Blockchain and DHT based lookup system aiming for alternative DNS

ICCCI 2020

Chukyo University : Kazuma Matsuoka, Tsunehiko Suzuki

Introduction

- We propose a lookup system using blockchain and DHT
    - our goal is this system will be an alternative to DNS

- Blockchain
    - guarantees the data integrity

- DHT
    - stores data among participating nodes distributedly
    - can retrieve and propagate any data efficiently

System Model

- Network Layer
  - has a mechanism to propagate transactions and blocks
- Consensus Layer
  - has a function to judge which transactions or blocks are valid
  - **No suitable consensus algorithm has been determined for this system, but we implemented it in PoW for now**
- Storage Layer
  - has a function as a global memory that stores data authenticated by the consensus layer
- View Layer
  - represents the state of participating nodes' data

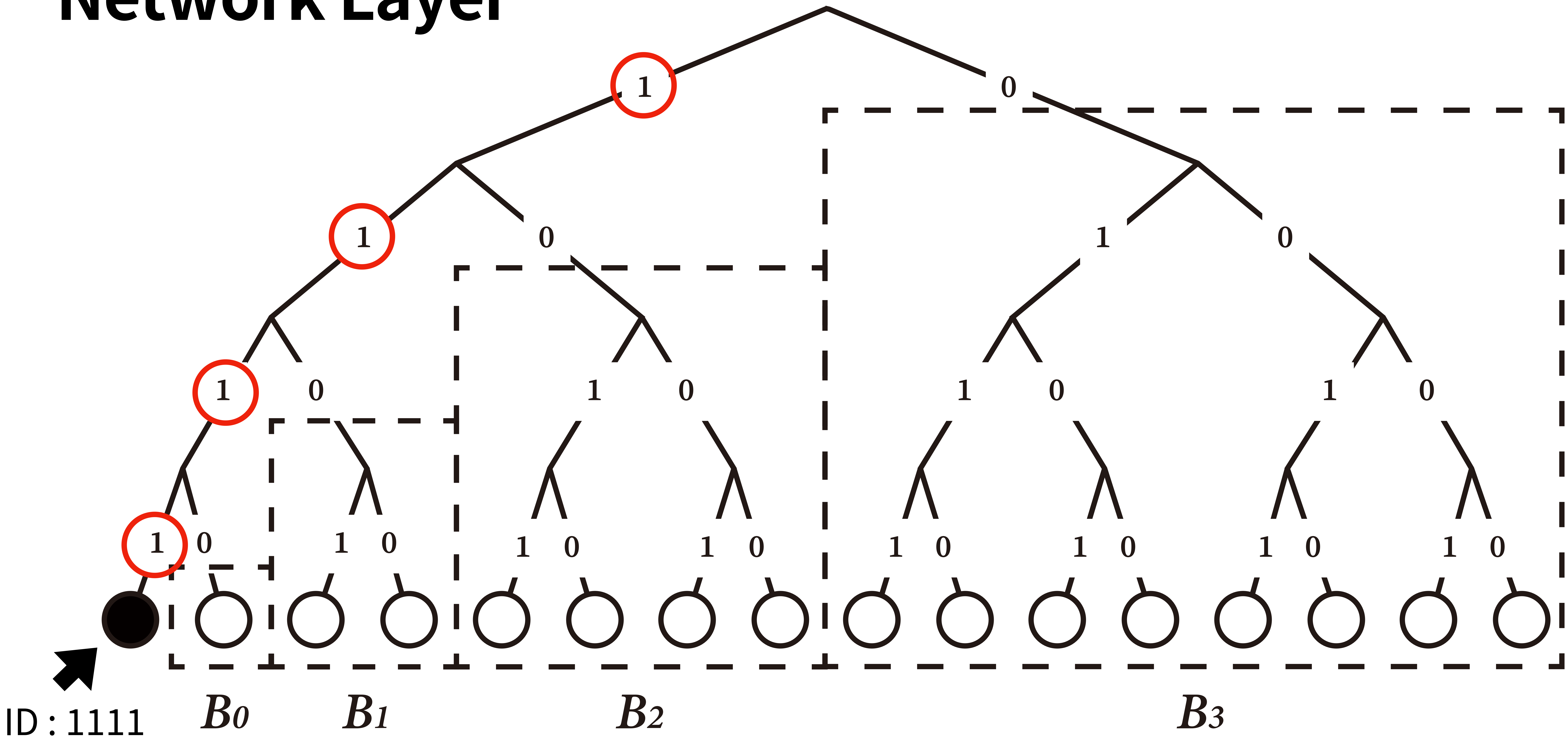
Network Layer

- Kademlia
  - all nodes, transactions, and blocks are inserted as the Kademlia overlay network nodes
  - **to enable all nodes to participate in mining, it is necessary to propagate transactions and blocks to all nodes**

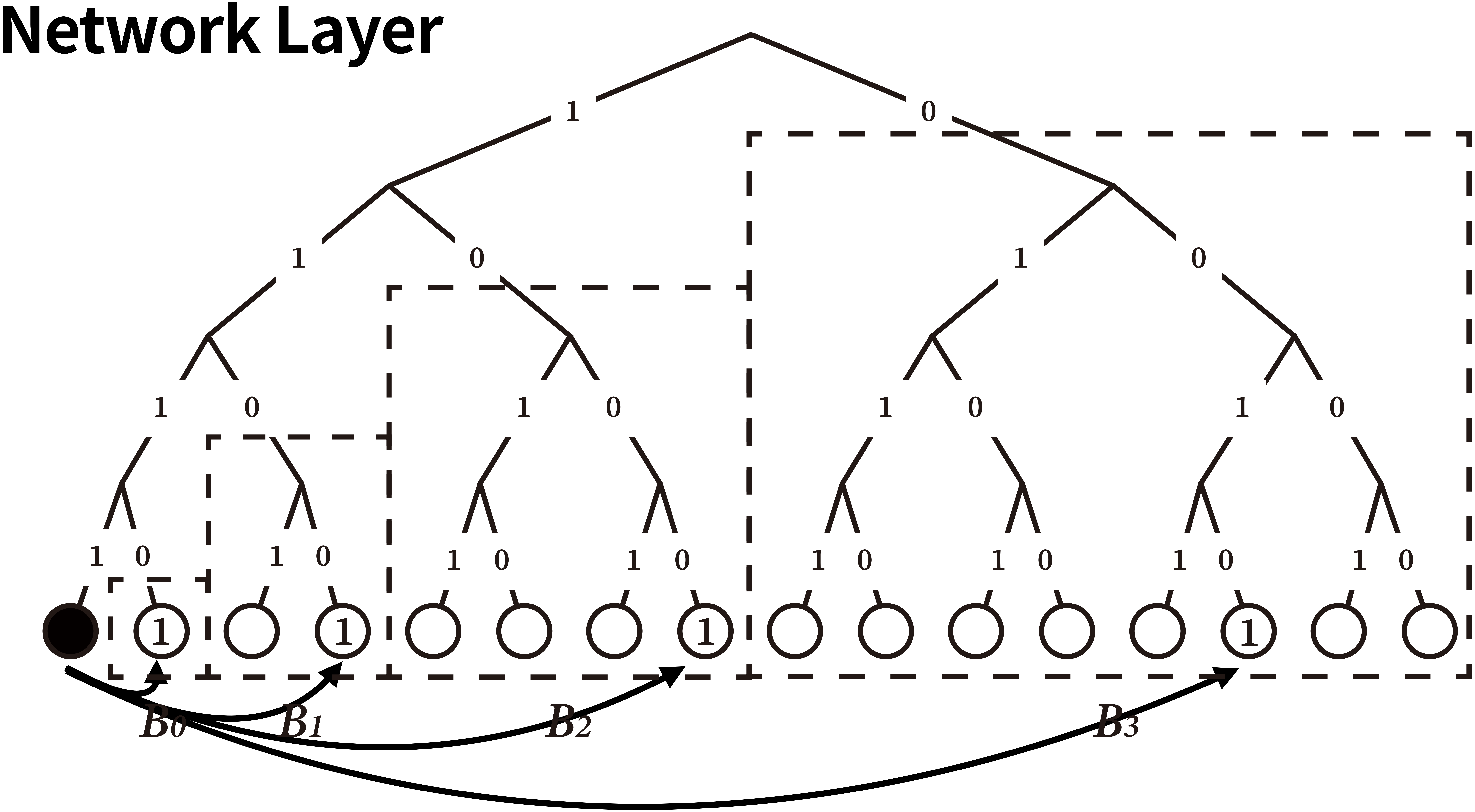
    → We use “Kadcast”

- Kadcast
  - any nodes can propagate data to all nodes efficiently

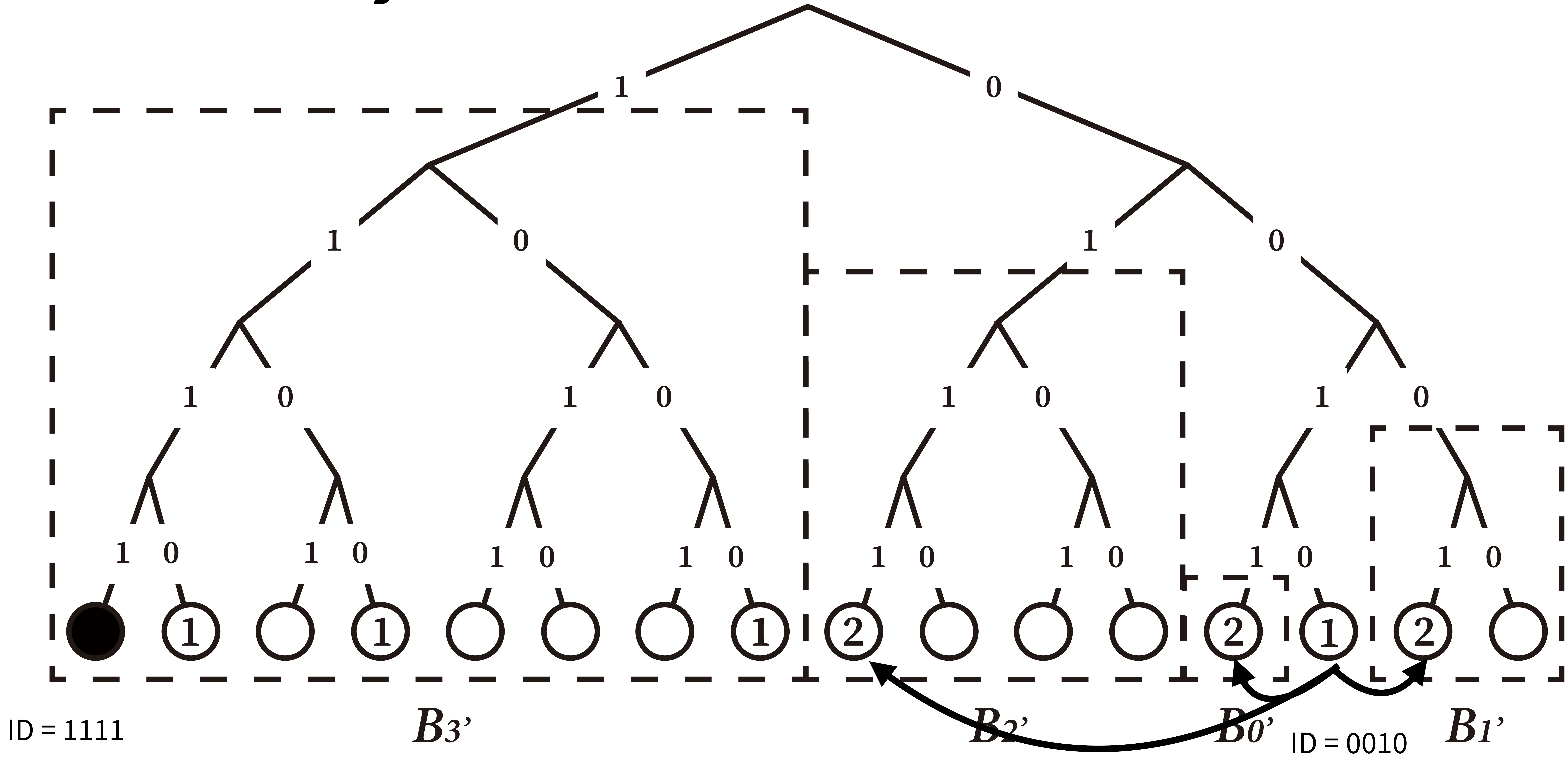
Network Layer



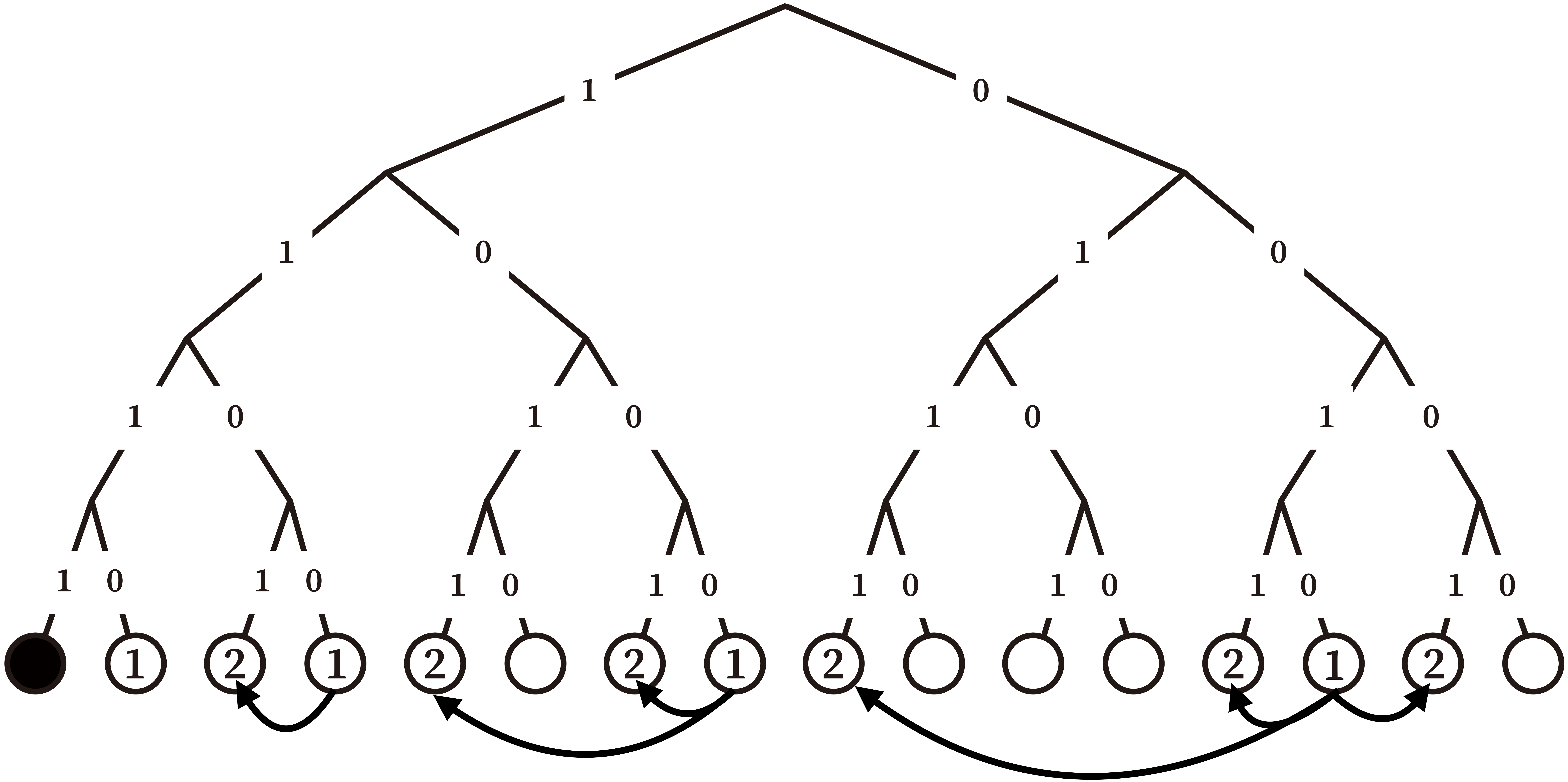
Network Layer



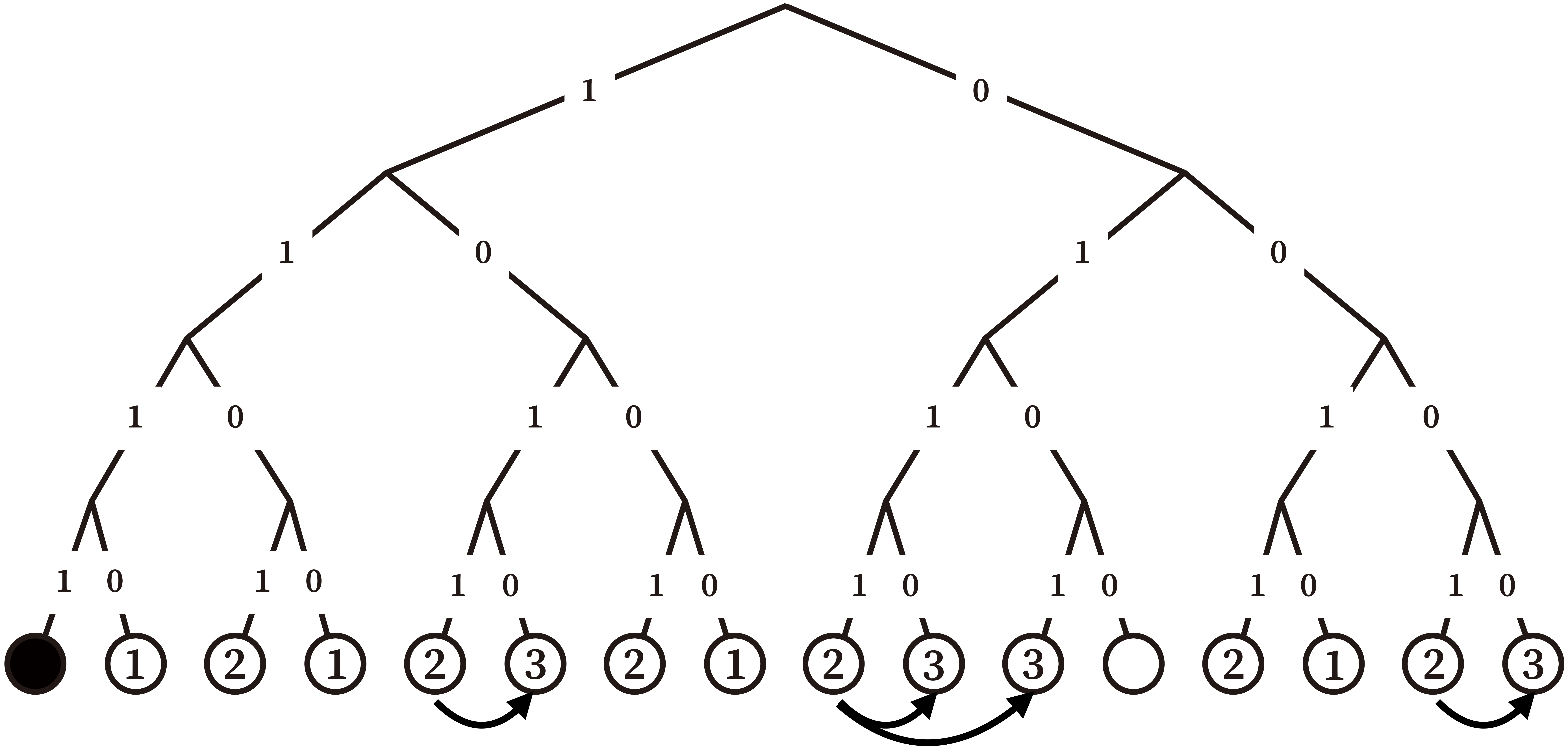
Network Layer



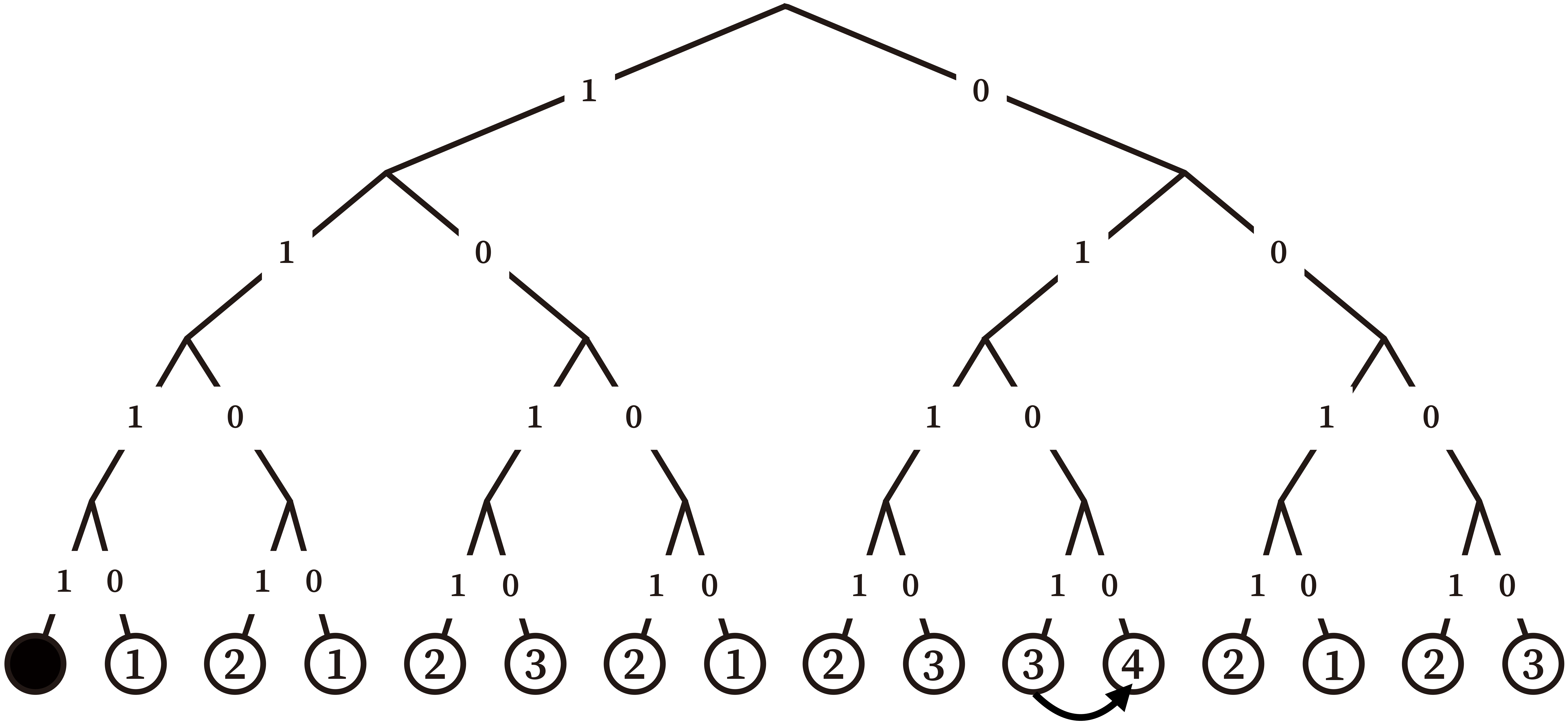
Network Layer



Network Layer



Network Layer



Consensus Layer

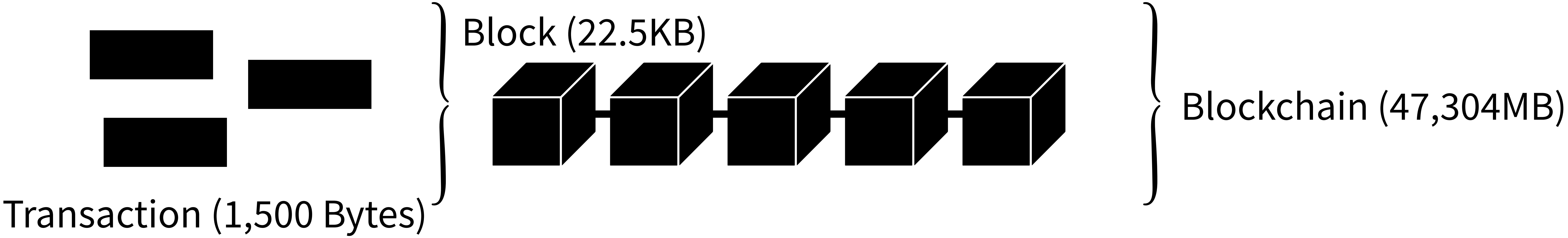
- $Tx = (key, value, hash, owner, pubkey, sig, block)$
  - $key, value$: contents to store
  - $hash$: hash value of the key and indicates which node holds this transaction
  - $owner$: ID of a node that issued this transaction
  - $pubkey, sig$: public key that $owner$ has and other nodes can verify the signature (sig)
    Designing a mechanism that guarantees the $pubkey$ is outside the scope of this paper
  - $block$: hash value of a block including this transaction

Consensus Layer

- $Block = (height, owner, nonce, prev\_hash, hash, txs)$
    - $height$: the order of the block
    - $nonce$: a number that proves the correctness of the block
    - $prev\_hash$: hash value of a block before this one
    - $hash$: hash value of this block
    - txs : list of transactions included in the block

Storage Layer

- Data estimation
  - We assume that transactions are issued every second and mining interval is 15s



Block (22.5KB)

Blockchain (47,304MB)

Transaction (1,500 Bytes)

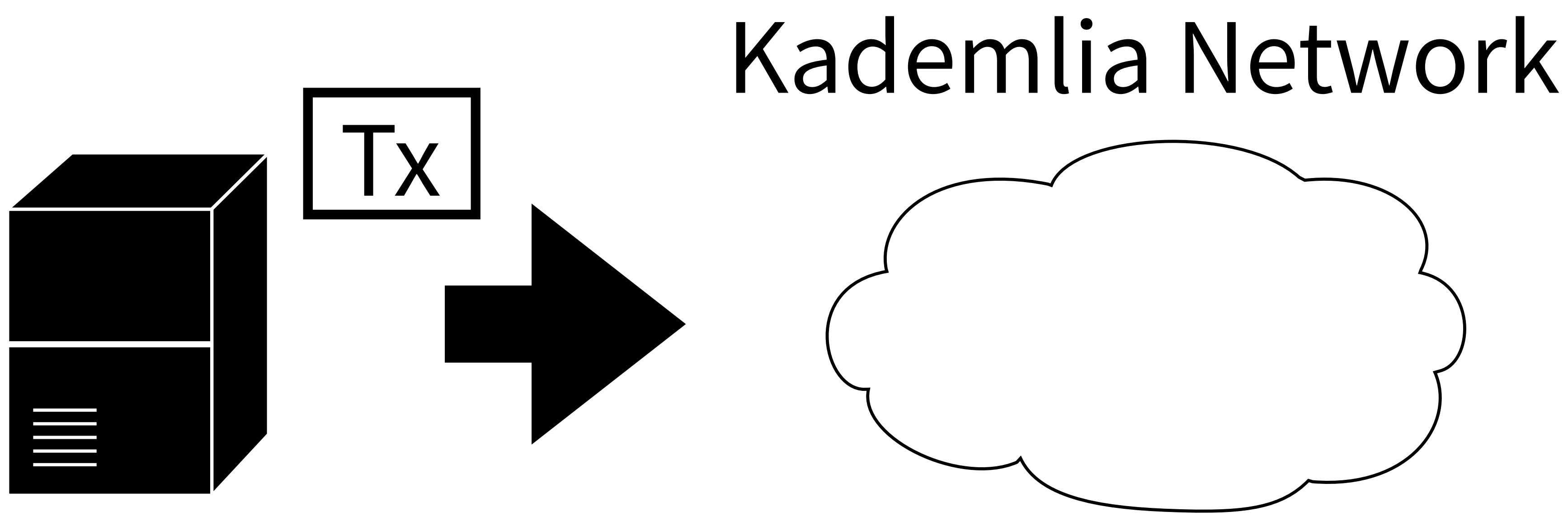
  - It is essential to store any data to multiple nodes, for all nodes are not always active

$$Data = \frac{47{,}304 \times (x + y + 1)}{N} \ \text{(MB)}$$

x	y	N	Data (MB)
5	5	1,000	520.344
10	10	10,000	99.338
20	20	10,000	193.946

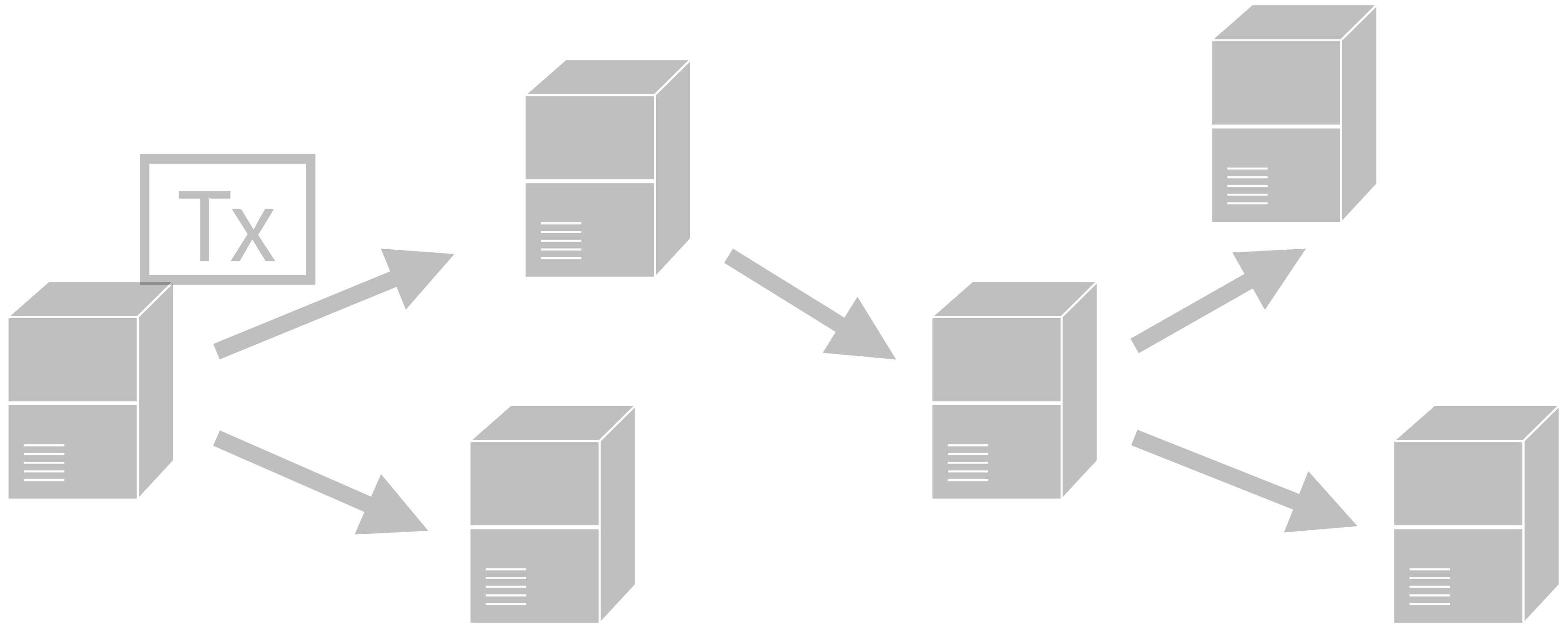
Store and retrieve for data

1



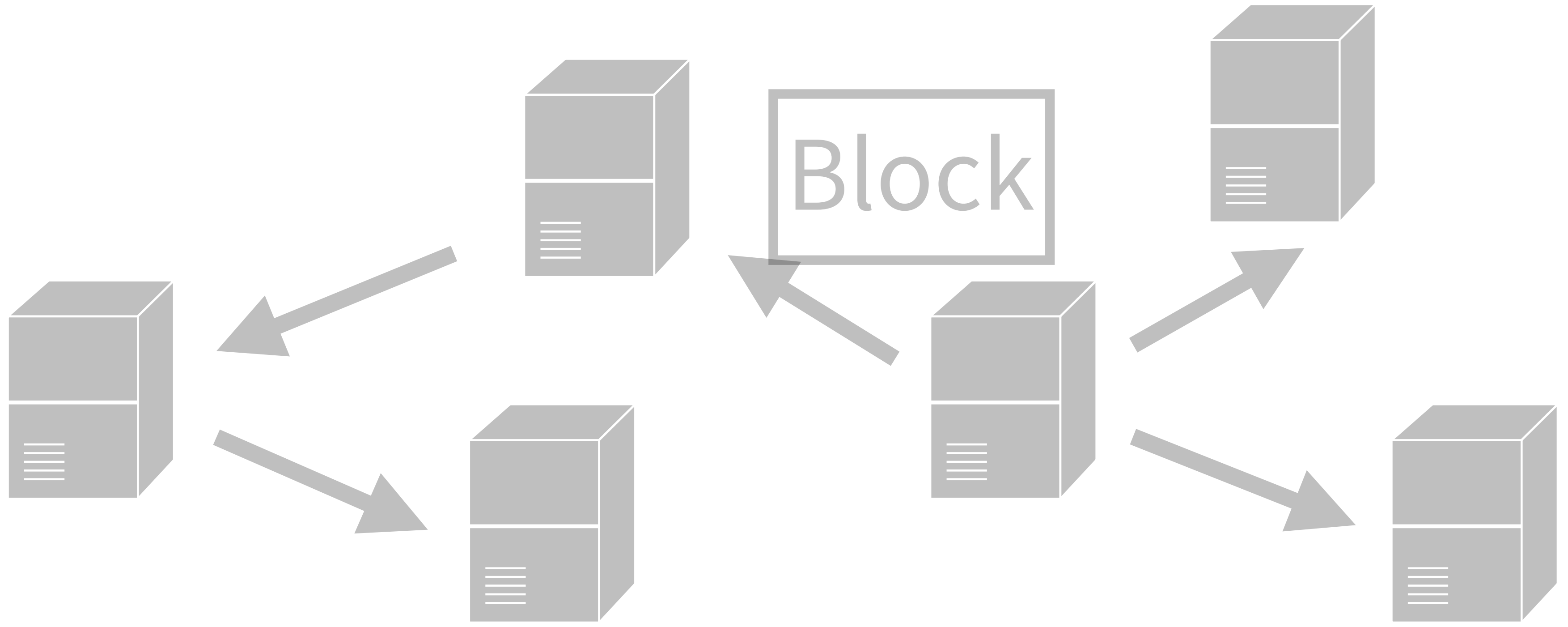
Issues a transaction using (key, value) data

2



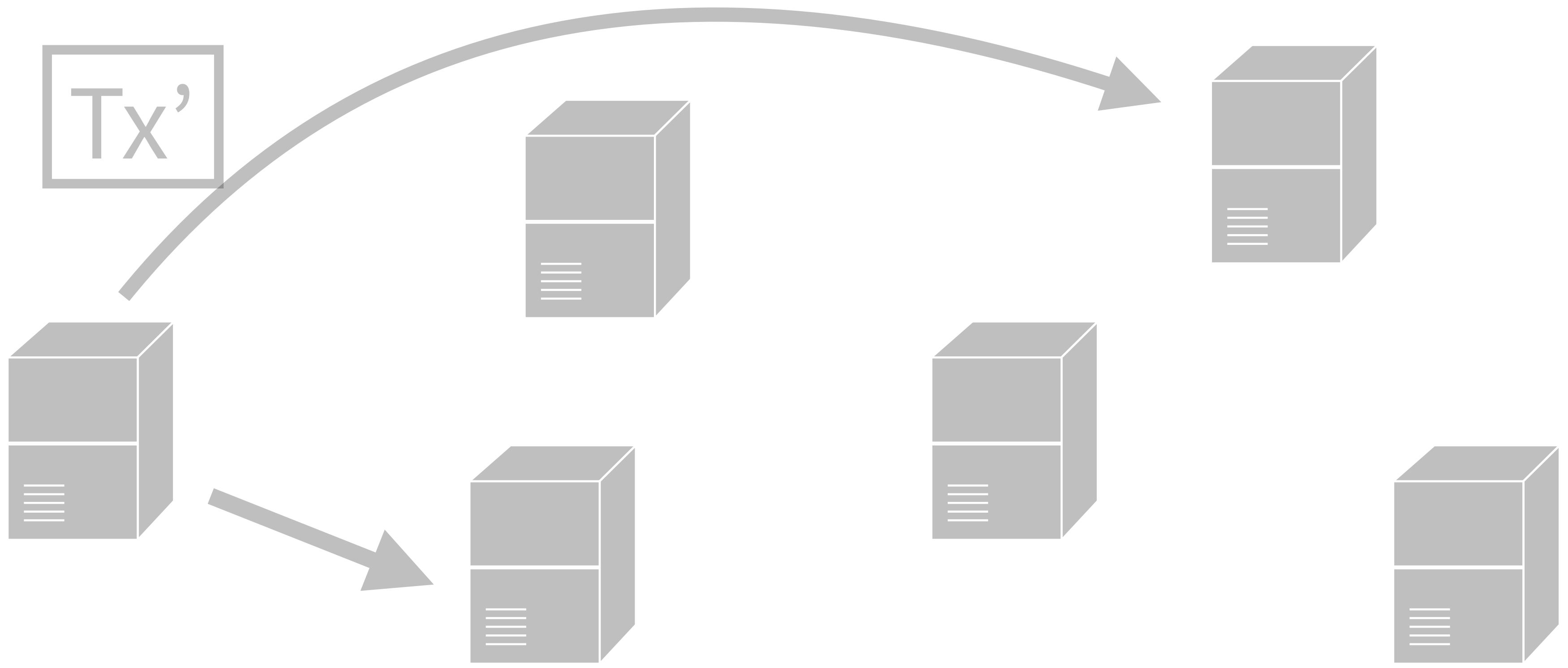
Tx is broadcasted and propagated to all nodes

3



If the Tx is legitimate, it will eventually be included in a block

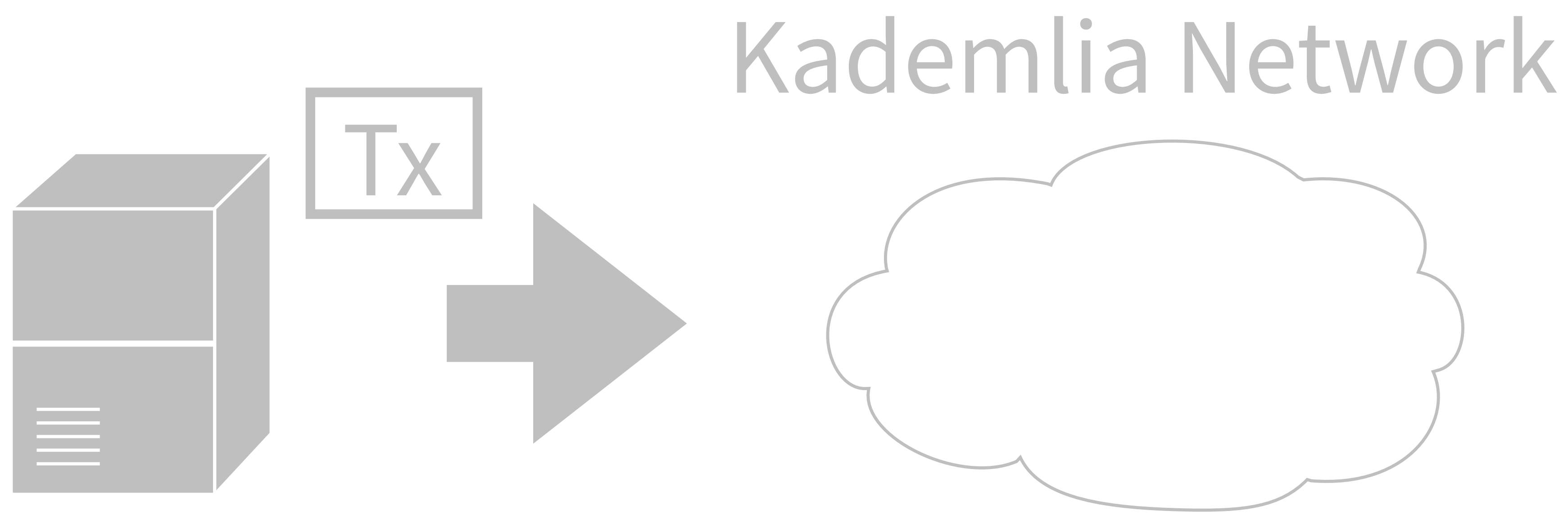
4



Send the transaction to some nodes to store

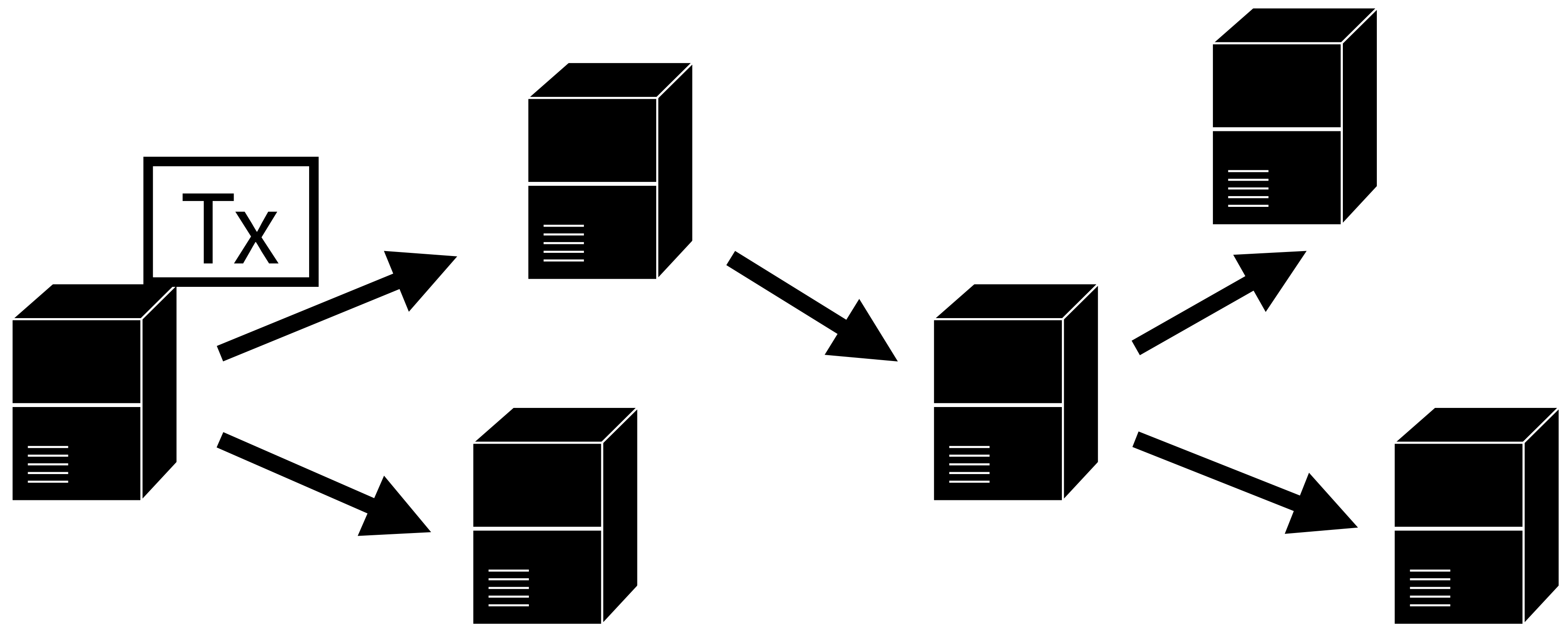
Store and retrieve for data

1



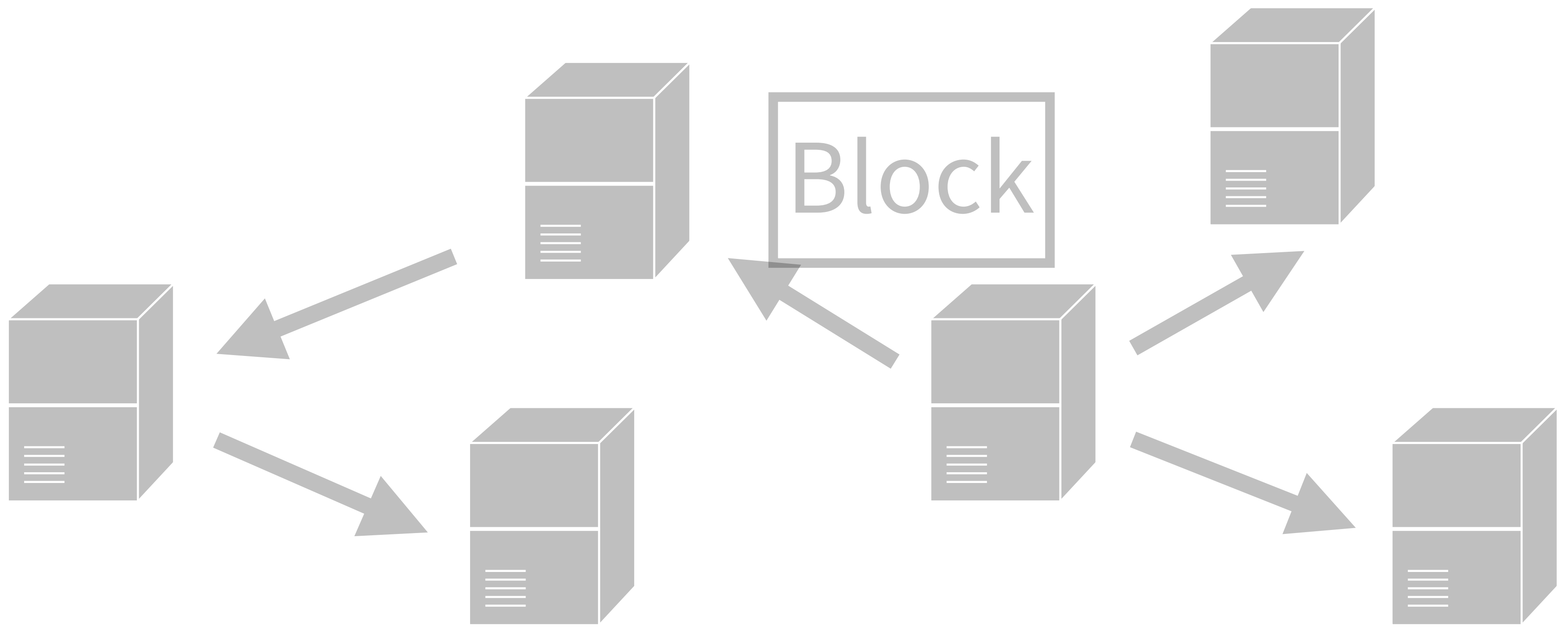
Issues a transaction using (key, value) data

2



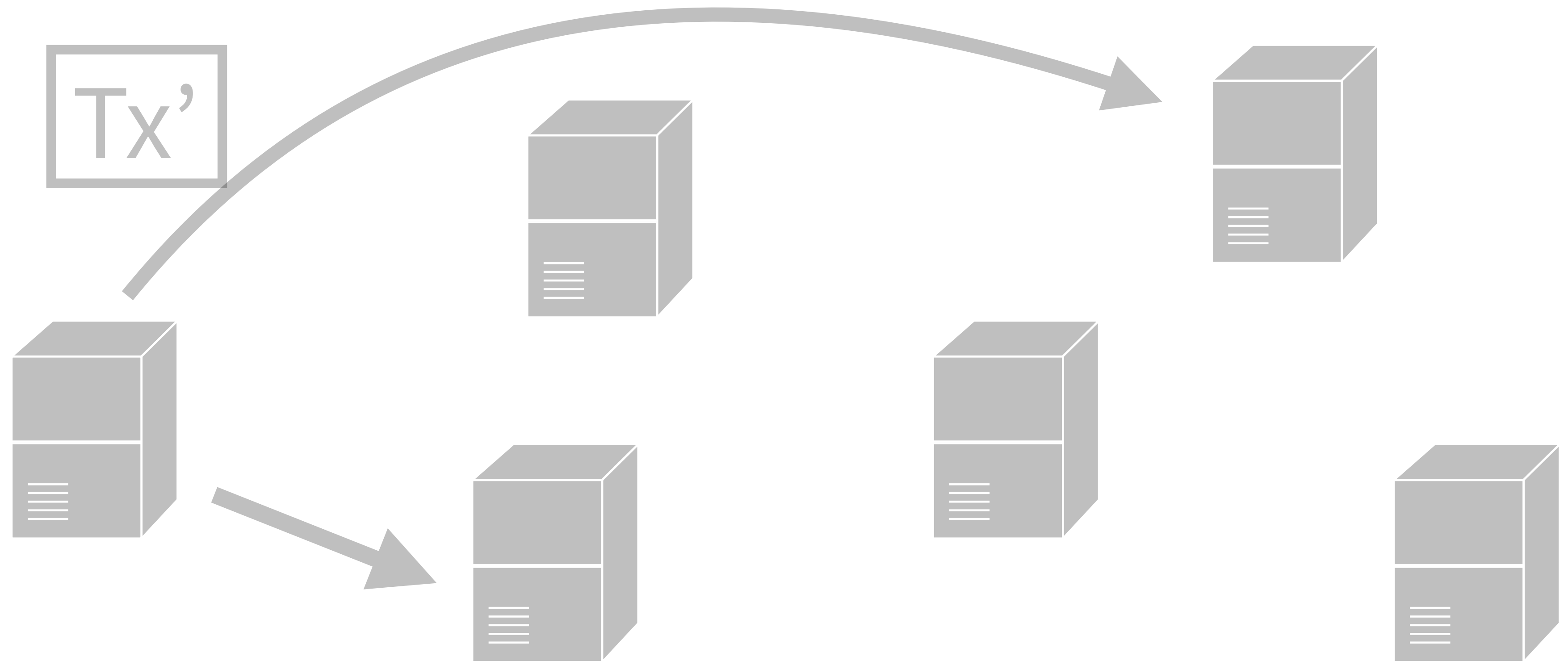
Tx is broadcasted and propagated to all nodes

3



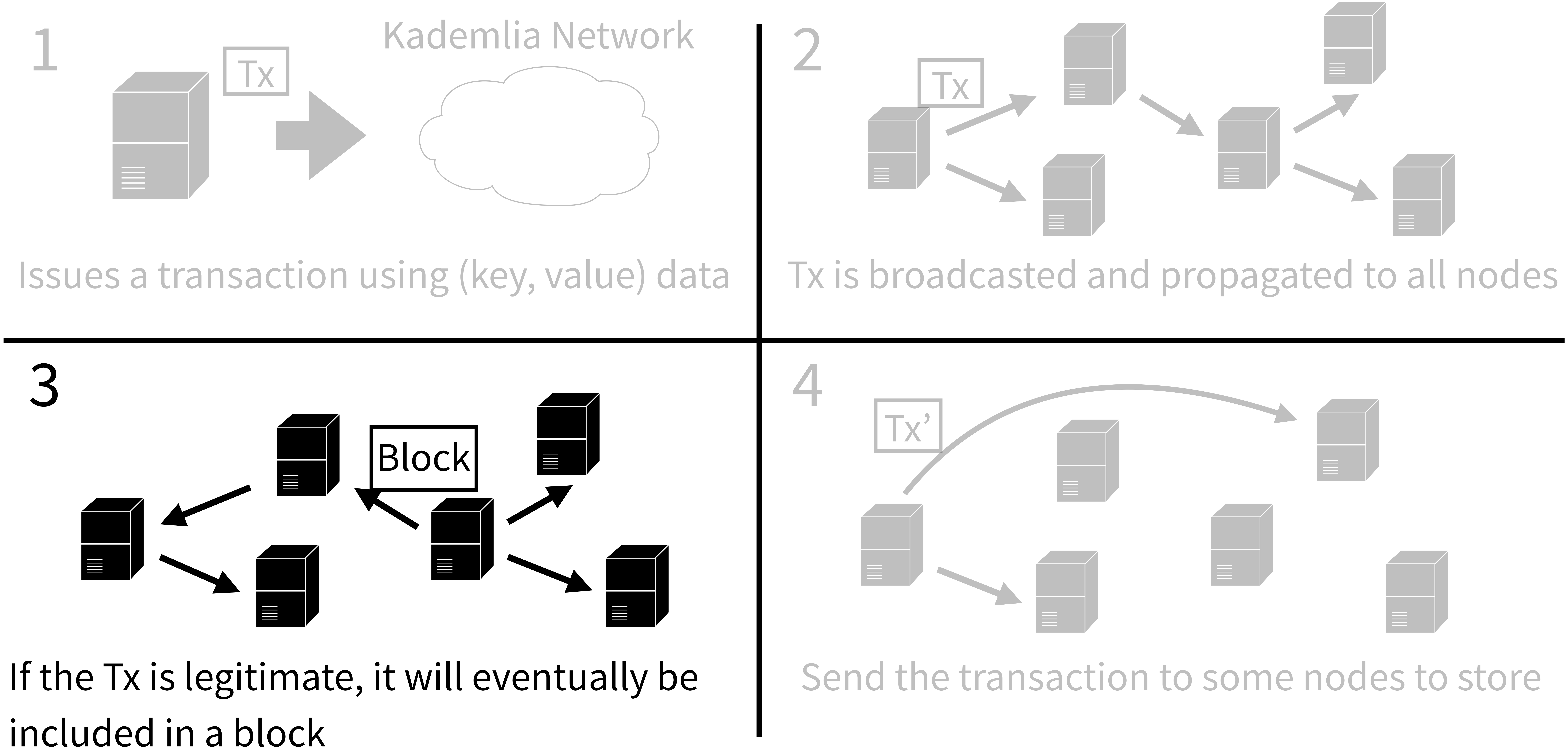
If the Tx is legitimate, it will eventually be included in a block

4



Send the transaction to some nodes to store

Store and retrieve for data



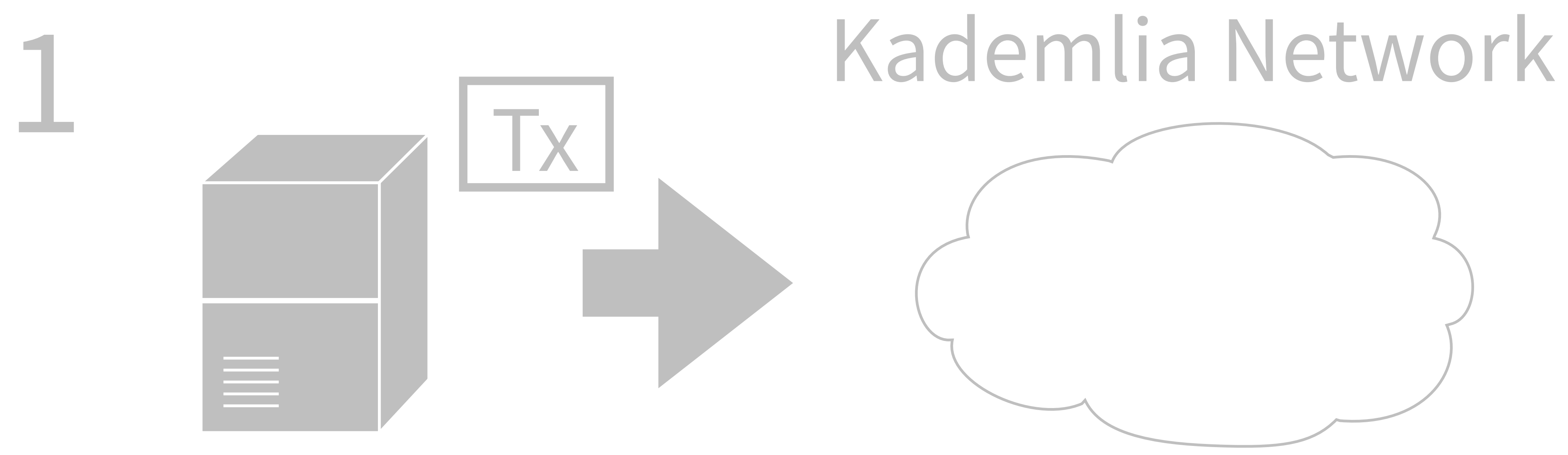
1 Issues a transaction using (key, value) data

2 Tx is broadcasted and propagated to all nodes

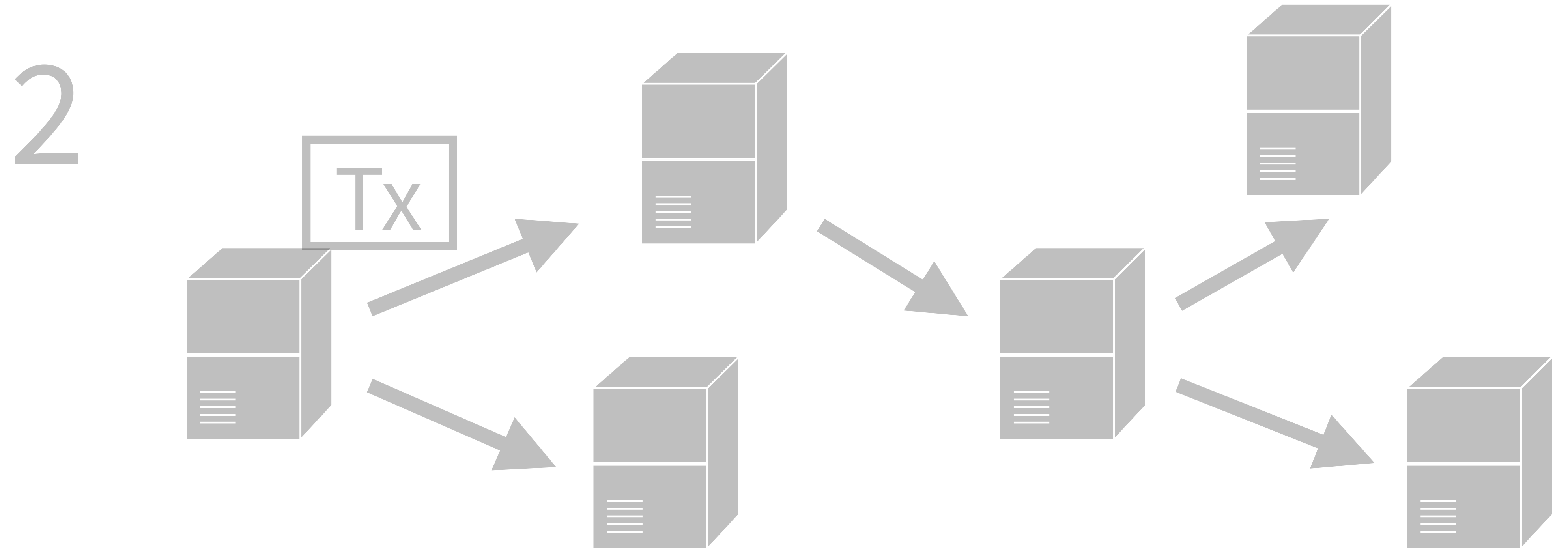
3 If the Tx is legitimate, it will eventually be included in a block

4 Send the transaction to some nodes to store

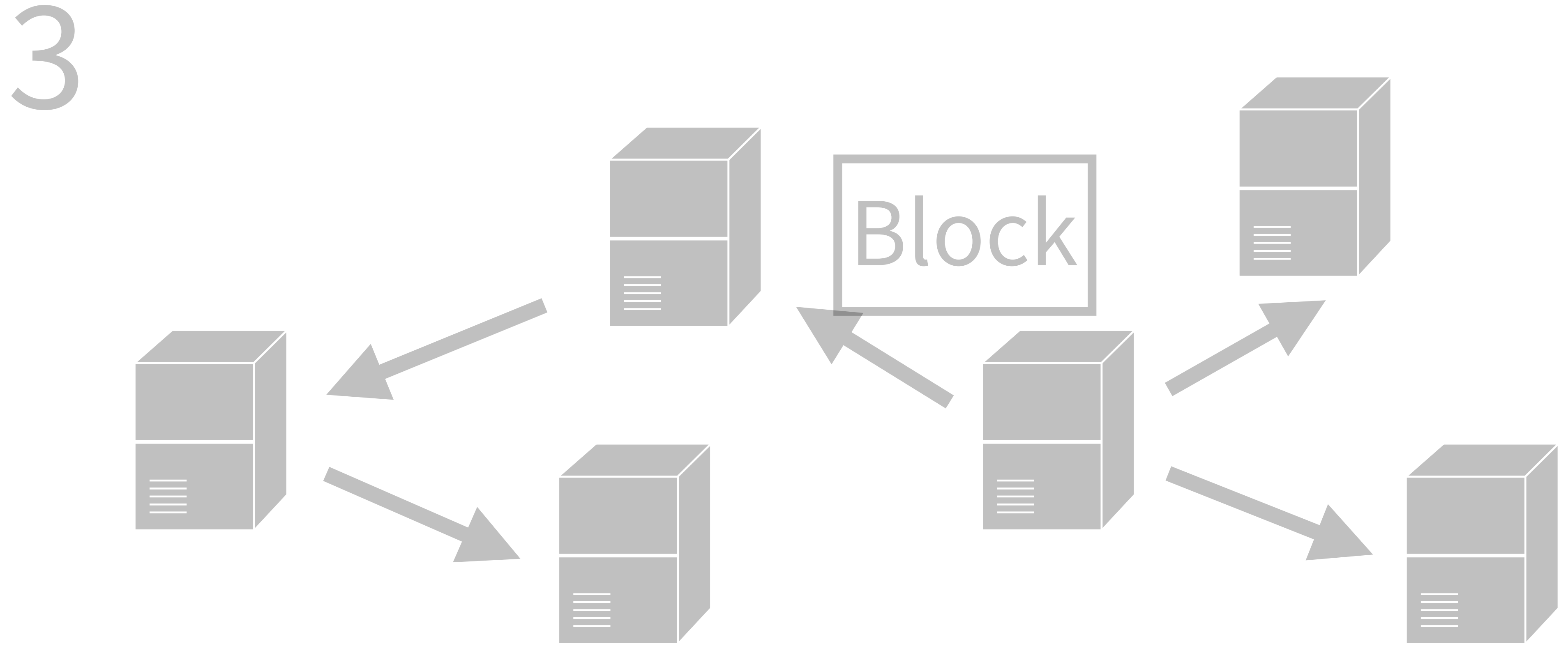
Store and retrieve for data



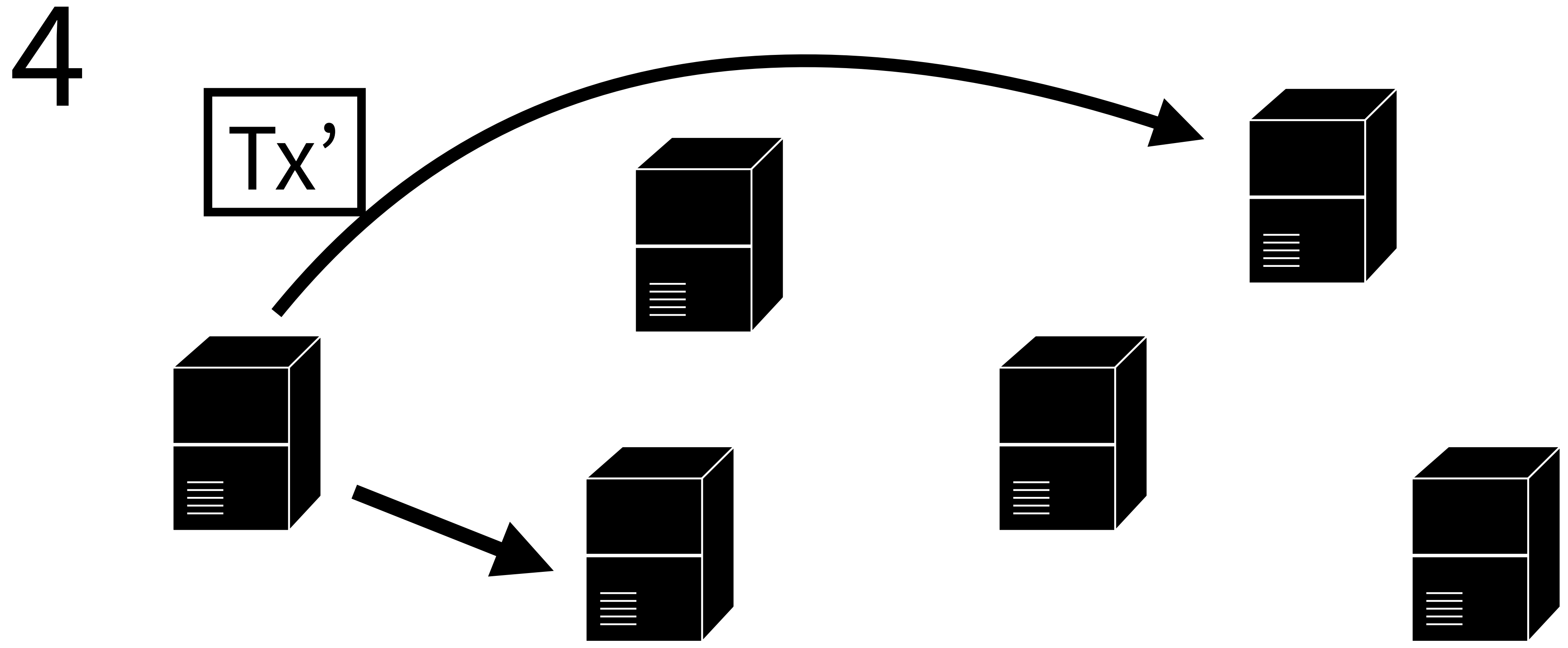
Issues a transaction using (key, value) data



Tx is broadcasted and propagated to all nodes

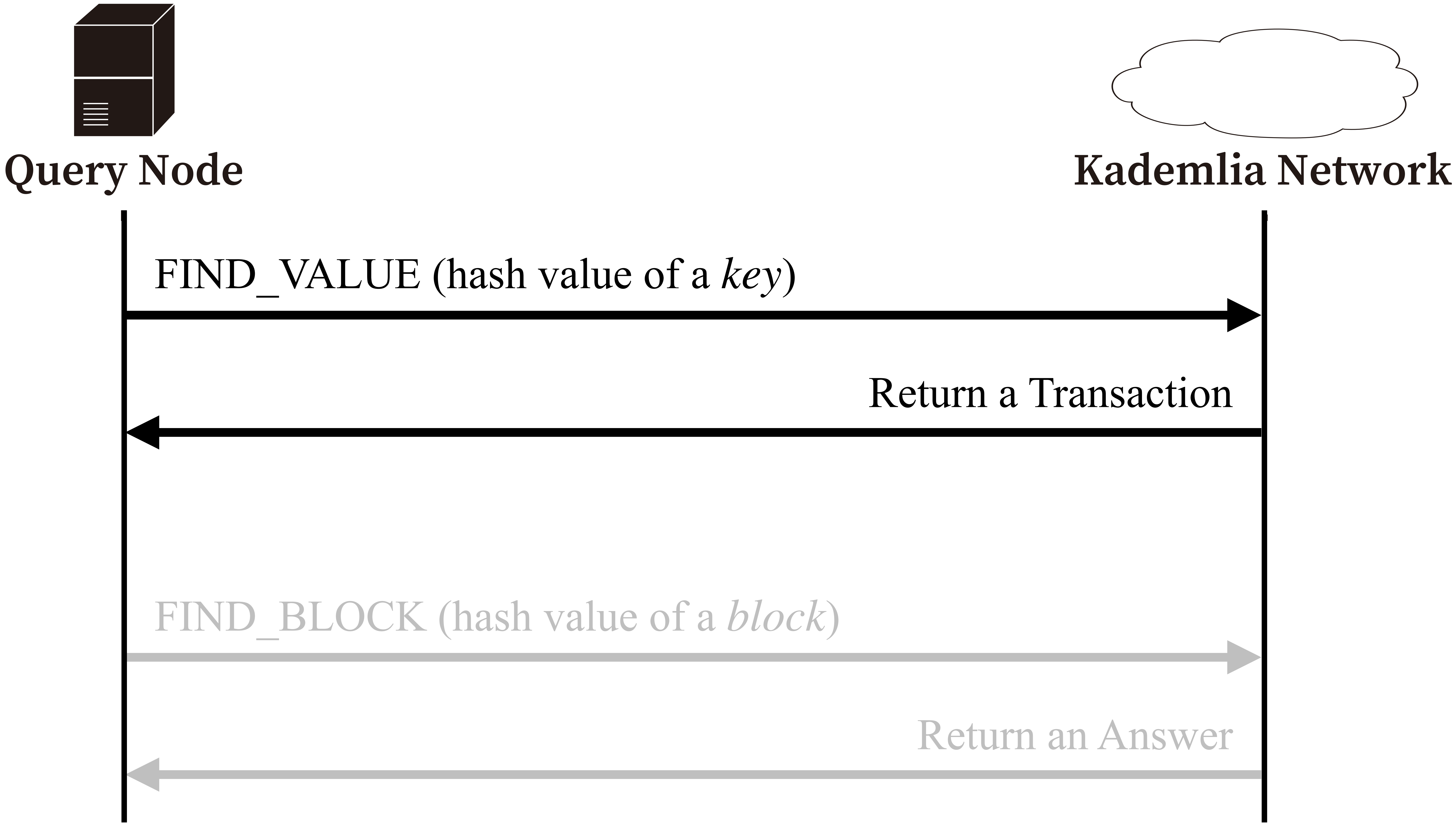


If the Tx is legitimate, it will eventually be included in a block

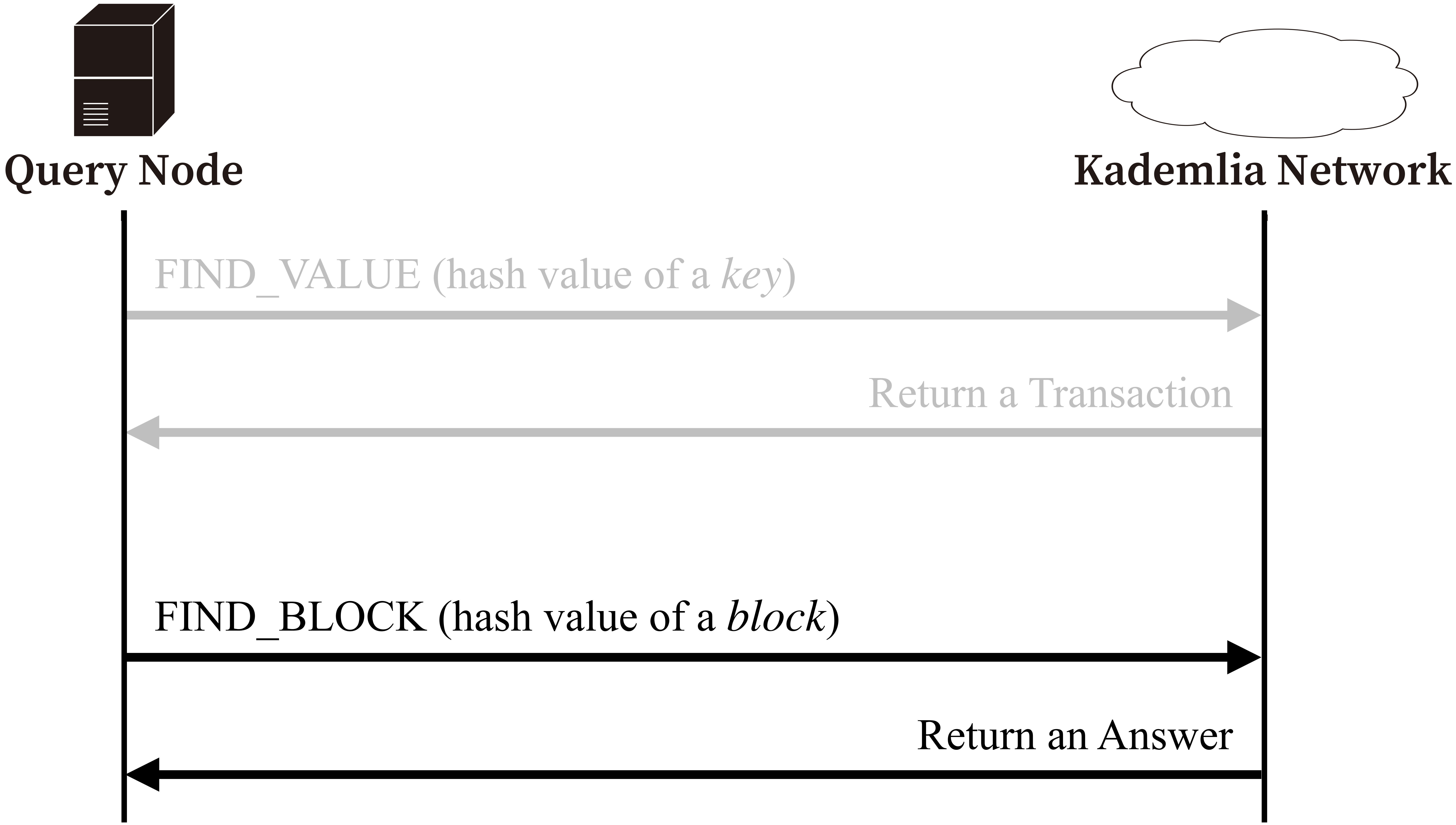


Send the transaction to some nodes to store

Store and retrieve for data


Query Node
Kademlia Network
FIND_VALUE (hash value of a *key*)
Return a Transaction
FIND_BLOCK (hash value of a *block*)
Return an Answer

Store and retrieve for data



Query Node

Kademlia Network

FIND_VALUE (hash value of a *key*)

Return a Transaction

FIND_BLOCK (hash value of a *block*)

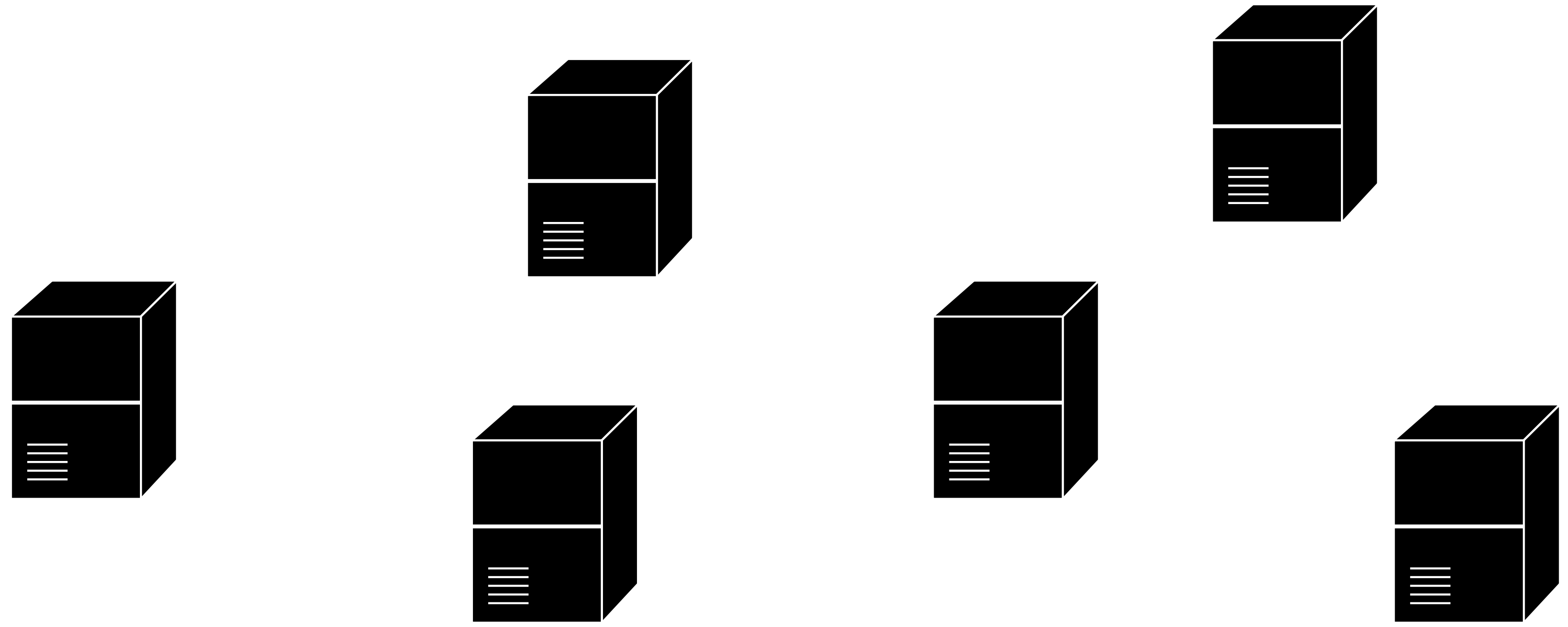
Return an Answer

Simulation

- We measured the query time to retrieve any data and success rate (while changing two parameters)
  - the number of nodes in the network
  - the ratio of off-line nodes

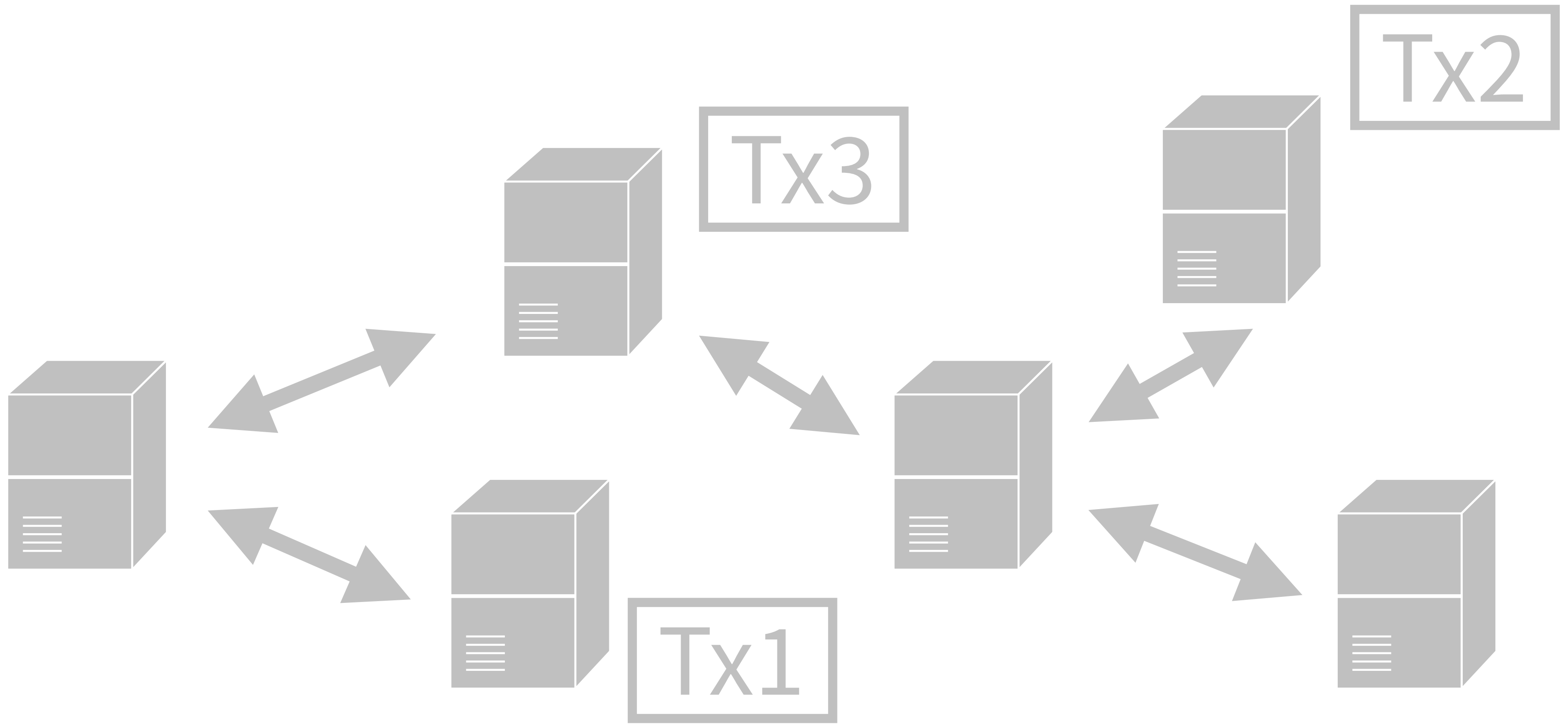
Simulation

1



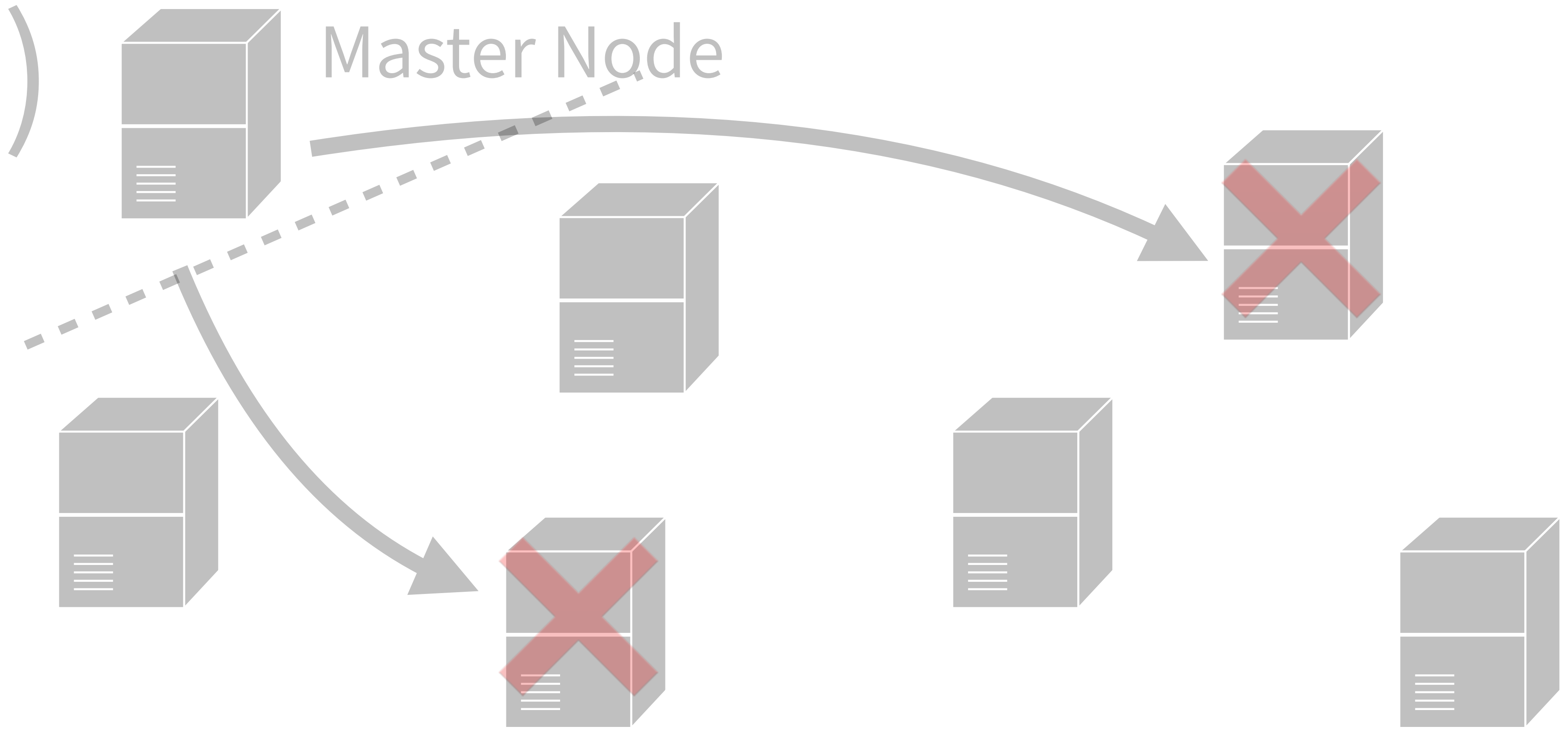
Prepare some nodes in a virtual network

2



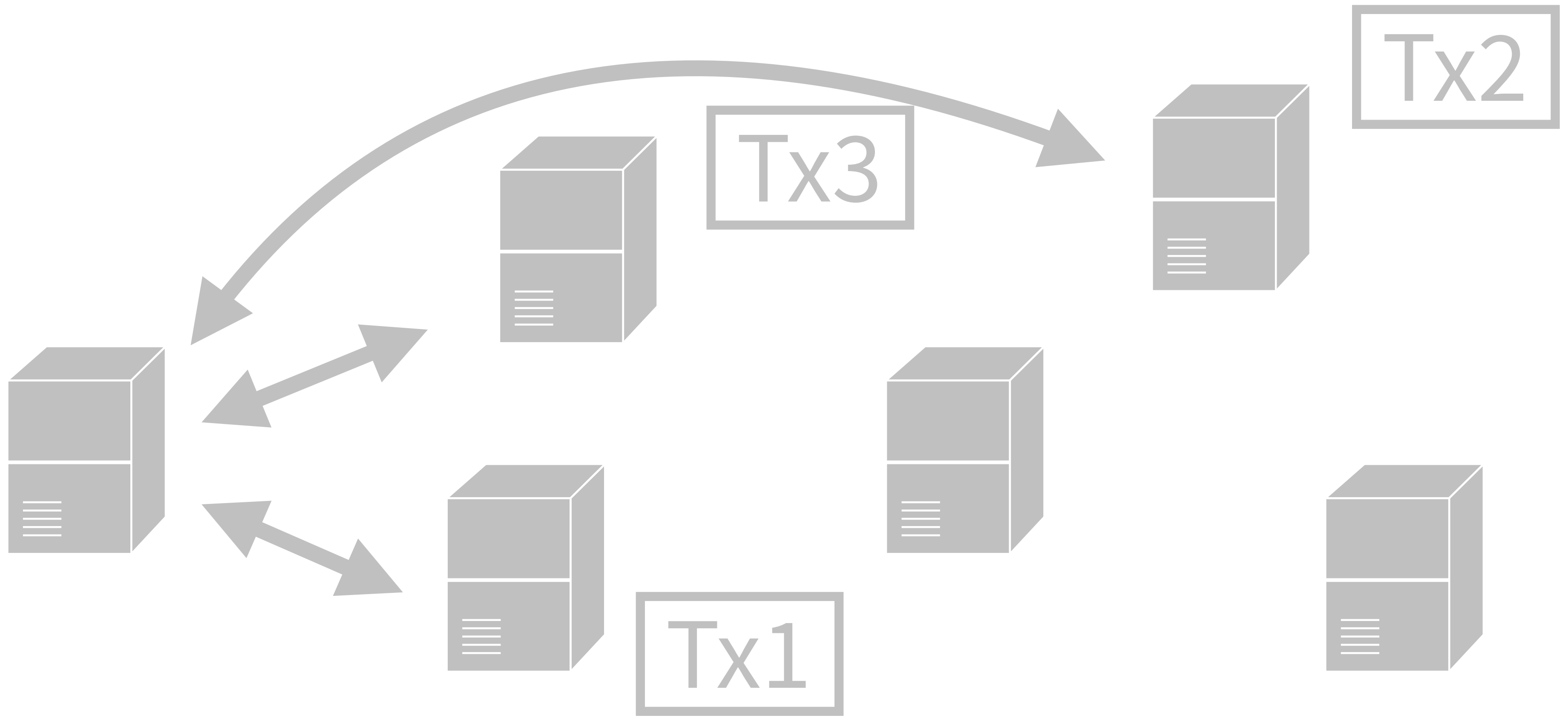
10 Txs are issued by nodes, and 1 block is mined

(3)



Send a message to nodes to be off-line

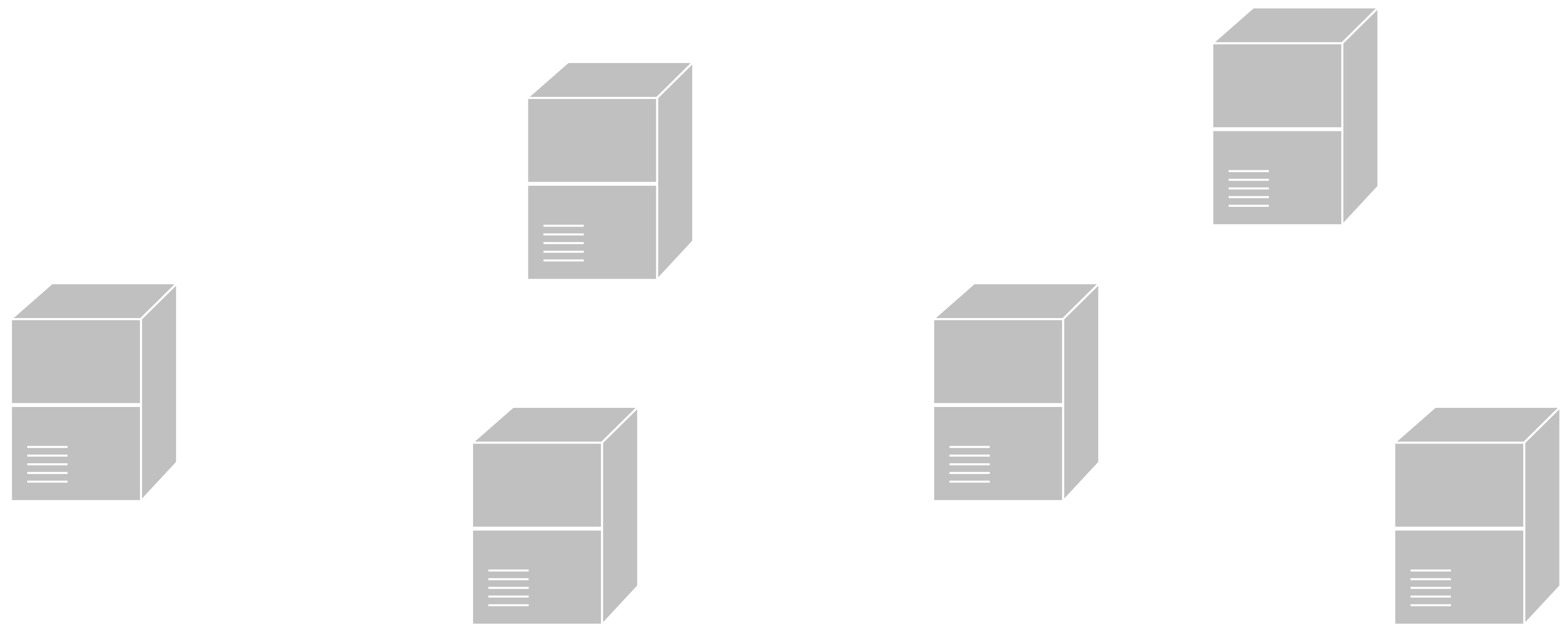
4



Execute a retrieval query and measure the query time

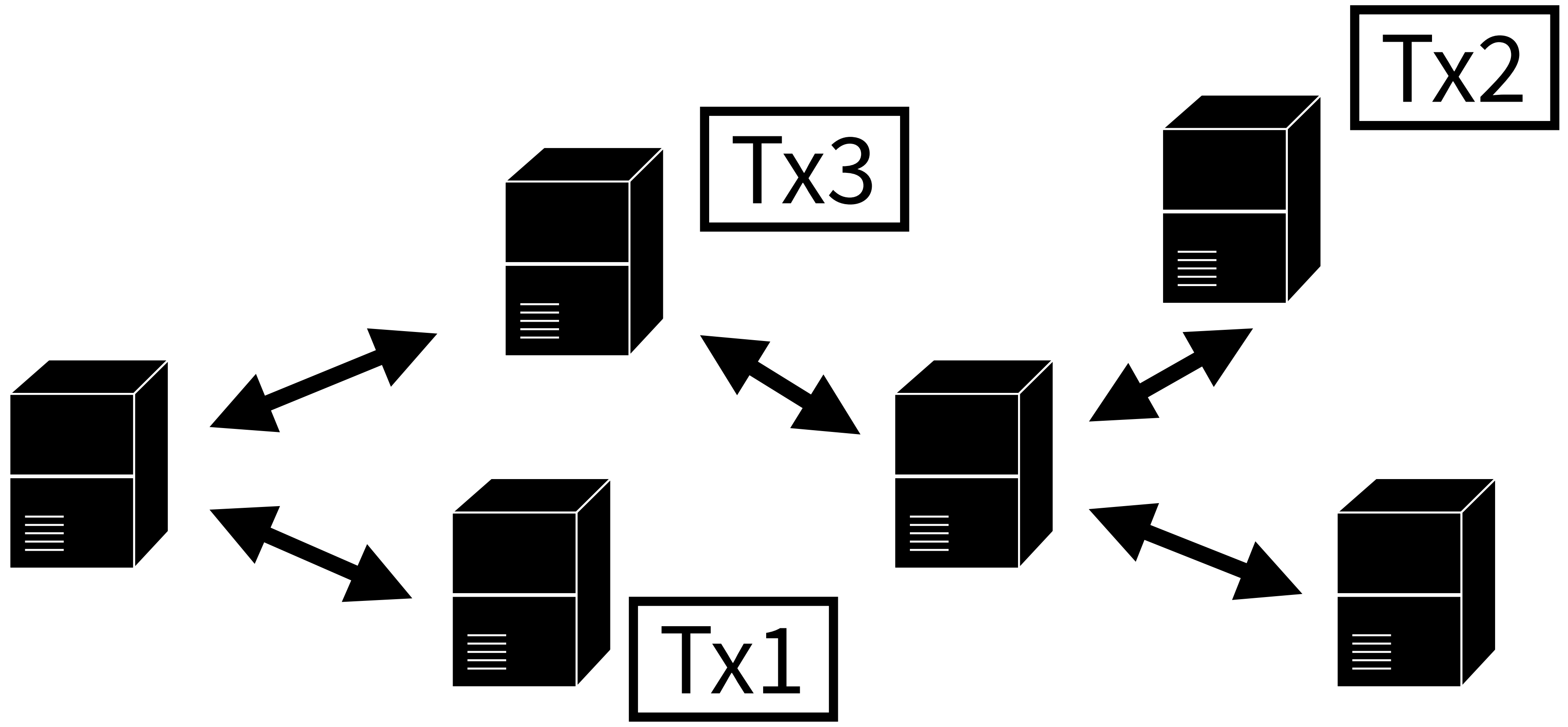
Simulation

1



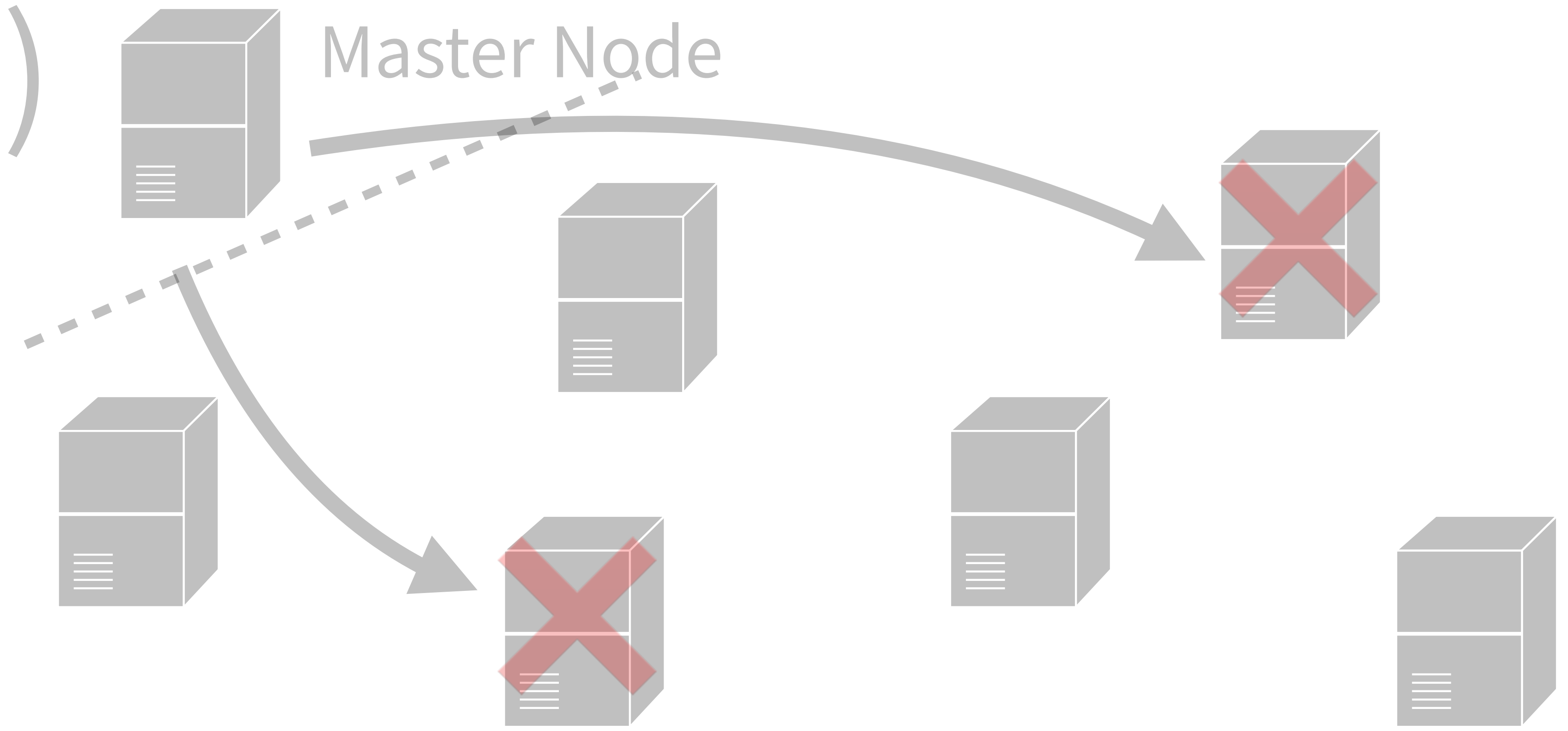
Prepare some nodes in a virtual network

2



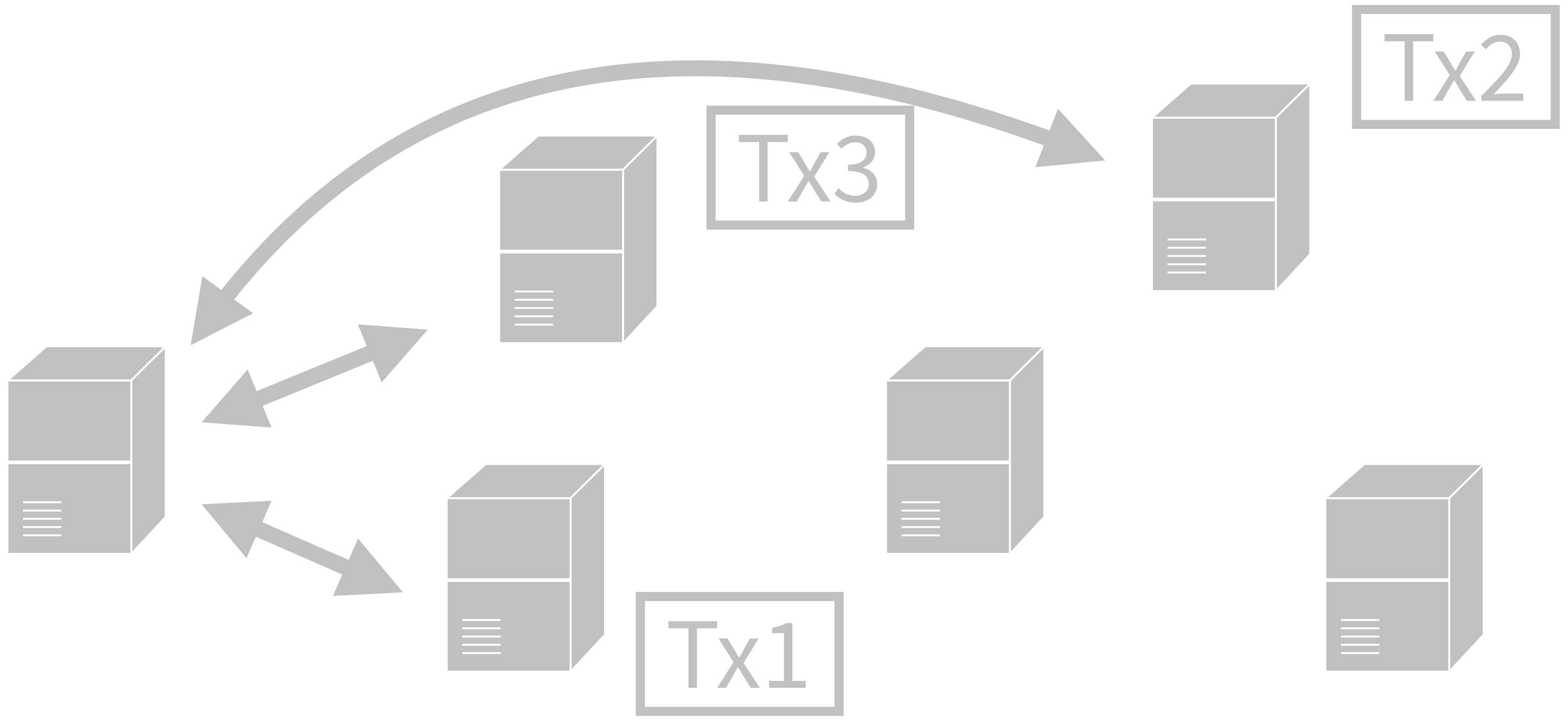
10 Txs are issued by nodes, and 1 block is mined

(3)



Send a message to nodes to be off-line

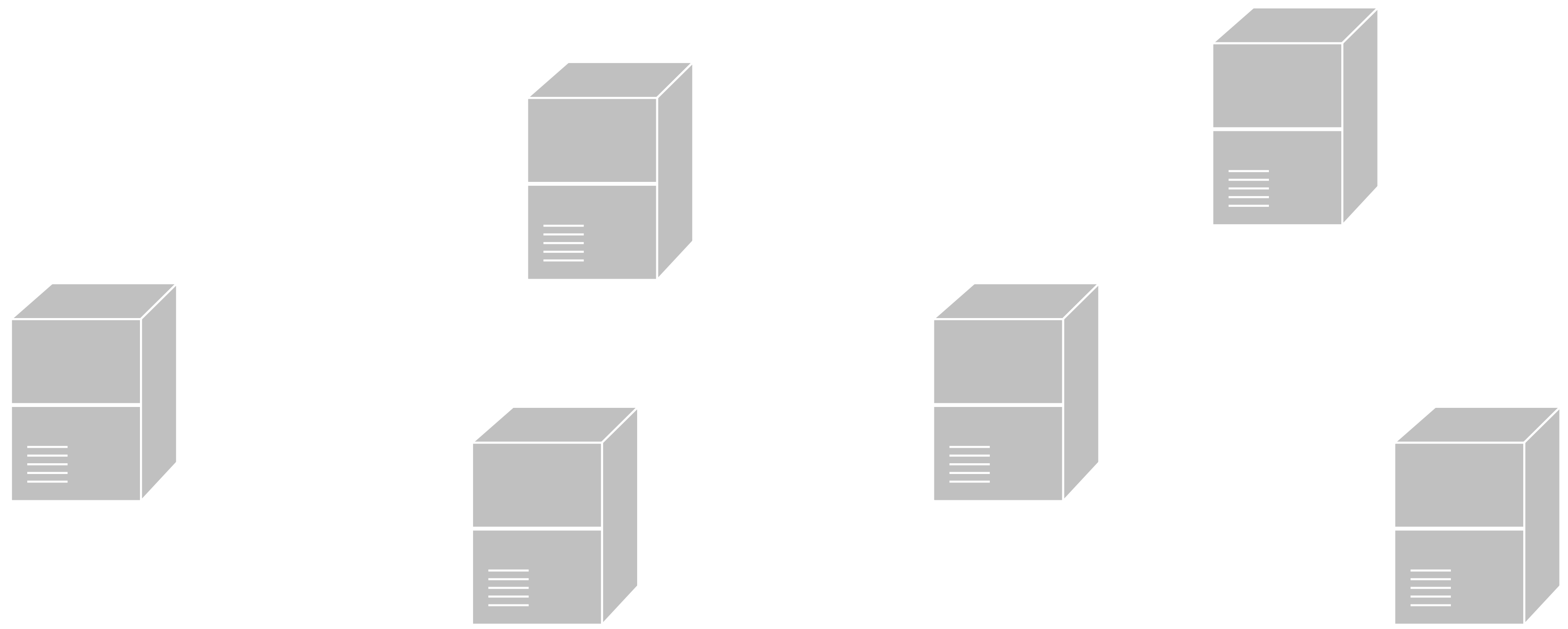
4



Execute a retrieval query and measure the query time

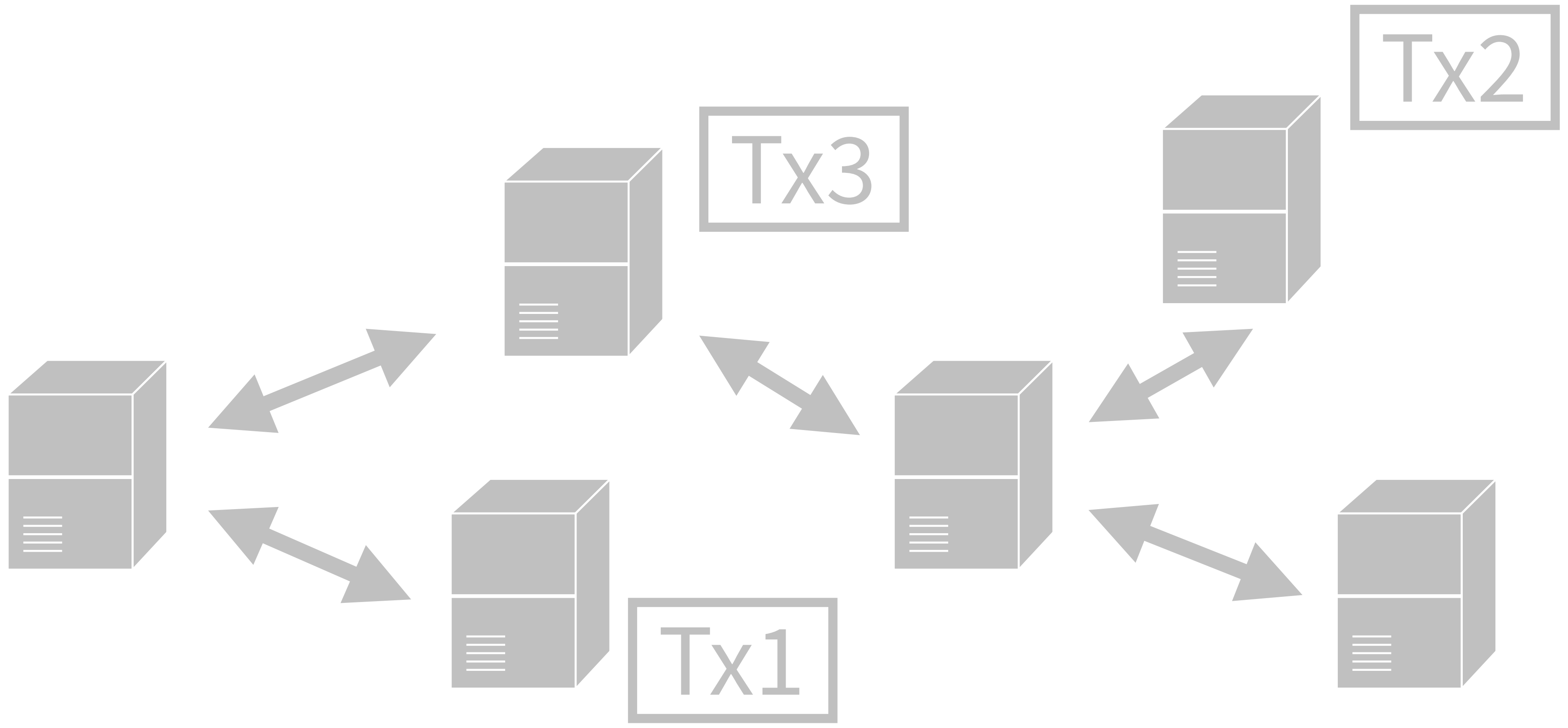
Simulation

1



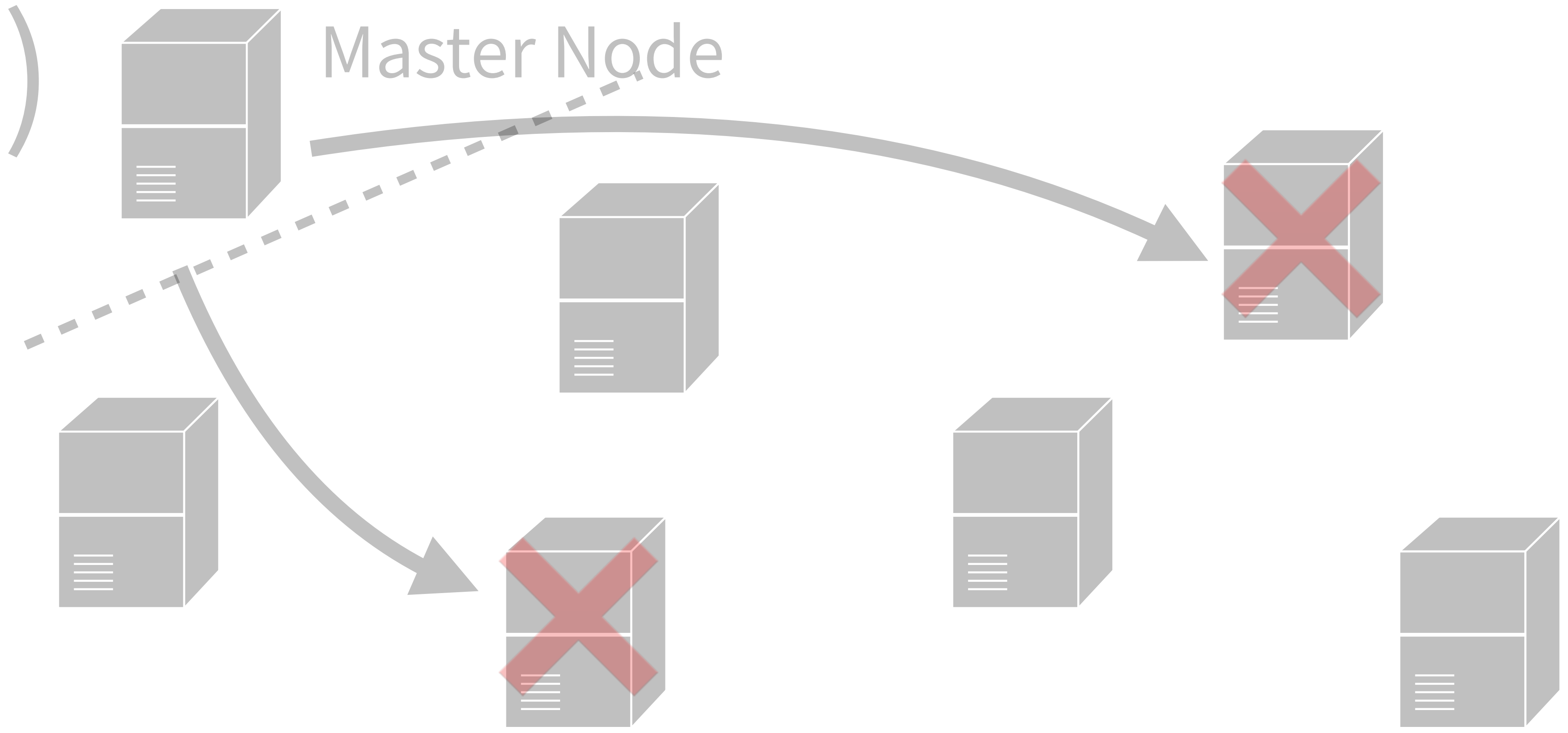
Prepare some nodes in a virtual network

2



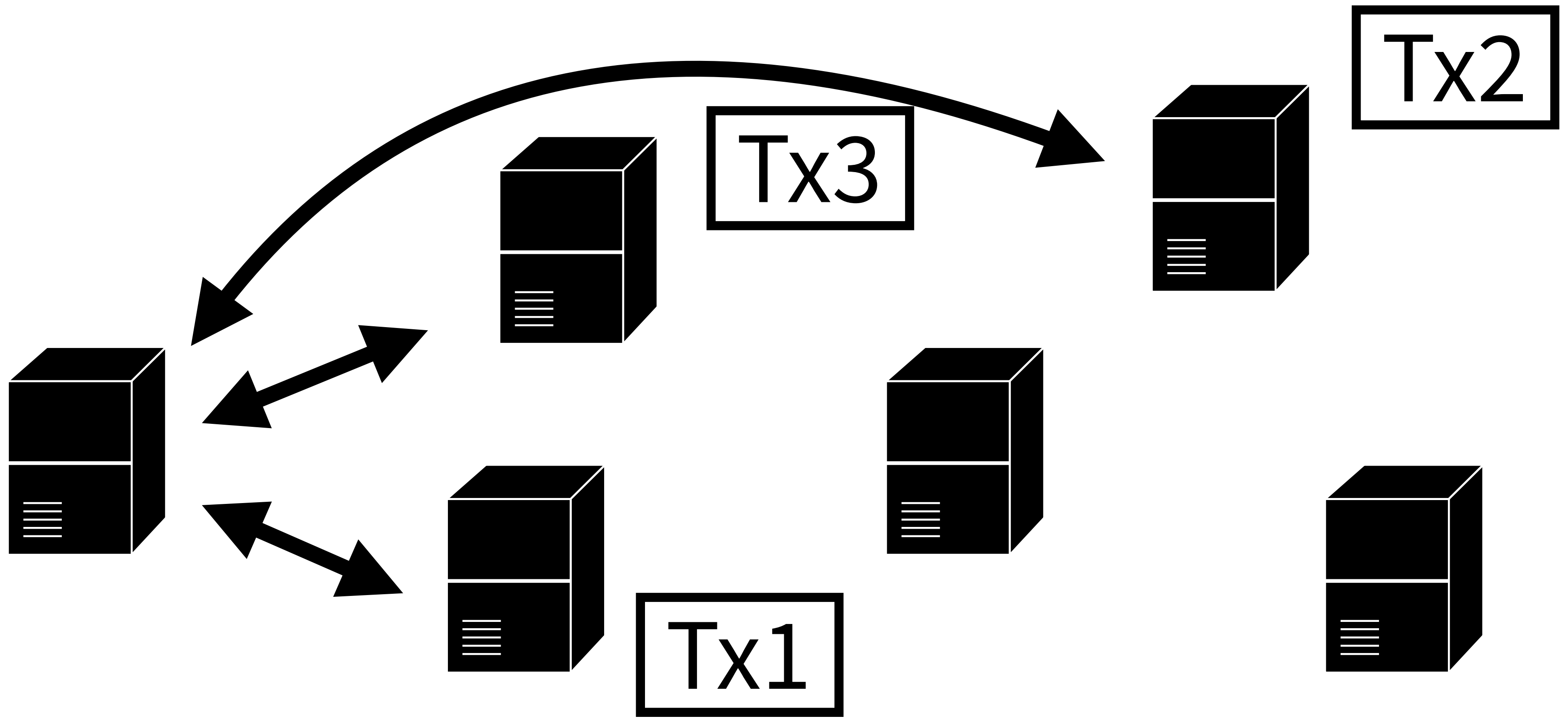
10 Txs are issued by nodes, and 1 block is mined

(3)



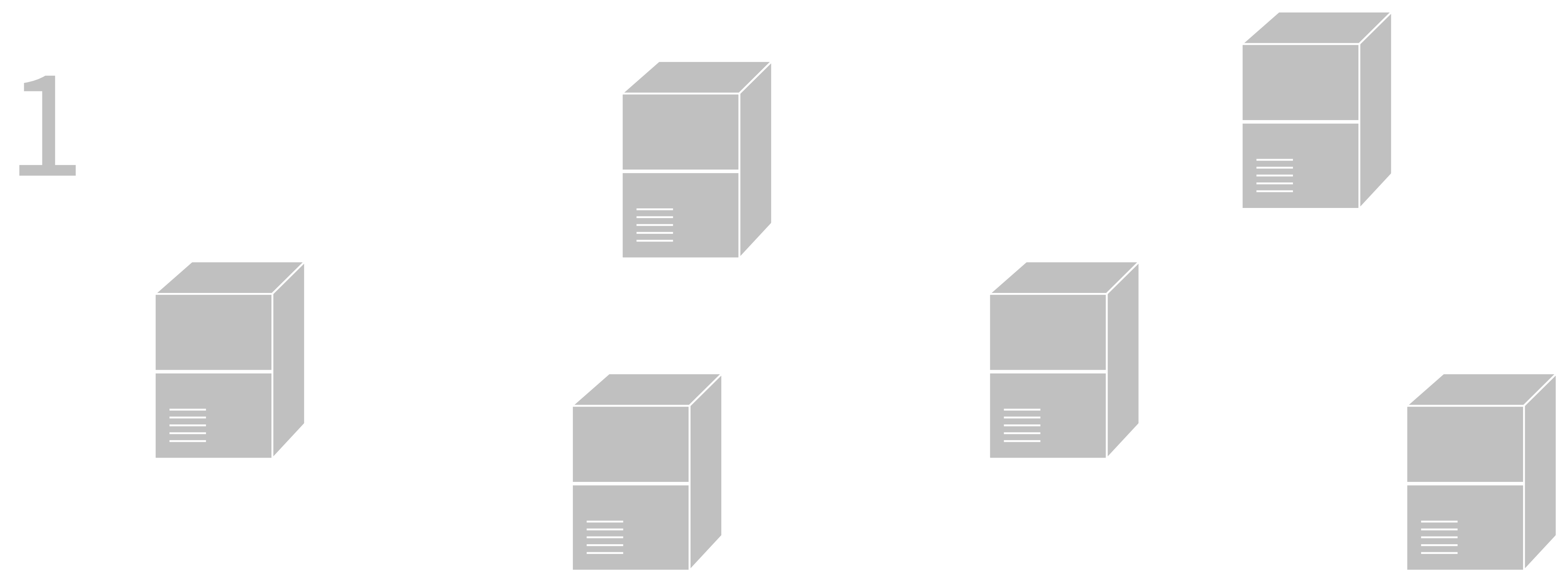
Send a message to nodes to be off-line

4

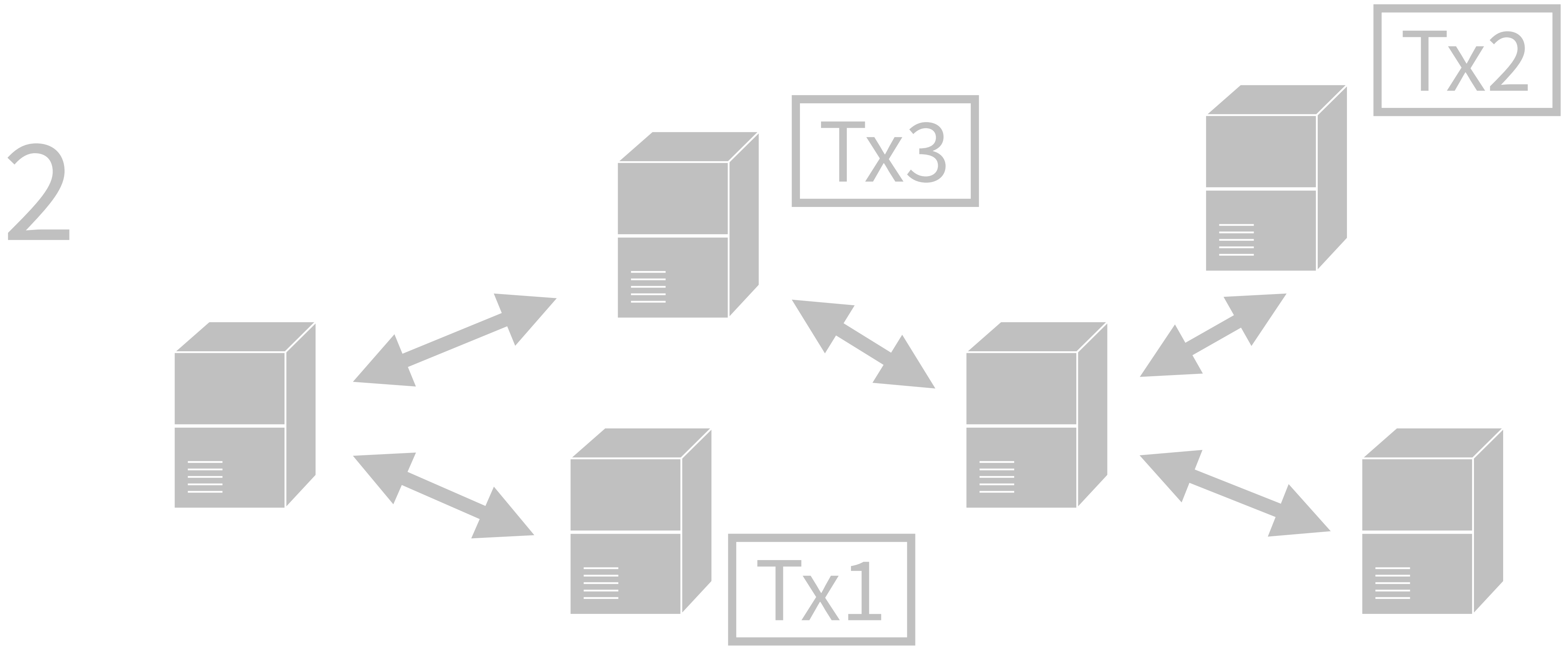


Execute a retrieval query and measure the query time

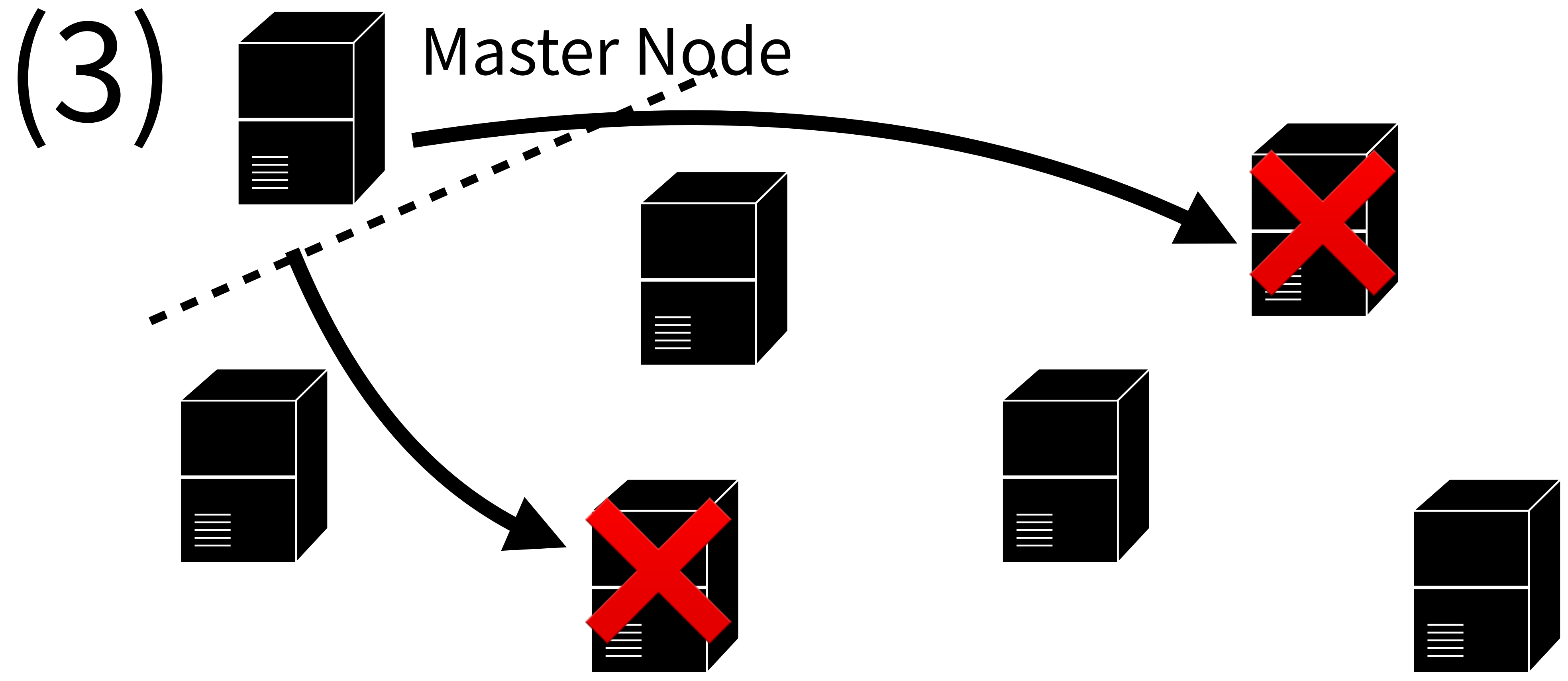
Simulation



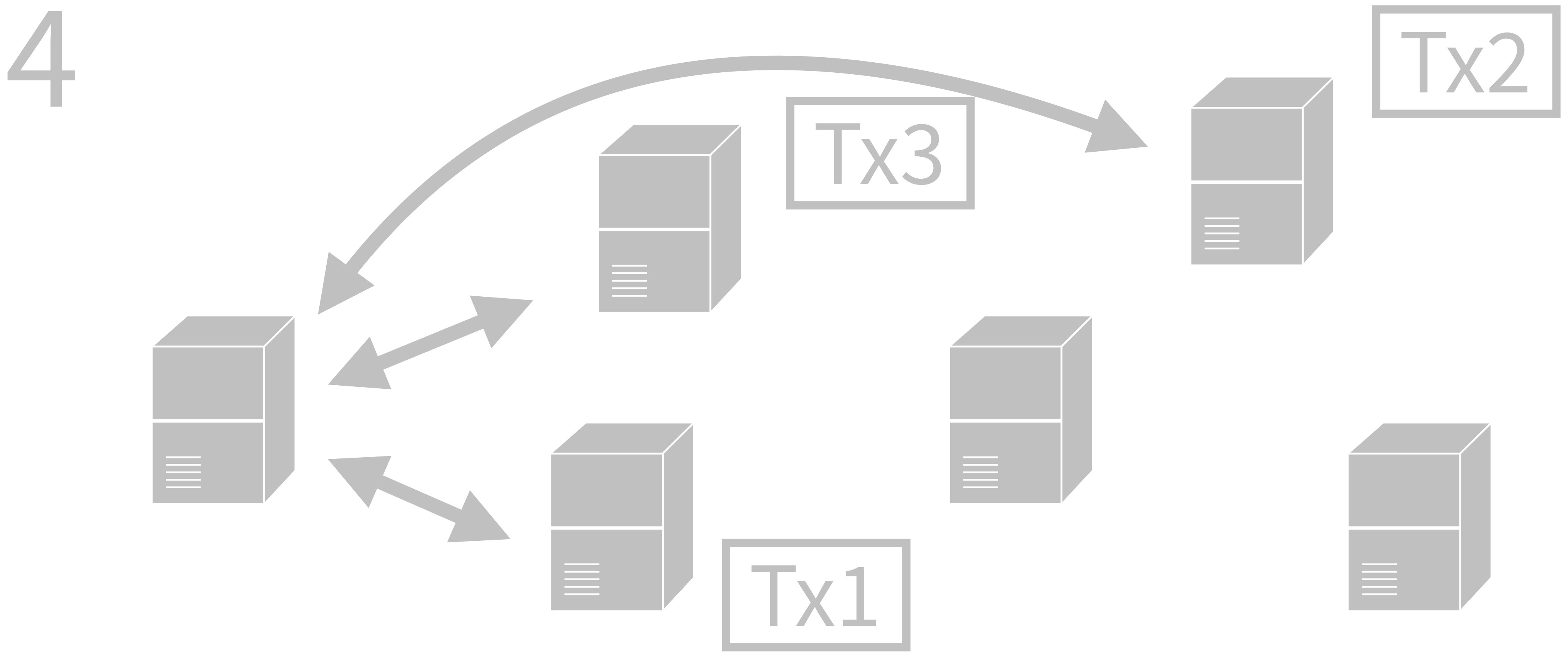
Prepare some nodes in a virtual network



10 Txs are issued by nodes, and 1 block is mined



Send a message to nodes to be off-line

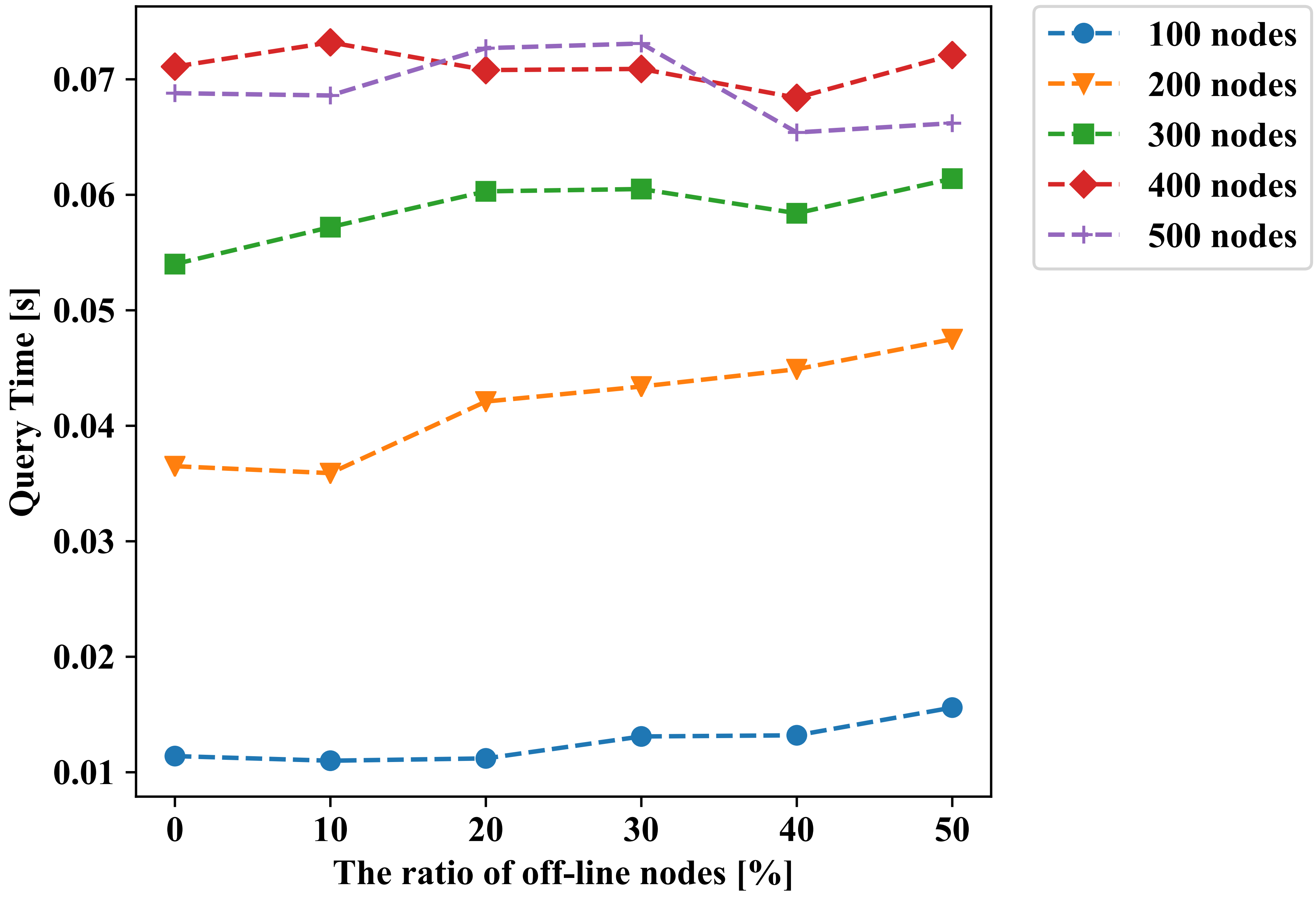


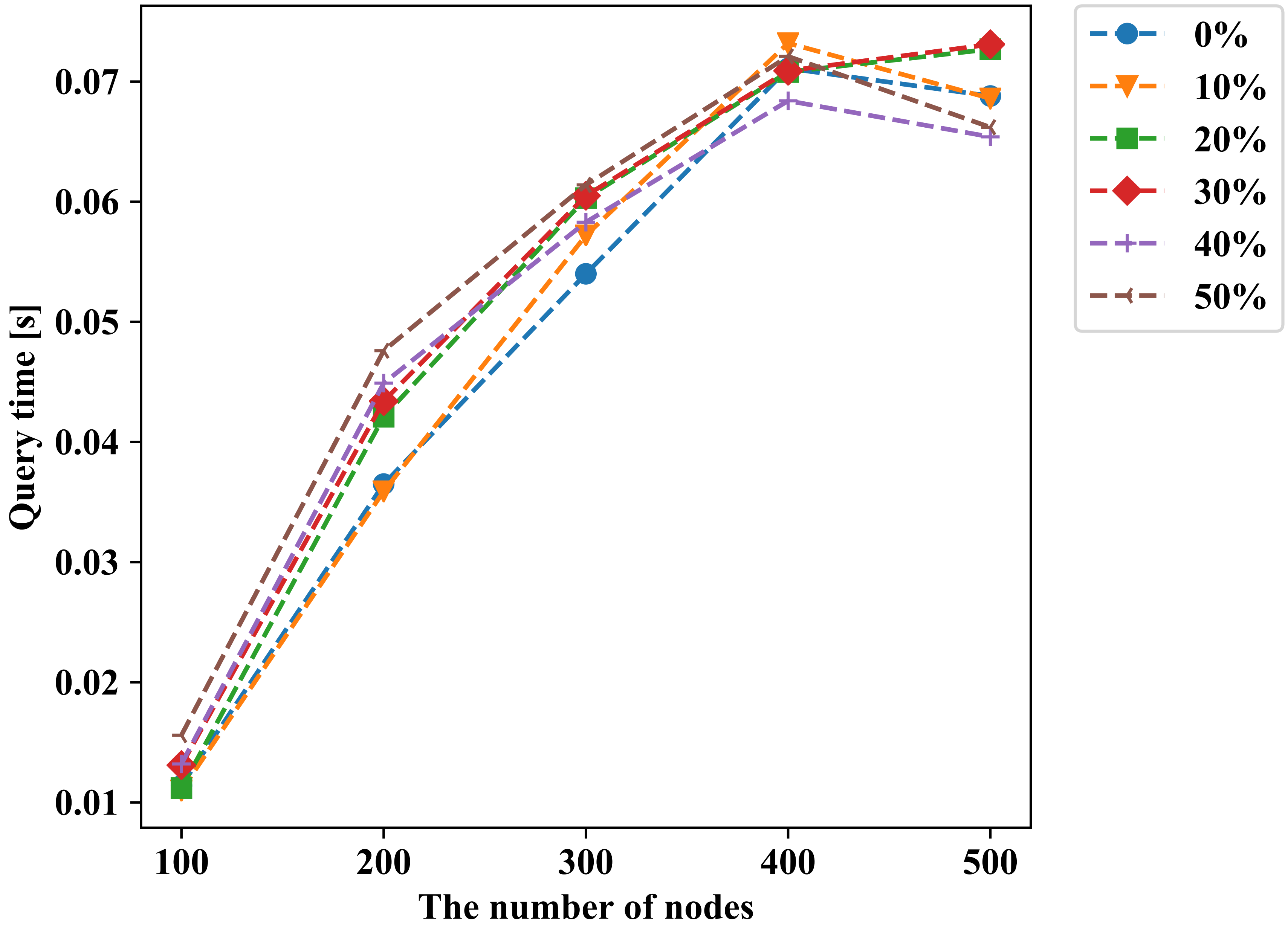
Execute a retrieval query and measure the query time

Query Time

- The query time increases as the ratio of off-line nodes
  - 100 nodes : the query time did not change significantly
  - 200 nodes : 3 to 4 times as long as 100 nodes
  - 300, 400 nodes : took more query time
  - 500 nodes : not much longer than with 400 nodes

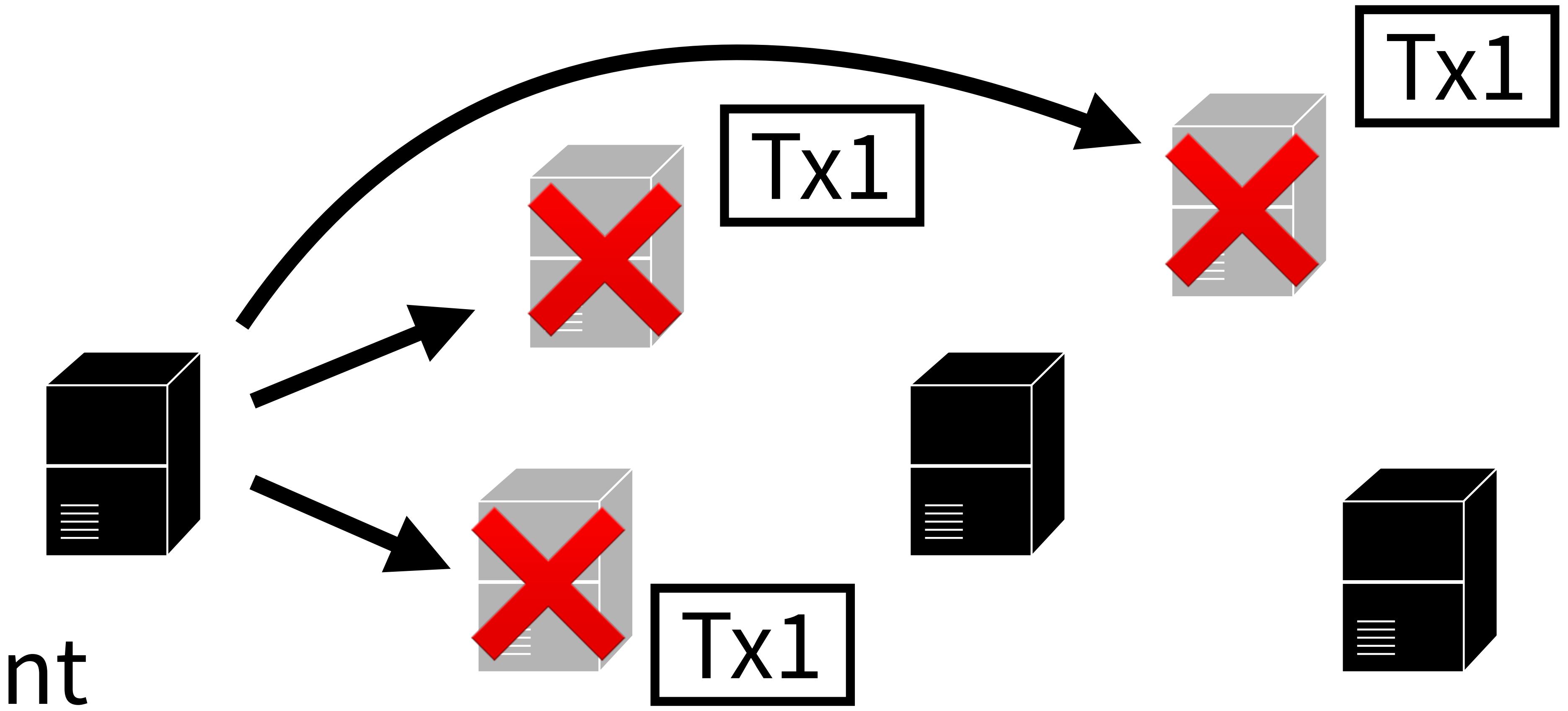
- The change in query time as the number of nodes increases





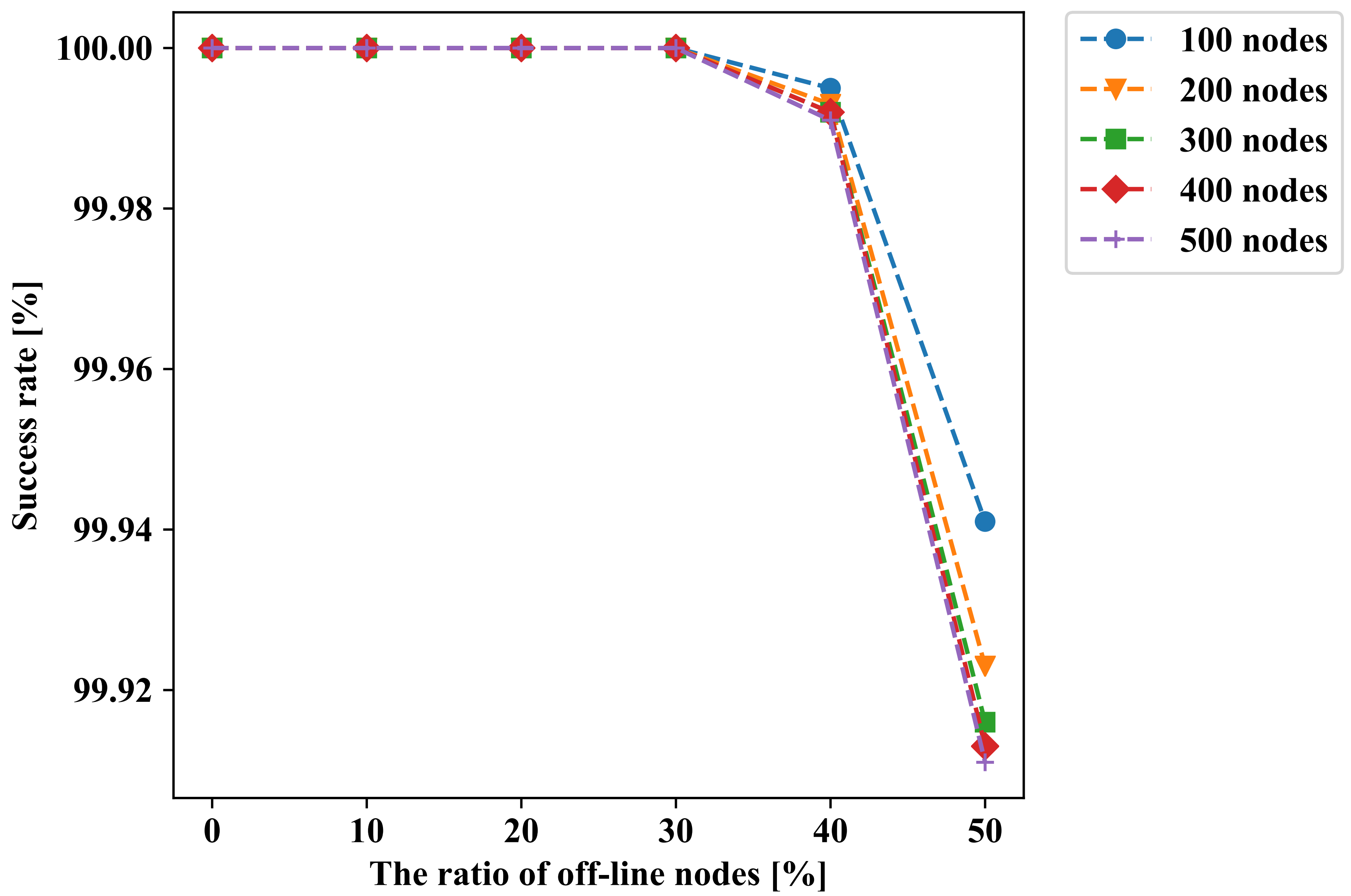
Success Rate

- All nodes storing specific content are off-line
  - This probability can be calculated from the number of off-line nodes
- $P_1 = \frac{{}_bP_c}{{}_aP_c}$
  - a : the number of total nodes
  - b : the number of off-line nodes
  - c : the number of nodes storing the same content
- Success rate is $P_2 = 1 - \frac{{}_bP_c}{{}_aP_c}$

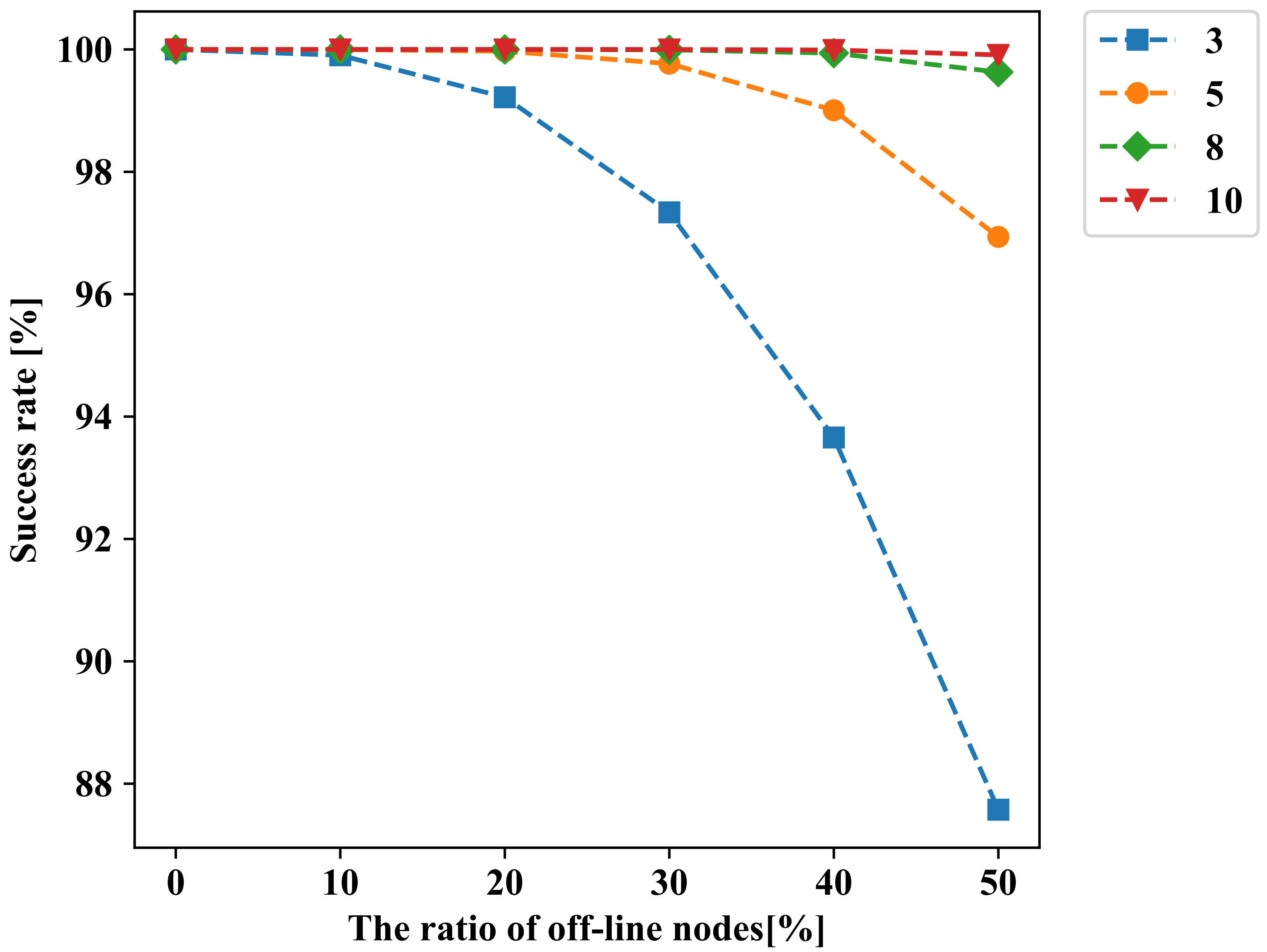


Success Rate

- The result of substituting some values into P_2 (fixed c to 10)
  - P_2 decreases when the ratio of off-line nodes exceeds 30%
  - the probability of failure for the search was less 1%



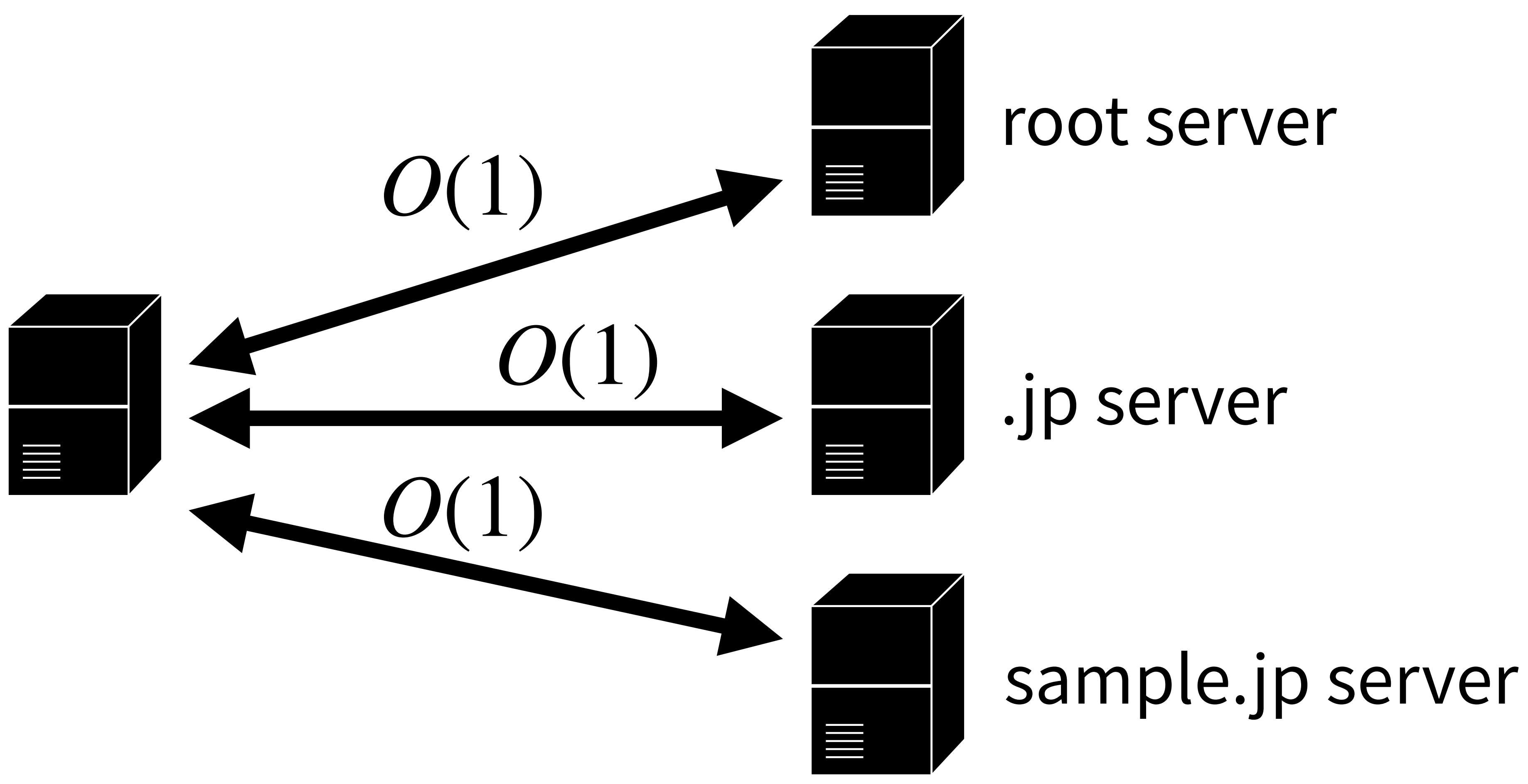
- The change in success rate (fixed a to 500)
  - if c is small, P_2 will be significantly reduced
  - if c is 10, sufficient search availability can be maintained



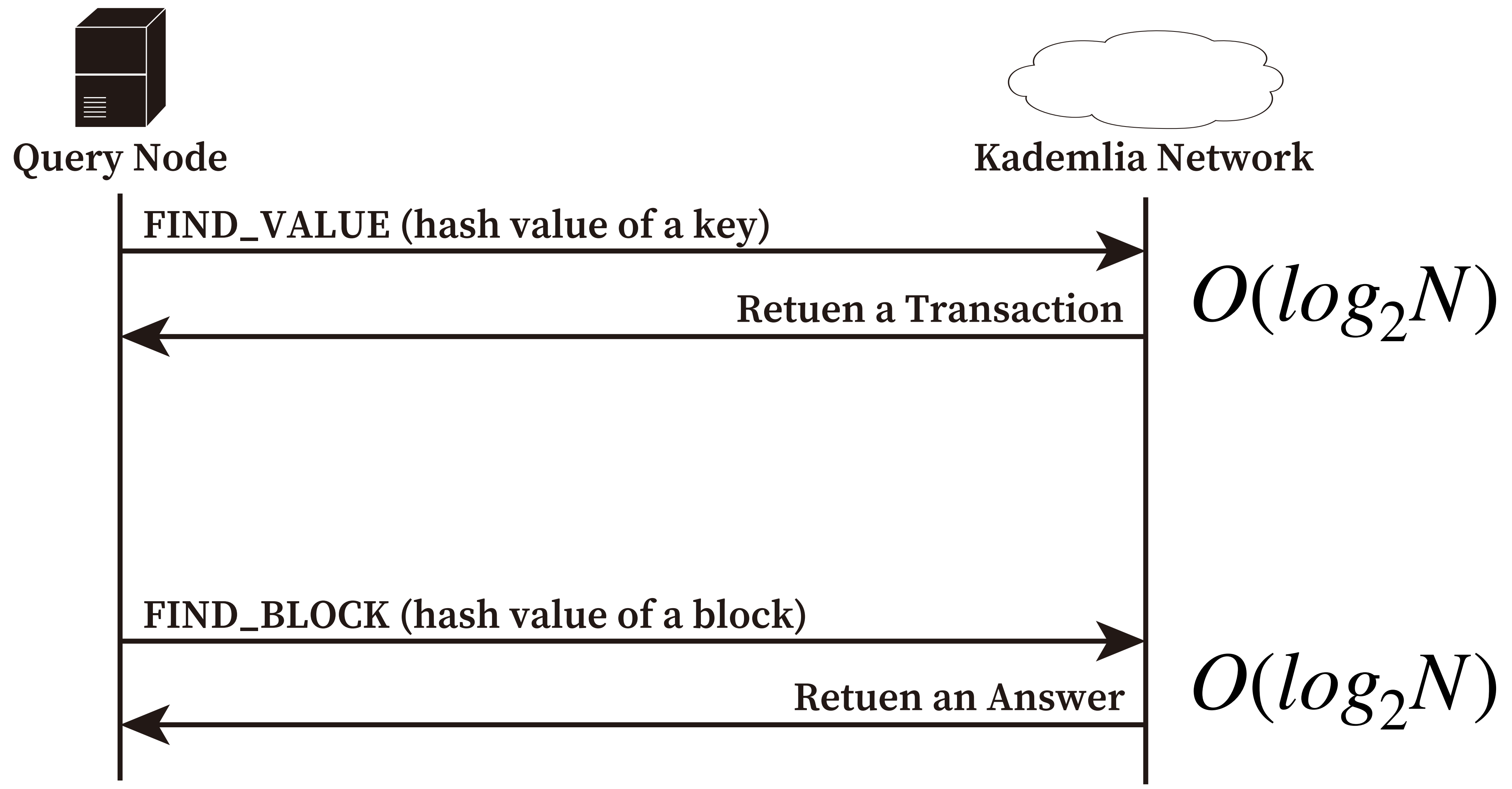
Comparison of our proposed system and DNS

- DNS has a mechanism to distribute administrators hierarchically
  - n-level domain name : $O(n)$
  - our proposed system : $O(2log_2N)$

    the expected value of each lookup decreases as the number of nodes storing the same data increases



DNS



Our proposed system

Conclusion

- In this paper, we described our proposed lookup system using DHT and blockchain

- Reported the result of measuring query time
  - the query time increases with the number of nodes
  - the success rate of any retrieval is almost 100% in the environment with no malicious nodes

Introduction

- DNS
  - domain name : a name given to resources on the network
  - name resolving : to find numbered address (IP address) corresponding to the domain name
  - none of several alternative to DNS are widespread
- DNSSEC (DNS Security Extensions)
  - guarantees data integrity
  - complex and requires many action from multiple parties
- This paper…
  - we propose a lookup system using “blockchain” and “DHT”
  - our goal is this system will be an alternative to DNS

Blockchain

- Bitcoin
  - allows online payments through a (P2P) network without a trusted third party
  - has a distributed ledger system to share all transactions -> blockchain
  - guarantees integrity
- Consensus algorithm
  - to judge which block is valid
  - PoW : finding a value called “nonce”

Blockchain's problem

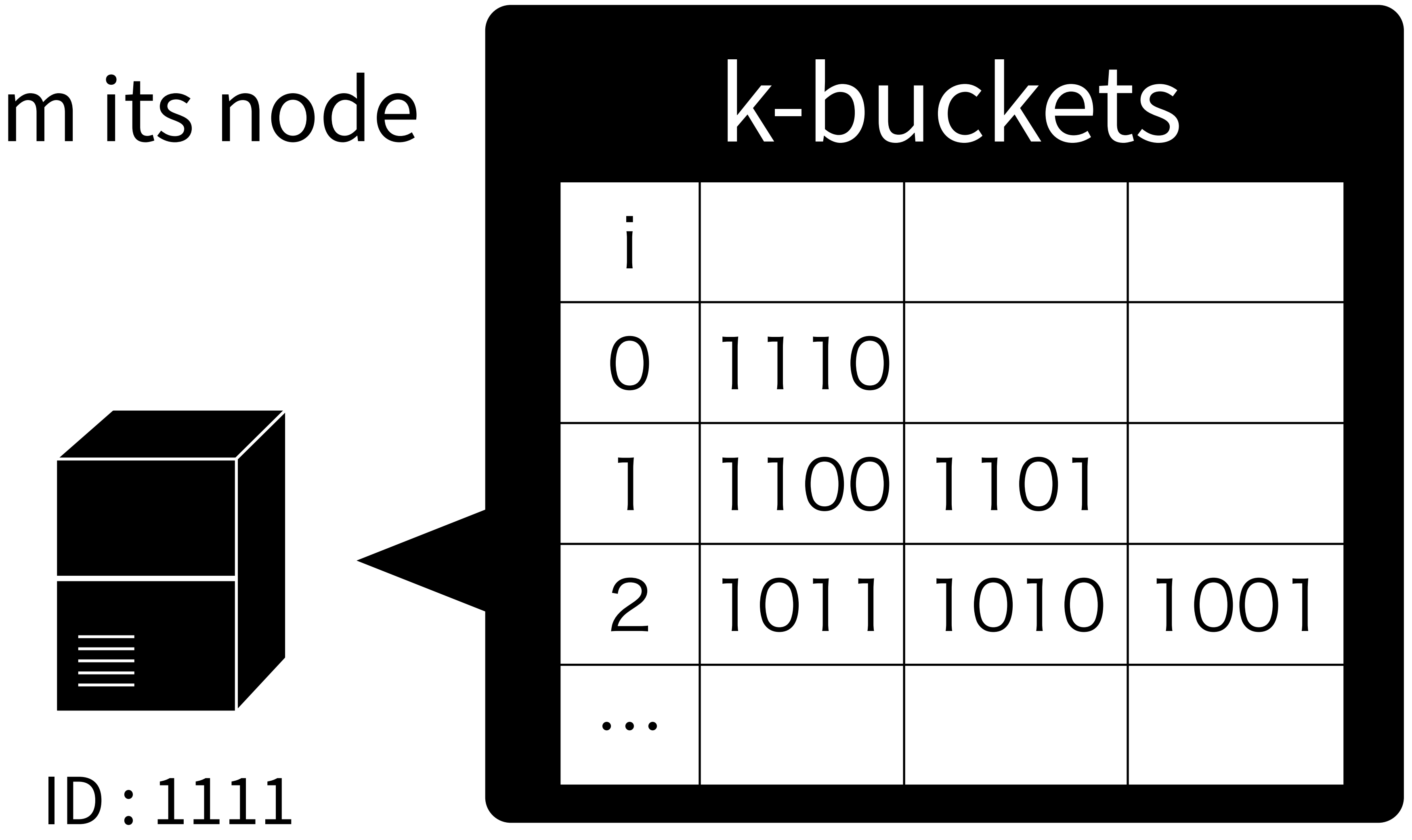
- PoW
  - needs much electric power to find nonce value
  - Bitcoin : 58 TWh / year
- Scalability
  - blockchain is an append-only database, Bitcoin nodes need over 260GB capacity
  - nodes have diversified with the increase in storage capacity

DHT

- Structured overlay network
  - nodes determine which node to link under a mathematical constraint
  - DHT has a scalability of node retrieval
  - Decrease the amount of data that each node holds and have fault tolerance
  - required longer query time to fetch data than DNS
- Hash table and ID space
  - ID space that the hash value of keys can take is divided and assigned to each node in charge
  - queries to other nodes that handle the ID space

Kademlia

- ID space is based on binary tree
- distance between two nodes is defined by an XOR of nodes' ID
    - distance between Node1(1101) and Node2(0001) is 1100 = 12 (1101 xor 0001)
- k-buckets
    - routing table of Kademlia
    - i-th list contains nodes that are $[2^i$ to $2^{i+1})$ away from its node



k-buckets

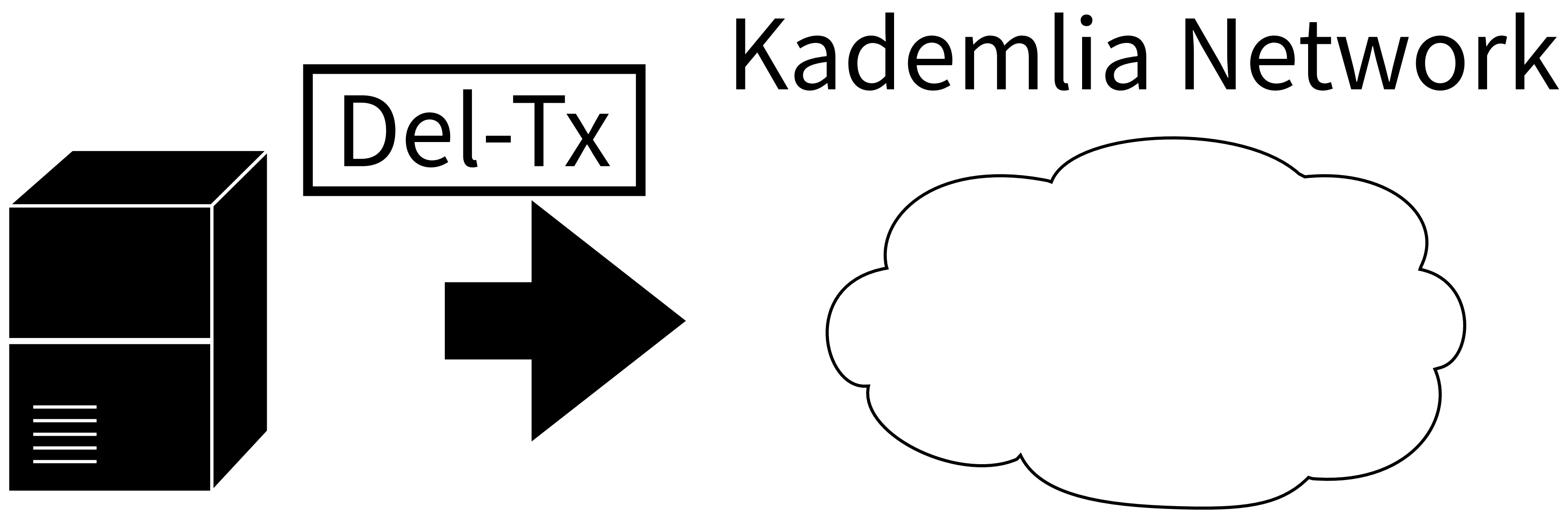
i			
0	1110		
1	1100	1101	
2	1011	1010	1001
…			

Kademlia

- PING
  - to confirm whether the recipient node is alive
- FIND_NODE
  - to search nodes closest to the value specified for the destination ID
- FIND_VALUE
  - to search nodes holding specific data
  - return nodes that holding it or closest to it
- STORE
  - to request storing data to nodes

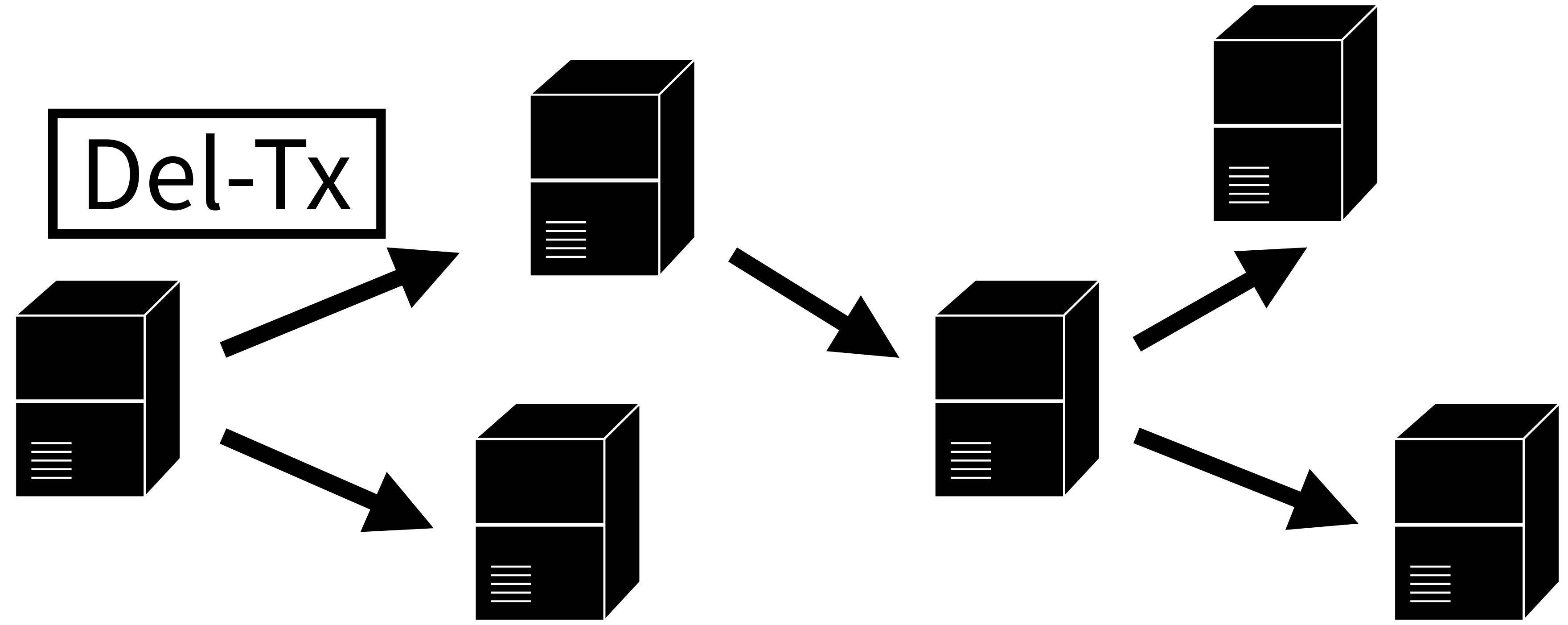
Update and delete for data

1



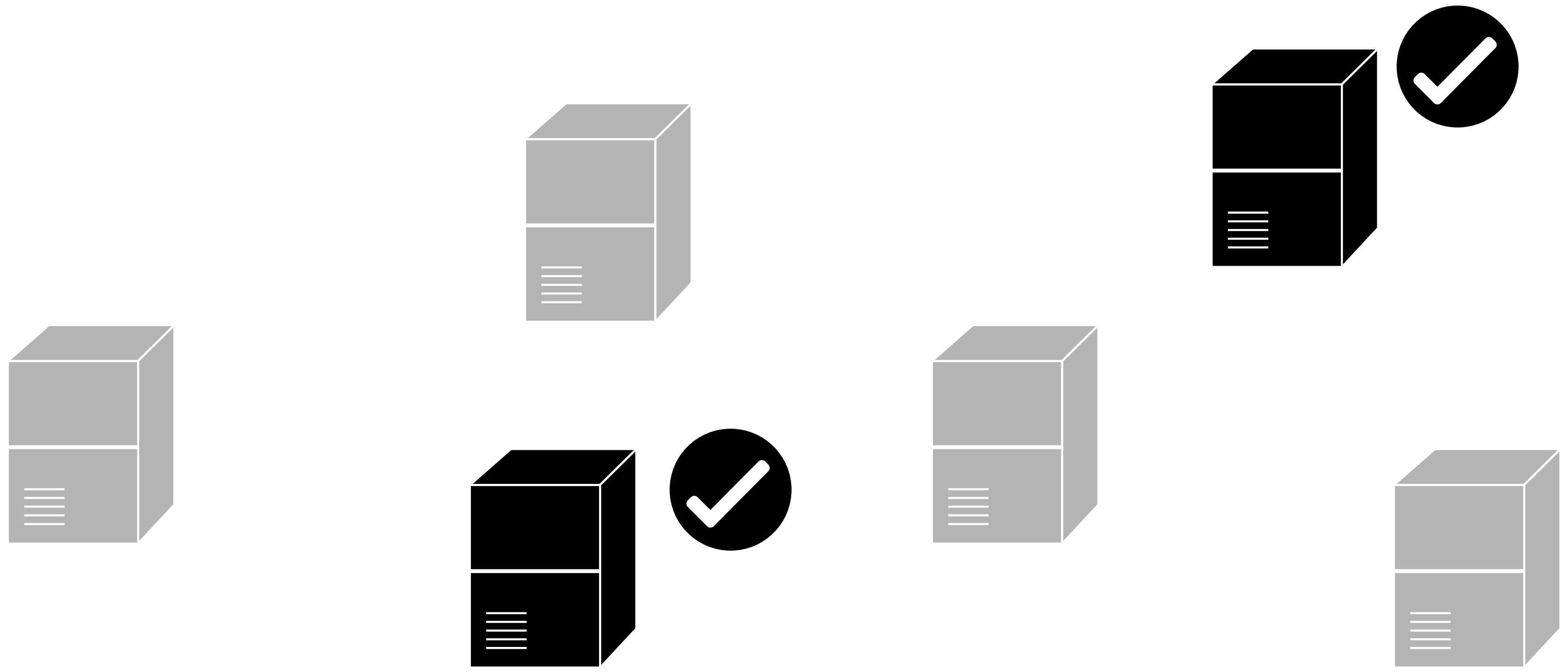
Owner node issues a Del-Tx for deleting

2



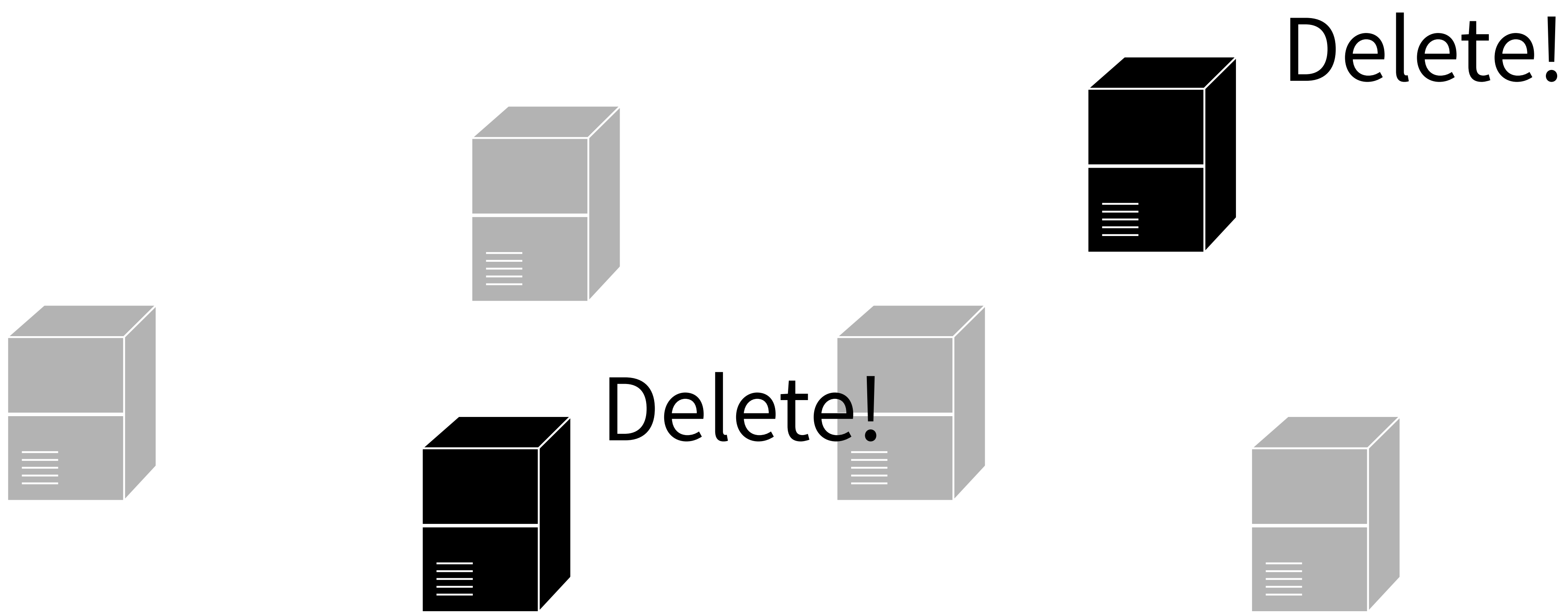
Del-Tx is broadcasted and propagated to all nodes

3



Verify whether Del-Tx is issued by owner node

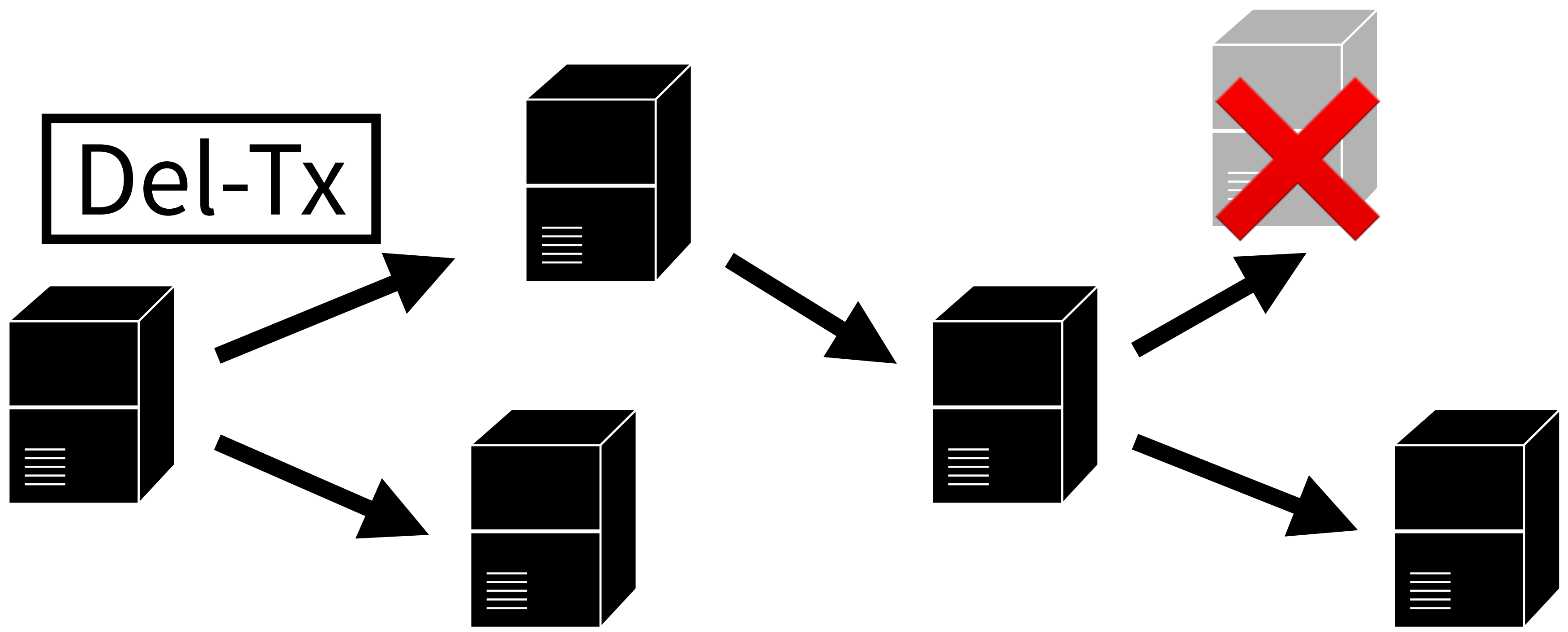
4



The result of verification is valid, it deletes the Tx

Update and delete for data

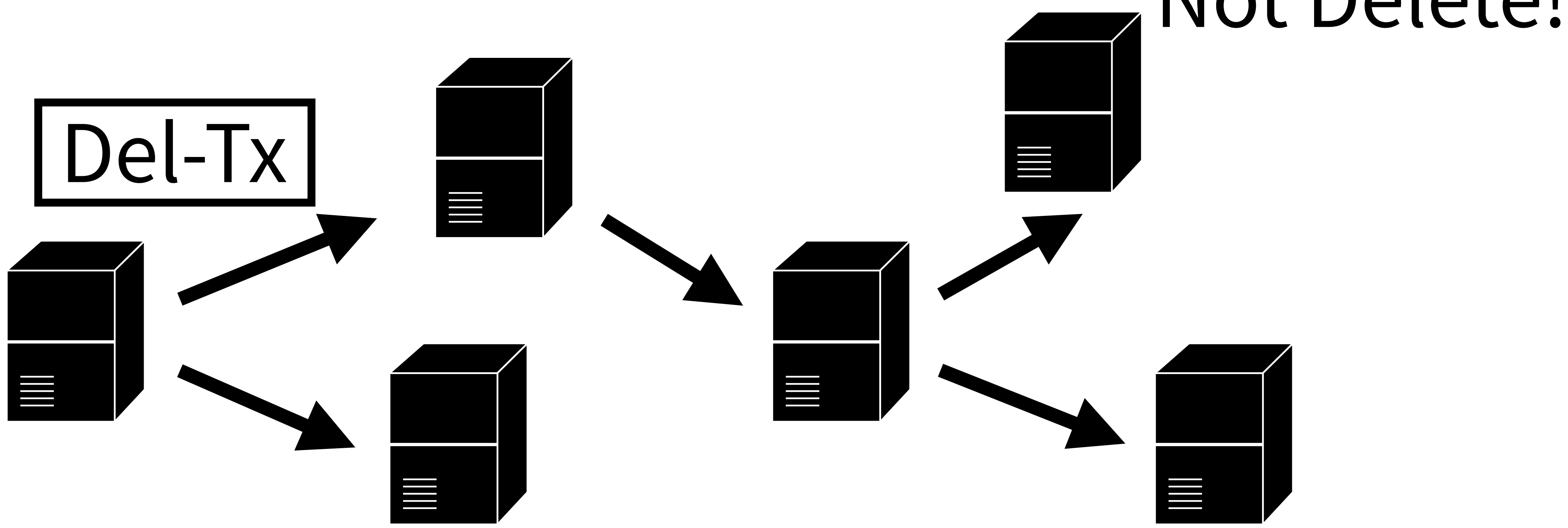
1



The node holding the data is off-line state

→ When it comes back online, it must search all blocks that occured while it was off-line

2



The node confirms a Del-Tx, but not intentionally followed

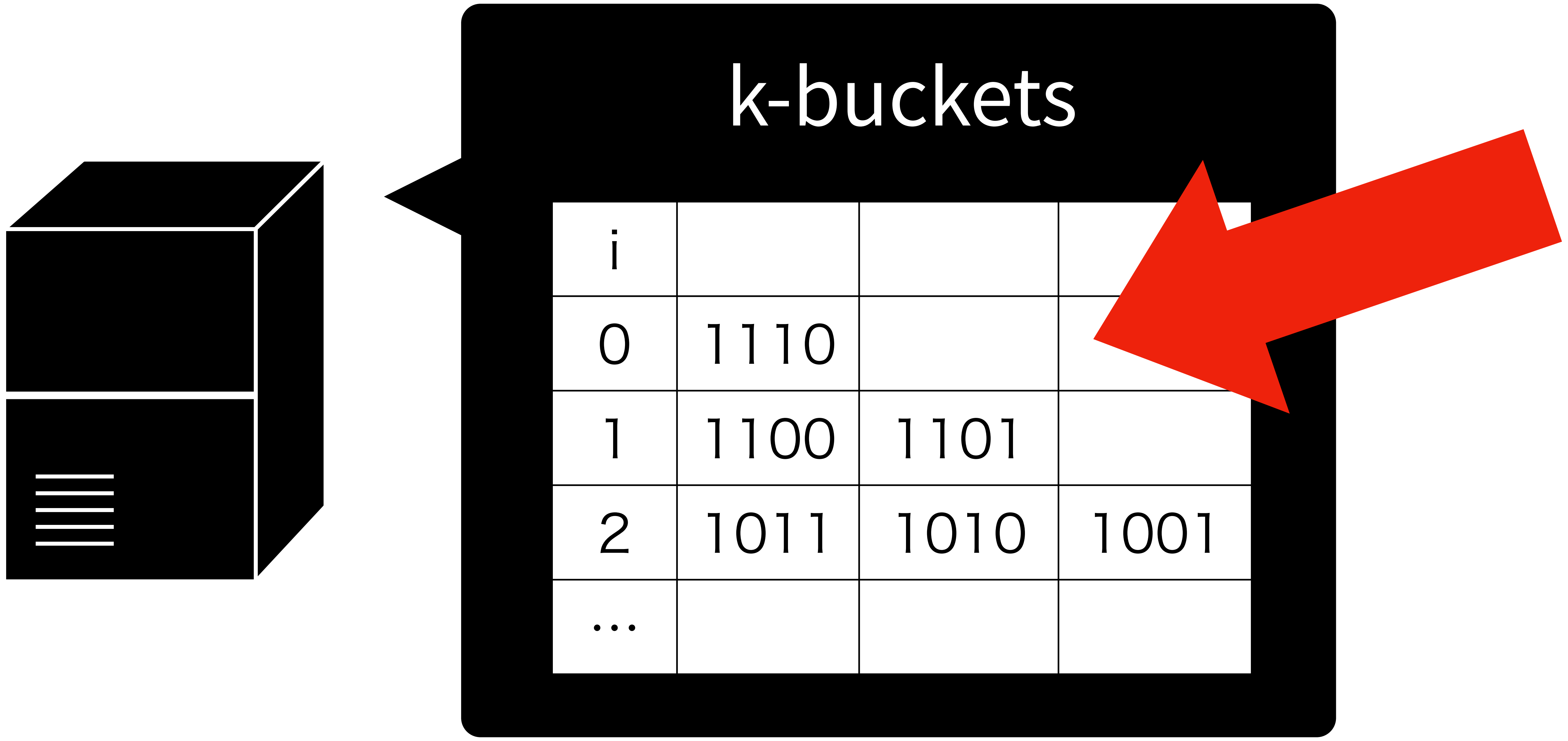
→ It is necessary to prepare an incentive for nodes to act honestly (or a mechanism to impose a penalty

Storage Layer

- Each node has three kinds of storages
  - to store data allocated within the network
  - to store transactions issued by all nodes (*Transaction pool*)
  - to store transactions and blocks generated by itself

Success Rate

- All nodes in a query node’s k-buckets are off-line
  - this probability cannot be computed since it is impossible to infer the k-buckets of a specific node



i			
0	1110		
1	1100	1101	
2	1011	1010	1001
…			

They are off-line