Mathematics of Cryptocurrency

A conceptual overview

What's the point?

Create a digital protocol for exchanging money that doesn't rely on a central point of trust. To figure out our protocol we will first consider a group of friends borrowing money from each other. The goal is to eliminate the need for a central authority while ensuring accurate tracking of transactions so that everyone is repaid correctly.

Basic Protocol

- Have a public ledger which anyone can add lines to.
- At the end of each month, check your transactions. If you spent more than you got, put money in. If you got more than you spent, take money out.

Ledger

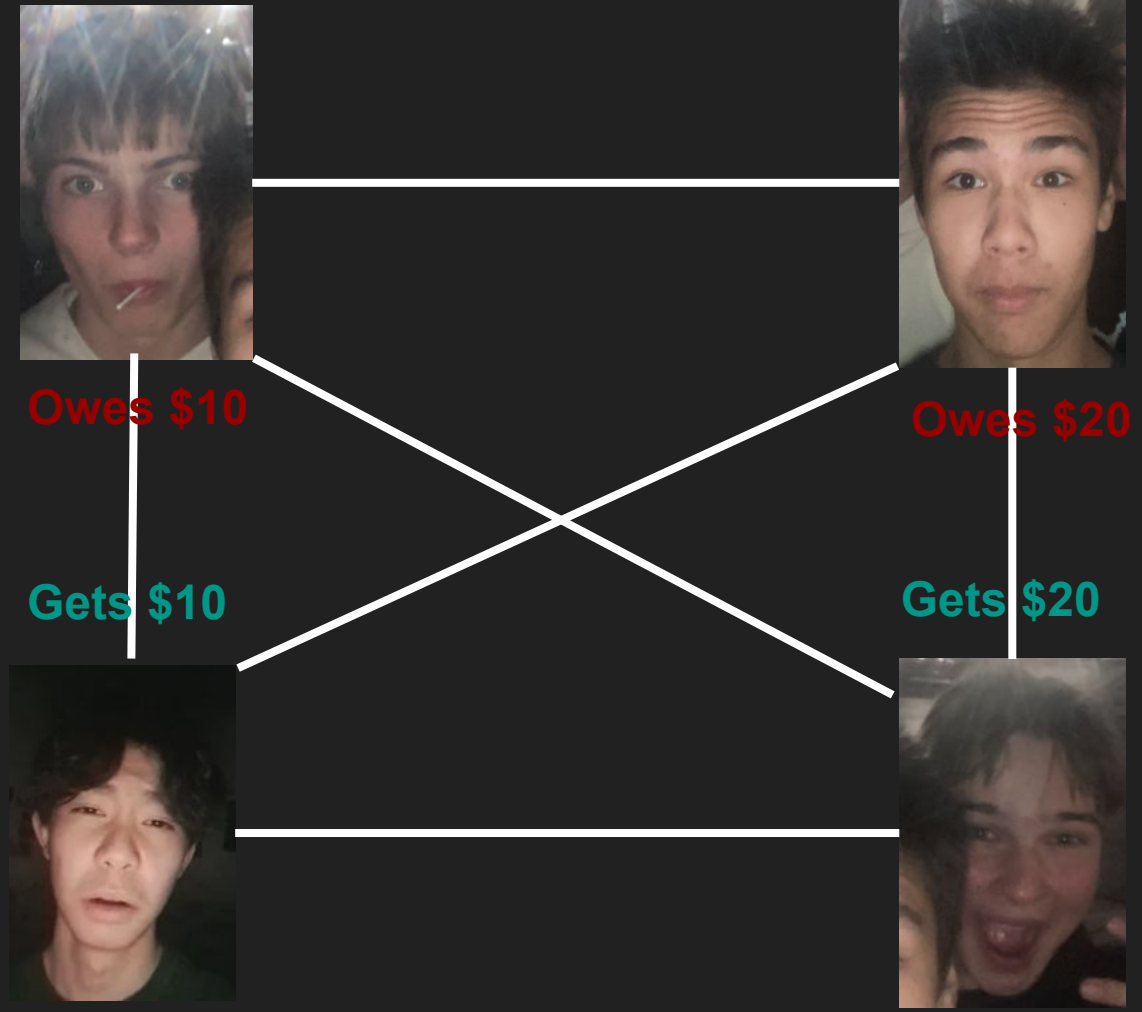
Andrei pays Markus $20

Markus pays Patrick $40

Patricks pays Syoma $30

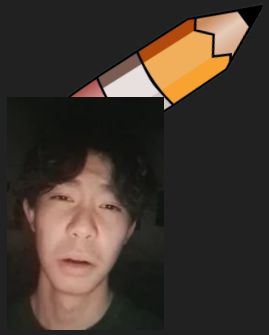
Syoma pays Andrei $10

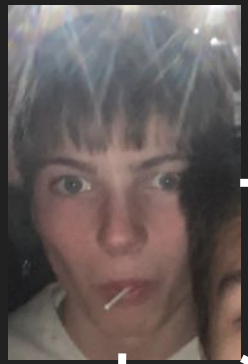
At the end of the month tally up the transactions. And put money in and out of the pot. Notice that the total debt is equal to the total profit.



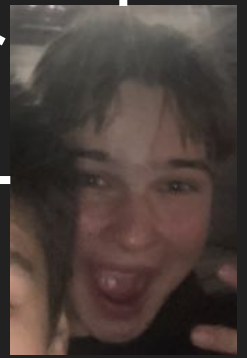
Ledger

MARKUS OWES PATRICK $1000





HEY!



Basic Protocol with Signatures

- Have a public ledger which anyone can add lines to. However any translation concerning payment from you must require a signature.
- At the end of each month, check your transactions. If you spent more than you got, put money in. If you got more than you spent, take money out.

Andrei pays Markus $20

Markus pays Patrick $40

Patricks pays Syoma $30

Syoma pays Andrei $10

Digital Signatures

- Remember our end goal is a digital protocol. You may wonder how digital signatures are possible. Since if it’s just a set of 1’s and 0’s, shouldn’t anyone be able to forge it?

Digital Signatures

- To create a digital signature you need two components which you put into a function.
  - A private/secret key which you keep to yourself (sk)
  - And the message you are signing
- Now this digital signature is only useful, if others can verify your signature.
- Thus we hand everyone an additional public key that everyone else knows (pk) which can be put into a function to verify a signature.
- Thus if we can construct these two functions we are one step closer to a decentralized systems.
  - Sign(Message, sk) = Signature
  - Verify(Signature, Message, pk) = True/False

Pk: 010000001….
Sk: 100101100….

Pk: 110110001…..
Sk: 111100101….

Digital Signatures

- Sadly we won't dive into the math of how these functions right now for the sake of time, and to focus on the big picture of cryptocurrency. The most famous of these algorithms is called RSA (which I may discuss in a later lecture) however Bitcoin uses ECDSA. The underlying mathematics of these algorithms use primes and number theory,
- Despite this, certain immediate concerns demand attention.
- If everyone can check signatures using the verification function, why can't someone just keep trying until they find yours? The sheer number of possible signatures, thanks to Bitcoin's 256-bit code (made up of 1s and 0s), prevents someone from randomly guessing and finding your signature when using the Verification function.
- To stop someone from copying your signature for a $100 transaction, we add a unique ID to each transaction. This makes every transaction have a unique signature, even if it looks the same as a previous one.

Ledger

1 Andrei pays Markus $20 67589175…

2 Markus pays Patrick $40 151501751…

3 Patrick pays Syoma $30 41480755…

4 Syoma pays Andrei $10 0273414514…

Sign(Message, sk) function

Verify(Signature, Message, pk) function.

Digital Signature protocol

- Have a public ledger which anyone can add lines to with each line having an associated id and signature created from a sign function.
- Have a corresponding verification function to verify signatures.
- At the end of each month, check your transactions. If you spent more than you got, put money in. If you got more than you spent, take money out.

Ledger

1 Patrick pays Markus $370 67589175…

2 Patrick pays Andrei $280 151501751…

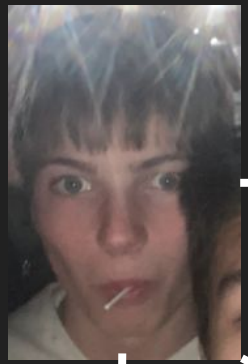
3 Patrick pays Syoma $100 41480755…

4 Patrick pays Andrei $604 0273414514…

Sign(Message, sk) function

Verify(Signature, Message, pk) function.

Issue: At the end of the month patrick just leaves.

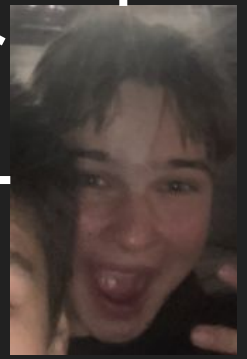


Gets $884

Gets $370

Gets $100

Owes $1354



Removing Cash

- The elimination of the requirement for in-person cash transactions is achieved by letting the ledger function independently, operating smoothly as long as individuals do not exceed their available funds.
- Initially, participants can contribute to a collective pool, enabling straightforward transactions that do not lead to overdraft situations.
- It's important to note that validating a transaction now involves examining the entire ledger history to ensure that no one goes into overdraft.
- An intriguing observation is that theoretically, if the entire global population adopts this ledger system, physical cash becomes unnecessary, and transactions would exclusively occur through the ledger. Consequently, traditional currencies like USD would be replaced by a new unit of currency, referred to as LD.
- Naturally, individuals have the liberty to exchange ledger dollars for USD, although it's crucial to note that such exchanges are not integrated into the protocol. Instead, they rely on the willingness of others to engage in the trade, akin to traditional currency exchange dynamics.

Ledger

1 Markus gets 100 LD

2 Andrei gets 100 LD

3 Patrick gets 100 LD

4 Syoma gets 100 LD

---

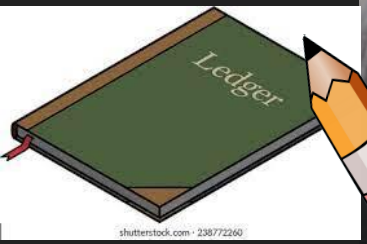
5. Syoma pays Markus 50 LD 3114174…

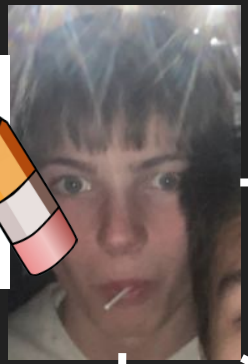
6. Syoma pays Andrei 50 LD 517059…

7. Syoma pays Patrick 20 LD 148914… OVERDRAFT INVALID

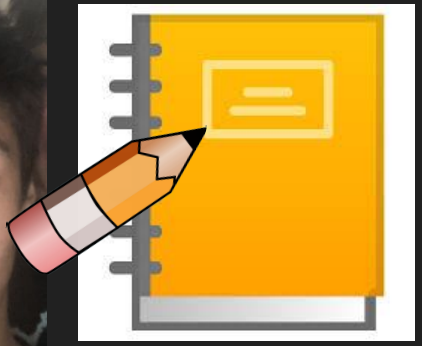
Have we achieved our goal?

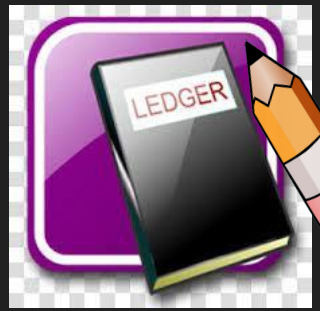
- Unfortunately, our objective remains unfulfilled. The current system hinges on a public ledger, assumed to be stored in some central location, introducing an element of trust towards the entity hosting the ledger.
- To mitigate this trust factor, we shift away from the concept of a singular ledger. Instead, each participant maintains their own copy of the ledger. To execute a transaction, the message, along with its unique ID and signature, is broadcasted to the entire network, allowing each participant to record the transaction on their individual private ledgers.

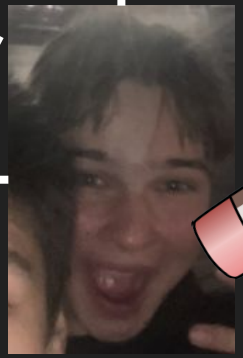




315. Markus Pays
Andrei 100 LD
873510975…







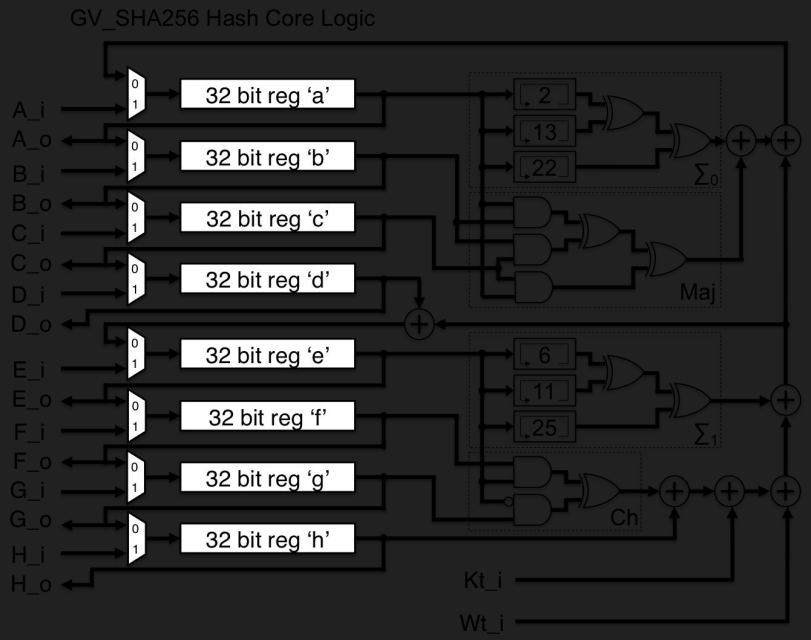
How do we ensure everyone has the same ledger?

Here lies the core of revolutionary technology in cryptocurrencies—the resolution to the ledger dilemma. A simplified explanation is that the protocol places trust in the ledger with the highest computational effort invested. This acts as a safeguard against fraud because attempting to deceive one individual into trusting a ledger different from the widely agreed-upon one would require an incomprehensibly large amount of computation, far surpassing that of the entire network. Although the concept of exploiting this vulnerability is termed a "51% attack," at this level of overview, we view it as practically infeasible. With 51% of the computational power you can send your friend money and then rewrite the ledger where you had never spent that money among other schemes such as reordering the transaction history.

Hash Functions

No one has been able to reverse SHA256

Hash functions operate by taking any message as input and producing a fixed-length string of bits known as the hash. The hash appears ostensibly random, yet it remains consistent for the same input (due to its functional nature). The key aspect of hash functions is their unpredictability—altering even a single character in the message can lead to a complete change in the hash. This cryptographic property renders the hash function entirely unpredictable and challenging to compute in reverse. In other words, given a hash like 1001111100111100...1001001110000100, there is no more efficient method than random guessing to determine the message that corresponds to that specific hash. Bitcoin uses SHA256 which gives a 256 bit hash, which again provides security as computing 2^256 guesses would take monstrously long.



My name is David

SHA256

```
10001110111010011010101011100011
00110001010010000010011100110000
00100111011110011100000010111011
00111010100101001100100110010000
01001110100111010000001010001101
00111111001000101100101001111010
00101101011111111101010011101000
10101110001000001001110101011100
```

My name is david

SHA256

```
10000101011001111001011011010110
10011011100110001001010010001110
00011100011000001101101001111110
11100111000001011111100101100000
01101101000100001110001110000011
01011110101011110100010000110111
01100111000110001100101110111000
01010000011111101110011011001100
```

My name is david wow

SHA256

```
01001111110001000110011010000110
01100000011011111111011111000010
01000100101111111111000110010010 5
00000010100100000010100100110100
11100001111010111100101001001101
10010101111001001101000010011110
11100011111110111100101001110110
01000001100100000100110001101011
```

My name is david.

SHA256

```
10000001001100001101010110111100
00011010010101111101010001101010
10011111000001111111010111101101
01000110101100001110101101101100
11010011101001111100010010101101
00110010101101000000001000111100
01000010110111100011110111000101
01011010000101101000000111001110
```

Proof of Work

We use hash functions as a computational challenge, where you input a list of transactions along with a number (initially set at 0) into the hash function. The challenge is to repeatedly adjust this number until the resulting hash begins with at least six zeroes. While achieving this is possible, it's a time-consuming process that primarily relies on guesswork. This is considered a proof of work. For example, if someone discovers a number that, when combined with the transaction list, results in 30 zeroes in the hash, the probability of this occurrence is 1 in 2^30, approximately 1 in a billion. This suggests that the person went through a substantial number of guesses (around a billion) to achieve this outcome. You can easily verify this effort by checking the hash, and we refer to the discovered number as a proof of work.

Rather than putting the entire ledger into the hash function every time we start by grouping transactions into blocks

Block 1:

Patrick pays
Markus 50 LD
<Signature>

Syoma pays
Patrick 40 LD
<Signature> …

Proof of work

Block 2:

Markus pays
Andrei 90 LD
<Signature>

Andrei pays
Patrick 40 LD
<Signature> …

Proof of work

Block 3:

Patrick pays
Markus 50 LD
<Signature>

Stoma pays
Patrick 10 LD
<Signature>….

Proof of work

Block 4:

Syoma pays
Markus 50 LD
<Signature>

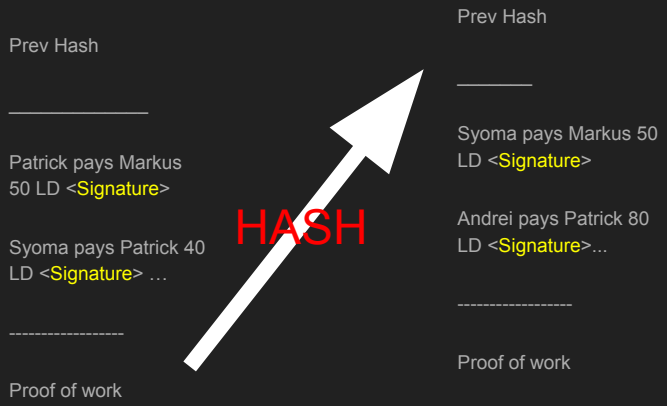
Andrei pays
Patrick 80 LD
<Signature>...

Proof of work

For now let 60 zeroes at the start of our hash be out proof of work threshold

Block Chains

To keep our blocks ordered, we add to the top of our blocks the hash of the previous proof of work. Because of this chain, instead of calling it a ledger, the ledger is commonly called a blockchain.



Block Creators (“Miners”)

Creating blocks for the protocol involves a significant amount of computational effort. The question arises: why would anyone participate in this process? Miners play a crucial role by actively listening for all broadcast transactions, gathering them into a block, and engaging in the challenging task of finding the proof of work number, typically requiring the hash to start with 60 zeros. As a reward for their efforts, the block creator earns a block reward, achieved through a special transaction that grants them ledger dollars. From the perspective of a crypto miner, each block represents a lottery, where miners rapidly guess to reach the zero threshold.

As we navigate through this, everyone keeps track of the generated blocks. In the event that two miners broadcast distinct block chains, a simple rule applies: trust the longer one more, as it signifies that more computational work has been invested in its creation.

Patrick tries to defraud Andrei.

Suppose Patrick aims to acquire something from Andrei for 1000 LD without actually spending the money. In this scenario, Patrick may try to send Andrei a block containing the line "Patrick pays Andrei 1000 LD" without disseminating the block to the wider network. Consequently, Andrei believes he has received payment, provides Patrick with the item, while Patrick retains the funds in everyone else's blockchain copy. The deception becomes apparent when Andrei, consulting the collective ledger, realizes he cannot validly spend those 1000 LD. However, this does not go to plan for Patrick.

For Patrick to defraud the system, he would need to accomplish a proof of work before any other miners—an exceptionally rare occurrence, though not impossible. Even if Patrick manages to transmit the fraudulent information to Andrei, other miners will still be broadcasting their blockchains to him, creating conflicting chains—one from Patrick and another from everyone else. To persist with the deception, Patrick must create a longer chain than all other miners, continuously misleading Andrei. However, due to the combined computational resources of other miners, they can confirm new blocks faster than Patrick. As Patrick's computational power is far from 50% of the total network, his fraudulent blockchain copy eventually becomes significantly shorter than everyone else's. Consequently, Andrei rejects Patrick's deceptive claims.

Final notes

Explore blockchains by visiting https://www.blockchain.com/explorer.

It's worth noting that with the advent of mining, there's no prerequisite for an initial buy-in, as funds can be generated through this process. Take Bitcoin, for example, which sets its proof of work challenge based on the number of zeros, ensuring the entire network requires approximately 10 minutes to discover a new block. This dynamic adjusts automatically to the increasing number of miners, making the hash challenge more demanding.

Moreover, every 210,000 blocks, the miner reward, which started at 50 bitcoin, undergoes a halving process, currently standing at 6.25 bitcoin as a block reward. Due to this halving process you can compute with the fact that we have a geometric series, that there will not be more than 21 million bitcoin in existence. While the 6.26 bitcoin reward might seem substantial individually, many individuals contribute to mining pools where they collaborate, share the rewards, and mitigate the effort required. Miners may also charge a transaction fee, prioritizing inclusion of your broadcast in the block they are attempting to validate through proof of work. Given Bitcoin's slower transaction speed, users might find it beneficial to offer a fee to a sizable mining pool to ensure their transaction is included in the blocks being validated, in addition to the regular transaction fee.

Sources

Heavily inspired by https://youtu.be/bBC-nXj3Ng4