Cryptocurrencies: The Mathematics of Bitcoin

Poonam Sahoo

November 2024

§1 Introduction

Cryptocurrencies have become a widely discussed topic, with increased momentum recently driven by the Trump administration's support [10] and Bitcoin reaching all-time high prices [11]. Understanding how cryptocurrencies work, what differentiates them from traditional currencies, and recognizing that they rely on cryptographic methods is essential. Using Bitcoin as an example, this paper will explore the mathematical foundations of cryptocurrencies. We will start from the basics of digital currency and slowly build up to the different features that make Bitcoin a viable digital currency system, including proof of work, the process of Bitcoin mining, and a mathematical proof for why it is infeasible to hack Bitcoin via a Gambler's Ruin problem formulation.

§2 Context: Digital Cash and Blind Digital Signatures

One aspect of cash that is typically not preserved online is third-party knowledge of a person's transactions. For example, my bank account contains statements of different places I send my money to. In order to anonymize digital currency transactions so that it is "identityless" like cash, we can use **blind digital signatures**. The motivation behind a blind digital signature is that even when a document is *concealed first* and the signer does not know its contents, the signature can be *verified* against the original document. Blind digital signatures were invented in 1982 by David Chaum [5], and can be used to provide unlinkability as well – so the blinded message that the signer sees is hard to link to an unblinded version of the message. This can be great for a situation where someone wants to transact with a bank, for example, needing a bank to approve a money transfer, but does not want the bank to know any other information about the transfer such as who they're sending the money to. This system of "digital cash" preserves some of the original aspects of the relative anonymity of cash.

Here's an example:

Example 2.1 (Blind Digital Signatures Example)

Suppose Alice has some money she wants to send to Bob in the form of a message M, and she wants the bank to approve the transaction without knowing who she's sending it to or what amount. Using RSA signatures as an example, the bank can create a blind RSA signature on M to approve the transaction as follows:

- Suppose that the bank has RSA signing triple (N, e, d) where $N = pq$ with p, q large (secret) primes, e public verification exponent, and d signing exponent satisfying $de \equiv 1 \pmod{(p-1)(q-1)}$.
- Alice picks random number $R \pmod{N}$, and uses the bank's public verification key e to compute $M' \equiv R^e M \pmod{N}$
- The bank uses their private signing key d to compute the signature

$$S \equiv (M')^d \equiv (R^v M)^d \equiv R^{ed} M^d \equiv RM^d \pmod{N}$$

- Since Alice knows R, she can compute $R^{-1}S \equiv M^d \pmod{N}$, which is the signature on her transaction M

In the example above, the bank has signed off on the transaction without ever seeing the original message M. However, this system relies on the assumption of a trusted central authority that is responsible for issuing cash and making sure that the system is secure.

§3 Motivation for Bitcoin

Bitcoin stands apart from traditional financial systems because it relies on "cryptographic proof" [6] instead of trusting a third-party centralized bank. Previously, financial transactions depended on intermediaries, like banks, to verify and manage vast amounts of information, which left room for fraud and errors. Bitcoin introduces a new paradigm: it ensures the security of transactions by making it computationally infeasible to reverse them and establishes a computational proof of transaction ordering, which protects buyers and minimizes risks without needing an intermediary.

What does this mean? In order to understand each aspect of Bitcoin more deeply, we will walk through the process of trying to design a digital currency and address different aspects of how Bitcoin works via this exercise.

§4 Proof of Intent: Signatures First!

If currency is completely digital, one important part of ensuring security is making sure that the person sending money intended to send it. A common way to scam people these days is to "accidentally" send people money via Venmo, CashApp, etc. and then demand that the reciever send their own (legitamately) earned moeny back [2].

Suppose Alice wants to send Bob a digital coin (we can call it a *Digicoin* for short). Bob can rest assured that Alice has sent it by verifying the signature on the message Alice sends. Again, this can be where **digital signatures** come in – Alice can add her private key to the transaction she is sending Bob, and Bob can use Alice's public key to verify that Alice is the one sending the Digicoin.

For Bitcoin in particular, digital signatures are appended using the **Elliptic Curve Digital Signature Algorithm (ECDSA)**. We can describe it as follows:

Theorem 4.1 (ECDSA)

Public parameter creation: A trusted party chooses a finite field $\mathbb{F}_p$, an elliptic curve $E/\mathbb{F}_p$, and a point G in $E(\mathbb{F}_p)$ of large prime order q.

For **key creation**, Alice chooses a secret signing key s, $1 < s < q-1$, computes the verification key $V = sG$ in $E(\mathbb{F}_p)$, and then publishes V.

For **document signing**, Alice chooses document d modulo q, a random element $e \pmod q$, computes eG in $E(\mathbb{F}_p)$ and publishes signatures s_1, s_2 satisfying

$$s_1 \equiv x(eG) \pmod q$$

and

$$s_2 \equiv (d + ss_1)e^{-1} \pmod q.$$

For **verification**, Bob computes s_1^{-1}, $s_2^{-1} \pmod q$ to find v_1, v_2 satisfying $v_1 \equiv ds_2^{-1} \pmod q$ and $v_2 \equiv s_1 s_2^{-1} \pmod q$. Then he can compute $v_1G + v_2V \in E(\mathbb{F}_p)$ and verify that $x(v_1G + v_2V) \equiv s_1 \pmod q$

Proof of ECDSA. Showing that the verification step verifies a valid signature, we show that $v_1G + v_2V = eG$ in $E(\mathbb{F}_p)$. We have that

$$v_1G + v_2V = ds_2^{-1}G + s_1s_2^{-1}(sG) = (d + ss_1)s_2^{-1}G = (es_2)s_2^{-1}G,$$

which upon simplifying gives us our desired eG. Therefore we have that

$$x(v_1G + v_2V) \equiv x(eG) \equiv s_1 \pmod q,$$

so the signature is valid as desired.

□

The specific elliptic curve, prime, and point used for Bitcoin signatures is called **secp256k1**, corresponding to elliptic curve

$$y^2 = x^3 + 7,$$

and prime

$$p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1.$$

More details can be found in [9].

Like before, we can view sending and receiving digital currency as a form of cryptographic messaging. However, although we have a great signature setup with the system proposed above, there are a couple of issues – Alice could send Bob duplicate messages, or other people could duplicate Alice's public message after she's sent it to forge it. We can use unique serial numbers to distinguish between different messages. But how would we create a trustworthy source of unique serial numbers? This is typically when banks come in as a trusted third party to verify both ends of a transaction.

§5 What if we were the bank?

Instead of the bank keeping track of transactions and issuing serial numbers, we want a system where everyone can hold others accountable. This calls for public records of transactions. This is where the **blockchain** comes in:

Definition 5.1 (Blockchain)**.** The blockchain is a shared public ledger that shows transactions. It consists of blocks chained in sequential order, which generates the order of serial numbers. Each block records a maximum of $2,400$ transactions.

Instead of Bob just verifying the Digicoin that Alice sends him on his own, he can get the rest of the network to also verify it with him. So instead of having some centralized public record, each person can keep their own history of transactions, track changes, and work to verify transactions that get broadcasted across the network. How do we incentivize users to do this? This is where proof of work and Bitcoin mining come in.

§5.1 Proof of Work

Since everyone has their own ledger and record of transactions, how can we determine which ledger is the correct one if there are discrepancies? At any point, we choose to accept the system with the most **computational work** in it. By computational work, we mean number of calculations and number of blocks. The motivation for this is that it becomes computationaly infeasible to the point of impossibility for fraudulent ledgers and transactions to propogate throughout the network. For Bitcoin, this is where the **SHA-256** hash function comes in.

§5.1.1 SHA-256

Definition 5.2 (SHA-256)**. Secure Hash Algorithm 256** is a cryptographic hash function which outputs a hash with 256 bits. It is s a one way hash computationally difficult to reverse without brute force calculations.

The most important property of the hash function is that it is completely random – even a small change in the message one hashes will result in an extremely different hash.

Example 5.3

The SHA-256 hash for **Poonam Sahoo** is: `d4e225018e0e6a198c795fd5b35f594dea38f25d89e9ba08a05ef9672294e9ad`. The SHA-256 hash for **Poonam Sahoo.** is: `4627b41a4843a6403f1880c63b1c599cec3481a4d9b348d465d1dfab6939cd6c`.

These strings differ by 160 bits, so only $\frac{96}{256} = 37.5\%$ of bits are the same between the two hashes!

This phenomenon is called the **avalanche effect** and is a desired quality of cryptographic hash functions [12]. Otherwise, the hash function is poorly randomized and will be easy to reverse and decode.

§5.1.2 Block Creation

Definition 5.4 (Proof of Work for Bitcoin)**.** The proof of work is the value of a nonce that, when hashed with the hash of a previous block and all of the transactions on a current block, has a result of a certain number of zeros Z in the front when hashing with the **SHA-256** hash function.

Remark 5.5. Z changes periodically so that the average block time (time in between creating new blocks) is every ten minutes.

Finding the proof of work is what creates a new block. Since proof of work involves taking the hash of the previous block, the blocks are chained together in a sequential matter. Since the SHA 256 hash function is so random, as explained before, it essentially takes an extremely large number of brute force computations to find the proof of work that matches the conditions outlined above.

The longest chain in the network will have the most proof of work invested into it – this is the chain that everyone accepts as the truth if there's any discrepancies.

With this context, it is easier to see why it is harder to fake transactions or act maliciously in this system. There aren't many ways to fake transactions to begin with due to the nature of the system (i.e. the ECDSA discussed earlier); the most likely way is for an attacker to "take back" money that they recently spent. It may be possible for an attacker to generate an alternate block that dictates transactions to his liking and add it to the chain, but because other people are also working on generating proof of work, as time goes on it becomes increasingly computationally infeasible to maintain this malicious blockchain and make it longer than the true blockchain. We will discuss this more concretely in Section 5.3.

§5.2 Bitcoin Mining

So why would anyone care about supporting and verifying other transactions in the network? To reward people who create blocks for their computational and energy expenditure, when the block is created via proof of work, the person who created the block receives some bitcoin in return. This is a **block reward** and is how increasing money supply works. This process of creating blocks and getting bitcoin as a reward is often called "bitcoin mining" in common popular speech.

The block reward decreases exponentially overtime. For example, between January 2009 and November 2012, the block rewards were 50 BTC, whereas currently they are 3.125 BTC [1]. This reward undergoes a halving every $210,000$ blocks. Hence, we can conclude that the total number of bitcoin will be

$$(50 + 25 + 12.5 + \dots)210,000 = \frac{50}{1 - \frac{1}{2}} \cdot 210,000 = 21,000,000$$

Hence, the total number of bitcoin is capped at 21 million. However, Bitcoin's inventor Satoshi Nakamoto proposed that Bitcoin miners can still earn money via transaction fees [6]. People can include transaction fees as part of the money they send to other people; they are incentivized to do this so that Bitcoin miners include their transaction on the block that they create.

§5.3 Feasibility of Hacking Bitcoin

As promised, we dive deeper into the scenario of an attacker attempting to generate an alternate malicious chain faster than the true blockchain. The race between the true blockchain and the malicious chain can be considered as a Binomial random walk where a "success" event is the blockchain being extended by one block and a "failure" event is the malicious chain being extended by one block. Since the ground truth blockchain is the longest chain with the most proof of work in it, we can think of the attacker as trying to generate an alternate chain that will eventually beat the length of the current

chain. The probability of an attacker catching up from being N blocks behind the true blockchain can be viewed as similar to a Gambler's Ruin problem.

Once we understand the classical Gambler's Ruin problem, extending it to consider the likelihood of an attacker successfully hacking Bitcoin is easier.

Theorem 5.6 (Classical Gambler's Ruin)

Consider a gambler who starts with an initial fortune of \$1 and then on each successive gamble either wins \$1 or loses \$1 independent of the past with probabilities q and $p = 1 - q$ respectively. The gambler's objective is to reach a total fortune of $\$N$, without first getting ruined (running out of money). Let q_i be the probability that the gambler wins when starting from i dollars. Then

$$q_i = \begin{cases} \frac{1-(p/q)^i}{1-(p/q)^N}, \text{when } p \neq q \\ i/N, \text{when } p = q = 0.5 \end{cases}.$$

Proof. We know, by how the problem is defined, that $q_0 = 0$ (ending the game at a loss) and $q_N = 1$. Then note for each i, we can define q_i as a recurrence relation in terms of q_{i+1}, q_{i-1} :

$$q_i = q(q_{i+1}) + p(q_{i-1})$$

Since $p + q = 1$, we also have that that we can rewrite the above equation as

$$(p)q_i + (q)q_i = q(q_{i+1}) + p(q_{i-1}).$$

Rearranging the terms on each side gives us

$$q_{i+1} - q_i = \frac{p}{q}(q_i - q_{i-1}).$$

From this equation, we can eventually derive a formula for $q_{i+1} - q_i$. First we have that $q_2 - q_1 = \frac{p}{q}(q_1 - q_0)$. For $q_3 - q_2$, we can substitute in the prior result to get that $q_3 - q_2 = \frac{p}{q}(q_2 - q_1) = \left(\frac{p}{q}\right)^2 q_1$. Finally, the general closed form is

$$q_{i+1} - q_i = \left(\frac{p}{q}\right)^i q_1.$$

We use this result to obtain the following: $q_{i+1} - q_1 = \sum_{k=1}^{i}(q_{k+1} - q_k) = q_1 \sum_{k=1}^{i} \left(\frac{p}{q}\right)^k$. Therefore, we have that $q_{i+1} = q_1 \sum_{k=0}^{i} \left(\frac{p}{q}\right)^k$. Note that this geometric series can be expressed as $\frac{1-\left(\frac{p}{q}\right)^{i+1}}{1-\frac{p}{q}}$ for $p \neq q$, and $i + 1$ for $p = q = 0.5$. Since we know $q_N = 1$, we have that $q_1 \left(\frac{1-\left(\frac{p}{q}\right)^N}{1-\frac{p}{q}}\right)$ if $p \neq q$ and $q_N = q_1(N) = 1$ if $p = q = 0.5$. From there, we can solve for q_1 and plug into our initial formula to get that

$$q_{i+1} = \begin{cases} \frac{1-\left(\frac{p}{q}\right)^{i+1}}{1-\left(\frac{p}{q}\right)^N}, & \text{if } p \neq q \\ \frac{i+1}{N}, & \text{if } p = q = 0.5 \end{cases}.$$

Hence, for any integer i, q_i can be computed as:

$$q_i = \begin{cases} \frac{1-\left(\frac{p}{q}\right)^i}{1-\left(\frac{p}{q}\right)^N}, & \text{if } p \neq q \\ \frac{i}{N}, & \text{if } p = q = 0.5 \end{cases},$$

as desired.

□

Now we can use this classical problem to estimate the likelihood of a blockchain attack.

Theorem 5.7 (Gambler's Ruin Blockchain Attack)

In our formulation, the gambler (attacker) has infinite credit, starts at a deficit of N (i.e. the malicious chain is N blocks behind), and "wins" by breaking even and equaling the length of the honest chain. Let p is the probability that an honest user finds the next proof of work and adds a block to the honest blockchain and q is the probability that the attacker finds the next proof of work and adds a block to their malicious chain. Note that $p = 1 - q$. Let P_N be the probability that the attacker successfully catches up to the honest blockchain after starting N blocks behind.

Then

$$P_N = \begin{cases} 1, & \text{if } p \leq q \\ \left(\frac{q}{p}\right)^N, & \text{if } p > q \end{cases}.$$

Proof. Gambler's ruin is typically formulated in terms of a limit on the amount of money the gambler can lose, but in this scenario the attacker has unlimited credit. We can solve this issue by letting y be the maximum amount of blocks the attacker can be behind by, then take the limit as y goes to infinity.

Formulating this as a Gambler's Ruin problem, we can state this as being equivalent to the attacker starting with y dollars, and the game ending either when the attacker reaches \$0 (a loss) or $y + N$ dollars (a win). Then, in terms of the last problem, we have that $q_0 = 0$ and $q_{y+N} = 1$. Substituting into the formula we found before, we have that

$$q_y = \begin{cases} \frac{1-(p/q)^y}{1-(p/q)^{y+N}}, & \text{if } p \neq q \\ \frac{y}{y+N}, & \text{if } p = q = 0.5 \end{cases}.$$

Note that when $p < q$, we have that

$$\lim_{y\to\infty} \frac{1-(p/q)^y}{1-(p/q)^{y+N}} = 1.$$

Further, when $p = q$, we have that

$$\lim_{y\to\infty} \frac{y}{y+N} = 1.$$

For the case where $p > q$, we can factor

$$\frac{1-(p/q)^y}{1-(p/q)^{y+N}} = \frac{(p/q)^y((p/q)^{-y}-1)}{(p/q)^y((p/q)^{-y}-(p/q)^N)} = \frac{(p/q)^{-y}-1}{(p/q)^{-y}-(p/q)^N}.$$

Then, taking the limit as y goes to infinity, we have that

$$\lim_{y\to\infty} \frac{(p/q)^{-y}-1}{(p/q)^{-y}-(p/q)^N} = \frac{-1}{-(p/q)^N} = \left(\frac{q}{p}\right)^N$$

when $p > q$.

Therefore we have successfully derived that $P_N = \begin{cases} 1, & \text{if } p \le q \\ \left(\frac{q}{p}\right)^N, & \text{if } p > q \end{cases}$.

□

Given that the number of honest users will vastly outnumber the malicious attacker, it is likely that $p > q$. In that case, the probability that the attacker will catch up from N blocks behind is $(q/p)^N$, a probability that drops exponentially as N increases. As time passes on, the gap between the length of the chains will only widen.

§6 Conclusion

Cryptocurrencies are getting more popular than ever; over the course of writing this paper, Bitcoin's price has increased by 44%. With all of the increasing relevance Bitcoin has in our society, it is more important than ever to understand the mechanics of cryptocurrency, the mechanisms that protect this decentralized system, and also its environmental impact! Global Bitcoin mining consumed 173.42 Terawatt hours of energy from 2020 to 2021; that is more energy than the energy consumption of Pakistan, which has a population of 230 million [3].

Hopefully this paper has been an illuminating and understandable introduction to the mathematics of Bitcoin. Through this paper, we covered the basics of blind digital signatures and the underlying mathematical procedures such as SHA-256 and Elliptic Curve Digital Signature Algorithms. We also formulated the likelihood of a malicious attack on the blockchain as a Gambler's Ruin problem to show how computationally infeasible it is to hack Bitcoin.

References

[1] 3Blue1Brown. *But how does bitcoin actually work?* July 2017. URL: https://www.youtube.com/watch?v=bBC-nXj3Ng4.

[2] Lynda Baquero. *Venmo warns of scam targeting users — here's how to avoiding falling victim.* Nov. 2023. URL: https://www.nbcnewyork.com/better-get-baquero/venmo-warns-of-scam-targeting-users-heres-how-to-avoiding-falling-victim/4839164/.

[3] Sanaz Chamanara, S. Arman Ghaffarizadeh, and Kaveh Madani. "The Environmental Footprint of Bitcoin Mining Across the Globe: Call for Urgent Action". en. In: *Earth's Future* 11.10 (Oct. 2023). ISSN: 2328-4277. DOI: 10.1029/2023ef003871. URL: http://dx.doi.org/10.1029/2023EF003871.

[4] CuriousInventor. *How Bitcoin Works Under the Hood.* July 2011. URL: https://www.youtube.com/watch?v=Lx9zgZCMqXE.

[5] Jeffrey Hoffstein, Jill Pipher, and J.H. Silverman. *An Introduction to Mathematical Cryptography.* en. Springer, Aug. 12, 2008. ISBN: 9780387779935. URL: https://books.google.com/books/about/An_Introduction_to_Mathematical_Cryptogr.html?hl=&id=hqwNDQEACAAJ.

[6] Satoshi Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System". In: (May 2009). URL: http://www.bitcoin.org/bitcoin.pdf.

[7] Michael Nielsen. *How the Bitcoin protocol actually works*. Dec. 2013. URL: https://michaelnielsen.org/ddi/how-the-bitcoin-protocol-actually-works/.

[8] A. Pinar Ozisik and Brian Neil Levine. *An Explanation of Nakamoto's Analysis of Double-spend Attacks*. 2017. arXiv: 1701.03977 [cs.CR]. URL: https://arxiv.org/abs/1701.03977.

[9] Certicom Research. *SEC 2: Recommended Elliptic Curve Domain Parameters*. Jan. 2010. URL: https://www.secg.org/sec2-v2.pdf.

[10] Tony Romm. *Trump eyes pro-crypto candidates for key federal financial agencies*. Nov. 2024. URL: https://www.washingtonpost.com/business/2024/11/11/trump-crypto-regulation-bitcoin/.

[11] MacKenzie Sigalos. *Over $2.8 billion bet on Bitcoin topping $90,000 as it hits all-time high*. Nov. 2024. URL: https://www.cnbc.com/2024/11/10/over-2point8-billion-bet-on-bitcoin-topping-90000-as-it-hits-all-time-high.html.

[12] Wikipedia contributors. *Avalanche effect — Wikipedia, The Free Encyclopedia*. [Online; accessed 5-December-2024]. 2023. URL: https://en.wikipedia.org/w/index.php?title=Avalanche_effect&oldid=1189887371.