Cryptocurrency Mining: Proof-of-Work Consensus

Nadir Akhtar
LECTURE OUTLINE
Decal Lecture 2

1. Intro and Terminology
2. Bitcoin and Consensus
3. Bitcoin Concepts
4. Mining Sketch
5. Bonus Content: Merkle Trees and Consensus Updates
INTRODUCTION
TERMINOLOGY

Intro and Basic Concepts

Bitcoin is the technology that started it all
- Bitcoin is a cryptocurrency

Blockchain is the technology underlying Bitcoin
- Enables distributed consensus

Community terminology
- "crypto", "cryptocurrency" - Bitcoin, Ethereum, more technical
- "private blockchains", "permissioned ledgers", or just "blockchain"
- "distributed tech" or "decentralized tech" - umbrella term
2 BITCOIN AND CONSENSUS
SATOSHI NAKAMOTO'S INNOVATION

Decal Lecture 2

Bitcoin was created by Satoshi Nakamoto in 2009
- Decentralized, trustless system for transactions
  - A low cost financial system that only requires an internet connection
- Nakamoto solved the Double Spending problem
  - Prevent someone from spending the same asset twice
  - Solution? The blockchain + Proof-of-Work

Dorian Satoshi Nakamoto
(not actually Satoshi Nakamoto)
v1

Alice writes and signs a message describing her transaction

“I, Alice, am giving Bob one bitcoin.”
Alice sends her message to the world
Alice sends five identical messages
v2
Introducing uniquely identifiable serial numbers
v2
Where do serial numbers come from?
v2
A central bank manages transactions and balances
Centralization

v2
v3

Making everyone the bank.
Everyone has a complete record of transactions
v3

Alice sends her transaction to Bob
v3
Bob announces the transaction to the world
v3
Alice double spends on Bob and Charlie
v4
Everyone verifies transactions
v4

Alice is prevented from double spending
v4

Alice sets up multiple identities
v4

Alice double spends with her multiple identities

**Sybil Attack**: Creating many fake identities to subvert a system
v5
Proof-of-work
v5

Other users add to list of pending transactions

1. I, Tom, am giving Sue one bitcoin, with serial number 3920.
2. I, Sydney, am giving Cynthia one bitcoin, with serial number 1325.
3. I, Alice, am giving Bob one bitcoin, with serial number 1234.
v5
Verifying transactions

1. Check blockchain
2. Solve puzzle
3. Announce block

Slide by Viget
v5
Why the math?
v4

Alice double spends with her multiple identities
v5

Proof-of-work as a competition
## Summary

<table>
<thead>
<tr>
<th>Version</th>
<th>Major feature</th>
<th>Value added</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Signed messages announced to the network</td>
<td>Basis of entire system</td>
</tr>
<tr>
<td>2</td>
<td>Serial numbers</td>
<td>Uniquely identifiable transactions</td>
</tr>
<tr>
<td>3</td>
<td>The block chain</td>
<td>Shared record of transactions</td>
</tr>
<tr>
<td>4</td>
<td>Everyone verifies transactions</td>
<td>Increased security</td>
</tr>
<tr>
<td>5</td>
<td>Proof-of-work</td>
<td>Prevents double spending</td>
</tr>
</tbody>
</table>
BASIC CONCEPTS - IDENTITY IN BITCOIN

Decal Lecture 2

- Send money between pseudonyms
  - pseudonym == address == public key
- Cryptographic primitives
  - digital signature scheme (ECDSA: Elliptic Curve Digital Signature Algorithm)
    - public key/private key pair; like email address + password
  - one-way hash function (SHA-256)
- Bitcoin is hidden in the large amount of public keys
  - Users can generate arbitrarily many key pairs
  - Example Address: 1FtQU9X78hdshngJiCBw9tbE2MYpx87eLT
  - $2^{160}$ possible addresses
    - (1,461,501,637,330,902,918,203,684,832,716,283,019,655,932,542,976 addresses)
  - Grains of sand on earth: $2^{63}$
  - $2^{126}$ is actually only 0.0000000058% of $2^{160}$
• Bitcoin exists as software
  ○ Transactions are conducted through wallet software
  ○ Wallet creation generates a Bitcoin address
• To receive money, you share your address
  ○ Sender specifies address and amount
• The transaction is broadcast to the network, where "miners" verify it and add it to the transaction history
Account-based
- must track every transaction affecting Alice
- Requires additional maintenance, error-prone

Bitcoin is a transaction-based ledger (triple-entry accounting).
Features:
- Change addresses - Required since tx outputs only spent once
- Efficient verification - only read recent history
- Joint payments - Alice + Bob form 1 tx

(Credit for content organization and figures goes to Princeton textbook)
Mineral Sketch

Decal Lecture 2

Notes on **Proof-of-work (PoW)**
- Proof-of-work is the solution to the mining problem
- Proof-of-work is an example of a "Byzantine consensus algorithm"
- Proof-of-work is one of a plethora of consensus algorithms
- Private blockchains tend to use alternative algorithms, but are not completely trustless

**Mining** functions as:
- A **minting mechanism** that ensures coins are distributed in a fair way
- An incentive for people to help secure the network
- Key component that enables you reach consensus in a decentralized currency
MINING SKETCH - FINDING BLOCKS

Decal Lecture 2

- Finding the PoW => 'found' a block; can add block to blockchain
  - Miner who found block adds "coinbase transaction"
    - contains mining reward (currently 12.5 BTC)
  - Miner broadcasts block
  - Other nodes verify, then add to their own copy of the blockchain
- Timeline + stats
  - This happens roughly every 10 minutes
    - Difficulty of the problem adjusted every 2 weeks
  - Block reward halving every 4 years (last halved on July 9th)
    - Bitcoin is in limited supply - 21 million bitcoins by 2140
      - Deflationary
  - Each block can only contain 1 MB worth of transactions
  - 15.2 million bitcoins currently in circulation today
  - ~$9.6 billion market cap
  - Price is currently ~$920 per bitcoin
WHAT A MINER DOES

A Bitcoin miner must:

1. **Download** the entire Bitcoin blockchain to store the entire transaction history
2. **Verify** incoming transactions by checking signatures and confirming the existence of valid bitcoins
3. **Create** a block using collected valid transactions
4. **Find** a valid nonce to create a valid block header (the “mining” part)
5. **Hope** that your block is accepted by other nodes and not defeated by a competitor block
6. **Profit**!
GENERAL TIPS FOR GOOD SLIDES

- Highlight (bold, underline, or yellow highlight in B@B yellow) key words that people ought to remember. Because if people even remember what few words or phrase links to the topic, it's often enough to come up with a definition on their end. Readability is king for slides!

- Do some testing on readability, not just legibility (font size is good) when you're done with the slides. This is simply based on how things are laid out (bullets, highlights, bolds, more white space - these are things that help), so pop it up on the projector and go through it quick, maybe like how efficiencymaster would...
GENERAL TIPS FOR GOOD SLIDES
- Highlight (bold, underline, or yellow highlight in B@B yellow) key words that people ought to remember. Because if people even remember what few words or phrase links to the topic, it’s often enough to come up with a definition on their end. Readability is king for slides!
- Do some testing on readability, not just legibility (font size is good) when you’re done with the slides. This is simply based on how things are laid out (bullets, highlights, bolds, more white space – these are things that help), so pop it up on the projector and go through it quick, maybe like how efficiency master would...
BLOCK DIFFICULTY: ANALOGY

DECAL LECTURE 5

- Mining is like throwing darts at a target while blindfolded:
  - Equal likelihood of hitting any ring
  - Faster throwers ⇒ more hits / second
  - Target: within green ring
  - Difficulty inversely proportional to green ring size
    - Green ring adjusts depending on average time to produce valid result
  - If people get better at throwing darts, green circle needs to get smaller

\[
H(\text{nonce} \mid \text{prev\_hash} \mid \text{merkle\_root}) < \text{target}
\]
**Hash puzzles:** the requirement to find a nonce that satisfies the inequality in the lower left region beneath the target:

\[ H(\text{nonce} \ || \ \text{prev\_hash} \ || \ \text{merkle\_root}) < \text{target} \]

- **Hash puzzles need to be:**
  1. Computationally difficult.
     - If finding the proof-of-work requires little work, what’s the point?
     - That’s why we blindfold the dart-throwers.
  2. Parameterizable (variable) cost.
     - Allows for adjustments with global hashrate increases
  3. Easily verifiable.
     - Should not be a need for a central authority to verify nonce validity; instead, other miners can rehash the nonce to verify validity.
     - If darts fell out of the dartboard, how can we prove where it hit?
**BLOCK DIFFICULTY: ADJUSTMENT**

**DECAL LECTURE 5**

- Equation for difficulty:
  
  \[
  \text{difficulty} = \text{difficulty} \times \frac{\text{two_weeks}}{\text{time_to_mine_prev_2016_blocks}}
  \]

  - **Sanity check** (assume difficulty = 10):
    - What is the new difficulty when `two_weeks = time_to_mine`...?
    - How about when `time_to_mine = one_week`? When `time_to_mine = four_weeks`?

\[
H(\text{nonce} \ || \ \text{prev_hash} \ || \ \text{merkle_root}) < \text{target}
\]
Equation for difficulty:

difficulty = difficulty * two_weeks / time_to_mine_prev_2016_blocks

- **Sanity check** (assume difficulty = 10):
  - What is the new difficulty when two_weeks = time_to_mine...? (Answer: 10. Stays the same!)
  - How about when time_to_mine = one_week? When time_to_mine = four_weeks?

\[ H(\text{nonce} \mid \text{prev\_hash} \mid \text{merkle\_root}) < \text{target} \]
**BLOCK DIFFICULTY: ADJUSTMENT**

**DECAL LECTURE 5**

- Equation for difficulty:
  \[ \text{difficulty} = \text{difficulty} \times \frac{\text{two\_weeks}}{\text{time\_to\_mine\_prev\_2016\_blocks}} \]
  
- **Sanity check** (assume difficulty = 10):
  - What is the new difficulty when \( \text{two\_weeks} = \text{time\_to\_mine} \)?
    
    (Answer: 10. Stays the same!)
  
  - How about when \( \text{time\_to\_mine} = \text{one\_week} \)? When \( \text{time\_to\_mine} = \text{four\_weeks} \)?
    
    (Answers: 20 and 5. Difficulty is inversely proportional to \( \text{time\_to\_mine} \).)

\[ H(\text{nonce} \ || \ \text{prev\_hash} \ || \ \text{merkle\_root}) < \text{target} \]
MINING_REWARD = BLOCK_REWARD + TX_FEES
MINING_COST = HARDWARE_COST + OPERATING_COSTS

if MINING_REWARD > MINING_COST:
    miner.get_profit()
Components hashed together:
- Merkle Root
  - 'summary' of the transactions in the block
- Hash of previous block
- Nonce
  - Randomness of SHA-256 is useful here!

Formally:
- Output = SHA-256(Merkle Root + SHA-256(PreviousBlock) + Nonce)
- Solution (Proof-of-work): an output that contains a requisite number of leading 0 bits
  - The number of 0 bits is the **difficulty**
  - Difficulty adjusts every every 2016 blocks* to regulate block creation
    - *technically every 2015 blocks
A binary tree of hash pointers
- Blobs of data are hashed
- Hashes are hashed together

Merkle trees are a way to very efficiently commit to a large string of data and later prove that this string contains certain substrings.

To prove inclusion of data in the Merkle tree, provide root data and intermediate hashes
- To fake the proof, one would need to find hash preimages
  - Second preimage resistance meets this qualification
Transactions are leaves in the Merkle tree, includes a coinbase transaction

Two hash structures

1. Hash chain of blocks
   a. These blocks are linked together and based off of each other
      i. tamper evident

2. A Merkle tree of txs, internal to each block
   a. Detail: Merkle tree is always full - duplicate the last tx to fill in gaps

Princeton Textbook Figure 3.7/3.8
Previously, hash of:
- Merkle Root
- PrevBlockHash
- Nonce (varied value)

below some target value.

Actually two nonces:
1. In the block header
2. In the coinbase tx

Hash of
- PrevBlockHash
- Coinbase nonce (varied value)
  - Affects the Merkle Root
- Block header nonce (varied value)
What if there is no solution?

- Block header nonce is 32 bits
  - Antminer S9 hashes 14 TH/s
  - How long does it take to try all combinations?
    - $2^{32} / 14,000,000,000,000 = 0.00031$ seconds
    - Exhausted 3260 times per second

- Therefore, must change Merkle root
  - Increment coinbase nonce, then run through block header nonce again
  - Incrementing coinbase nonce less efficient because it must propagate up the tree

---

Princeton Textbook
51% ATTACKS

Decal Lecture 2

Major assumption of Bitcoin:

No more than 51% percent of the network is dishonest

An honest majority will always form the longest proof-of-work chain

51% Attack: Attempt to overwhelm the mining power of the network

51% ATTACKS – POOLS AND GAME THEORY

GAME THEORETIC PERSPECTIVE ON THE BLOCK SIZE LIMIT AND THE SECURITY OF THE BITCOIN NETWORK

Source: Martin Koppelmann presenting at SF Bitcoin Devs
Double Spend: Successfully spending the same money more than once.

Alice wants to buy an iPhone 0-day exploit from Bob on the black market for $1.5 million ~ 2350 BTC.

How can Alice double spend Bob, i.e., send the money to Bob and receive the goods while simultaneously sending the money to herself?
Double Spend - Confirmations

**Confirmations:** The number of blocks created on top of the block a txn is in.

\[ \text{= block depth} - 1 \]
Double Spend - (0)-confirmations Bob

Suppose Bob doesn’t wait for any confirmations on Alice’s transaction. He simply checks that the transaction is valid and immediately sends Alice the exploit.

Bob is vulnerable to a Race Attack!
Double Spend - \((z)\)-confirmations Bob

Clearly not secure if Bob doesn’t wait for any confirmations... What if Bob waits for \(z\) confirmations?

\([A \rightarrow B]\) transaction needs \(z\) confirmations before Bob sends the goods. In order to double spend Bob, Alice needs to mine on her private chain, mine \(z\) blocks on top of her block, then broadcast after Bob sends the goods.

Suppose Bob waits for \(z = 2\) confirmations

\(A \rightarrow B\)

Honest Network

\(z = 2\) confs, Bob sends goods

\(A \rightarrow A'\)

Alice mining blocks, withholding from honest network

Now, Alice has her money back, even though Bob already sent the goods!

Alice broadcasts her chain, which has a higher block height, so the honest network accepts Alice’s chain over the previous chain.
Suppose Bob waits for $z$ confirmations before sending the goods.

Alice has $h_A$ hash power.

The honest network has $h_H$ hash power.

Total network hashpower is then $h_A + h_H$.

The probability of the honest network finding the next block:

$$p = h_H / (h_A + h_H).$$

The probability of Alice finding the next block:

$$q = 1 - p = h_A / (h_A + h_H).$$

The honest network mines $z$ blocks, Bob sends the goods.

In the meantime, Alice has been hard at work mining on her chain. She will have an expected number of blocks mined equal to

$$\lambda = z \left( q / p \right)$$

which follows a Poisson Distribution, i.e., the probability that Alice generates $k$ blocks:

$$p_A(k) = (\lambda^k e^{-\lambda})/(k!)$$

What is the probability Alice can mine enough blocks in secret to successfully broadcast her chain with the double spend?
Double Spend - Security cont.

First, consider the related problem of Alice trying to catch up to a chain that is \( j \) blocks ahead of Alice’s chain.

What is the probability that Alice will ever catch up given an unlimited number of trials? This is while the honest network is simultaneously mining blocks on their chain.

This is an instance of the **Gambler’s Ruin** problem [1], which has probability

\[
p_c(j) = \begin{cases} 
1 & \text{if } q \geq p \text{ OR } j < 0 \\
\frac{1}{(q/p)^j} & \text{if } q < p
\end{cases}
\]

of Alice catching up if she’s \( j \) blocks behind.

Combining these two probabilities, we can compute the probability that Alice can catch up after \( z \) blocks mined on the honest chain.

We do this by considering the probability Alice mines \( k \) blocks and then manages to catch up to the honest chain which is \( z - k \) blocks ahead, summed over all possible values of \( k \).

\[
\sum_{k=0}^{\infty} p_A(k) \cdot p_c(z - k)
\]

\[
= \sum_{k=0}^{\infty} \frac{(\lambda^k e^{-\lambda})}{(k!)} \cdot \begin{cases} 
1 & \text{if } k > z \\
\frac{1}{(q/p)^{z-k}} & \text{if } k \leq z
\end{cases}
\]

Double Spend - Security cont.

To avoid infinite tail, discrete summation, we look at the inverse probability that Alice can’t catch up to the chain $j$ blocks ahead

$$= 1 - \sum_{k=0}^{\infty} \frac{(\lambda^k e^{-\lambda})}{(k!)} \begin{cases} 0 & \text{if } k > z \\ 1 - \frac{q}{p}^{z-k} & \text{if } k \leq z \end{cases}$$

$$= 1 - \sum_{k=0}^{z} \frac{(\lambda^k e^{-\lambda})}{(k!)} \cdot (1 - \frac{q}{p}^{z-k})$$

So… how many confirmations should Bob wait for, before sending Alice the goods?

Depends on how much hash power we assume Alice to control.
Double Spend - Bribing Miners

What if Alice controls more than 50% of the total network hash power?

Whenever Alice is $j$ blocks behind the honest network, she will always (in expectation) be able to catch up and out-produce the honest miners.

Therefore, the probability that Alice can successfully **double spend** with >50% hash power reaches 1!

Bribing Miners:

Alice might not physically control the mining hardware necessary to perform a double spend.

Instead, Alice can bribe miners or even entire pools to mine on her withheld chain.
Double Spending - “Gold Finger” Attack

Why would Alice not want to double spend?

If the rest of the network detects the double spend, it is assumed that confidence in the cryptocurrency and exchange rate would plummet.

If Alice isn’t staked in Bitcoin she can short the currency to profit after her attempted double spend.

What if Alice is a hostile government / adversarial altcoin / large finance institution with significant capital available?

Alice can acquire enough mining ASICs or bribe enough miners / pools to achieve >50% effective hash power.

Alice can perform a so-called “Gold Finger” attack with the objective of destroying the target cryptocurrency, either by destroying confidence in the currency with a double spend or spamming the network with empty blocks.

Ex: Eligius pool kills CoiledCoin altcoin [3]
Hash Rate

The estimated number of tera hashes per second (trillions of hashes per second) the Bitcoin network is performing.

Source: blockchain.info