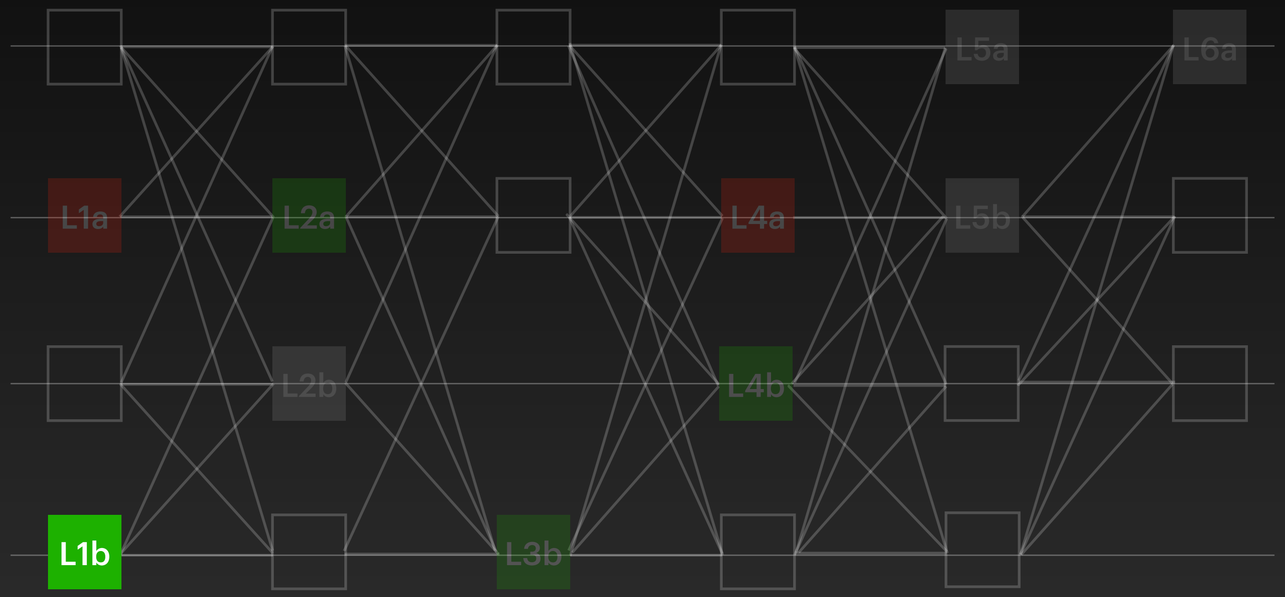
Final leader sequence



leaders sequence: L1b L2a

Commit Sequence

Commit sub-dag

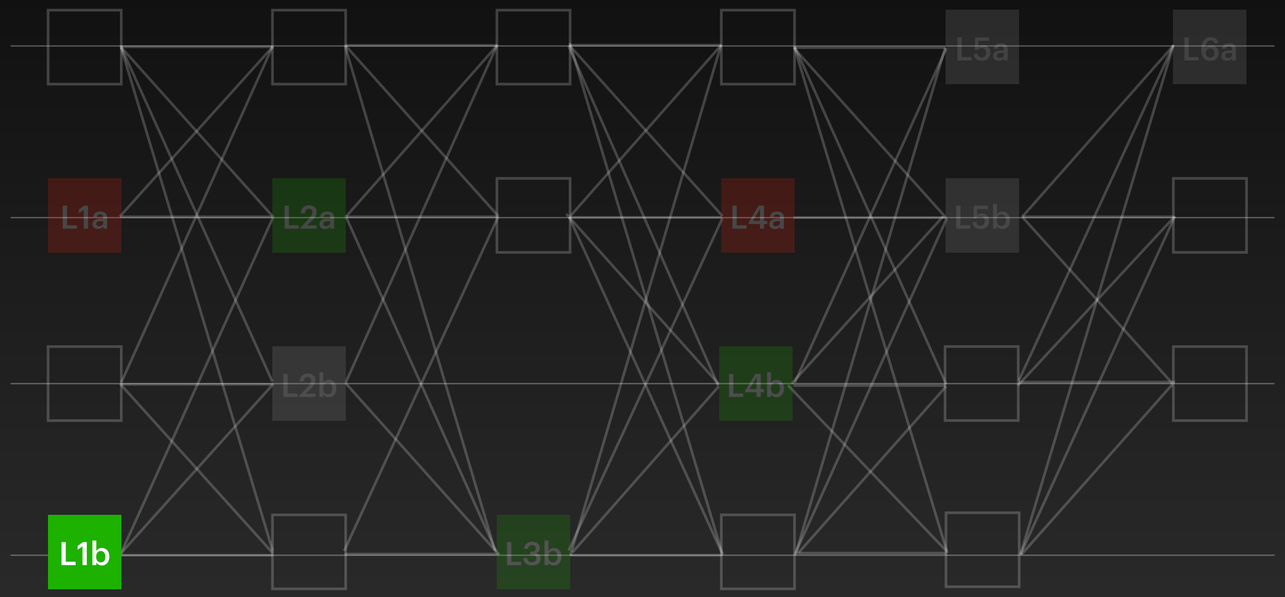


leaders sequence: L1b L2a

output sequence:

Commit Sequence

Commit sub-dag

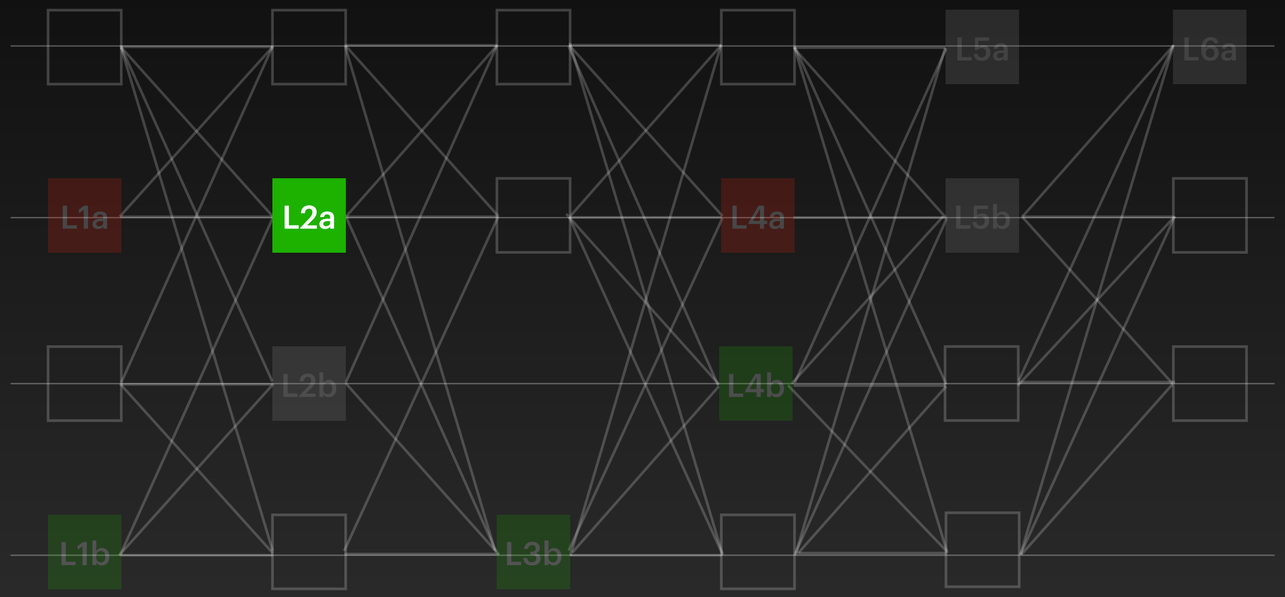


leaders sequence: L2a

output sequence: L1b

Commit Sequence

Commit sub-dag

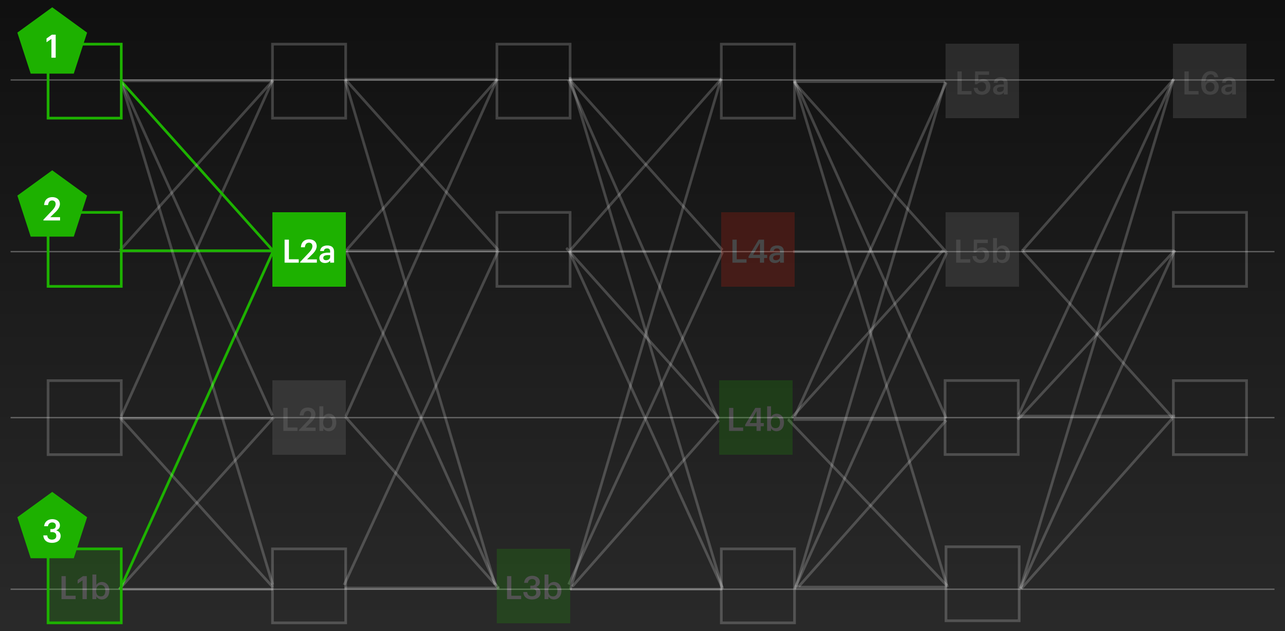


leaders sequence: L2a

output sequence: L1b

Commit Sequence

Commit sub-dag

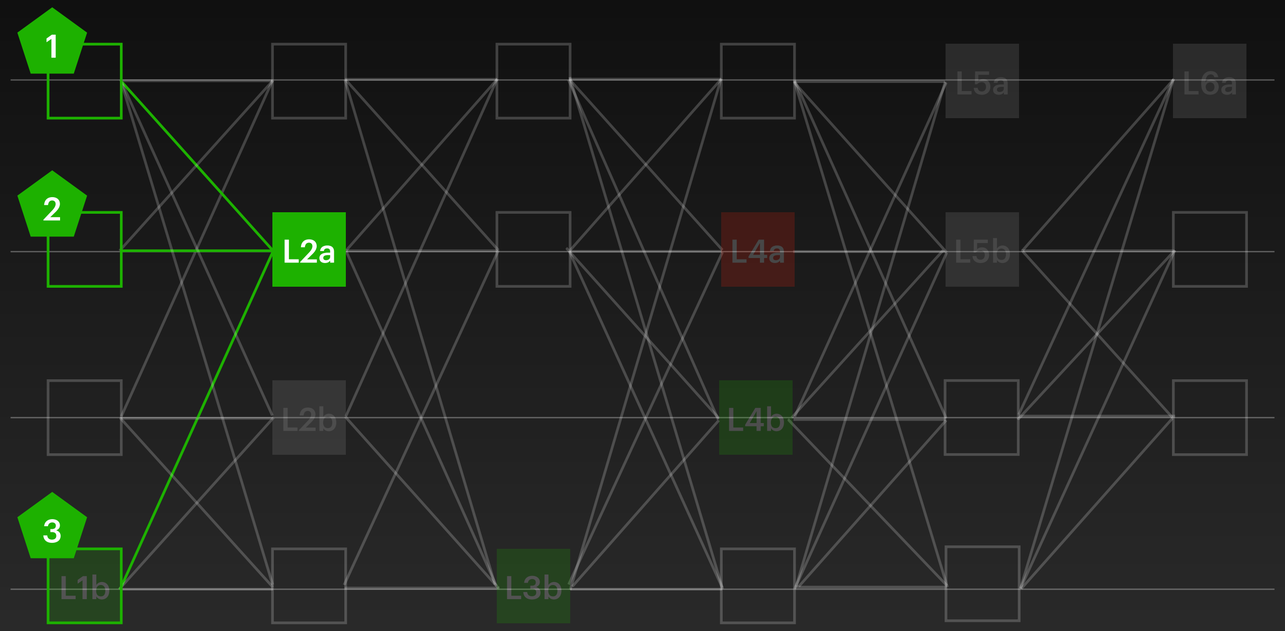


leaders sequence: L2a

output sequence: L1b

Commit Sequence

Commit sub-dag



leaders sequence:

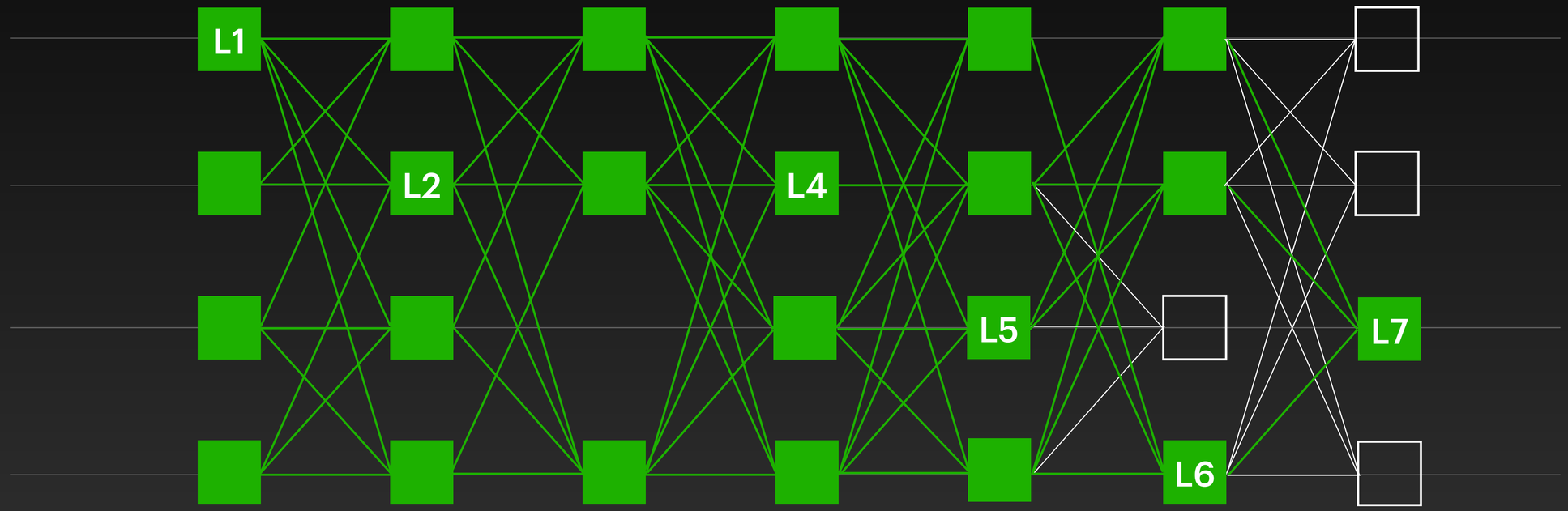
output sequence: L1b, 1, 2, L2a

Slow Leaders are Annoying

Suffer from them only when under attack or bad network

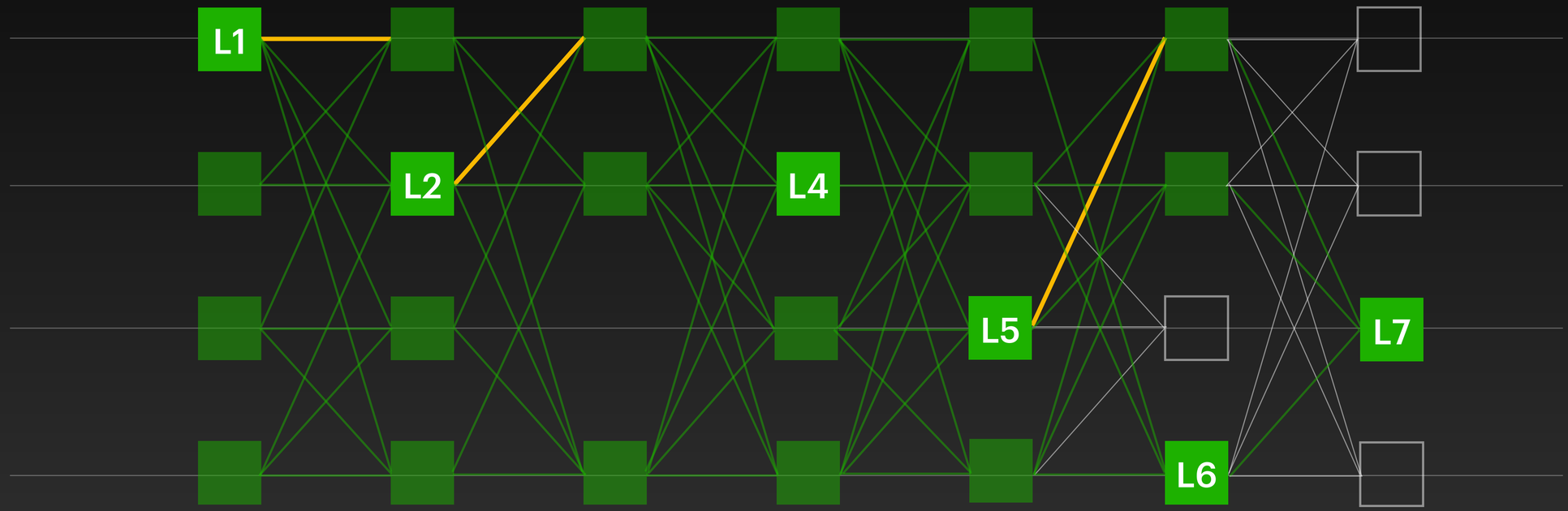
HammerHead

Compute Reputation Scores



HammerHead

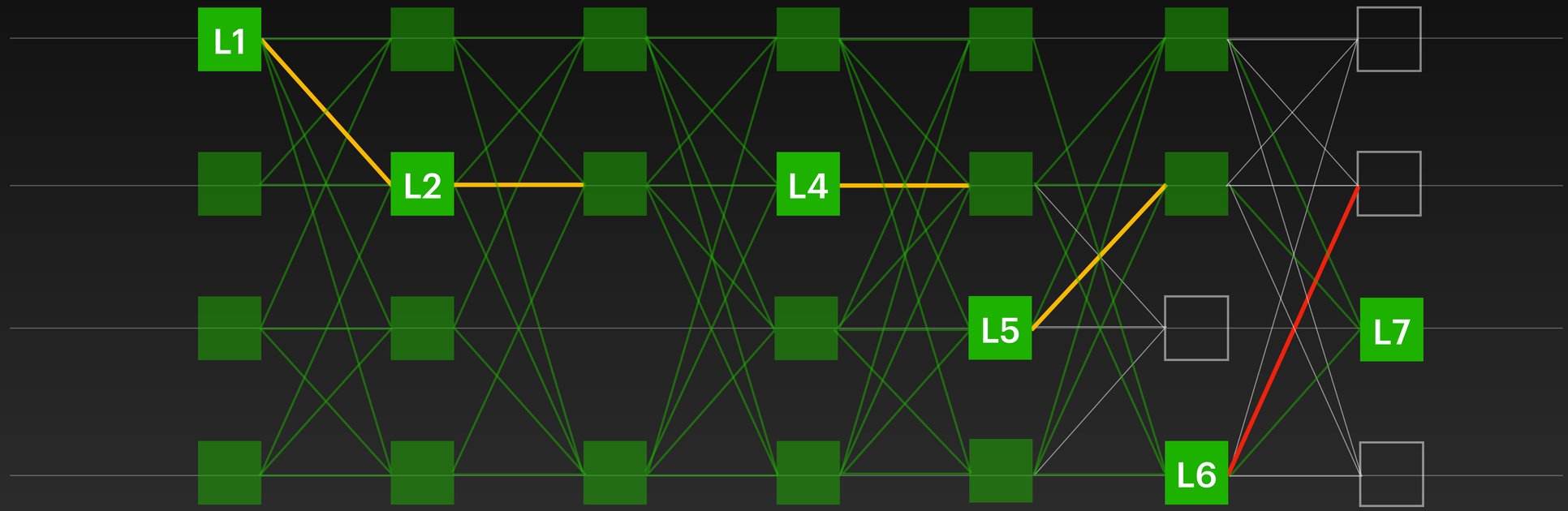
Compute Reputation Scores



node 1: 3

HammerHead

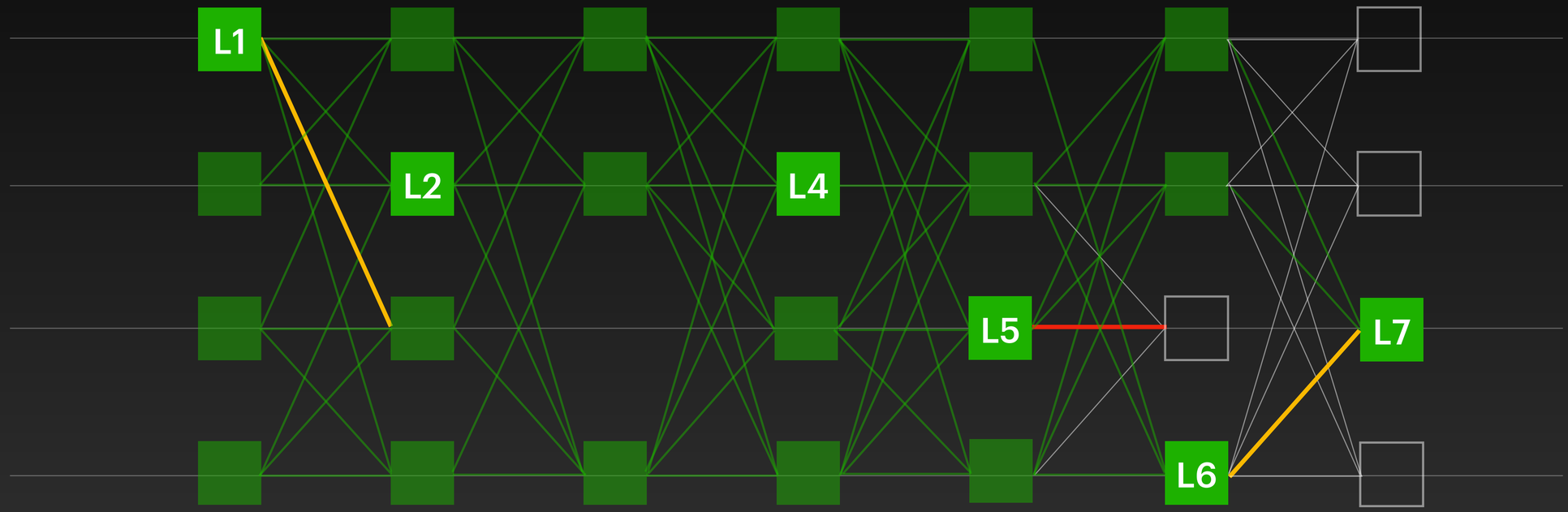
Compute Reputation Scores



node 1: 3 **node 2:** 4

HammerHead

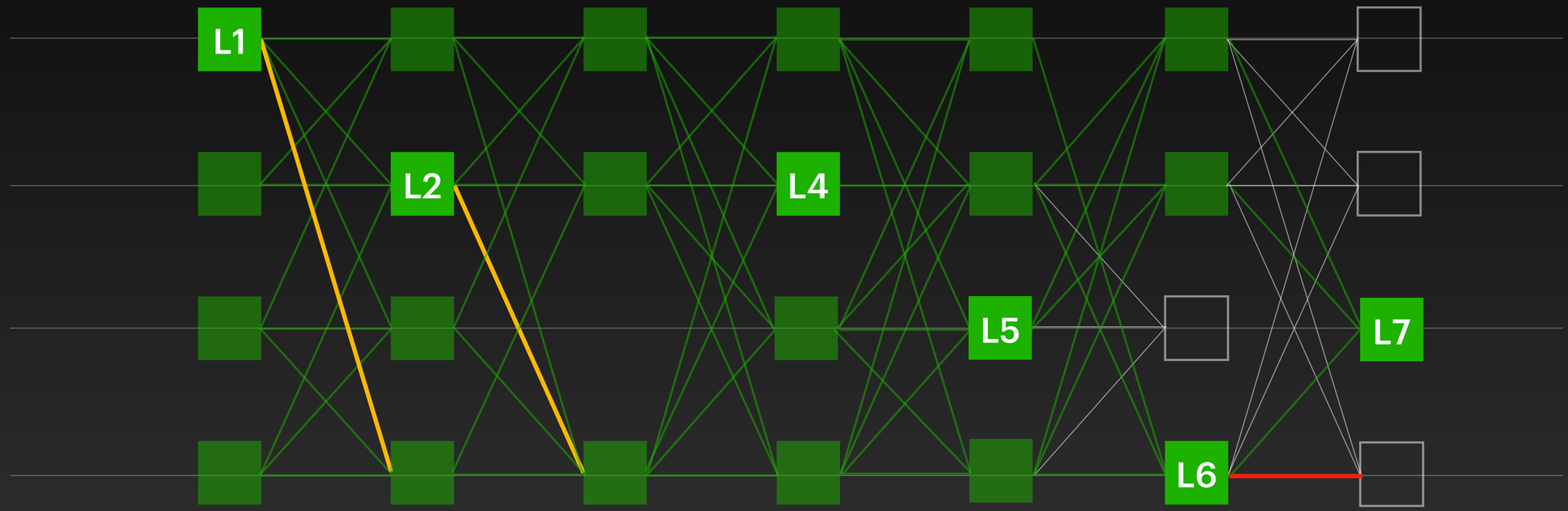
Compute Reputation Scores



node 1: 3 **node 2:** 4 **node 3:** 2

HammerHead

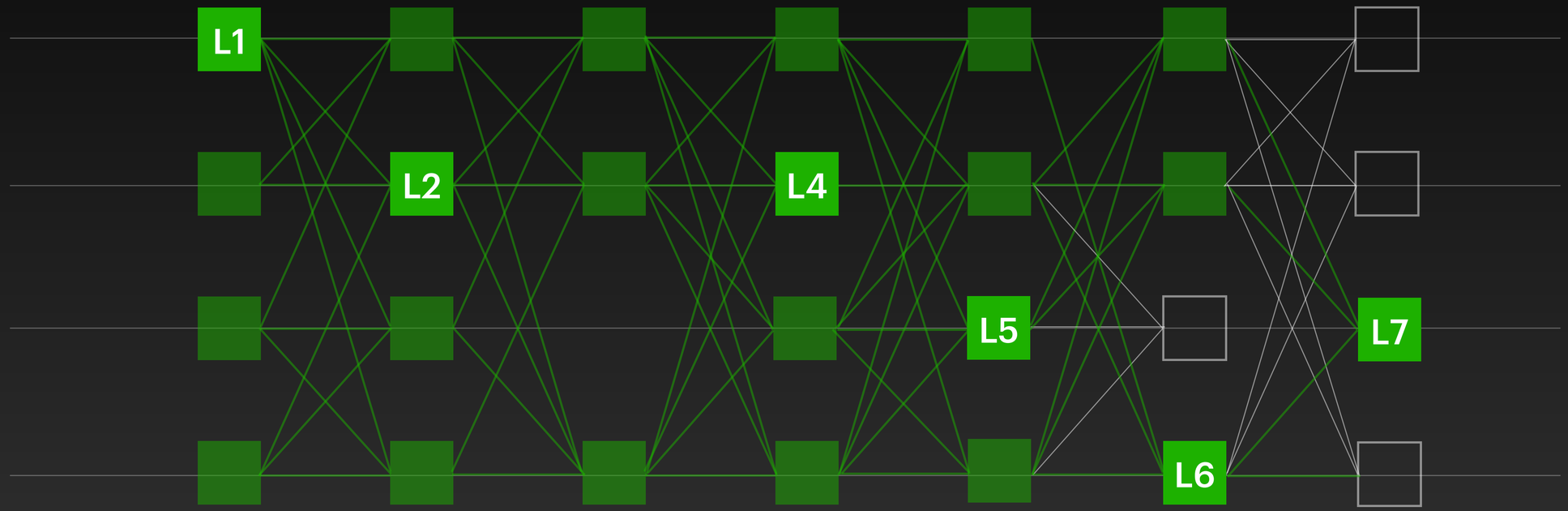
Compute Reputation Scores



node 1: 3 **node 2:** 4 **node 3:** 2 **node 4:** 2

HammerHead

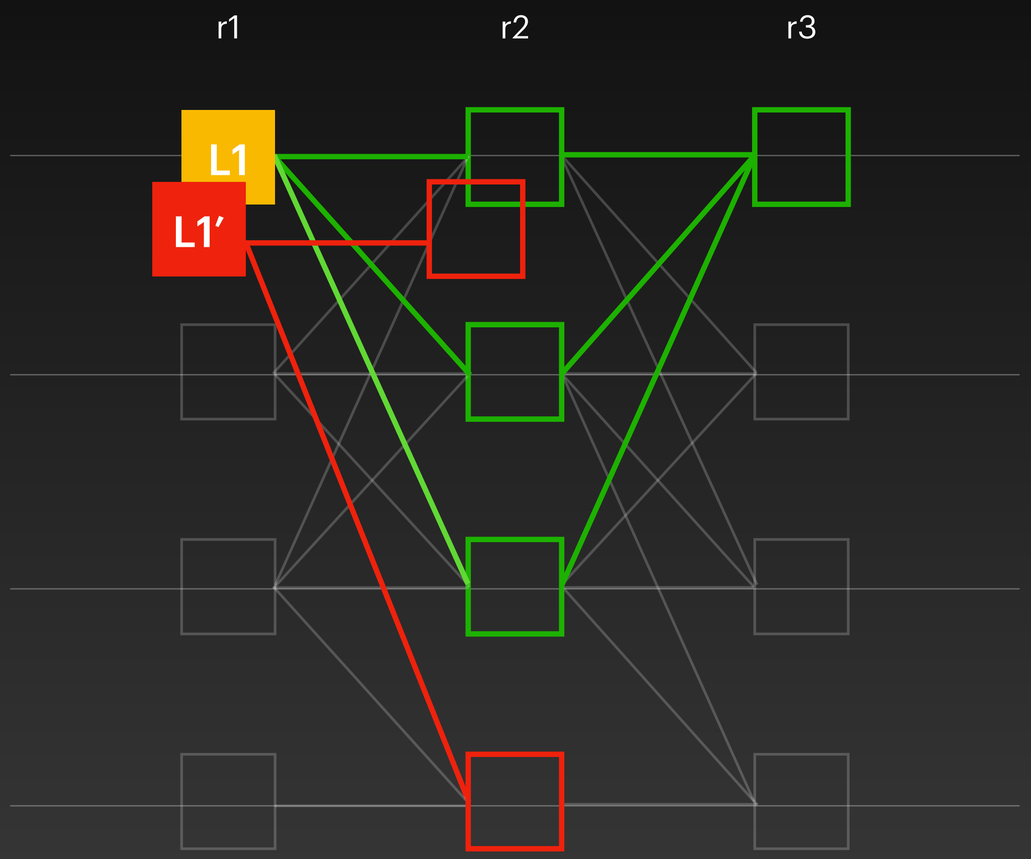
Future Leaders



node 1: 3 node 2: 4 node 3: 2 node 4: 2

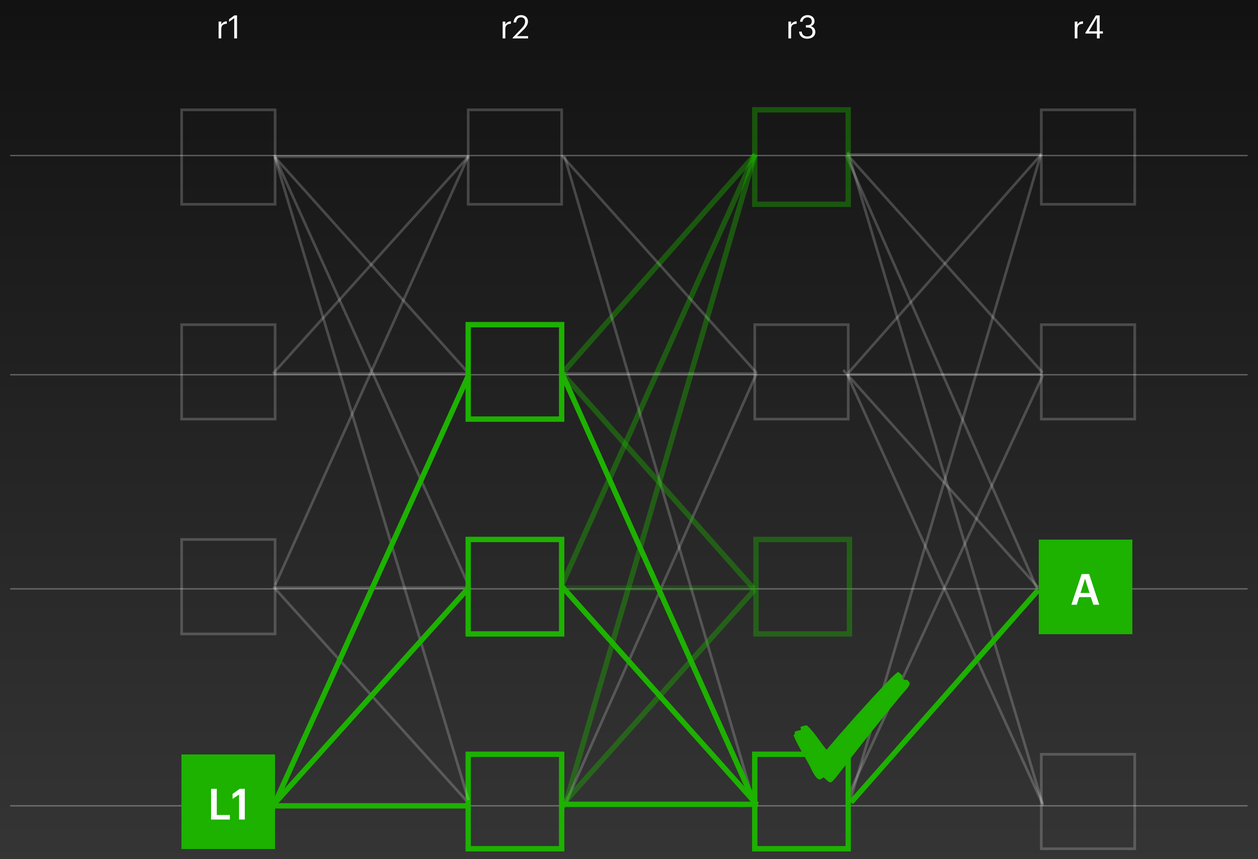
Security Intuition

Security Intuition



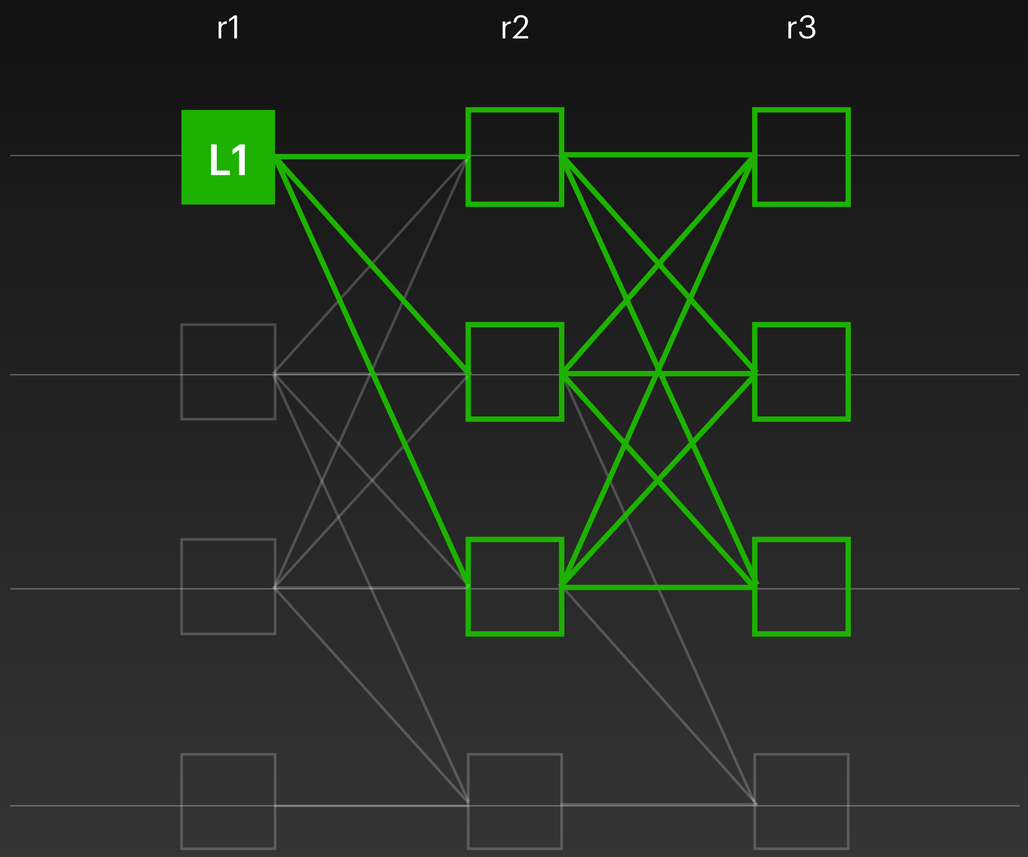
- At most **L1** or **L1’** can have a certificate pattern (quorum intersection)

Security Intuition



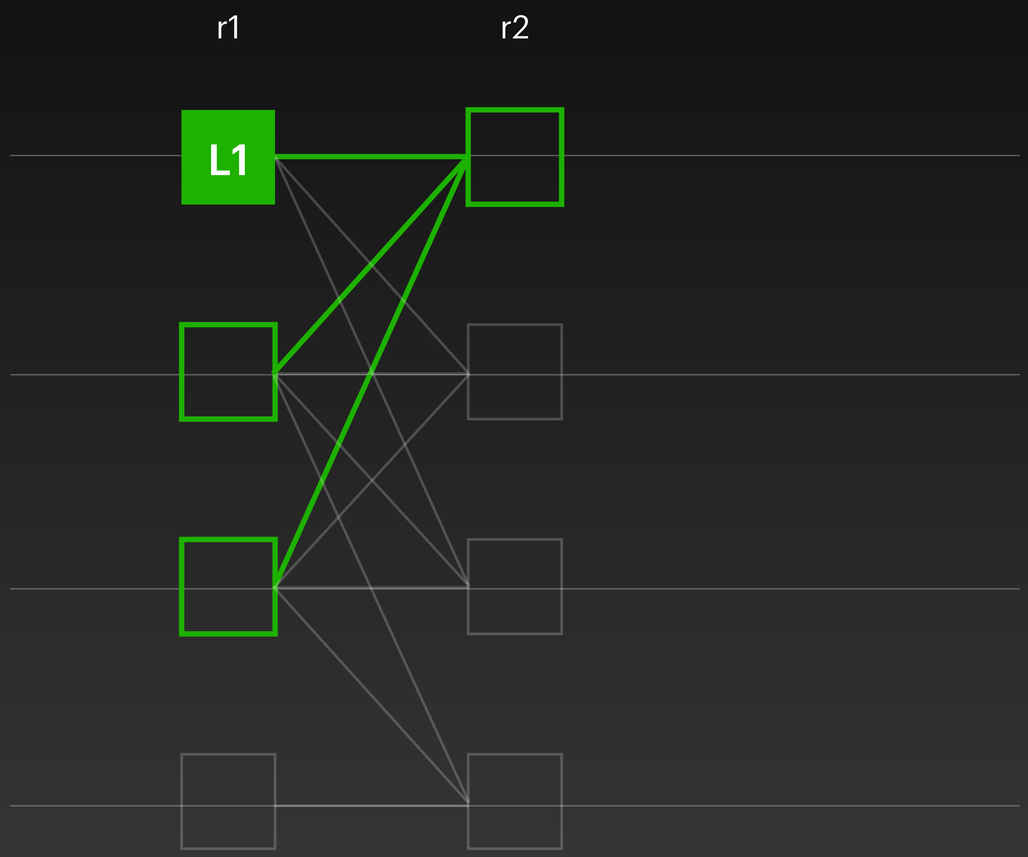
- At most **L1** or **L1'** can have a certificate pattern (quorum intersection
- If **L1** has 2f+1 certificate patterns, **A** always has a certified link to **L1**

Security Intuition



- At most **L1** or **L1'** can have a certificate pattern (quorum intersection
- If **L1** has 2f+1 certificate patterns, **A** always has a certified link to **L1**
- After GST, the direct decision rule **commits** a block

Security Intuition



Leader Timeout:

Wait for 2f+1 parents + 250 ms

Mysticeti-FPC

Adding a fast commit path

Consensus Not Required

Coins, balances, and transfers

NFTs creation and transfers

Game logic allowing users to combine assets

Inventory management for games / metaverse

Auditable 3rd party services not trusted for safety

...

Consensus Required

Increment a publicly-accessible counter

Auctions

Market places

Collaborative in-game assets

...

Object Type

Owned Objects

- Objects that can be mutated by a single entity
- e.g., My bank account
- **Do not need consensus**

Shared Objects

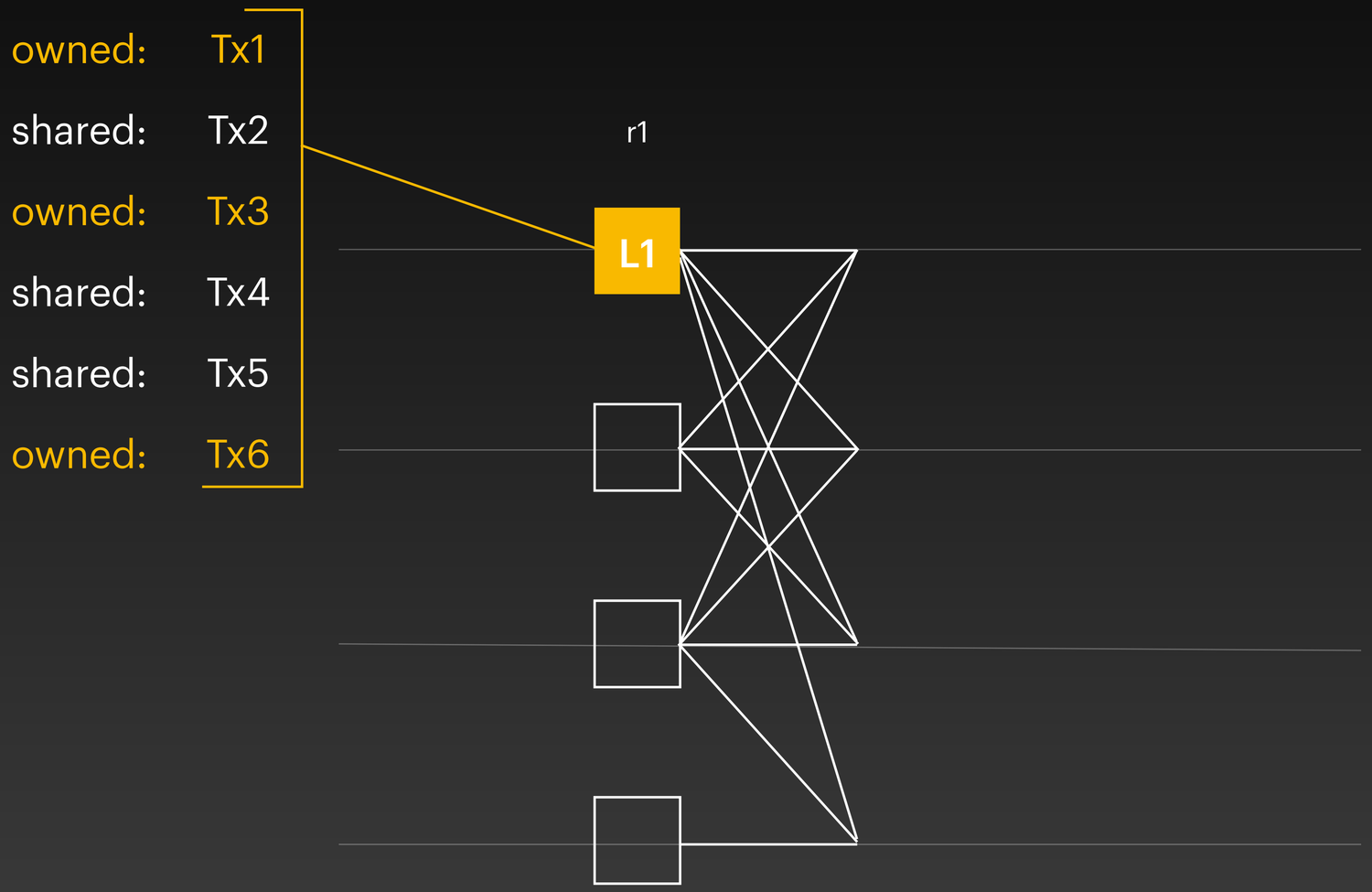
- Objects that can be mutated my multiple entities
- e.g., A global counter
- **Need consensus**

System State

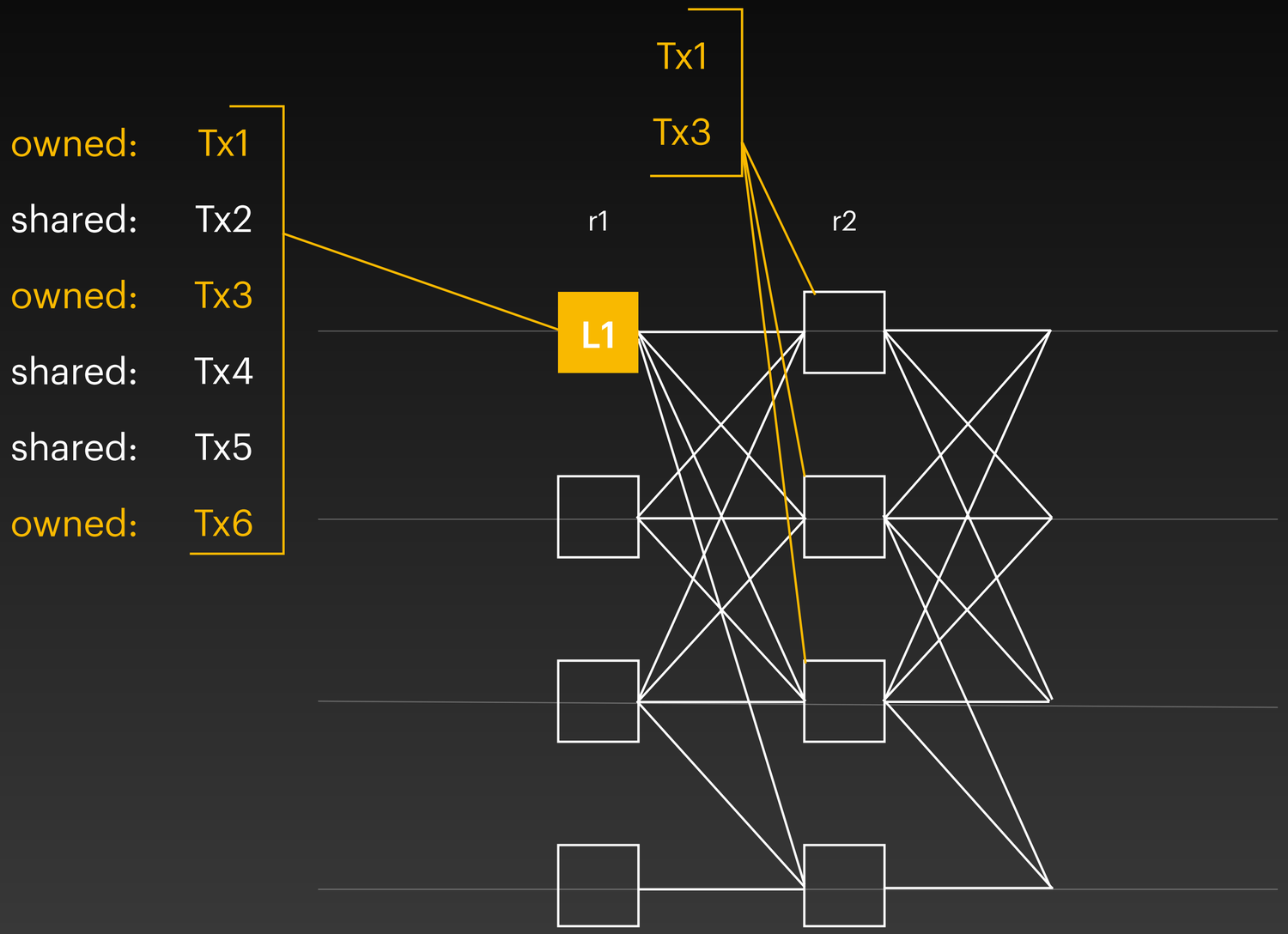
Objects:

- Unique ID
- Version number
- Ownership Information
- Type (shared, owned)

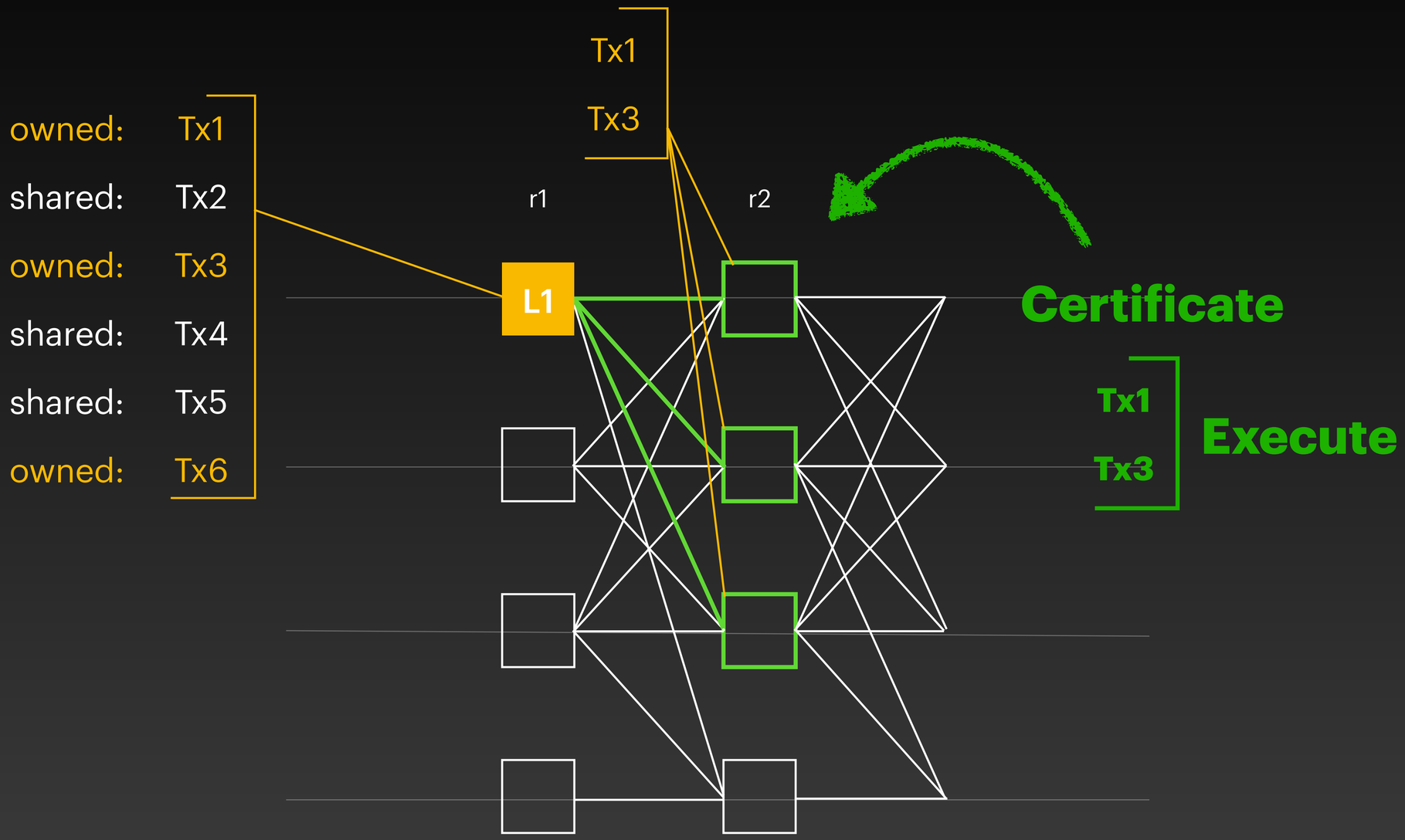
Fast Execution



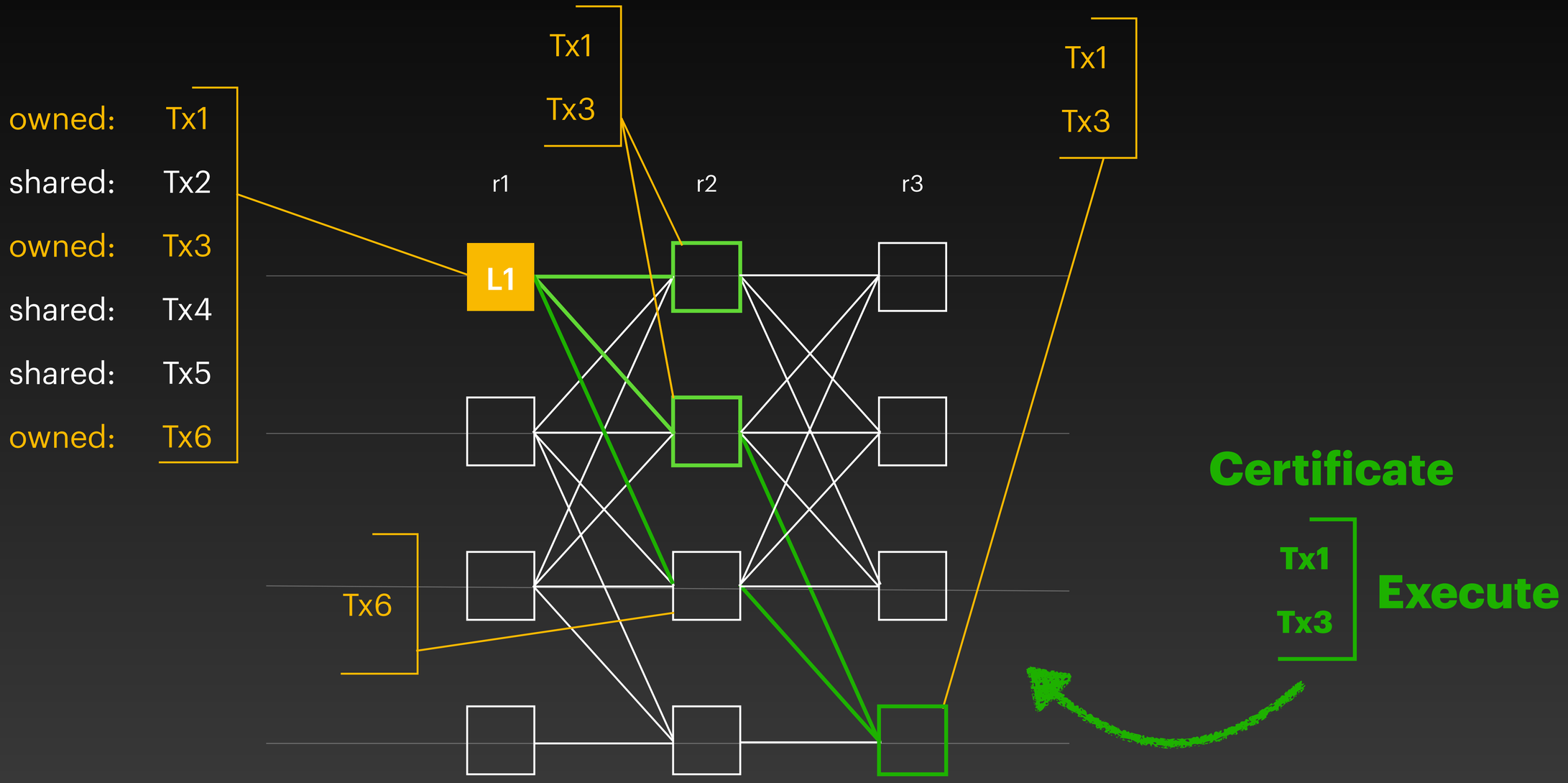
Fast Execution



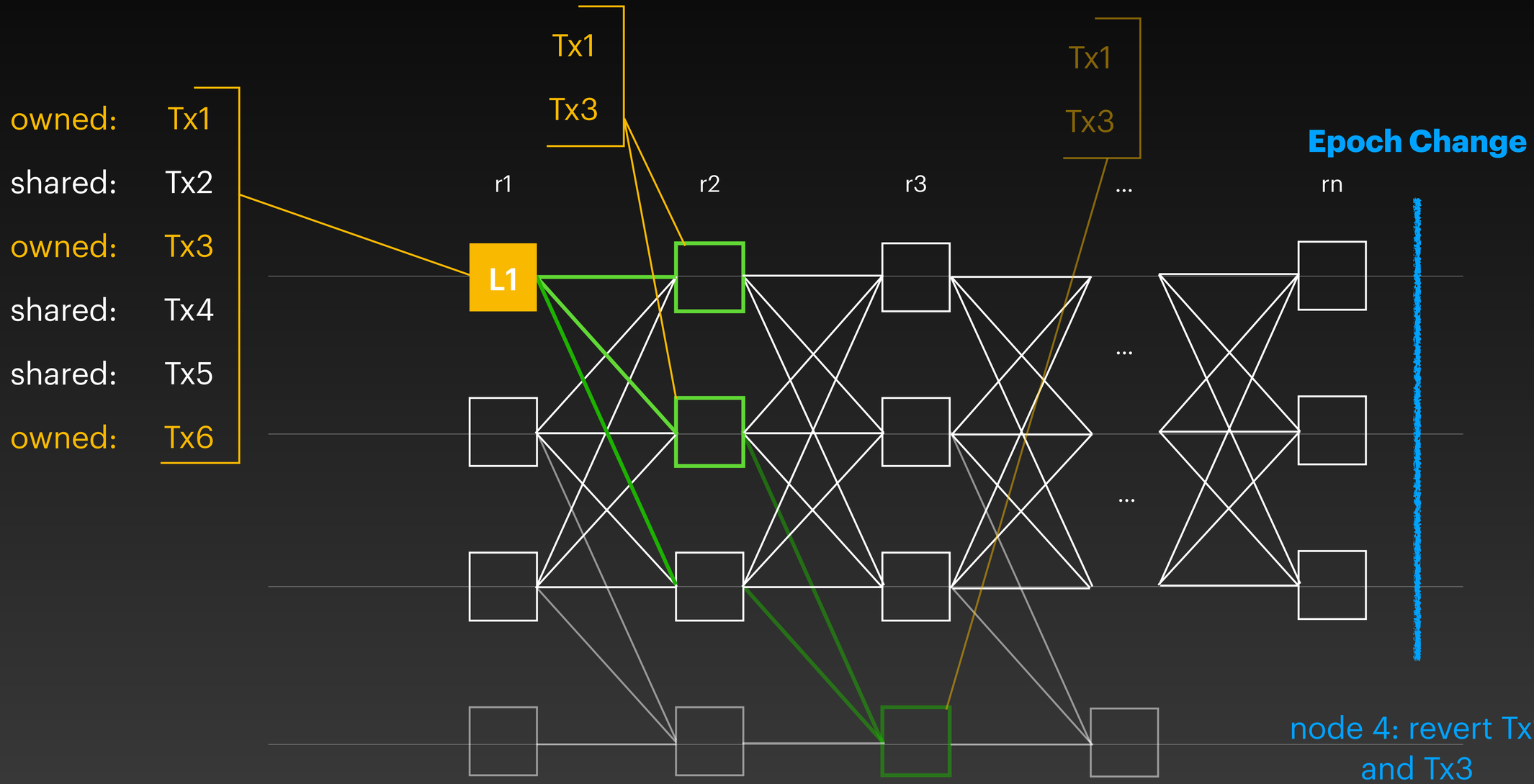
Fast Execution



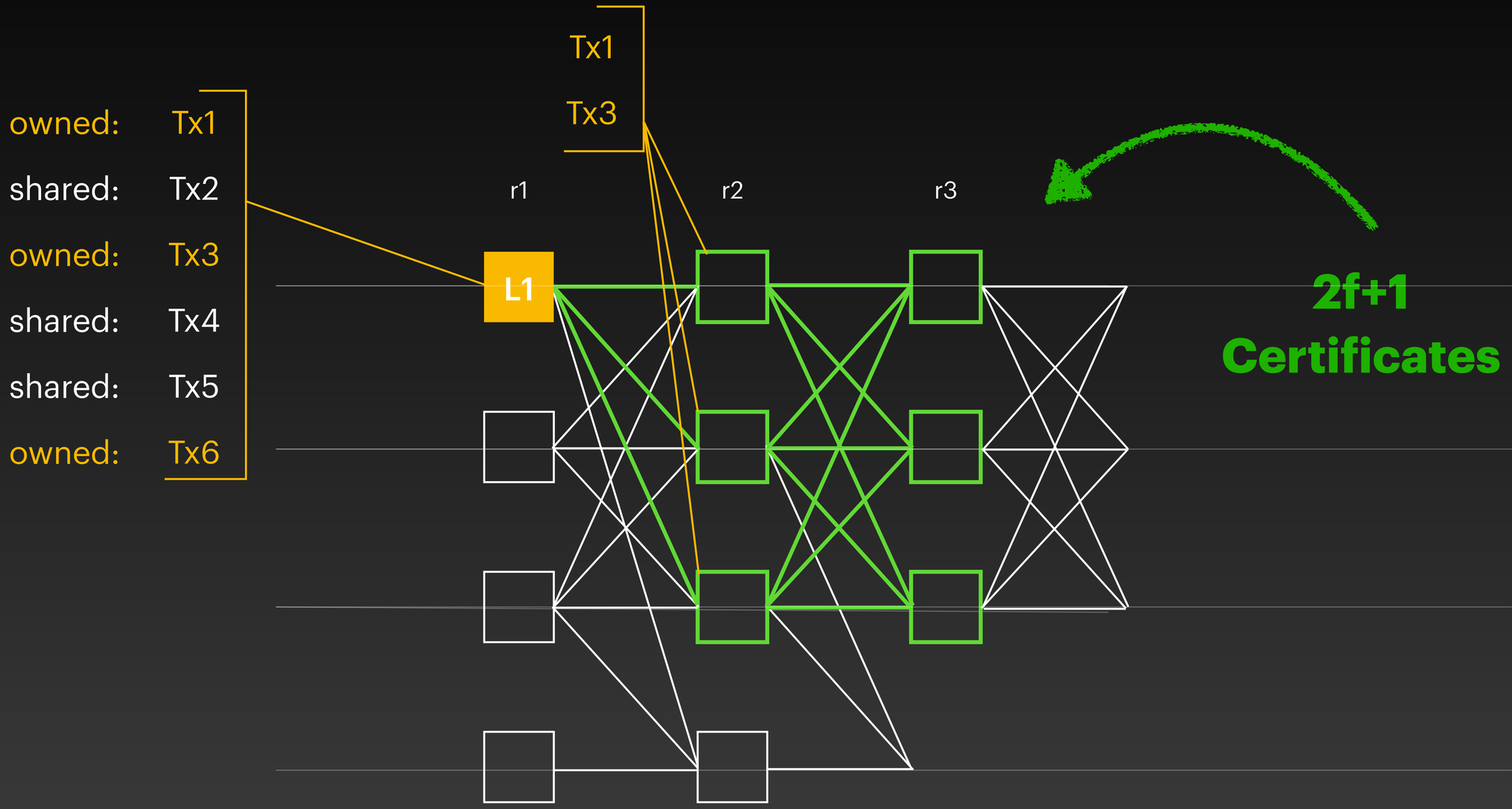
Fast Execution



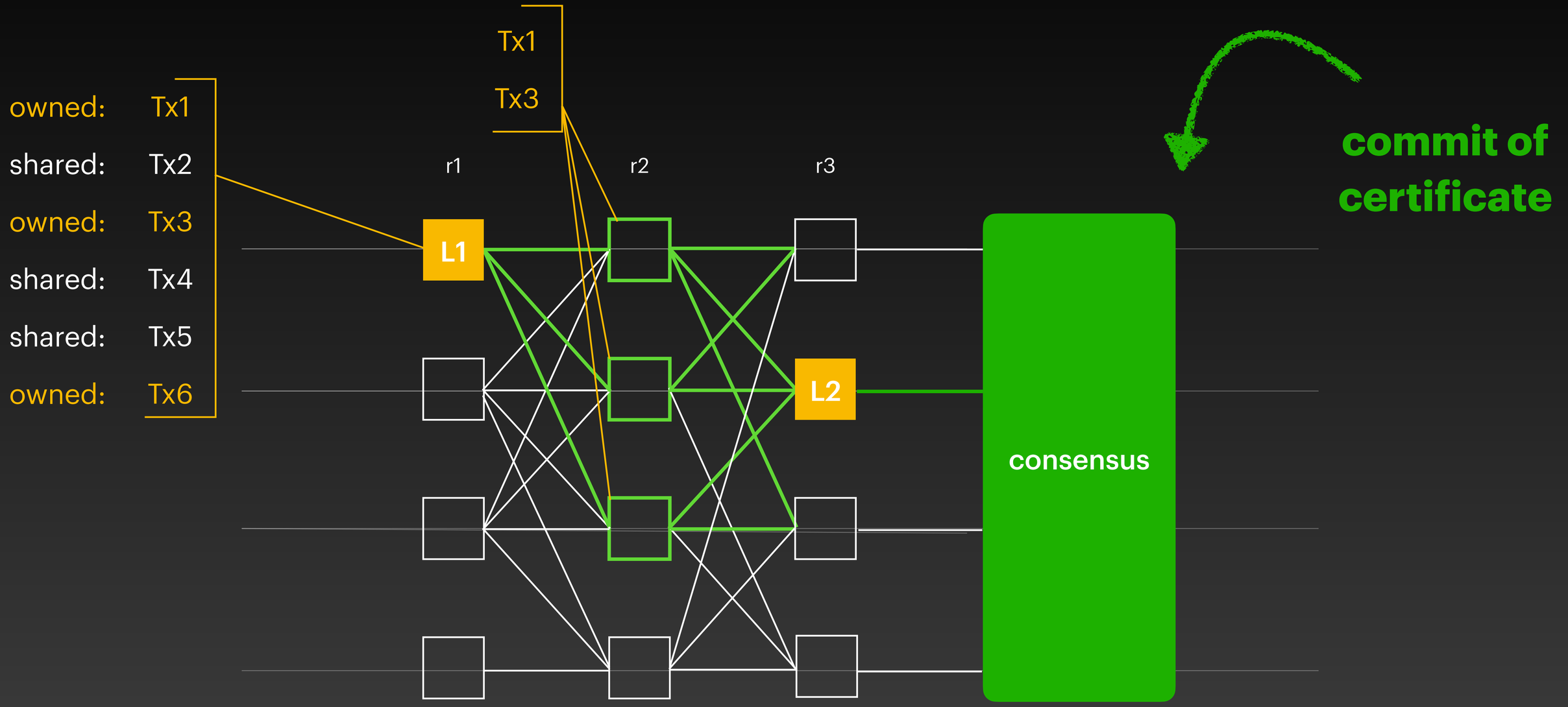
No Finality


owned: Tx1
shared: Tx2
owned: Tx3
shared: Tx4
shared: Tx5
owned: Tx6
Tx1
Tx3
Tx1
Tx3
r1
r2
r3
...
rn
L1
...
...
Epoch Change
node 4: revert Tx1 and Tx3

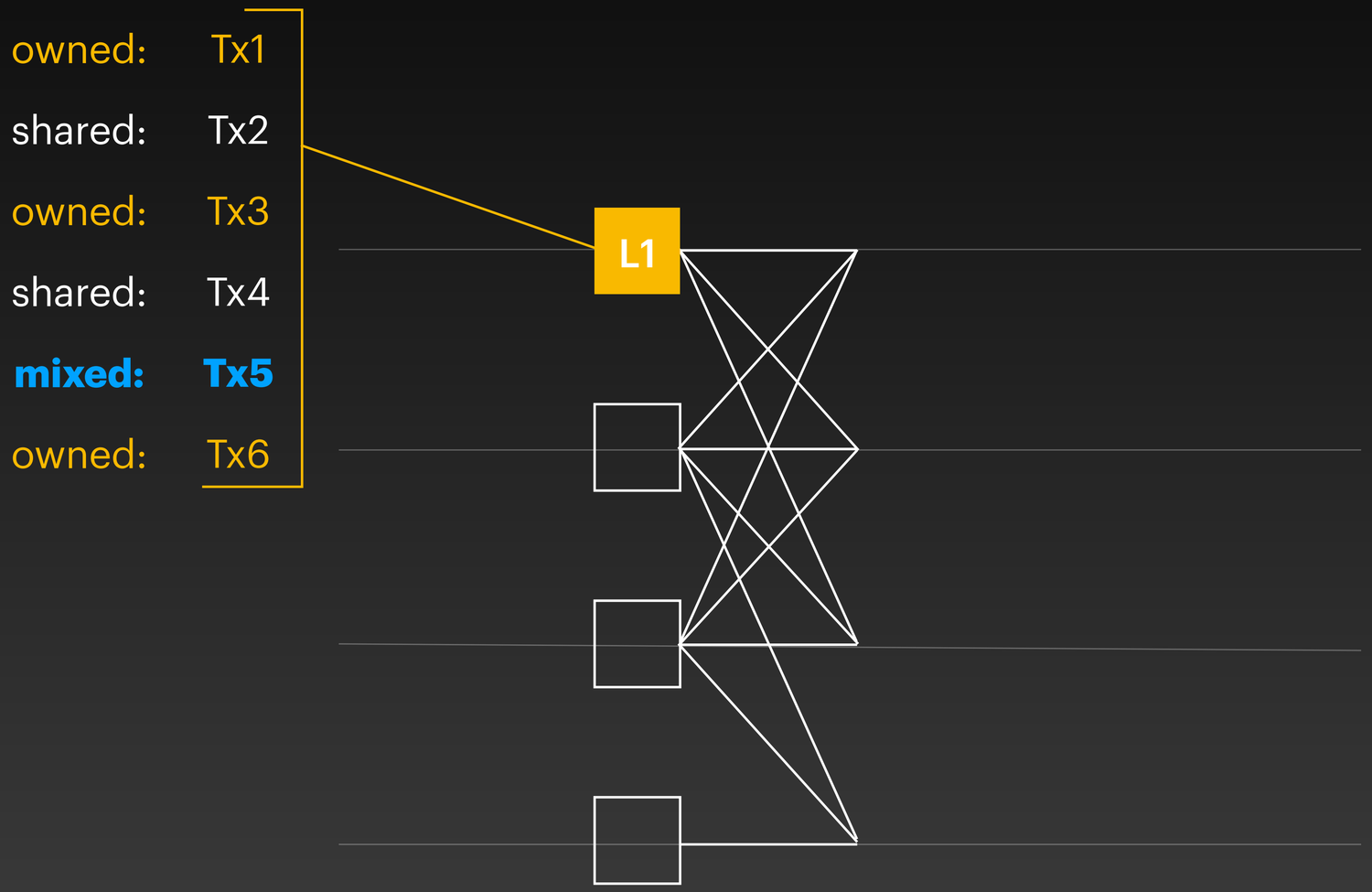
Fast Path Finality (1)



Fast Path Finality (2)

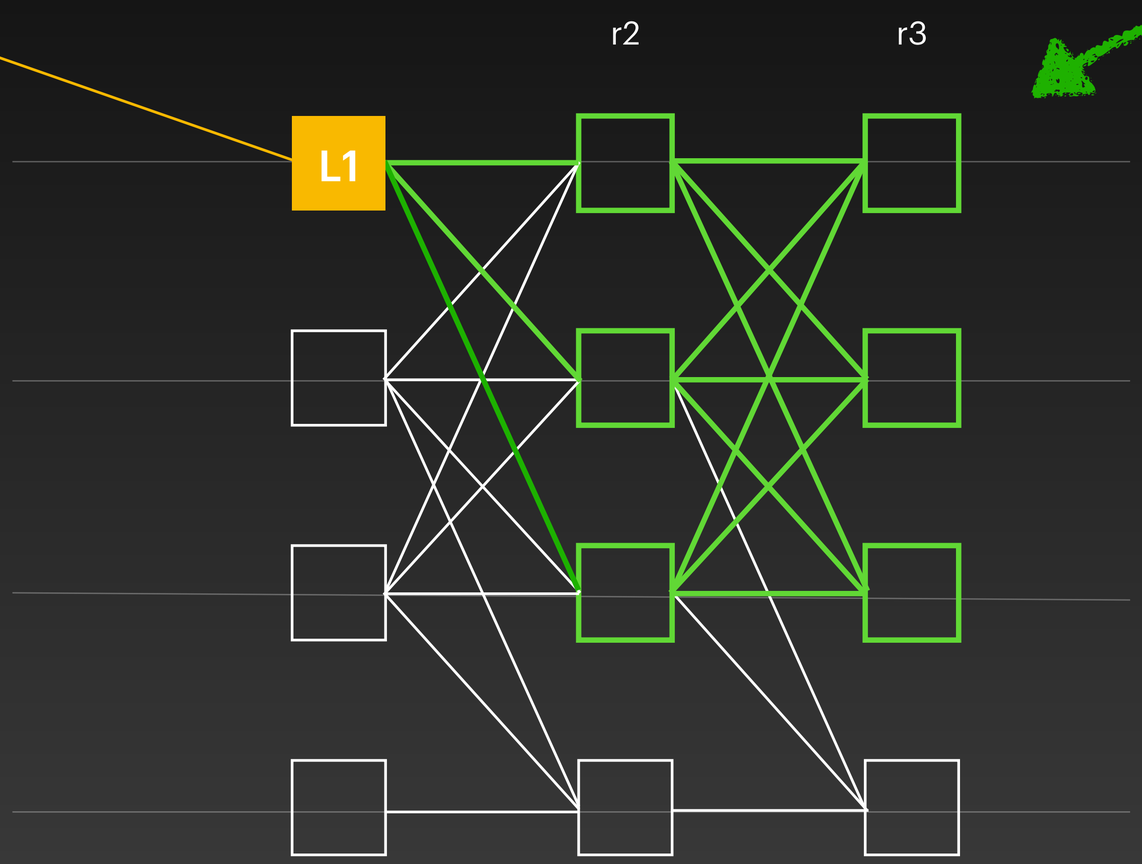


Mixed-Objects Transactions



Mixed-Objects Transactions

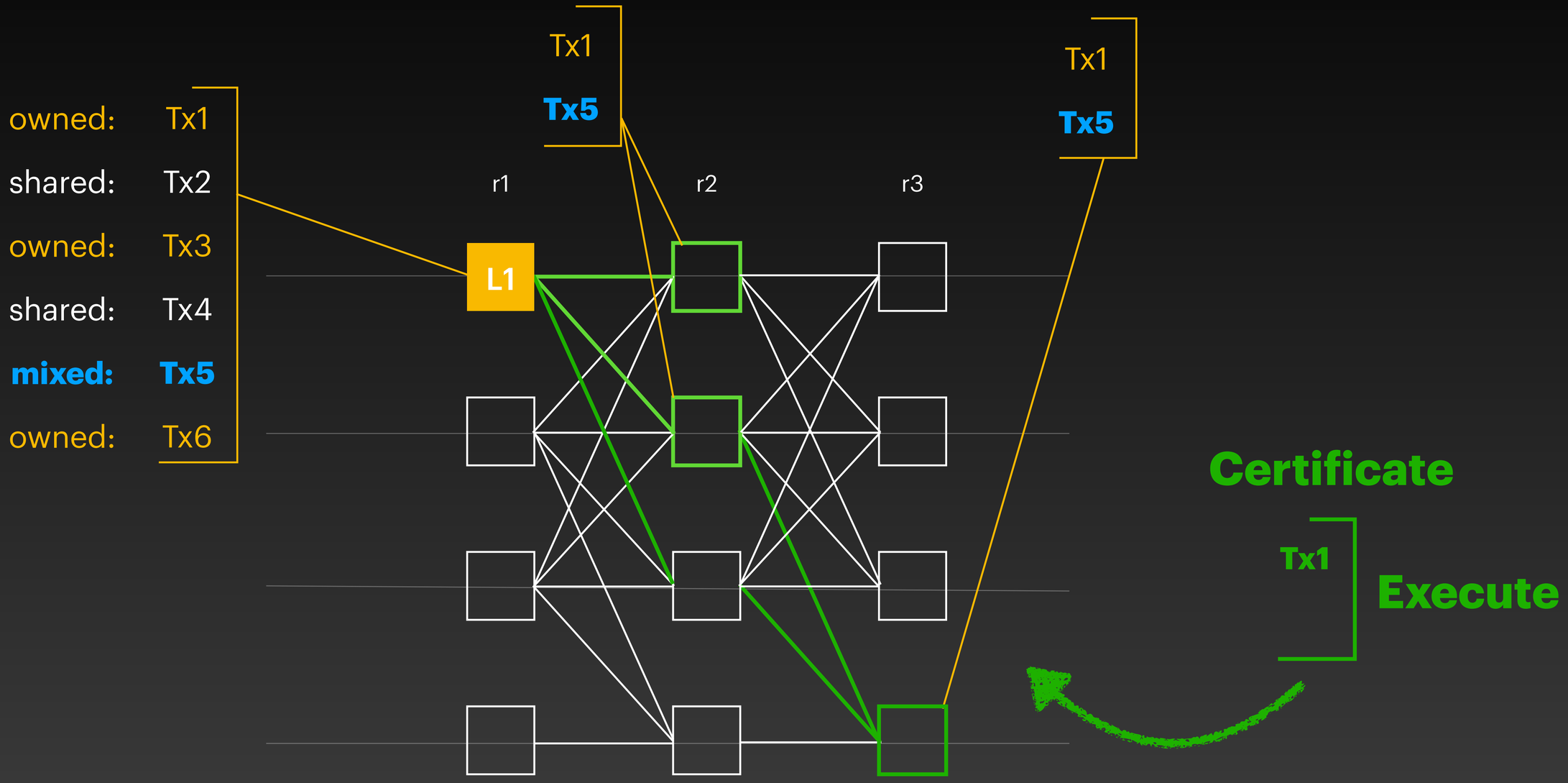
owned: Tx1
shared: Tx2
owned: Tx3
shared: Tx4
mixed: Tx5
owned: Tx6



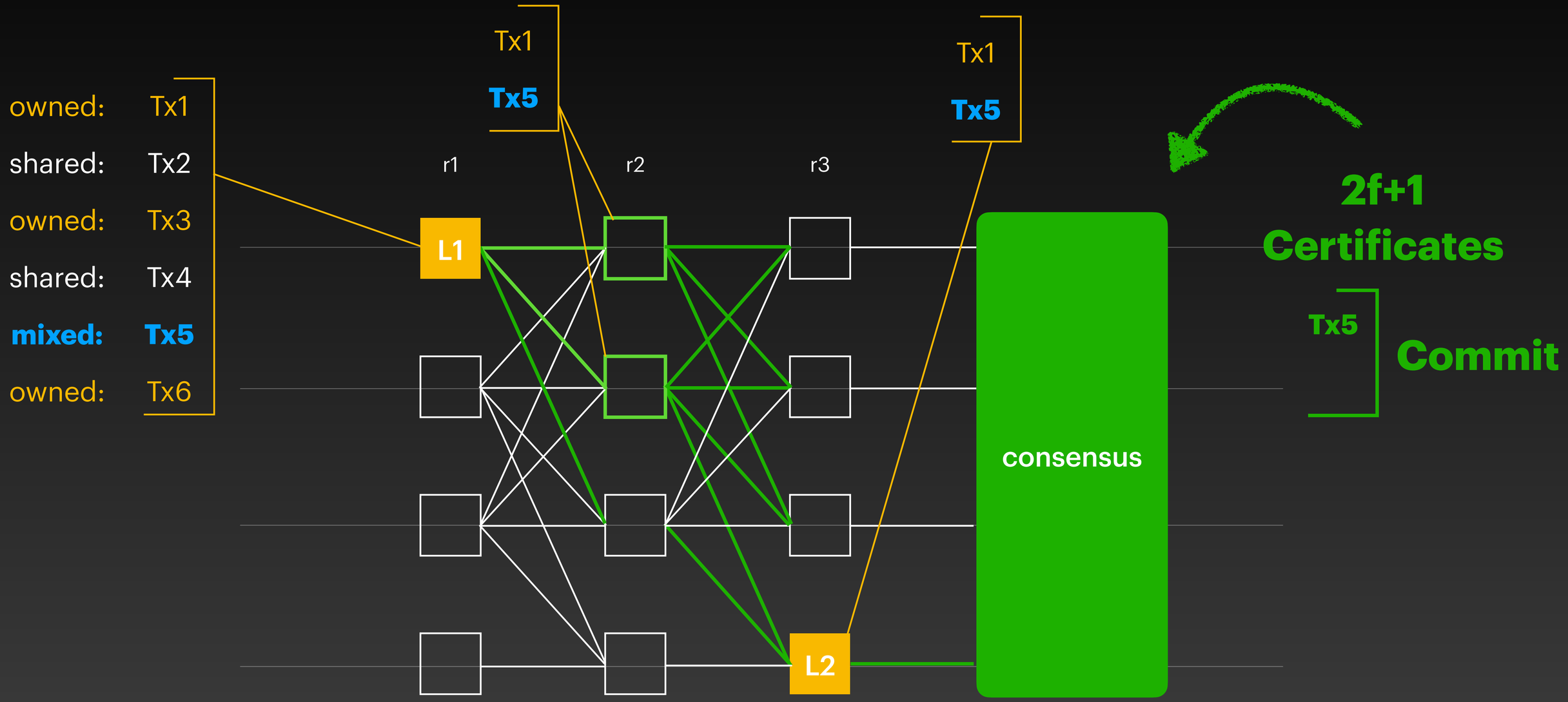
Commit

Tx2
Tx4

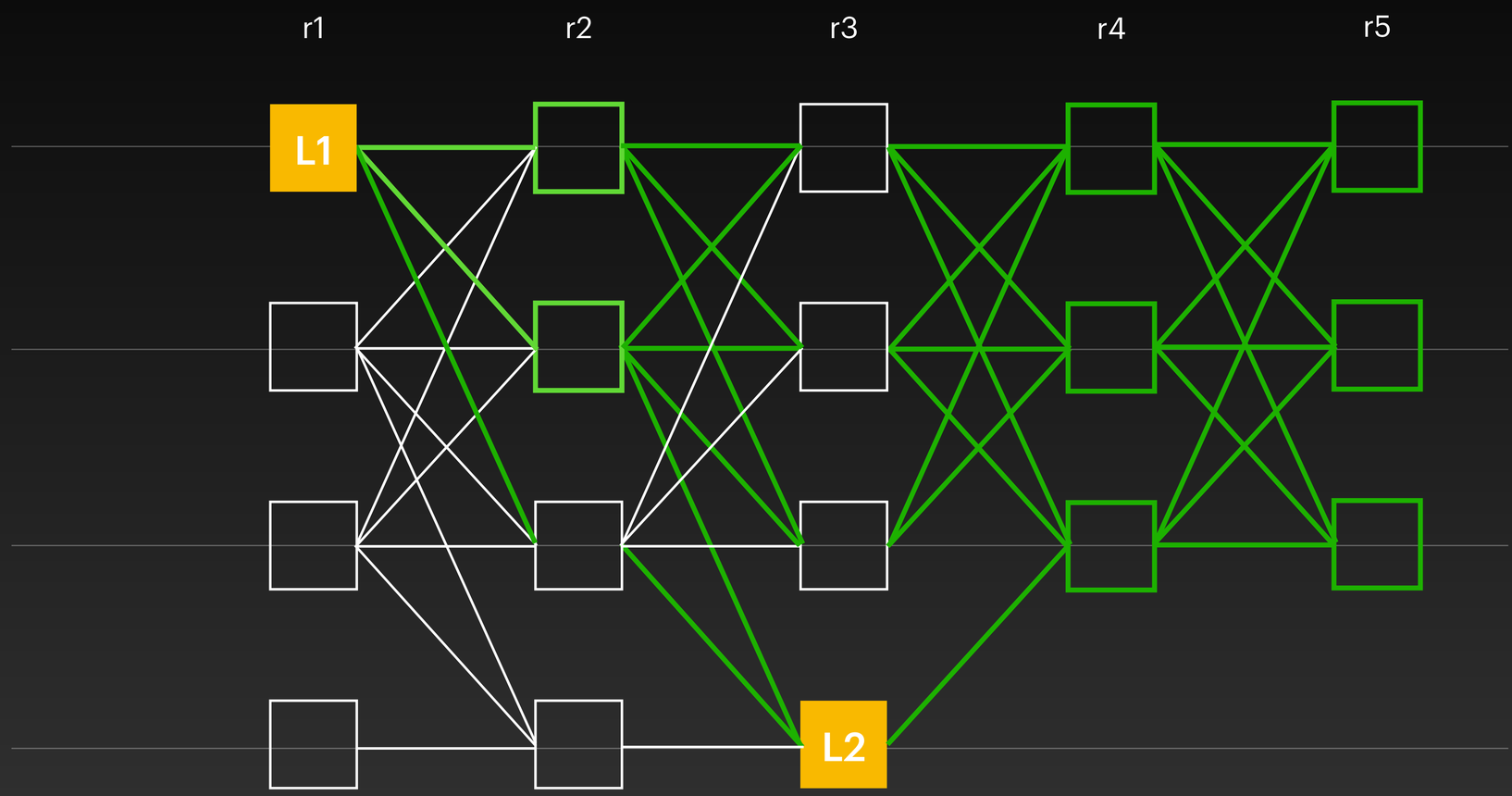
Mixed-Objects Transactions



Mixed-Objects Transactions



Mixed-Objects Transactions



lock owned objects

commit the lock on owned objects

Summary

Mysticeti

- A single message type
- Interpret patterns on the DAG

- **Paper:** https://sonnino.com/papers/mysticeti.pdf
- **Code:** https://github.com/mystenlabs/mysticeti

EXTRA

Preliminary Benchmarks

Implementation

- Written in Rust
- Networking: Tokio (TCP)
- Storage: custom WAL
- Cryptography: ed25519-consensus

https://github.com/mystenlabs/mysticeti

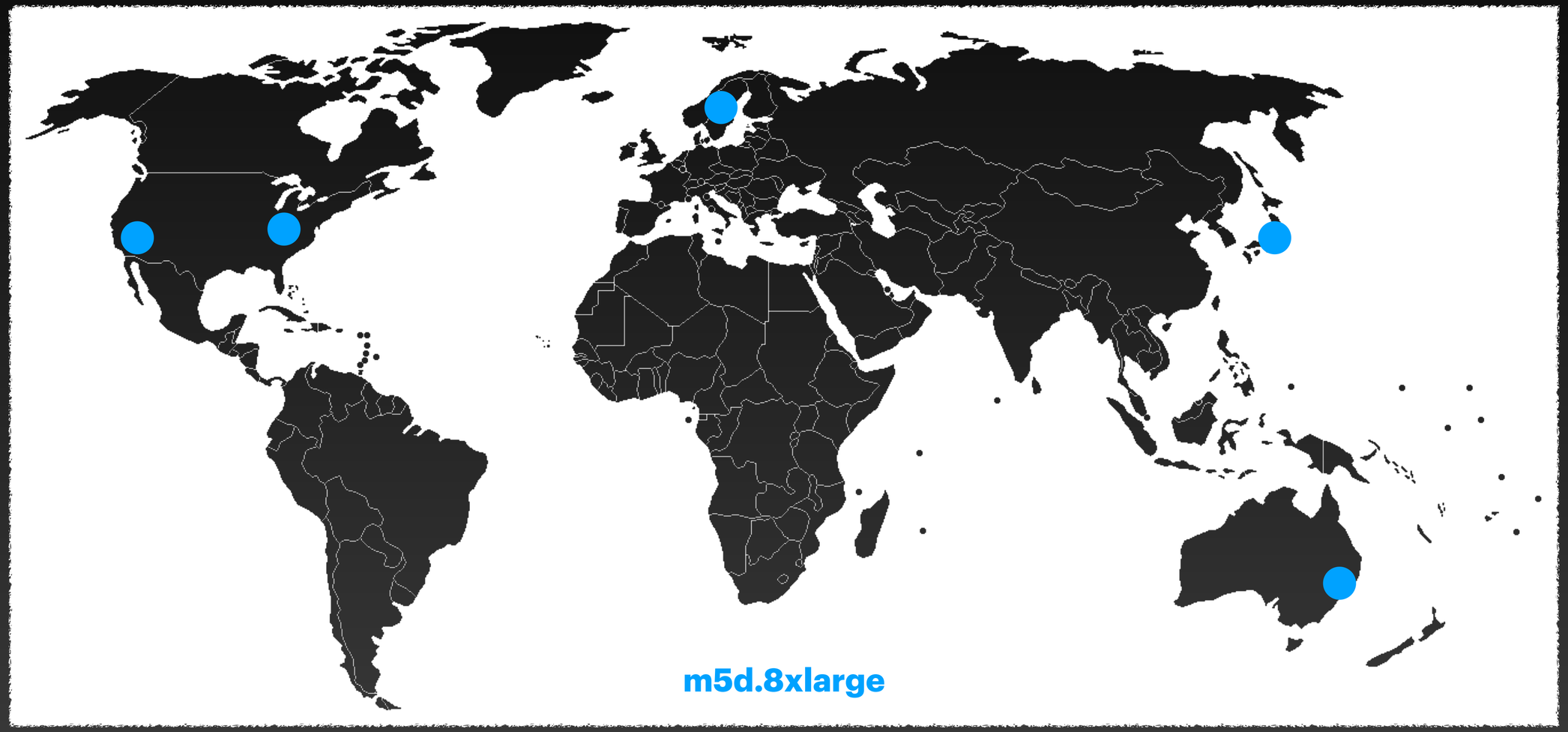
Implementation

- Synchronous core
- One Tokio task per peer (limiting resource usage)
- DTE simulator

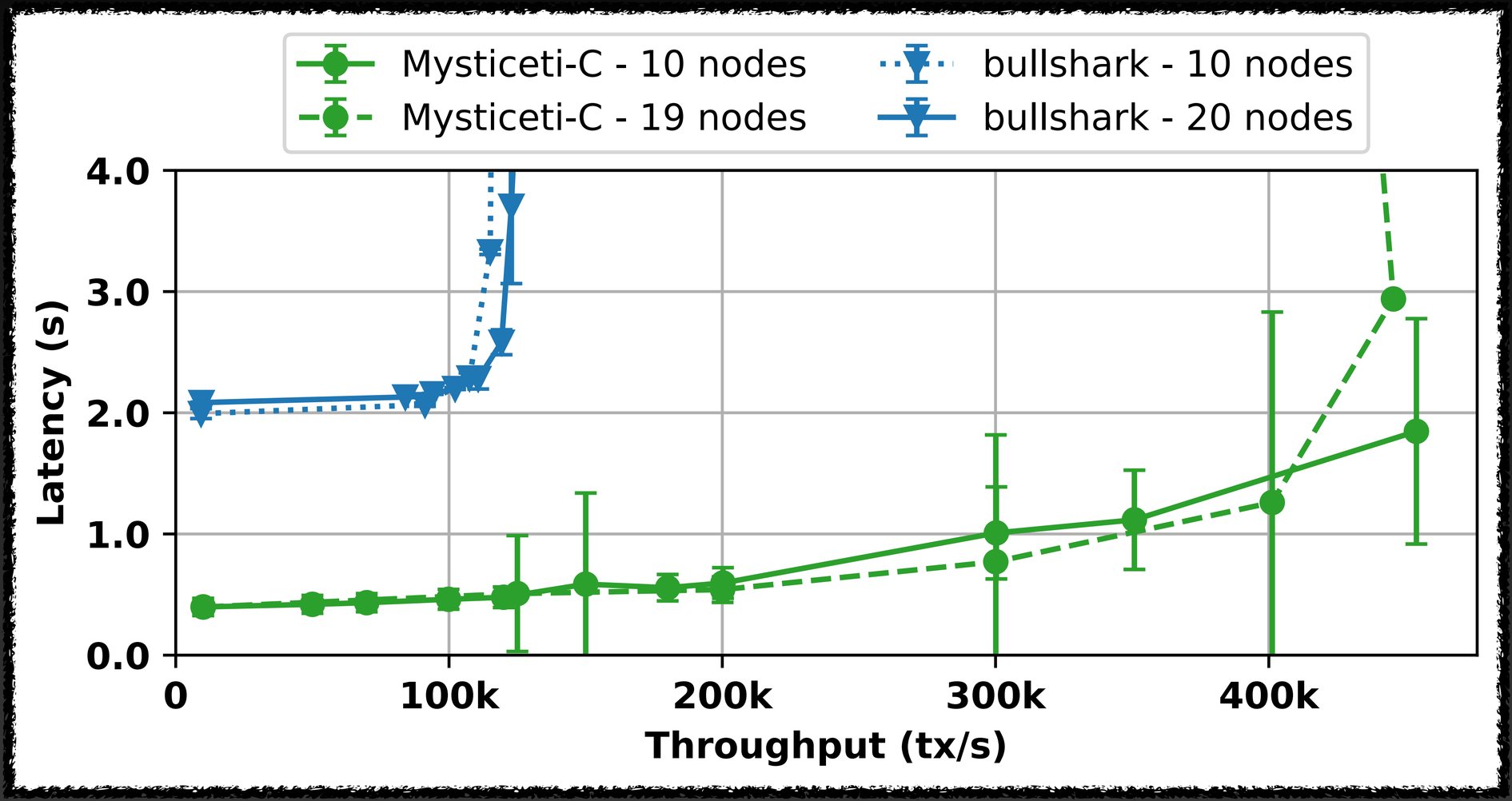
https://github.com/mystenlabs/mysticeti

Evaluation

Experimental setup on AWS



Preliminary Results



Engineering Benchmarks

Protocol	Committee	P50	P95
Bullshark	137	2.89 s	4.60 s
Mysticeti	137	650 ms	975 ms

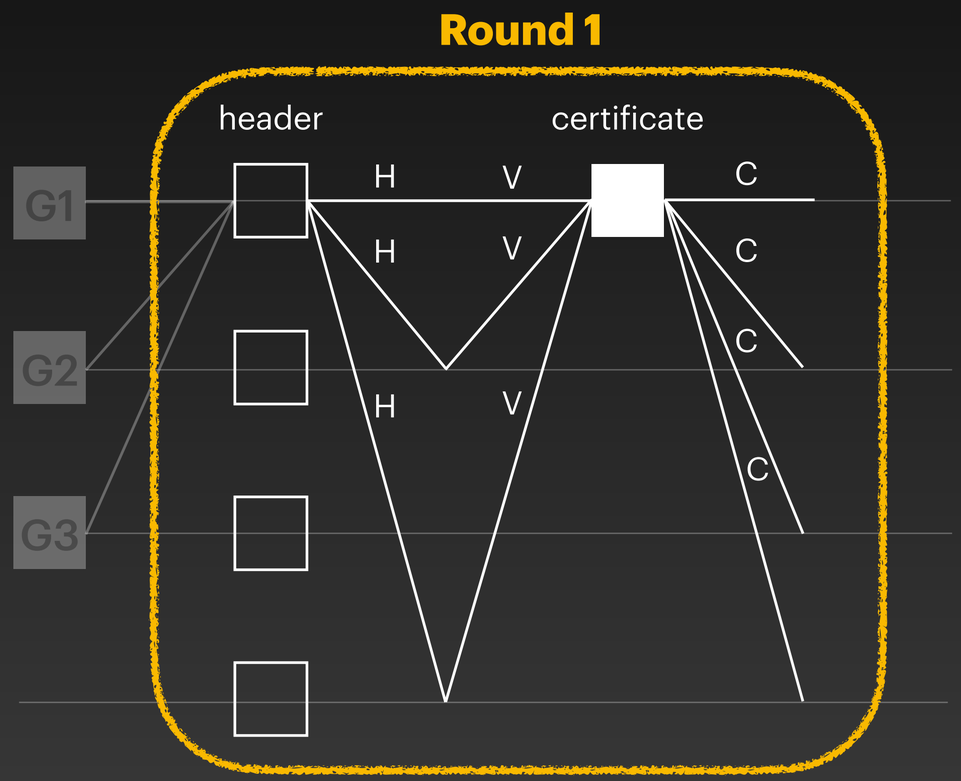
We run it at max load for 24h 👍

EXTRA

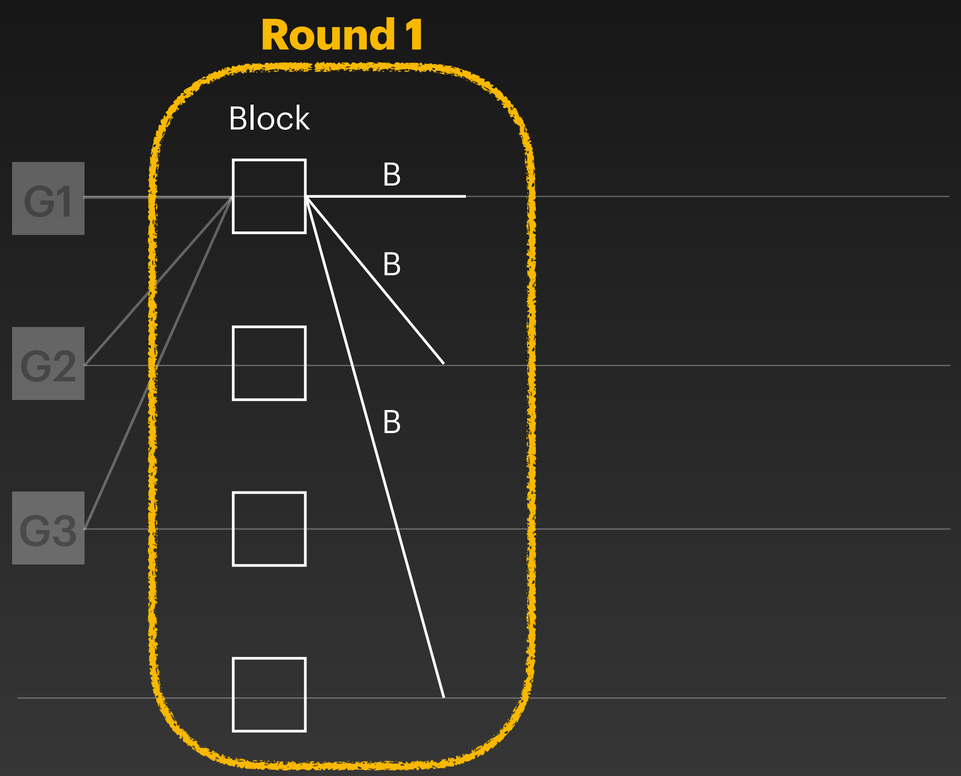
Narwhal vs Mysticeti

Narwhal vs Mysticeti

Narwhal

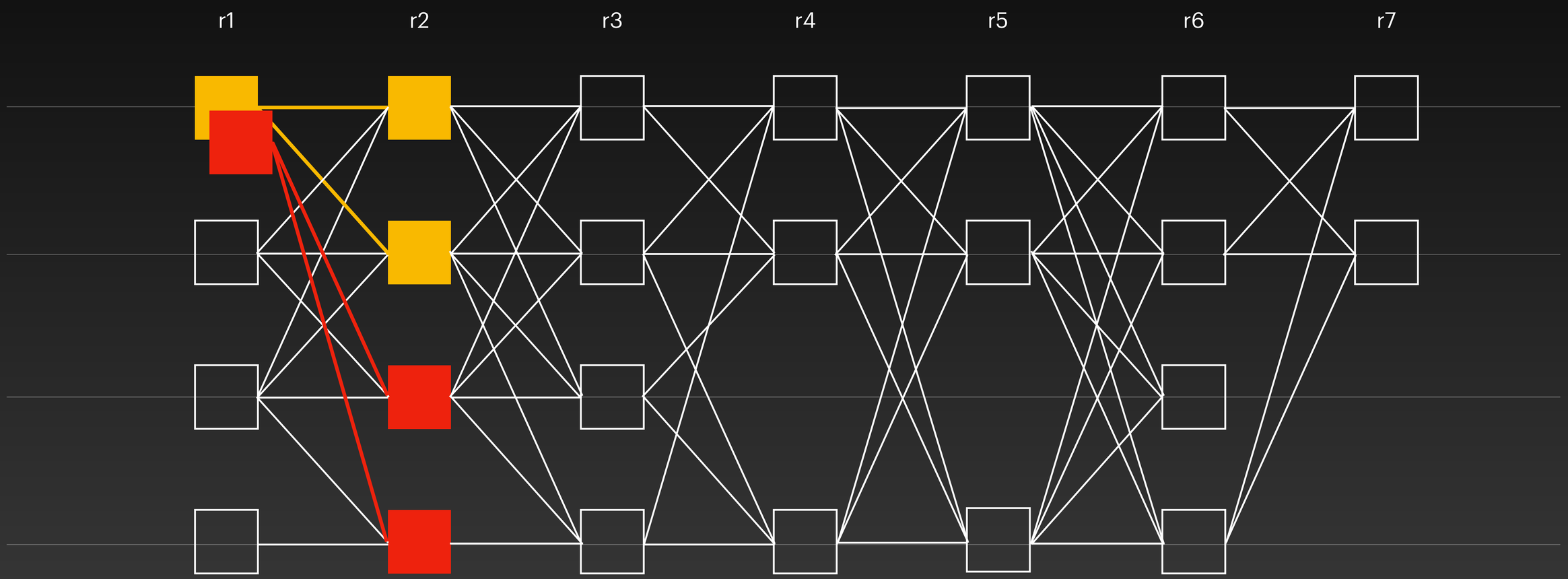


Mysticeti



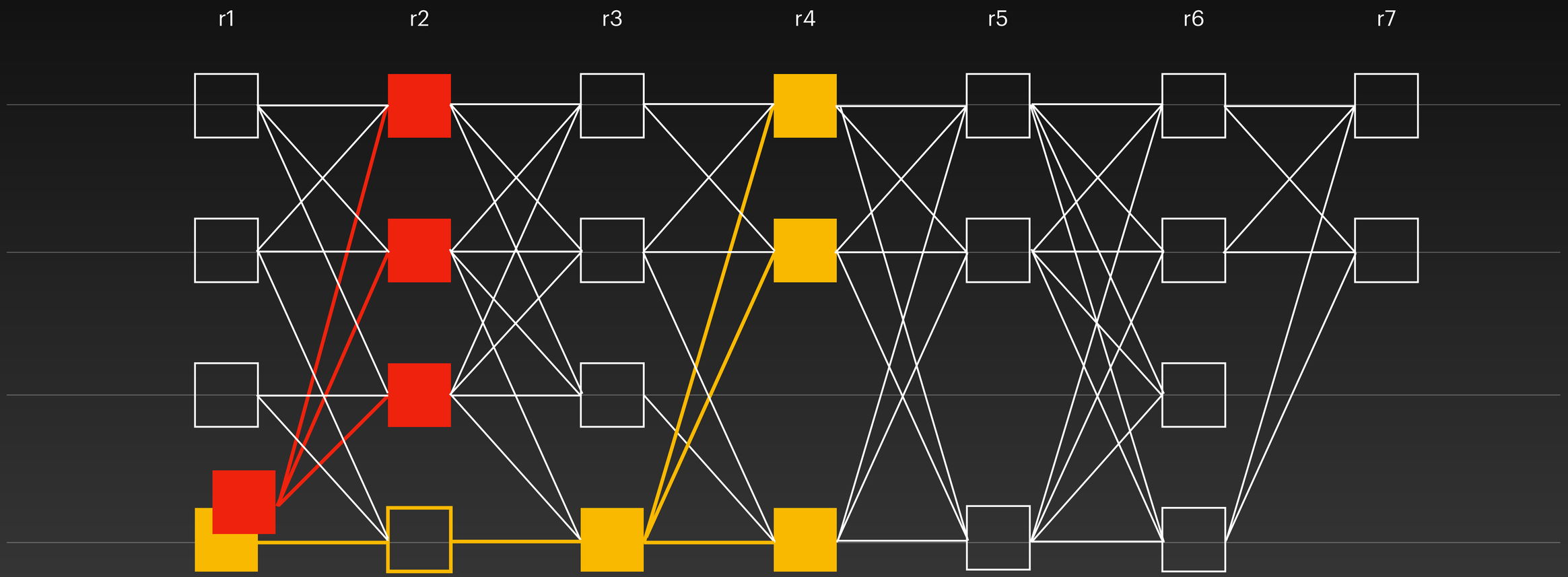
Main Challenge

Possible equivocations



Main Challenge

Possible equivocations (even with 2f+1 support)



Decision Rules

Upon interpreting the DAG...

Bullshark

- A leader is **Commit** or not
- Either directly or indirectly (recursion)

Mysticeti

- A leader is **Commit**, **Skip**, or **Undecided**
- Either directly or indirectly (recursion)

EXTRA

Linear vs DAG

Quorum-Based Consensus

Linear-Chain

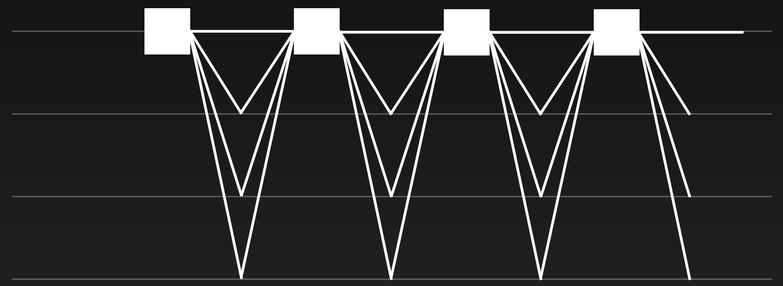
- Low latency
- Fragile to faults
- Complex leader-change

DAG-Based

- High latency
- Robust against faults
- No/Simple leader-change

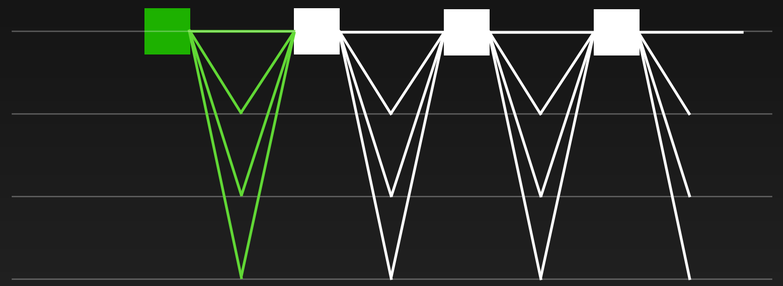
Linear-Chain Consensus

Rough overview



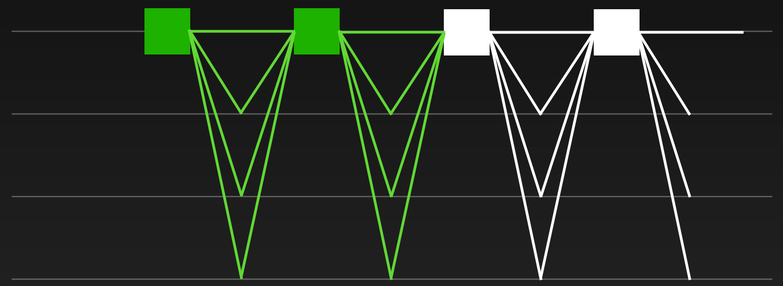
Linear-Chain Consensus

Rough overview



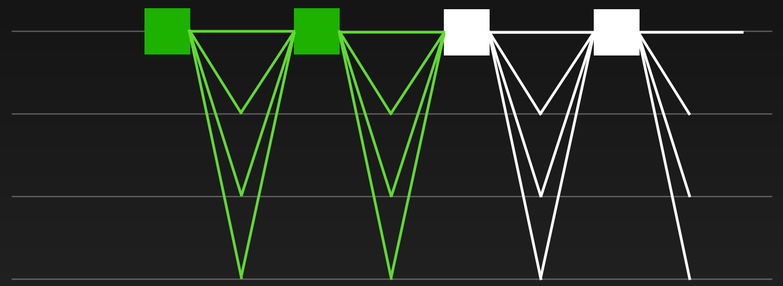
Linear-Chain Consensus

Rough overview



Linear-Chain Consensus

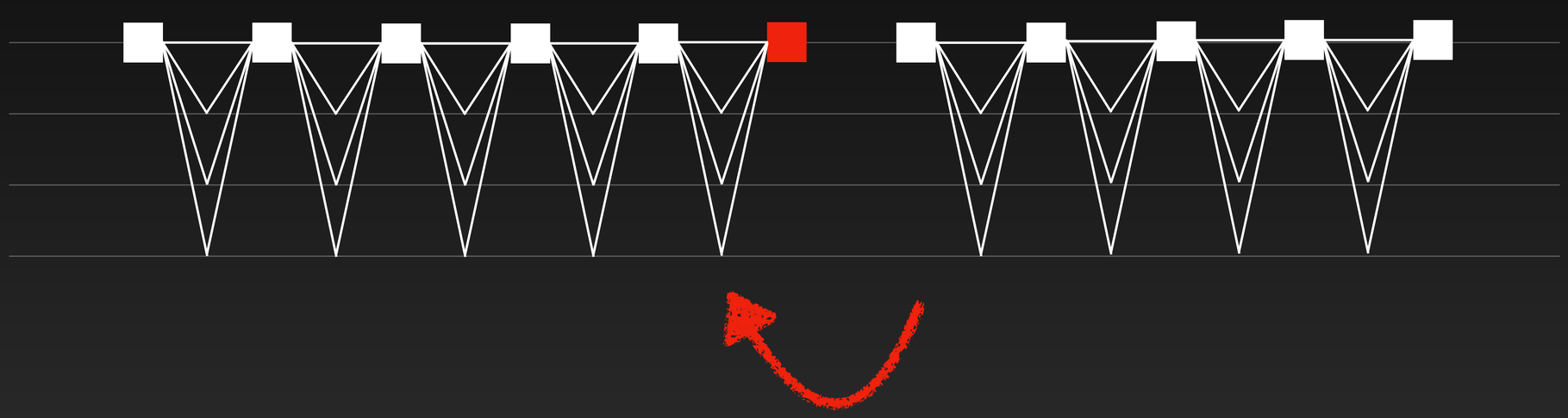
Rough overview



- The leader does all the work

Linear-Chain Consensus

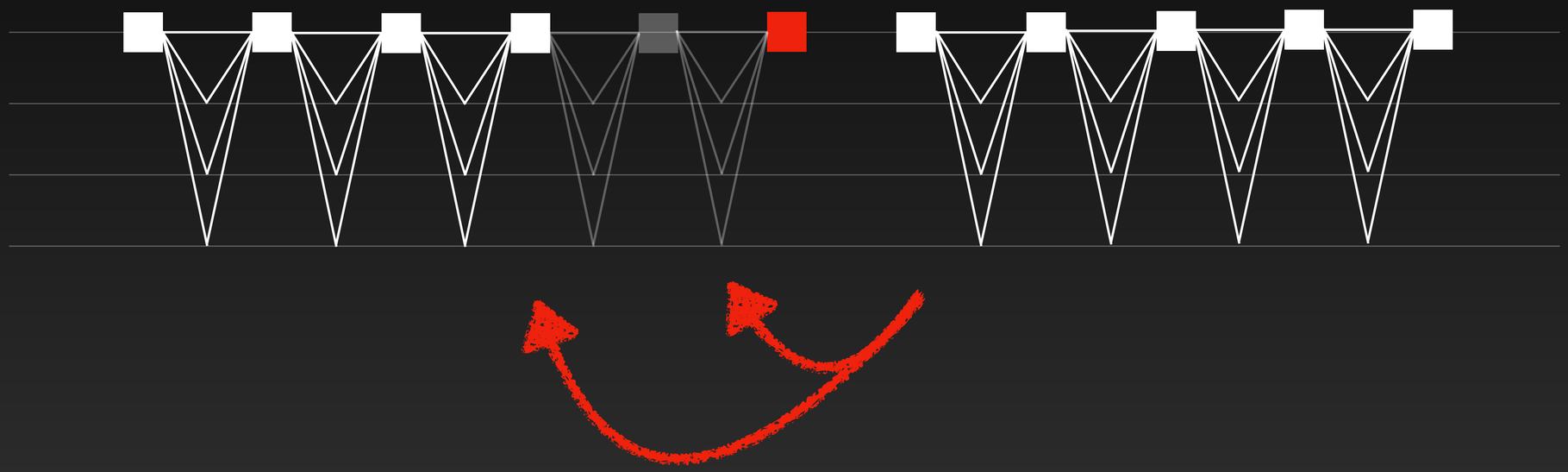
Rough overview



- The leader does all the work
- Complex leader-change

Linear-Chain Consensus

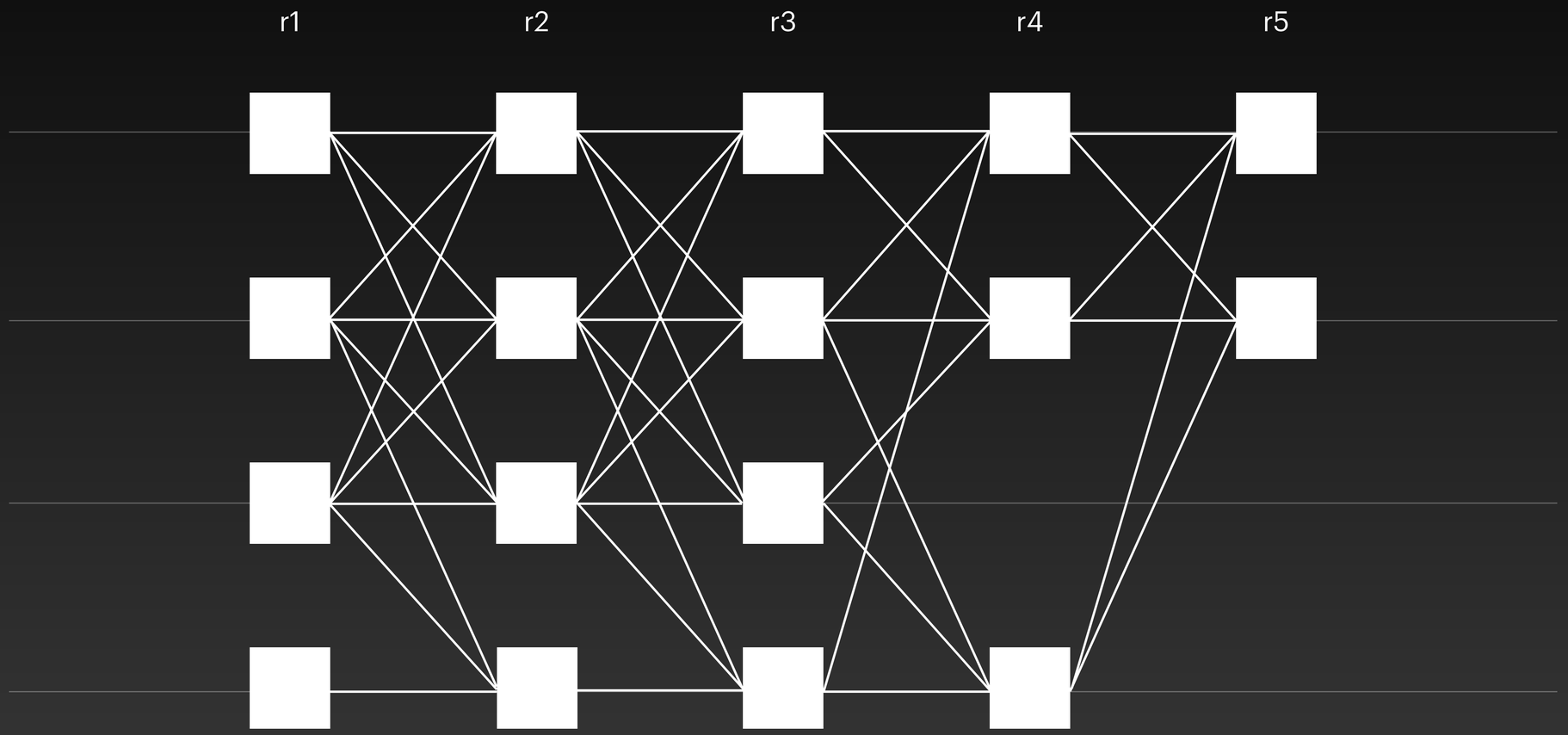
Rough overview



- The leader does all the work
- Complex leader-change

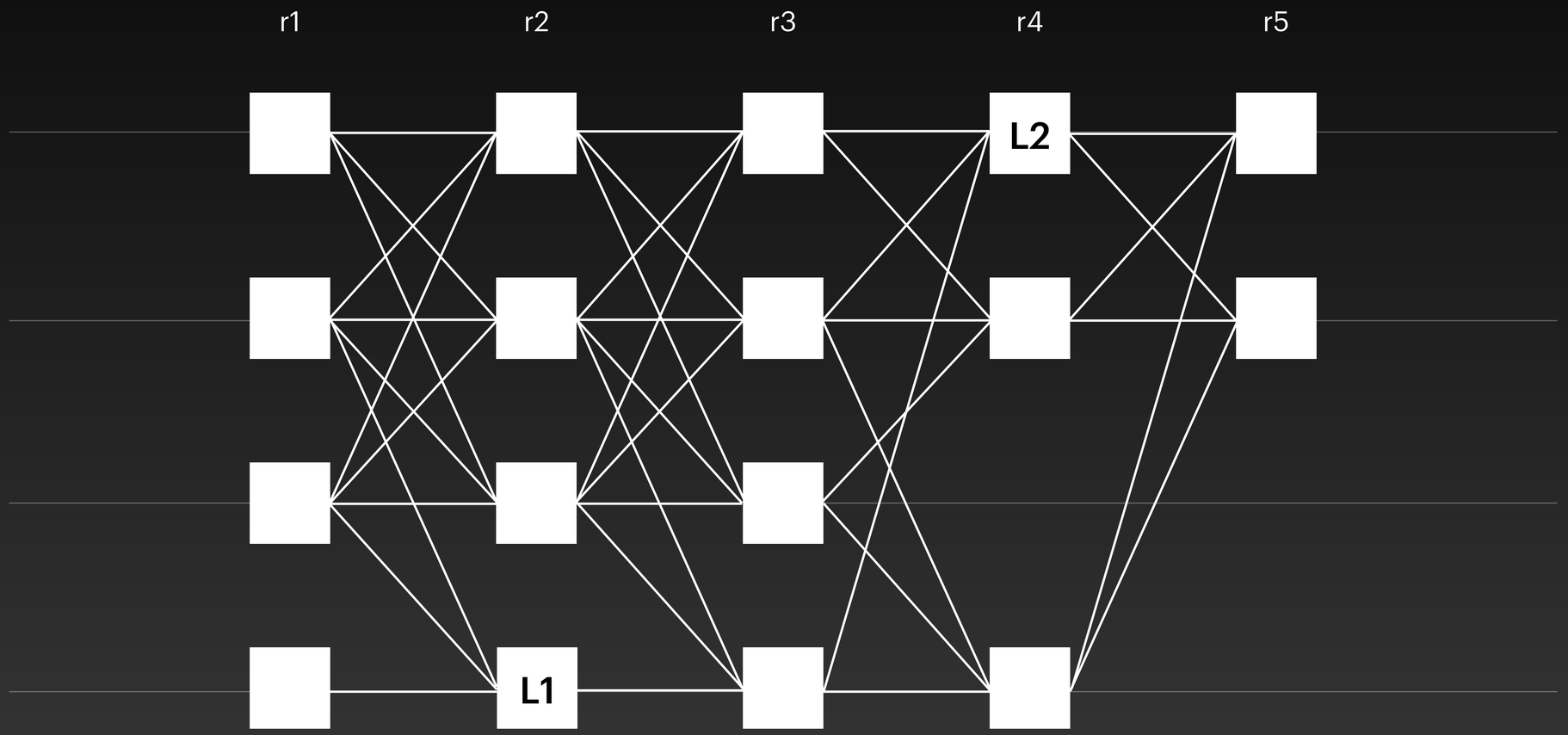
DAG-Based Consensus

Rough overview



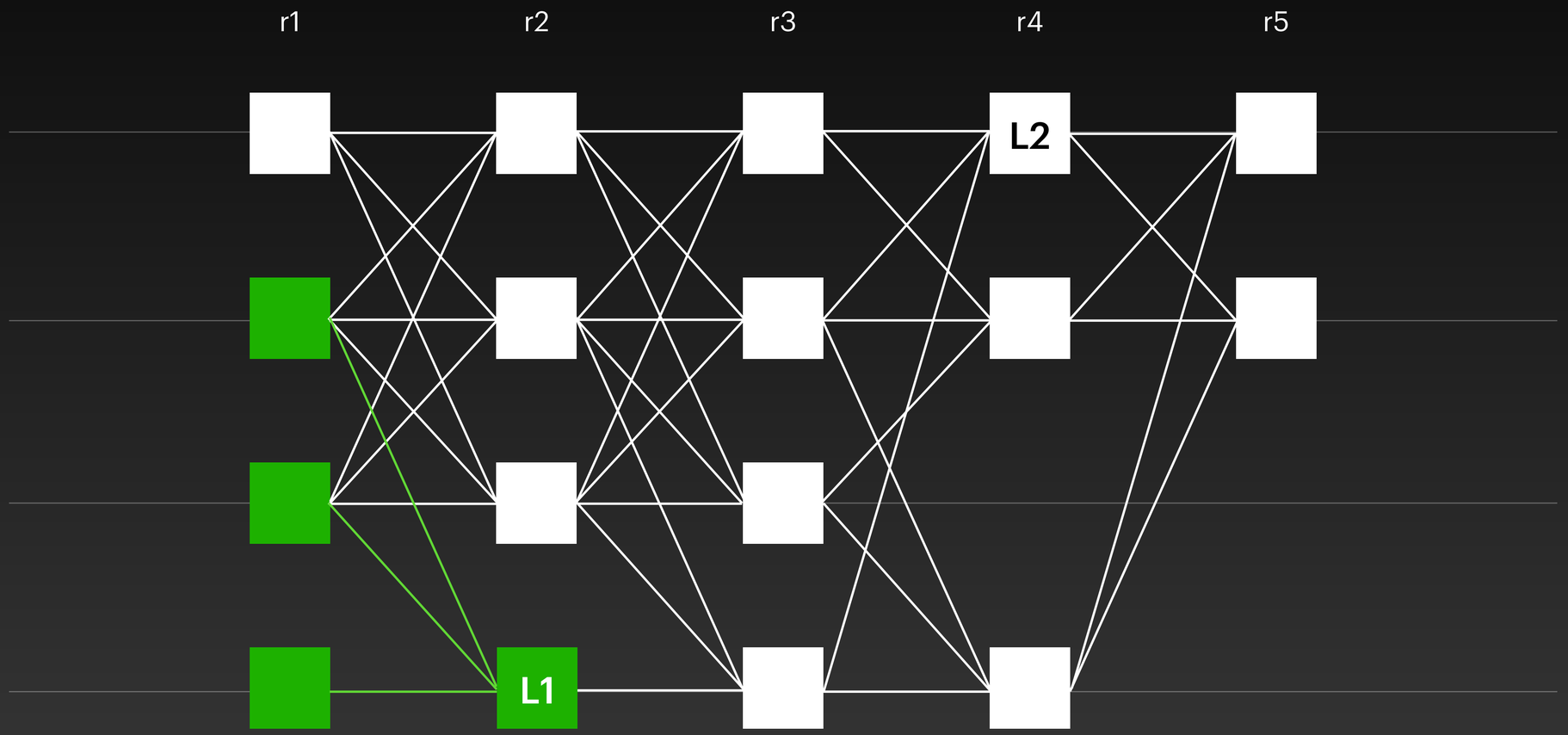
DAG-Based Consensus

Rough overview



DAG-Based Consensus

Rough overview



DAG-Based Consensus

Rough overview