A Workshop on

Blockchain Technology

Lecture 0

Introduction to Blockchain Technology

Know Your Level Of Abstraction

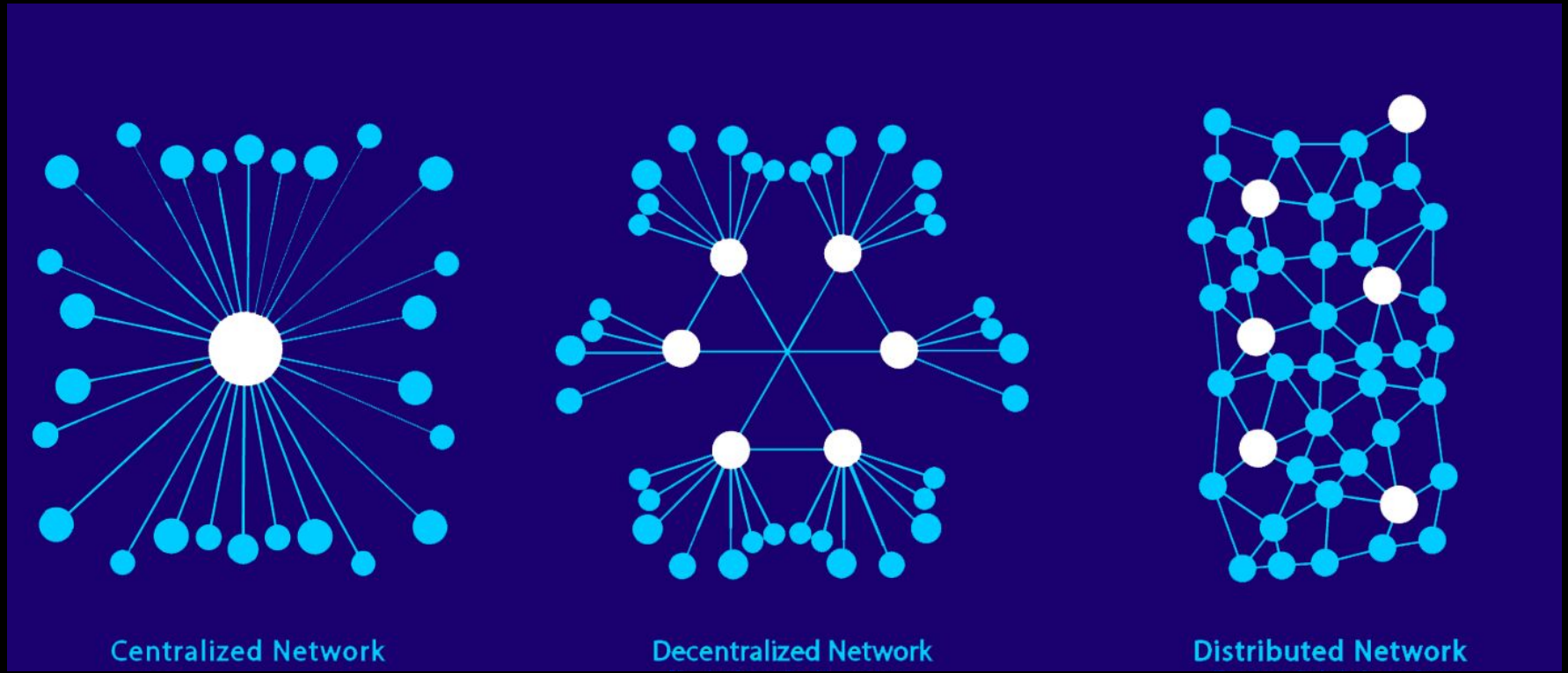
Lecture 0

Introduction To Blockchain

Centralized vs Decentralized vs Distributed

Lecture 0

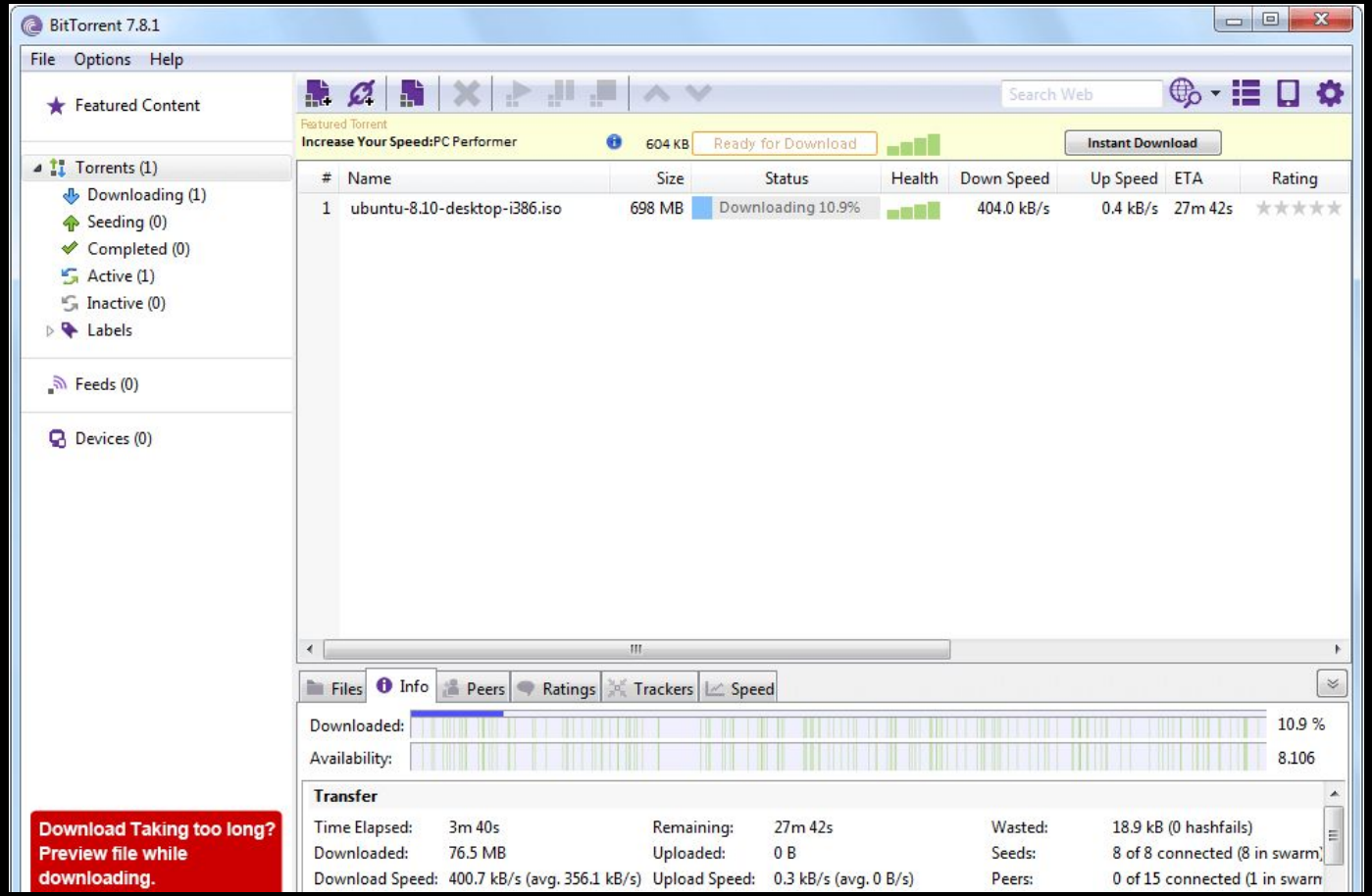
Introduction To Blockchain


Centralized Network
Decentralized Network
Distributed Network

WORLDLINK

Dear Valued Customer,

Your requested website has been blocked as per the directions received from Nepal Telecommunications Authority, Government of Nepal.


BitTorrent 7.8.1
File Options Help
Featured Content
Torrents (1)
Downloading (1)
Seeding (0)
Completed (0)
Active (1)
Inactive (0)
Labels
Feeds (0)
Devices (0)
Search Web
Featured Torrent
Increase Your Speed:PC Performer
604 KB
Ready for Download
Instant Download
Name Size Status Health Down Speed Up Speed ETA Rating
1 ubuntu-8.10-desktop-i386.iso 698 MB Downloading 10.9% 404.0 kB/s 0.4 kB/s 27m 42s
Files Info Peers Ratings Trackers Speed
Downloaded: 10.9 %
Availability: 8.106
Transfer
Time Elapsed: 3m 40s
Remaining: 27m 42s
Wasted: 18.9 kB (0 hashfails)
Downloaded: 76.5 MB
Uploaded: 0 B
Seeds: 8 of 8 connected (8 in swarm)
Download Speed: 400.7 kB/s (avg. 356.1 kB/s)
Upload Speed: 0.3 kB/s (avg. 0 B/s)
Peers: 0 of 15 connected (1 in swarm
Download Taking too long?
Preview file while
downloading.

Problems in P2P Systems: Trust and Integrity

Lecture 0

Introduction To Blockchain

What problems could arise if you removed the central bank completely from the equation?

Lecture 0

Introduction To Blockchain

Wait, so no one is actually in charge?

1. “I can’t trust an unknown user telling me that I won a lottery.”
2. “Wait, so I can agree to send money that I don’t have and there will be absolutely NO ONE to verify it?”
3. “Hold up, so I can just copy-paste the NFT and no one will actually know which is the original? Who owns it really?”
4. “What is to stop someone from simply disobeying the rules of the system in the absence of authority?”

Blockchain: The Solution

Lecture 0

Introduction To Blockchain

Implementing a Blockchain

1. Blockchain as **Data Structures**: Linked List (fundamentally)
2. Blockchain as **Algorithm**: Cryptographic Hash Functions

Keeping it as simple as possible.

Example blockchain: Manage records of assignments copied and originally done in the class.

Transactions (Assignments/Lab Report)

Lecture 0

Introduction To Blockchain

Objective: Transactions

- Define a way to store assignment details as ‘data’.

Assignment Doer	Supriya Khadka
Assignment Copier	Ranju GC
Number of Words (words)	300
Timestamp	Time Of Assignment Exchange (UNIX)

Defining a basic transaction

```python
# we are making our own coin 'words'
import time

# to be understood as assignment
# the copier owes the doer 'x' number of words
class Transaction:
    def __init__(self, doer, copier, words) -> None:
        self.timestamp = time.time()
        self.doer = doer # this person does the assignment
        self.copier = copier # this person copies the assignment
        self.words = words # number of words in the assignment
```

Making a transaction ‘unique’

```python
import time
import hashlib # library used to hash the transaction

# to be understood as assignment
# the copier owes the doer 'x' number of words
class Transaction:
    def __init__(self, doer, copier, words) -> None:
        self.timestamp = time.time()
        self.doer = doer
        self.copier = copier
        self.words = words
        self.hash = self.calculate_hash() # store a unique hash for each transaction to identify it

    def calculate_hash(self) -> str:
        transaction_string = str(self.timestamp) + str(self.doer) + str(self.copier) + str(self.words)
        return hashlib.sha256(transaction_string.encode('utf-8')).hexdigest() # use the sha256 hashing algorithm

```

Q: Making transactions on the behalf of others? How can you verify that the transaction was carried out by the actual owner?

Lecture 0

Introduction To Blockchain

Spoof!

```
tx1 = Transaction(doer='Supriya', copier='Ranju', words=300)
tx2 = Transaction(doer='Newton', copier='Sanskar', words=500)
```

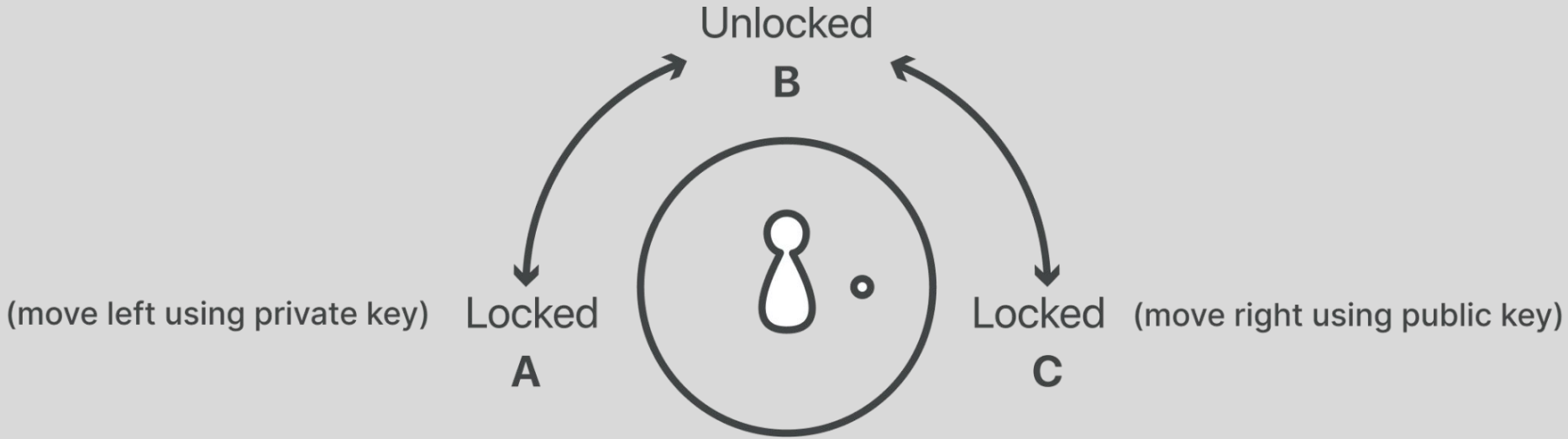
Neither transaction has been made by ‘Pranjal’, even though we don’t have a way to verify that it was actually not a valid transaction.

Public-private (asymmetric) encryption

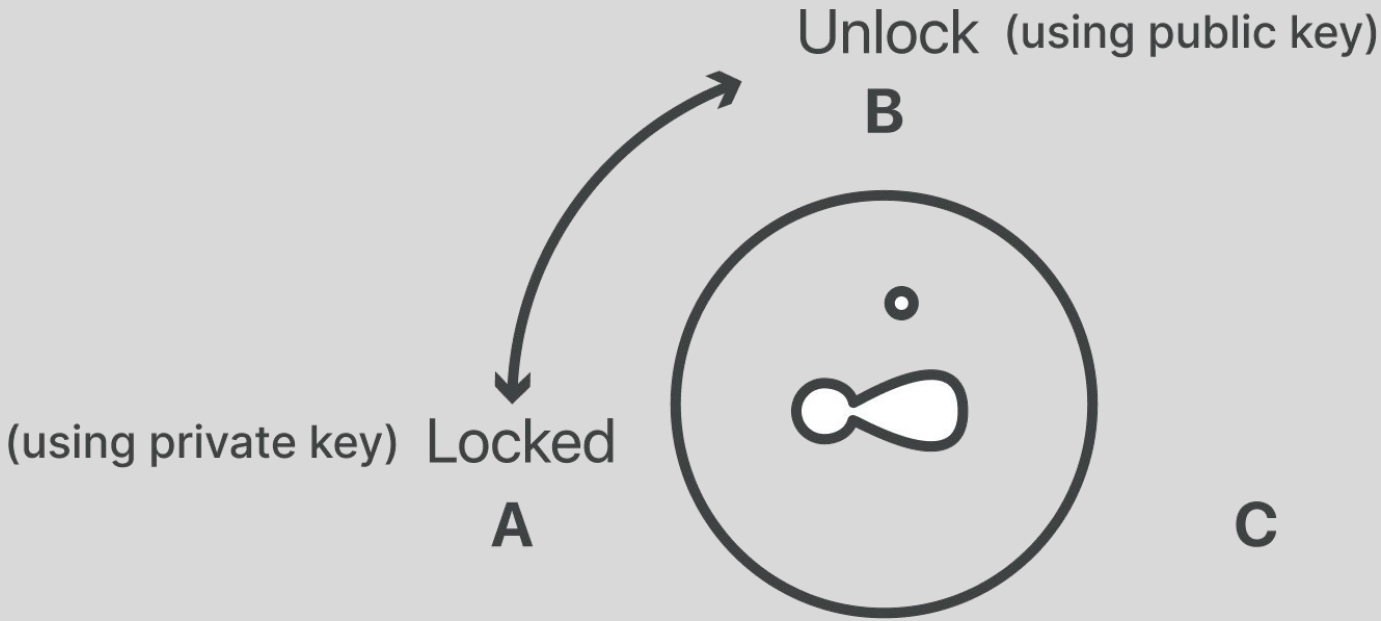
To keep things as simple as possible,

- We have two ‘keys’ - a private key and a public key derived from the private key
- Whatever is encrypted using the private key can be decrypted using the public key
- And vice-versa

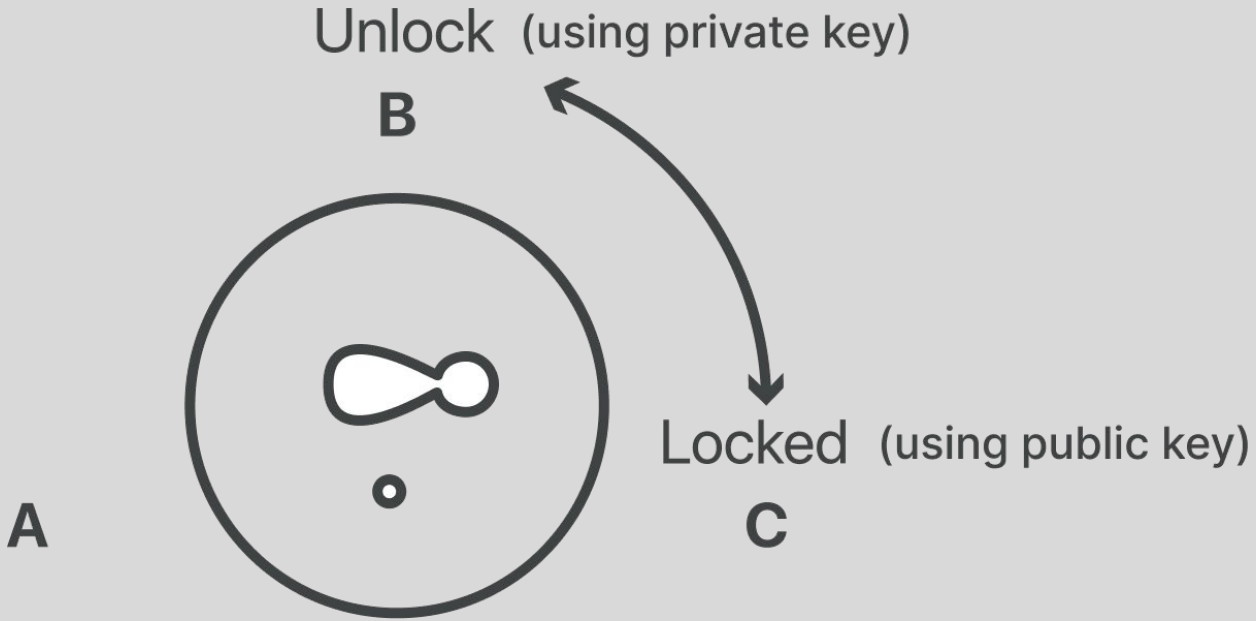
Public-private (asymmetric) encryption



Public-private (asymmetric) encryption



Public-private (asymmetric) encryption



Translating to blockchain

1. Private and public key pair are handled using a blockchain wallet (Electrum, Metamask)
2. The actual ‘lock’ in our case is the transaction hash.
3. Locking/signing a transaction: Encrypting the transaction hash with private key
4. If the user’s public key is able to decrypt the signed transaction hash, then the transaction is the user’s to spend.

Account Balance == Unspent Transactions (UTXO)

Lecture 0

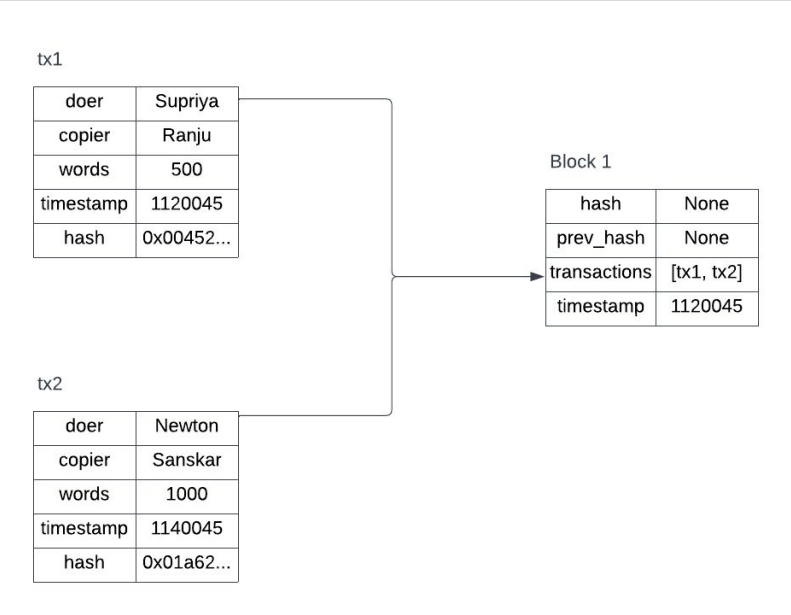
Introduction To Blockchain Technology

Blocks (Assignments Record - Page)

Lecture 0

Introduction To Blockchain Technology

Going from transactions to block



Transactions are added to blocks.

A block can contain one or more transactions.

Defining a basic block

```
# block holds transactions i.e. list of assignment details
You, 1 second ago | 1 author (You)
class Block:
    def __init__(self, pending_transactions: List[Transaction]) -> None:
        self.timestamp = time.time()
        self.transactions = pending_transactions # transactions that have yet to be added to a block
        self.hash = None
```

Calculating block hash

```
def calculate_hash(self, nonce: int) -> str:
    block_string = str(self.timestamp)        You, 1 second ago • Uncommitted cha

    # linearly combine transaction hashes into one common string
    for transaction in self.transactions:
        block_string += str(transaction)

    self.hash = hashlib.sha256(block_string.encode('utf-8')).hexdigest()
    return self.hash
```

Blocks in a Chain (Assignments Record - Book)

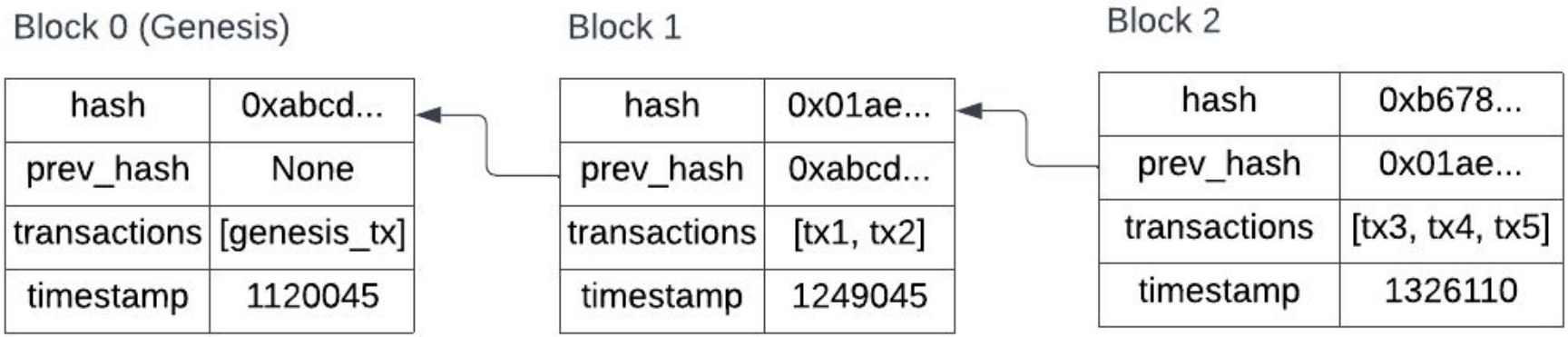
Lecture 0

Introduction To Blockchain Technology

Blockchain as Distributed Ledgers

- Solve a ‘cryptographic puzzle’ before adding a block to a chain.
- Computing the puzzle is time consuming but verifying the solution is easy (think solving linear equations).
- A simple proof of work algorithm is to make sure the block hash has number of leading zeros equal to a certain ‘difficulty’.

Going from individual blocks to chain



Adding previous hash into hash calculation

```
def calculate_hash(self) -> str:
    block_string = str(self.timestamp) + str(self.previous_hash)

    # linearly combine transaction hashes into one common string
    for transaction in self.transactions:
        block_string += str(transaction)

    self.hash = hashlib.sha256(block_string.encode('utf-8')).hexdigest()
    return self.hash
```

Q: Older blocks with modified information?

Lecture 0

Introduction To Blockchain

Proof of Work (PoW)

Lecture 0

Introduction To Blockchain

Proof of Work

- Solve a ‘cryptographic puzzle’ before adding a block to a chain.
- Computing the puzzle is time consuming but verifying the solution is easy (think solving linear equations).
- A simple proof of work algorithm is to make sure the block hash has number of leading zeros equal to a certain ‘difficulty’.

Proof of Work

- Consider difficulty is 2 i.e. the calculated block hash must have at least 2 leading hexadecimal zeros.

```
c86e3aa59b4bf0ae32cb4d0409c5b6891e2c2aa588e15697ec9a46e2b456b7f2
04a17750896f692a4e83c7575d63b9d8fcc517e0f3465c53b842434a02328e42
1c0295b649448d5d7ccbd6f871b42a18019a5cade64f04baad6a3ac2b5905973
137ab74927ccf023a4e8616c74f9fba1304bd11695b028cca0a3869033dcb708
000acf08362a2dd0a21711a613b597aeed66b9ef15450b7574dc031d643e43d6
```

Nonce

- **N**umber used **once**
- Added into block hash calculation function and is changed continuously until target difficulty is reached.
- Calculating a correct nonce value is called ‘mining’.
- Only way to find the value of nonce is through pure brute force.
- Reward provided for mining a block as new currency minted in the system.

Mining block

```
def mine_block(self, difficulty: int) -> None:
    nonce = 0 # change block hash by incrementing nonce
    while True:
        self.calculate_hash(nonce)
        print(self.hash) # printed current hash for aesthetic, not required
        hash_substring = self.hash[0:difficulty] # gets first n hex digits of hash equal to difficulty
        leading_zeros = '0' * difficulty
        if hash_substring == leading_zeros:
            self.nonce = nonce
            break
        nonce += 1      You, 5 seconds ago • Uncommitted changes
```

Calculating block hash based on nonce

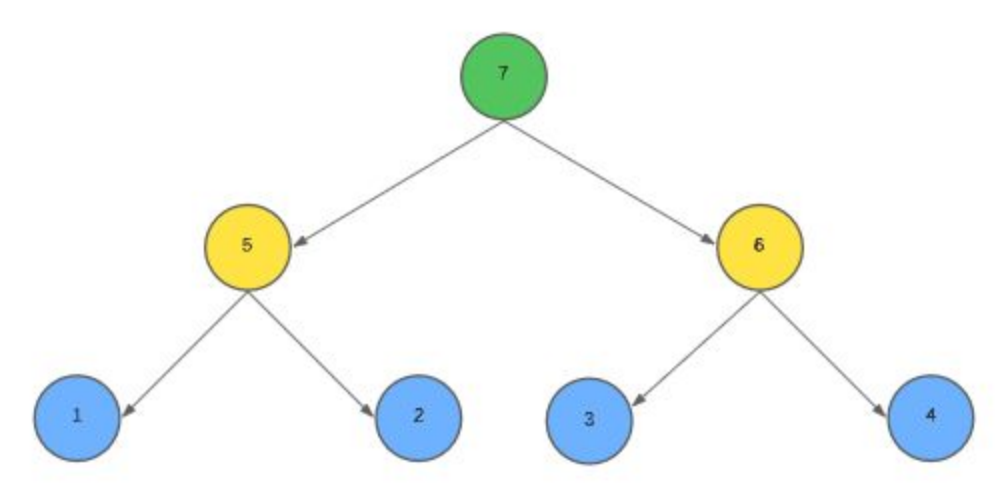
```
def calculate_hash(self, nonce: int) -> str:
    block_string = str(self.timestamp) + str(self.previous_hash) + str(nonce)

    # linearly combine transaction hashes into one common string
    for transaction in self.transactions:
        block_string += str(transaction)

    self.hash = hashlib.sha256(block_string.encode('utf-8')).hexdigest()
    return self.hash
```

Merkle Tree

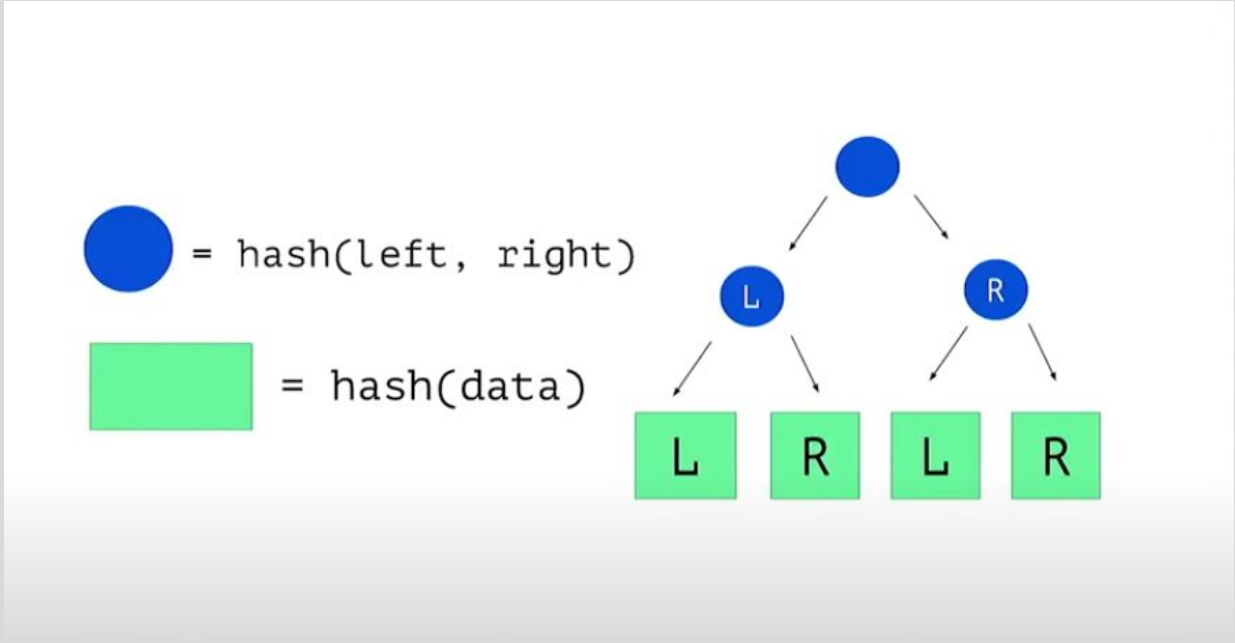
Lecture 0

Introduction To Blockchain

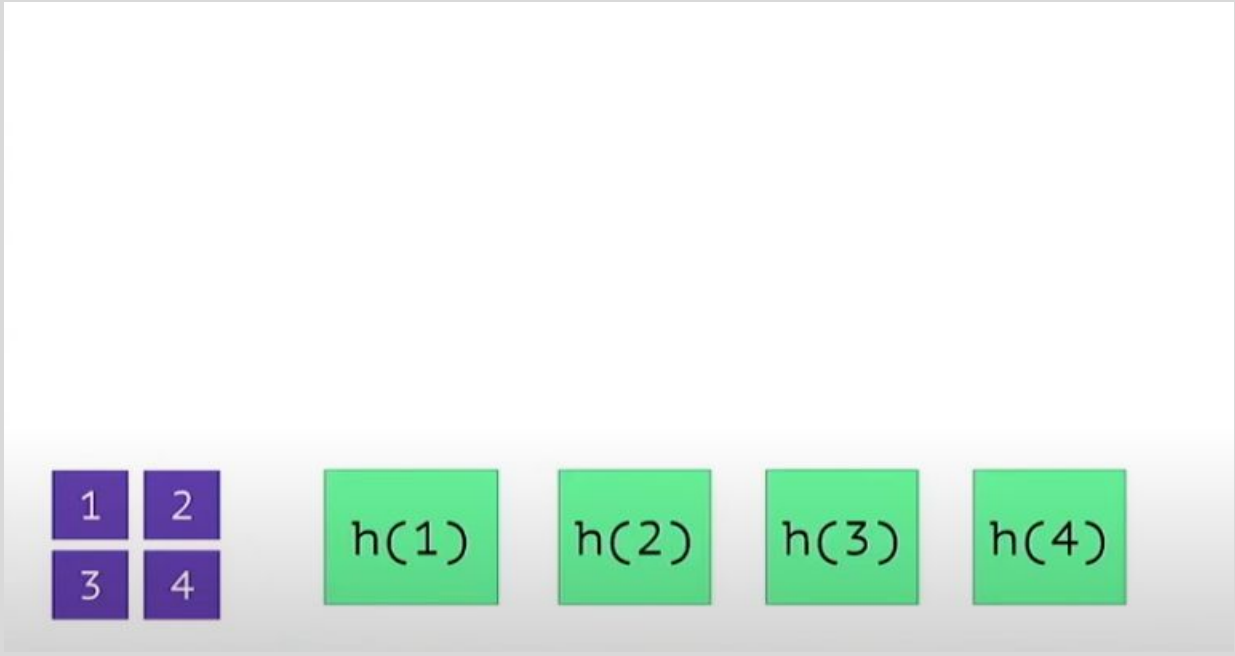
Regular Binary Tree



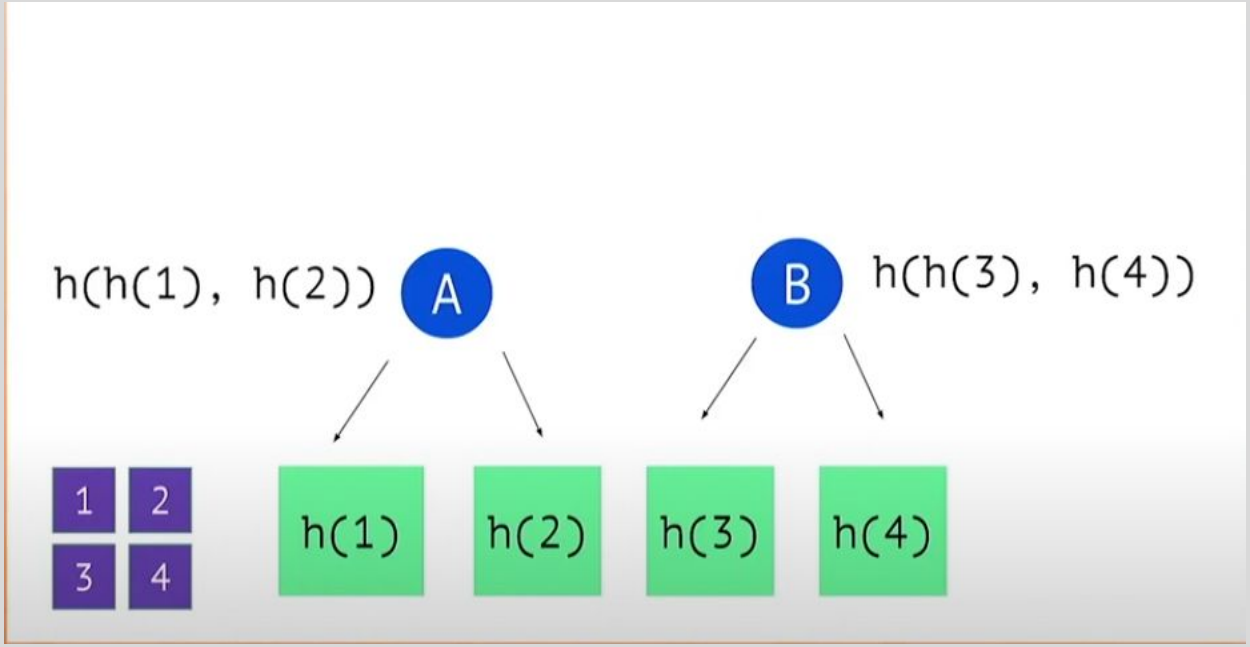
Merkel Tree



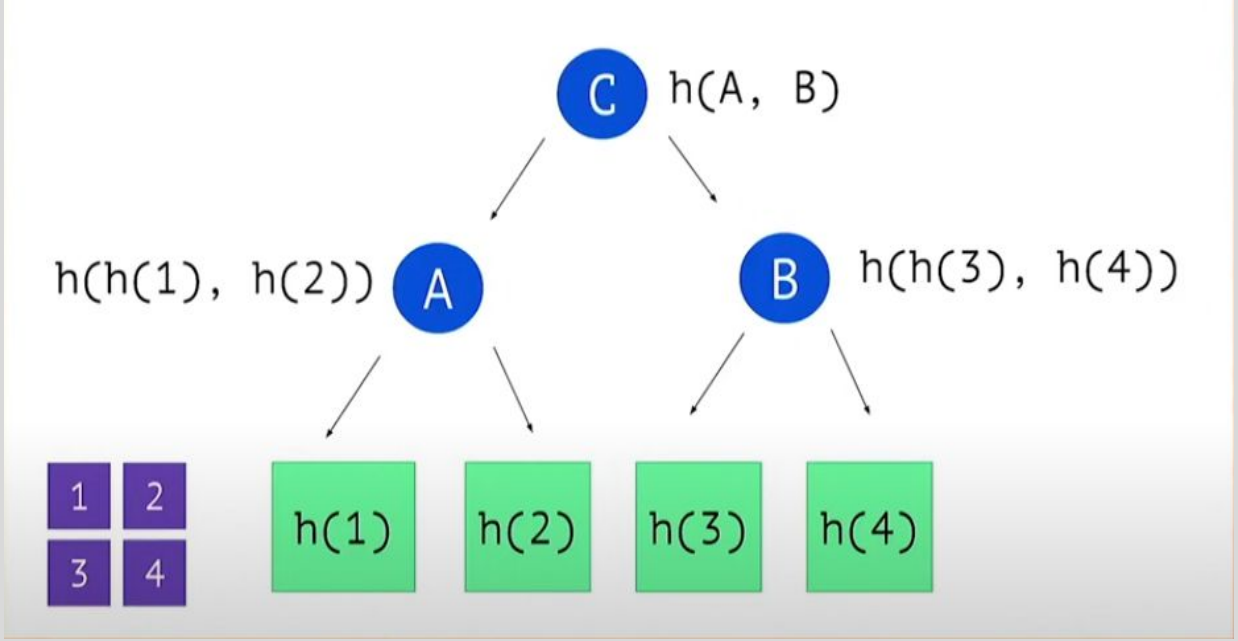
Merkel Tree

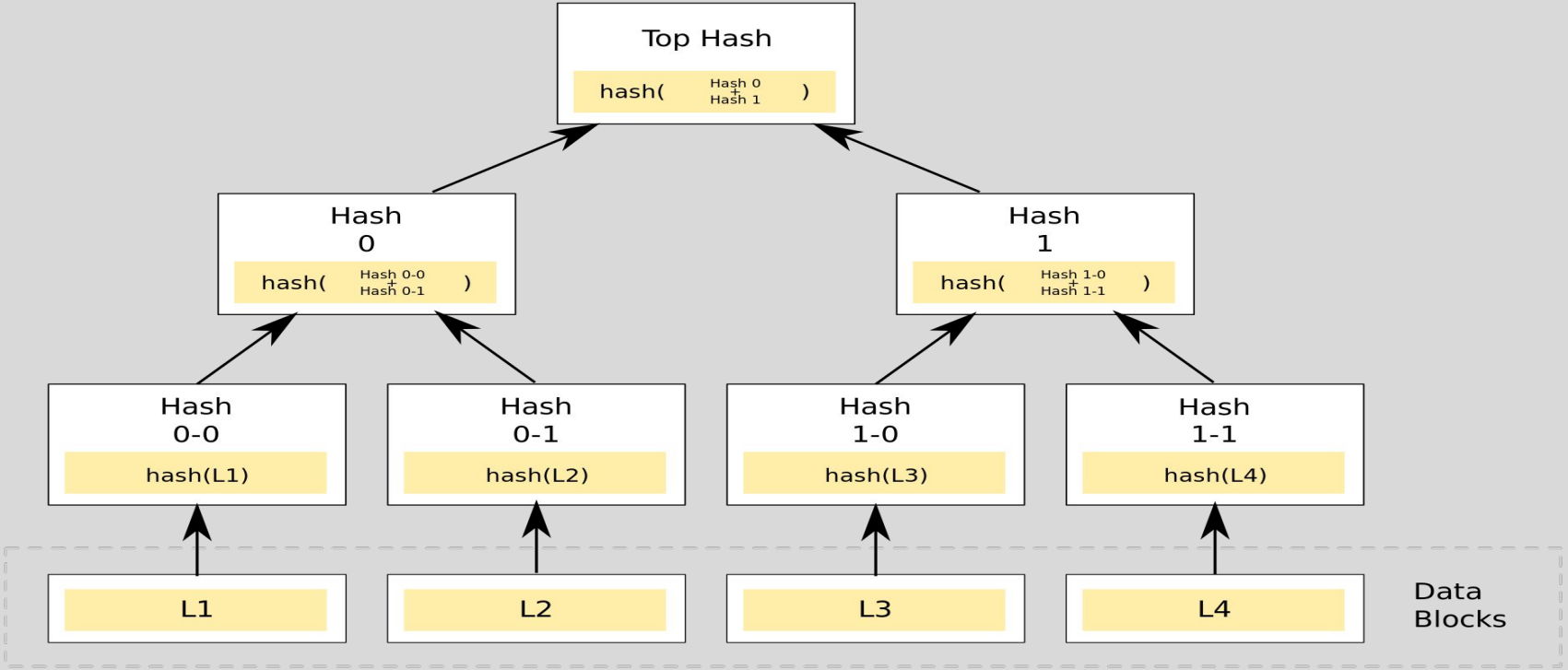


Merkel Tree



Merkel Tree



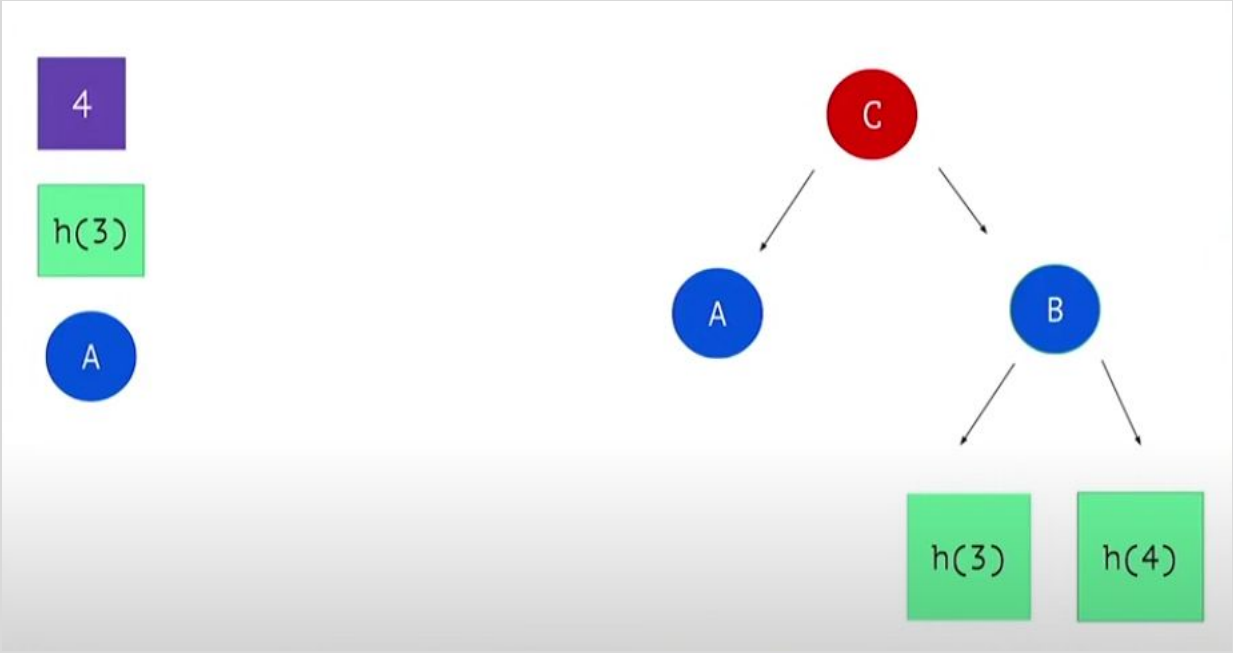

Top Hash
hash(Hash 0 + Hash 1)
Hash 0
hash(Hash 0-0 + Hash 0-1)
Hash 1
hash(Hash 1-0 + Hash 1-1)
Hash 0-0
hash(L1)
Hash 0-1
hash(L2)
Hash 1-0
hash(L3)
Hash 1-1
hash(L4)
L1
L2
L3
L4
Data Blocks

Q:Why create a tree when we can just concatenate transactions?

Lecture 0

Introduction To Blockchain

Partial Verification



Peer to Peer Network

Lecture 0

Introduction To Blockchain

Q: How do nodes communicate without a central entity?

Lecture 0

Introduction To Blockchain

Types of Node

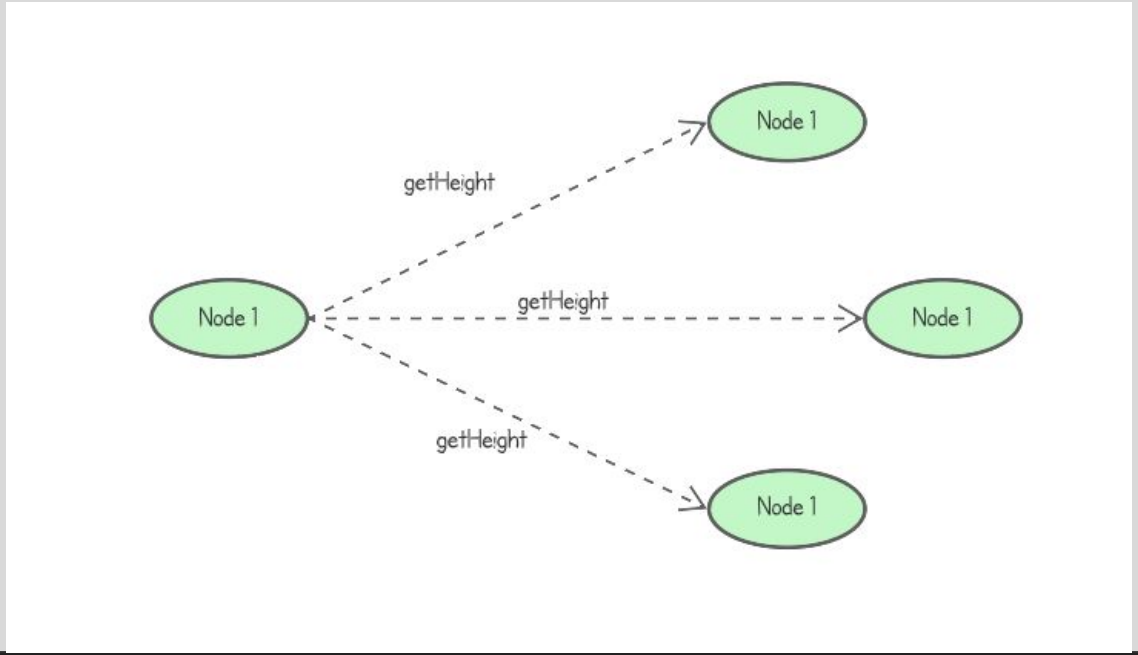
- Full Node
- Simplified Payment Verification (SPV) Node
- Miner Node

Q: How do you join to a P2P Network?

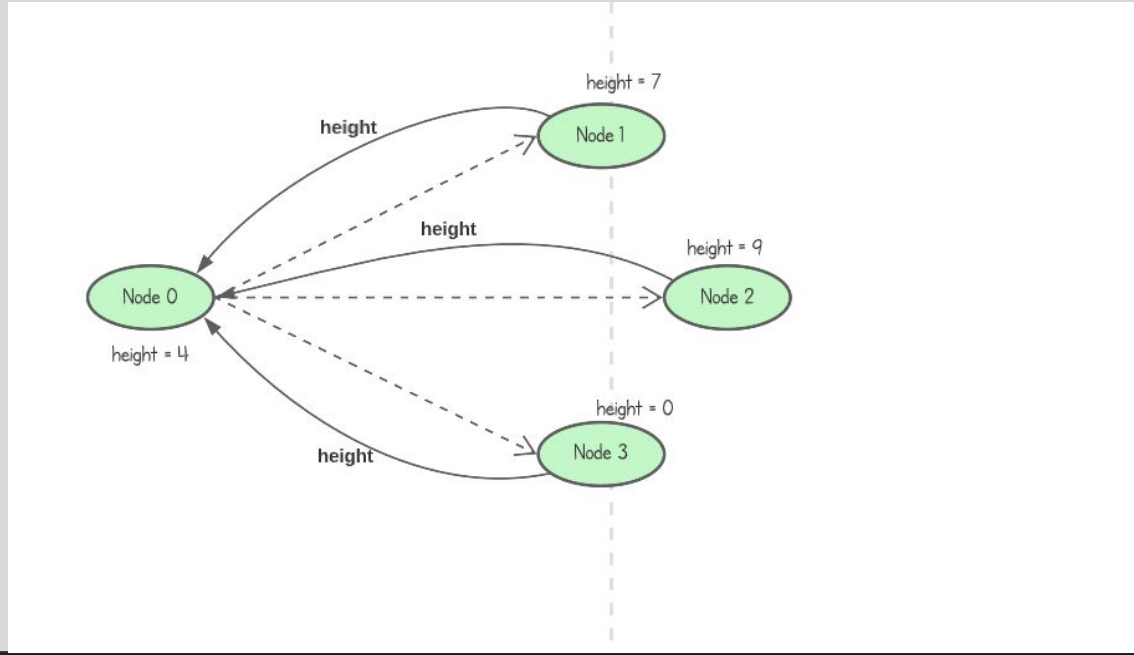
Lecture 0

Introduction To Blockchain

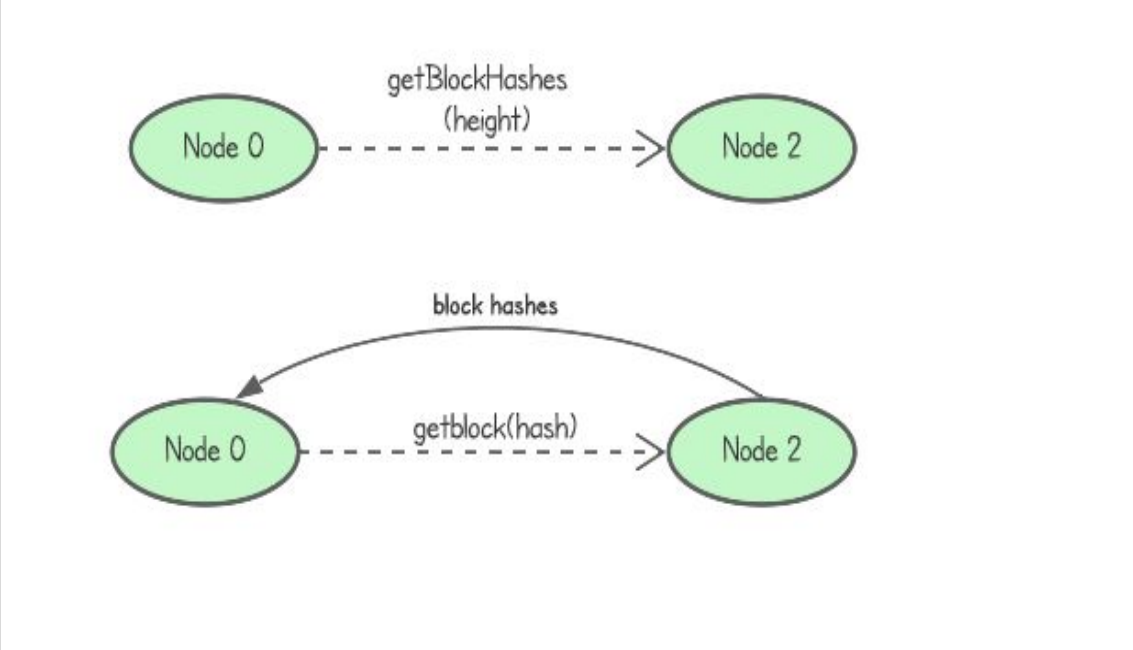
Communication Process



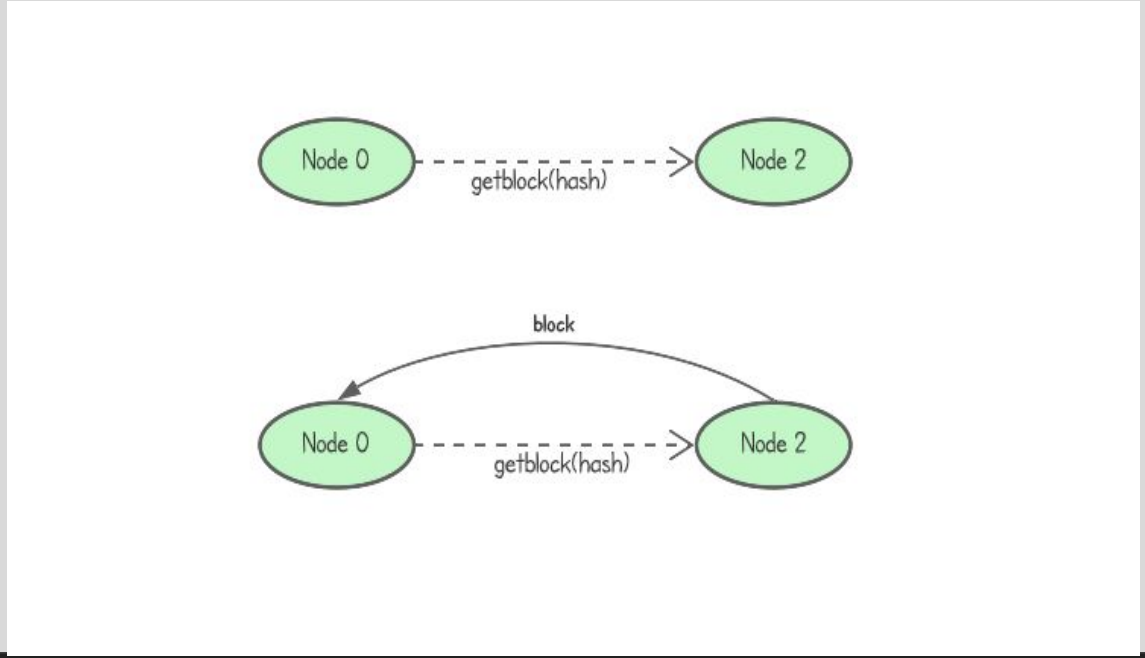
Communication Process



Communication Process



Communication Process



Consensus

Lecture 0

Introduction To Blockchain

Q: What if different computers have chains with different data?

Lecture 0

Introduction To Blockchain

References

Lecture 0

Introduction To Blockchain