Lecture 5: Stitching Bitcoin Components Together

https://web3.princeton.edu/principles-of-blockchains/

Professor Pramod Viswanath
Princeton University

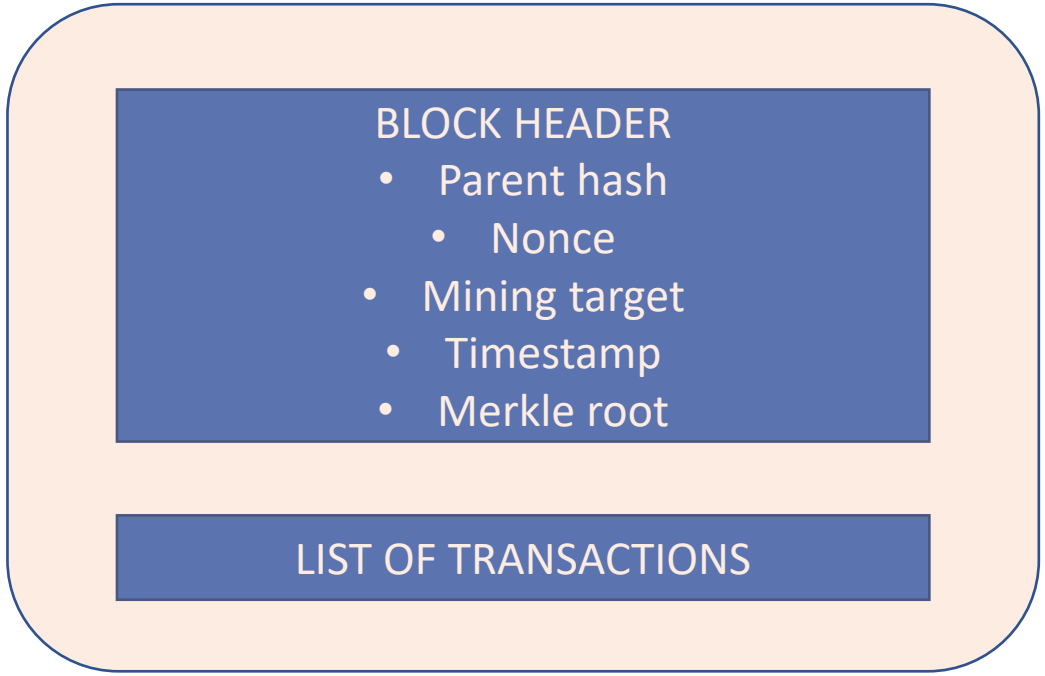
This lecture:
Components of the Bitcoin system
System View of Bitcoin

} lens of software client implementation

Parts of the Bitcoin Full Client Project

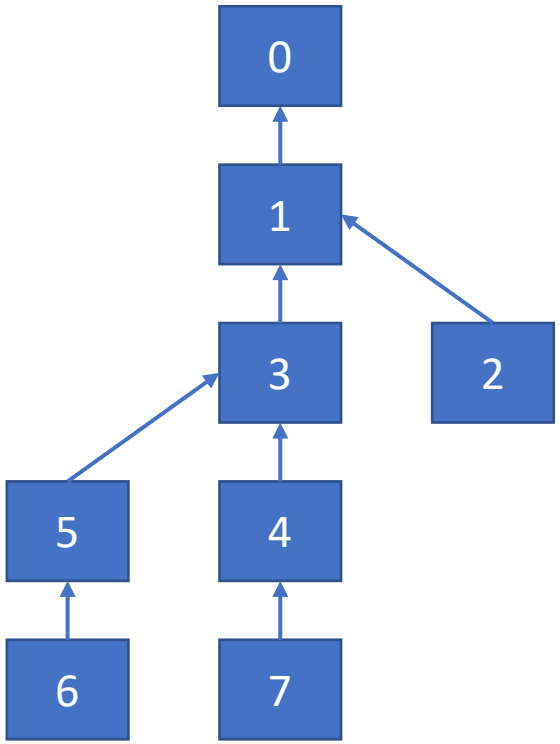
- Week 1 - Block and Blockchain structure
- Week 2 - Miner
- Week 3 - Network
- Week 4 - Putting together a data blockchain
- Week 5 - Transactions
- Week 6 - State validation

Week 1 - Block structure



- Merkle root of the list of transactions
- Difficulty: Mining target
- Timestamp: Local time when a block is mined

Week 1 - Blockchain structure



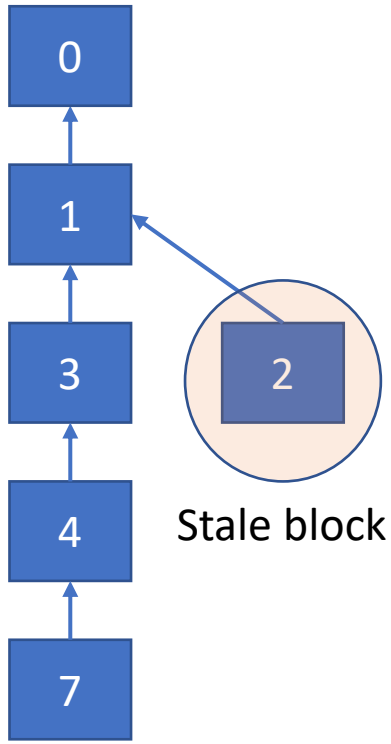
Longest chain

- If two chains have same length

Pick one randomly

7 Tip of the blockchain

Week 1 - Blockchain



- Store stale blocks as they may lead to a new longest chain
- Most bitcoin clients trim the blocktree after every update with an updated checkpoint

- It may be possible that block 7 was received before block 4 is received
- Parent of block 7 is missing, we don't know where to place block 7
- Store it in an orphan block buffer, and process the block once its parent is received and processed
- Orphan block buffer is of finite size, can you think of an attack?

Week 2 - Mining

BLOCK HEADER (New block)
- Parent hash(tip of blockchain)
- Nonce
- Mining target
- Timestamp
- Merkle root

- Try different nonce in block header until,

Hash(block header) < Mining target

- Equivalent to lottery due to uniform random output of Hash()
- Mining target set by the protocol
- Change timestamp, Merkle root(list of transactions)

if all options of nonce are exhausted

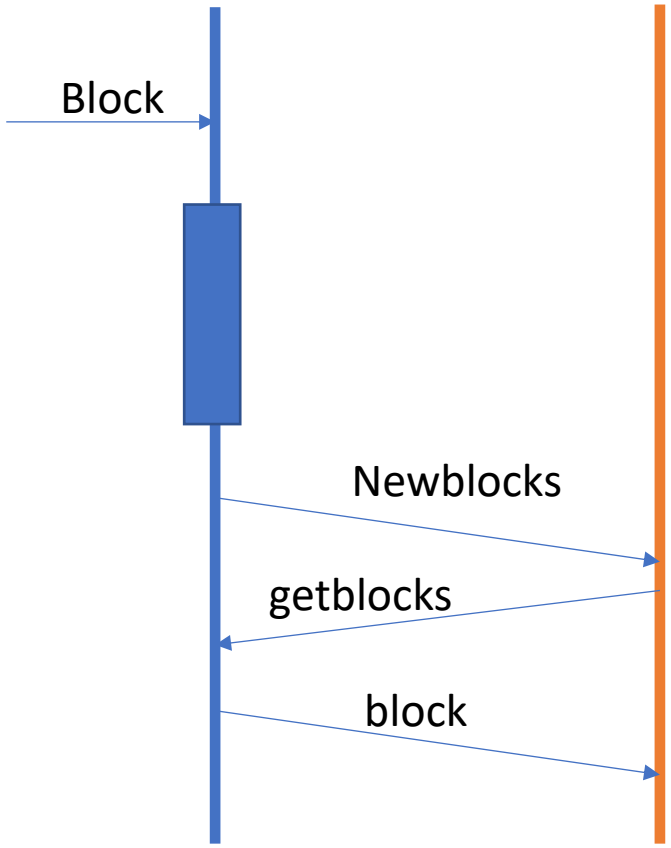
LIST OF TRANSACTIONS → (Coinbase transaction, all other transactions)

- Miner gets its mining reward through coinbase transactions
- Coinbase transactions: Thin air -> Miner's account (Public key)
- Coinbase transaction will be different for different miners

Mining Difficulty

- difficulty = difficulty_1_target / current_target
- difficulty_1_target = (first 32 bits zero in a 256-bit number)
- Updated every 2016 blocks (approx. 2 weeks)
- ΔT = Time taken by last 2016 blocks (in mins)
- New_difficulty = Old_difficulty * (2016*10/ ΔT)
- ¼ < New_difficulty/Old_difficulty < 4
- Maintains Inter block arrival time average of 10 mins
- Chain with the most work

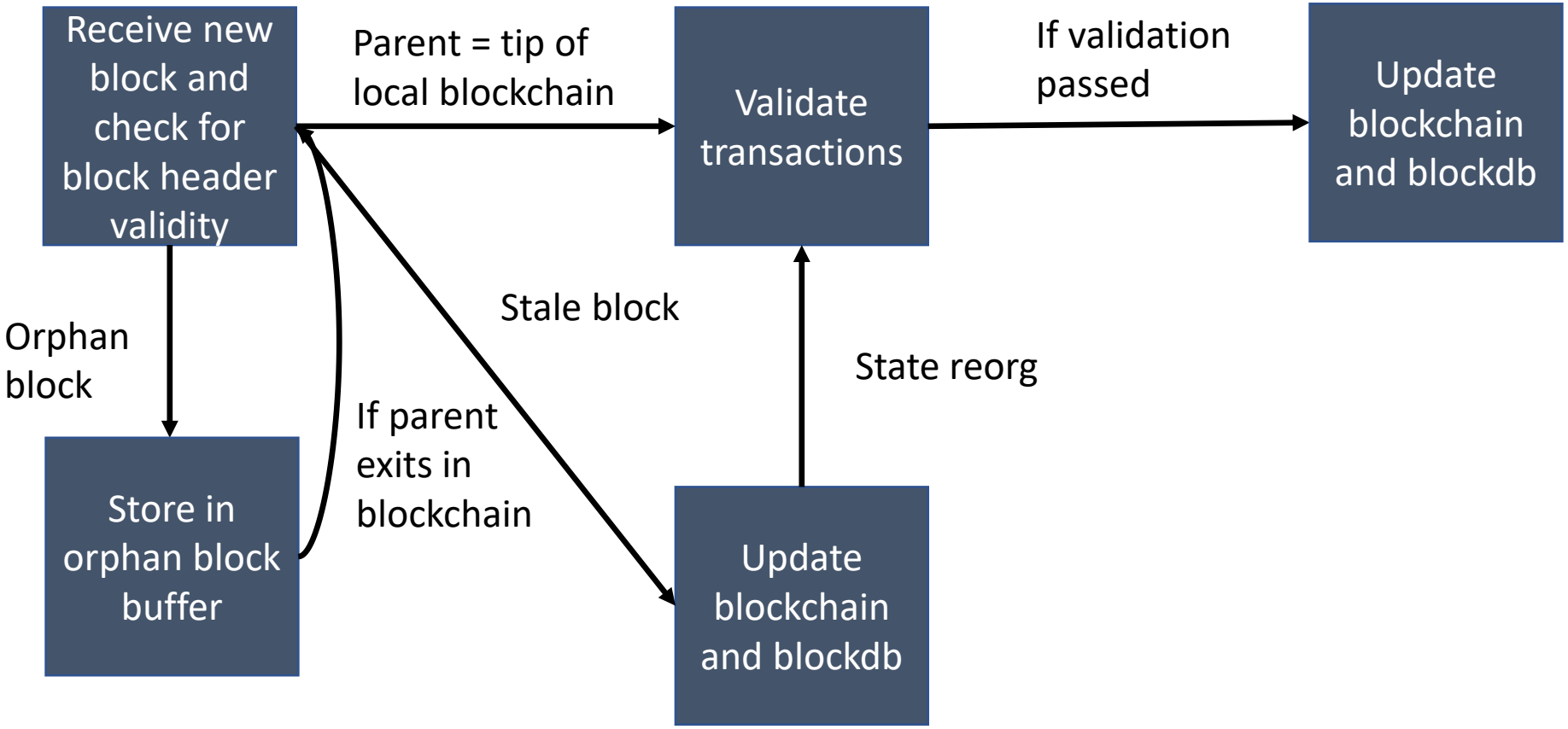
Week 3 - Network



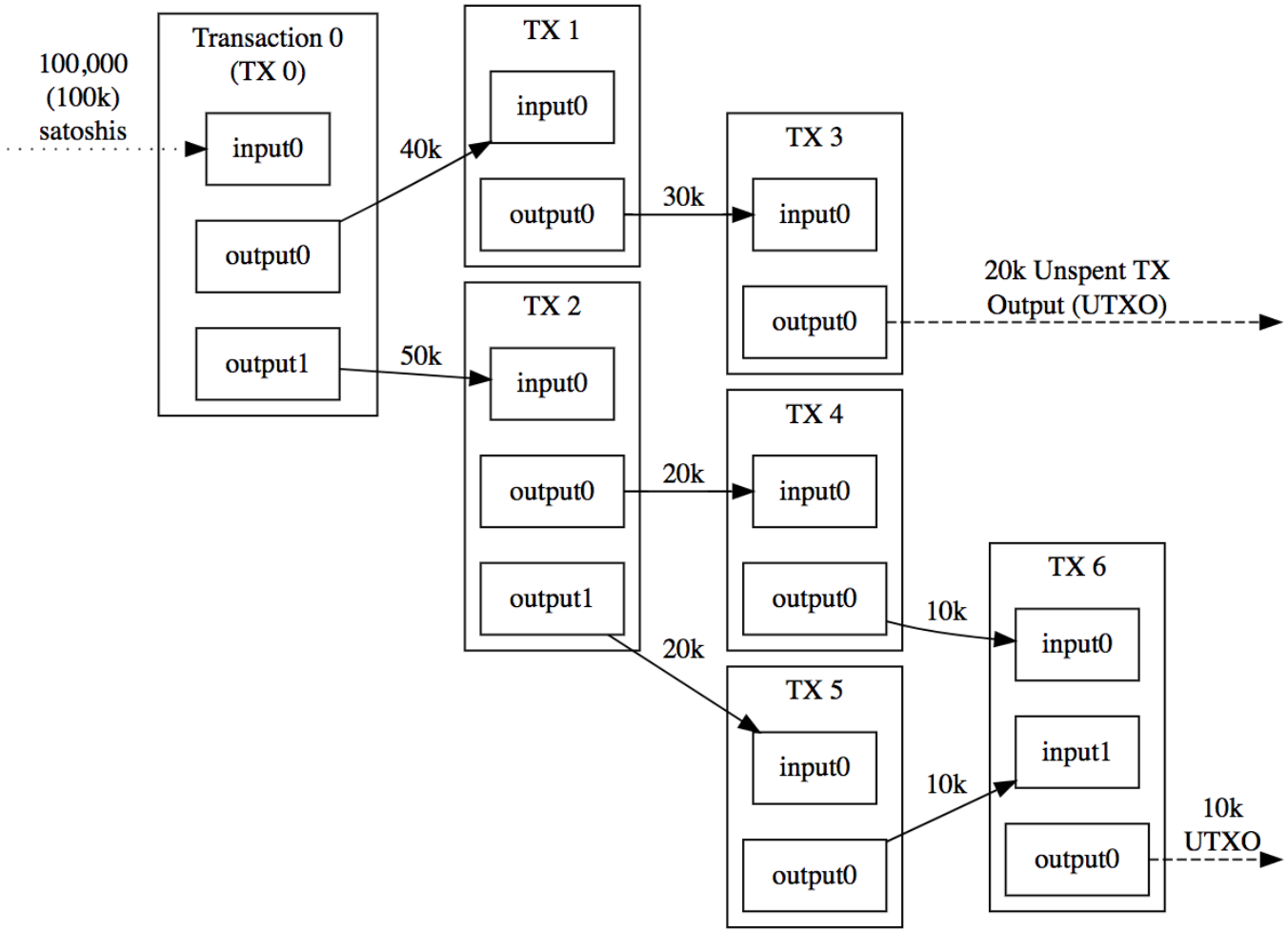
- Implement a Block First node
- Inv = NewBlocks(list of blockhash)
- Getdata = getblocks(list of blockhash)
- Block = Block header+ block content(transactions)

- Do not need to implement peer discovery protocol
- Create a d-regular topology and centrally feed peer addresses to each node

Week 4 - Putting together (Network + Blockchain)



Week 5 - Transactions-UTXO



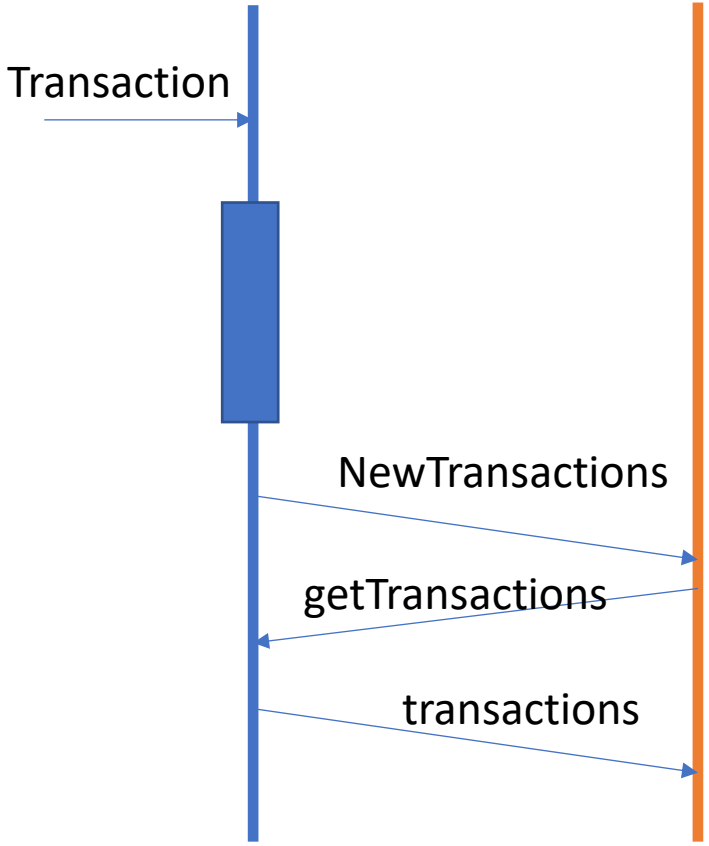
Triple-Entry Bookkeeping (Transaction-To-Transaction Payments) As Used By Bitcoin

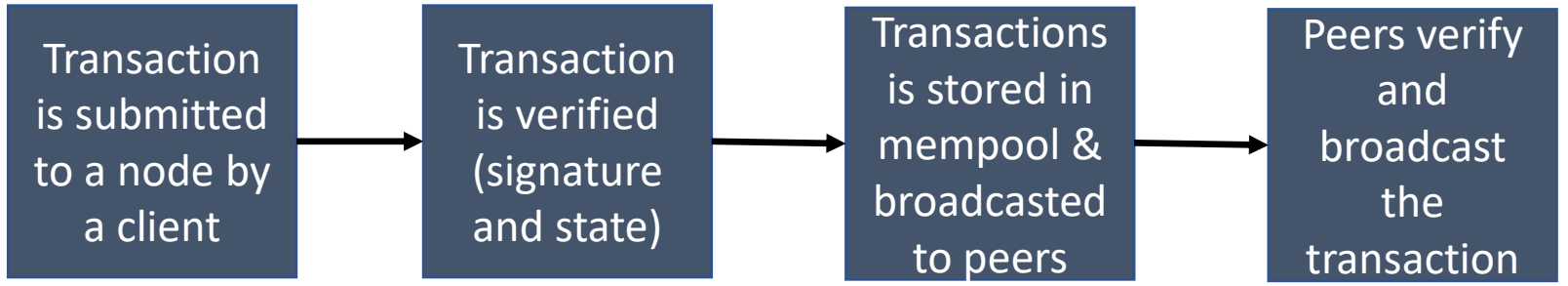
TRANSACTION
- INPUT((coinid, value, owner))
- OUTPUT((value, recipient))
- Signature(of all input owners)

- UTXO: Unspent Transaction Output
- UTXO are stored as coins: Coinid: Hash of tx, index of output
- Inputs must be UTXOs
- Alternate: account model

Source: bitcoin.org

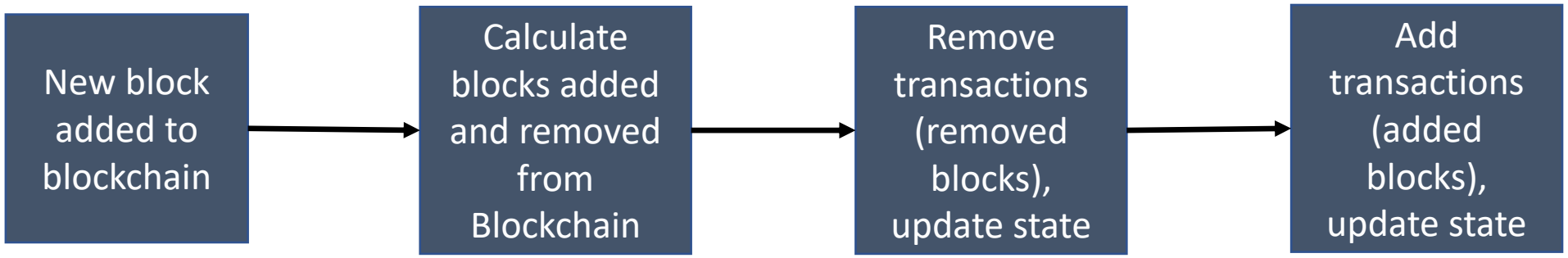
Week 5 - Transactions- network, mempool





- Transaction Mempool is of fixed size
- Miner collects transactions from mempool (priority typically given to higher fees) and forms a block

Week 6 - State handling and validation



- Each node maintains a UTXO database (state)
- Check if tx inputs are in UTXO database (state validation)
- When a new block arrives, it leads to reorganization of blockchain
- Some transactions are added (typically the scenario)
- Remove Input of the transaction from UTXO db, add outputs of tx to UTXO db
- Some transactions are removed (if a fork is now the longest chain)
- Remove output of transactions from UTXO db, add input of tx to UTXO db

Syntax of Transactions

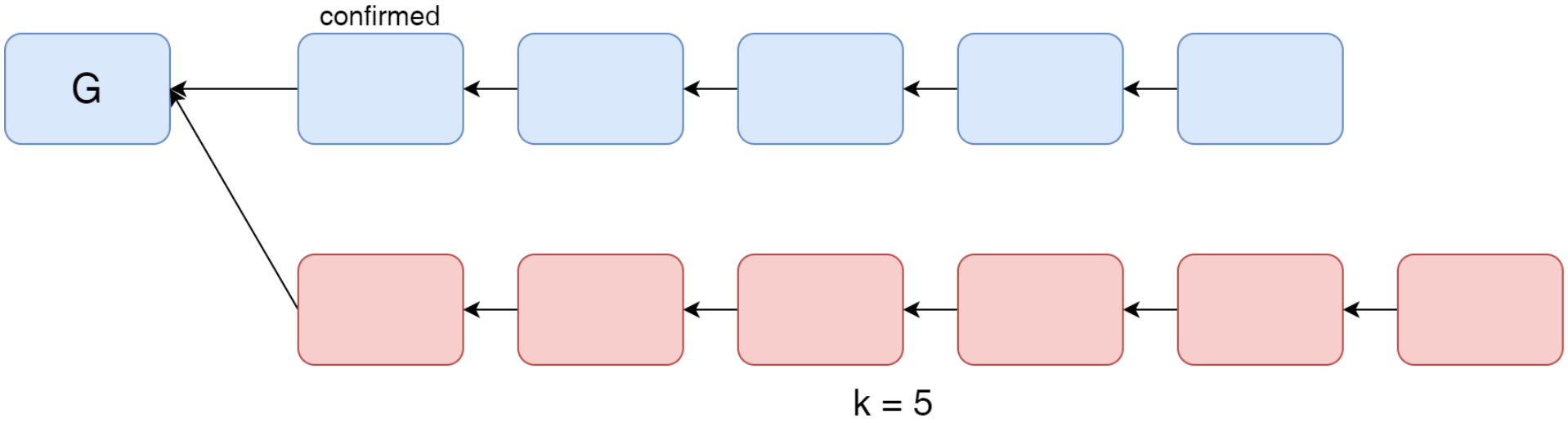
- Address vs Account
  - Address of a transaction is hash of a user's public key
- Input and Output of a transaction
  - Input = amount and address of sending BTCs
  - Output = amount of address of receiving BTCs
- Signature on a transaction
  - Signed by the private key of owner of BTCs in the input field
  - append public key for verification
- Validation
  - Every transaction input address must be the output address of a previous transaction
- UTXO model
  - Alternate: account based model

Transaction fees

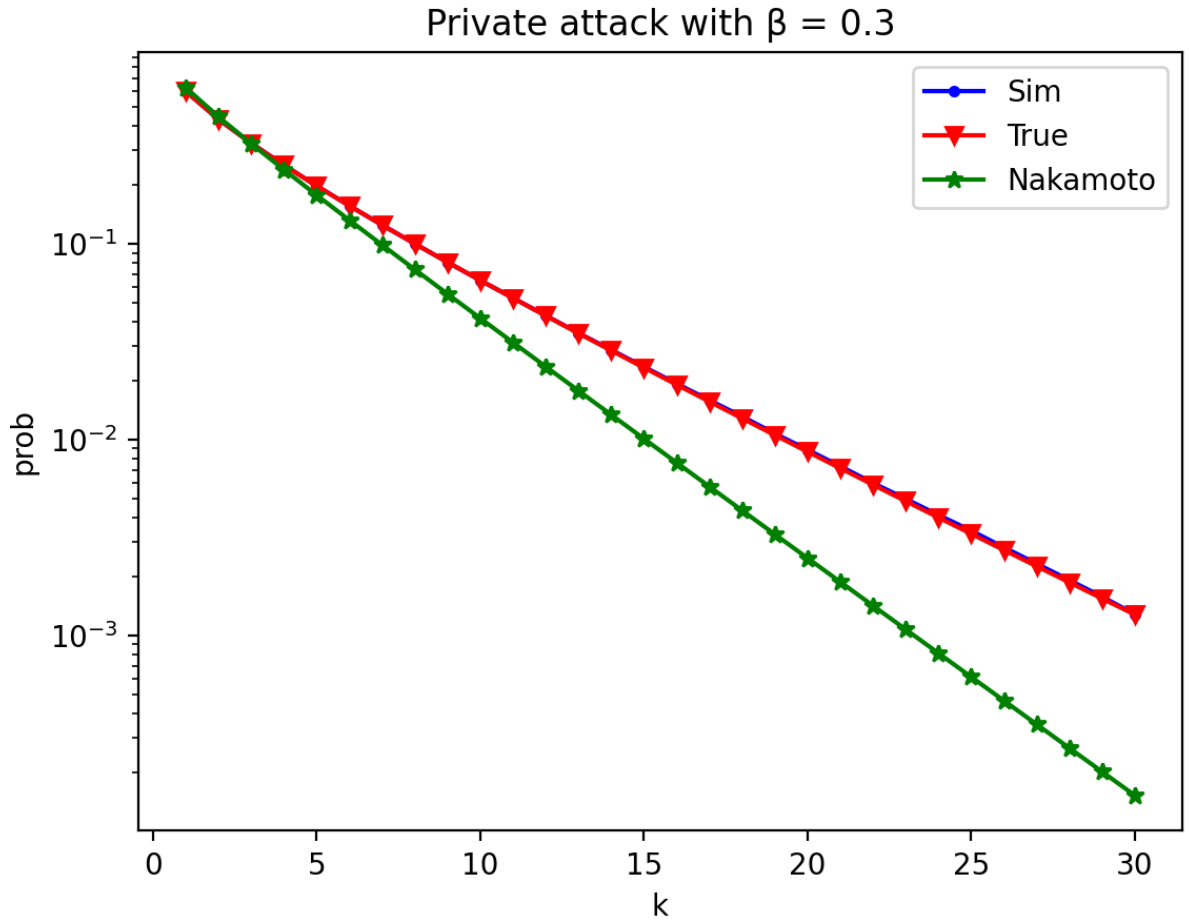
- Blocks have a maximum size limit hence can fit limited number of transactions
- Each transactions pays some tip to the miner who mines the transaction
- Necessary for DoS resistance
- Miners typically choose transactions with highest tip
- Transaction fees = (Input - Output)
- The fees are aggregated in the coinbase transaction

Confirmation Rule

- A block is confirmed if it is **buried k-deep in the longest chain**
- An attacker would need more than k blocks to double spend



Security vs Latency Tradeoff



Bootstrap: Initial Block Download (IBD)

- Connect to a full node (called sync node)
- Block-First IBD: Ask for all blocks in blockchain in order (using inv, getdata, blocks)
- Header-First IBD:
  - Ask sync node for header chain (using getheaders, headers)
  - Upon receiving the longest header chain, ask other peers for their longest chain
  - Ask peers for complete blocks(using getblocks) starting for genesis
  - Getblocks can be done in parallel

Scripting language

- Transactions more complex than input/output (sending money)
- May want to create complex contracts (like escrow)
- Bitcoin provides a scripting language
  - Stack based
  - Not Turing complete: No loops/recursion
  - 256 OPCODES
- Limited to prevent DoS attacks by transactions spending too much compute
- Ethereum proposed EVM state machine to allow loops, compute costs money(gas), hence avoids DoS

Simple Payment Verification

Lite Client

- SPV node
- Only wants to verify a transaction

Download chain of block headers from main node

- header: nonce, prev hash, Merkle root
- periodically poll main nodes
- update if there is a longer chain

SPV Operations

Transaction inclusion in a block

verify Merkle proof

Integrity of Block

Need to verify transaction with respect to UTXO state

Use block depth as proxy

Honest nodes need to control the network

Attendance : NFT Drop

https://poap.website/total-democrat-that

- Mint token to Metamask.
- Submit tx hash for attendance claim.

- Instructions in Ed pinned posts.