Building Bitcoin 2.0: Decentralized Corporations, Self-Enforcing Contracts and Universal Asset Exchange

Vires in Numeris

Vitalik Buterin

November 7, 2013

What is Bitcoin?

- Decentralized, peer-to-peer online digital currency
- Can be sent nearly instantly from anywhere to anywhere around the world
- $0-$0.03 transaction fee (compare Paypal: 2.9%-6.8% + $0.30)
- No trusted authorities - no one has the power to block or reverse your payments
- Higher degree of privacy than existing systems
- Used by hundreds of thousands of people around the world, right now

History

Oct 2008 - Satoshi Nakamoto publishes "Bitcoin: A Peer-to-Peer Electronic Cash System"

3 Jan 2009 - The Bitcoin network goes live

May 2010 - Guy buys a pizza for 10,000 BTC - first Bitcoin sale

Mar - Nov 2011 - First Bitcoin bubble, price spikes from $0.7 to $31.9 back to $2

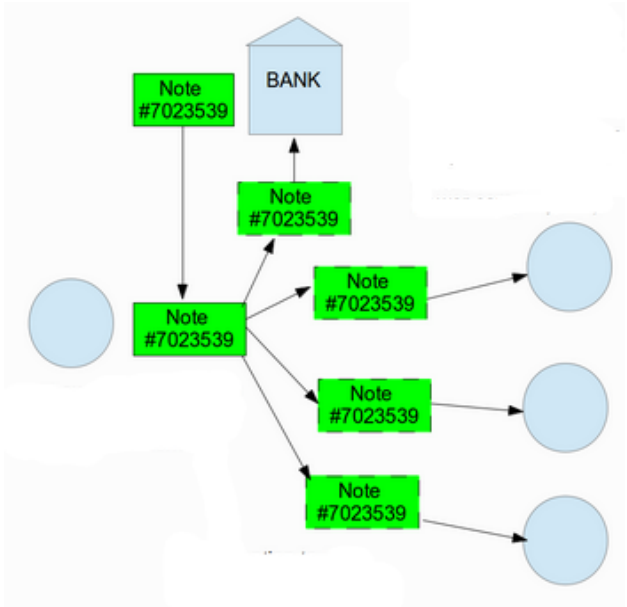
2012 - Bitcoin price stable at $5-$13, Bitcoin business attention grows

Jan - Apr 2013 - Second Bitcoin bubble, price spikes from $13 to $266 back to $50

Oct 2013 - Chinese interest in Bitcoin expands massively, start of third bubble (now $270)

Double spending: problems and solutions

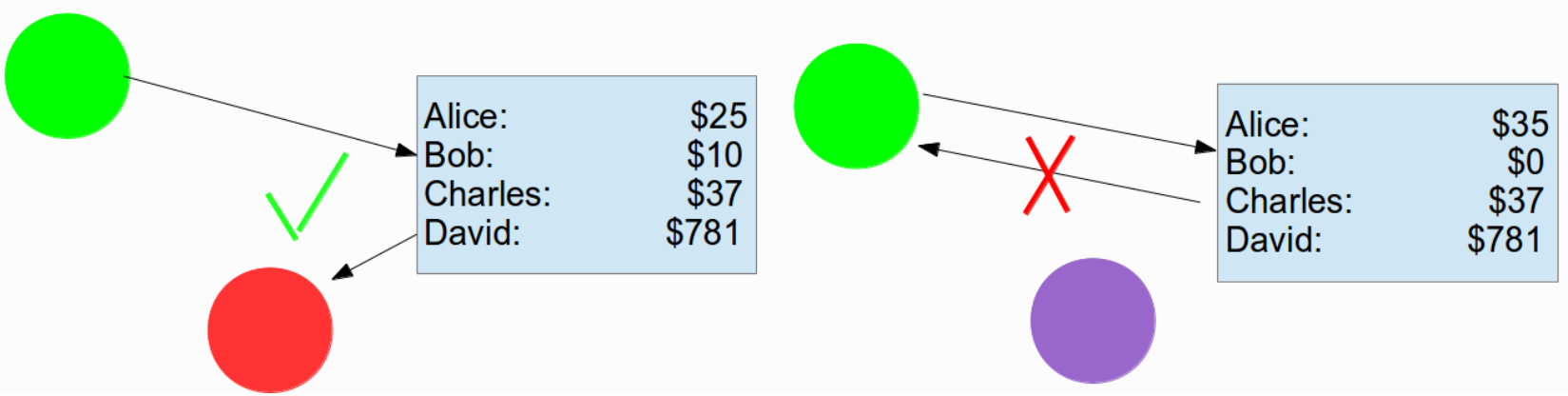
Physical currencies like dollars, euros and gold are rivalrous goods - if I have a given dollar that means that you don't have it.



Digital items, however, are non-rivalrous - if I file-share a song to you, I still have that song. This is good for music (despite what the RIAA thinks), but it makes for very bad money.

Solution I: Centralization

Idea: use a centralized database to keep track of balances

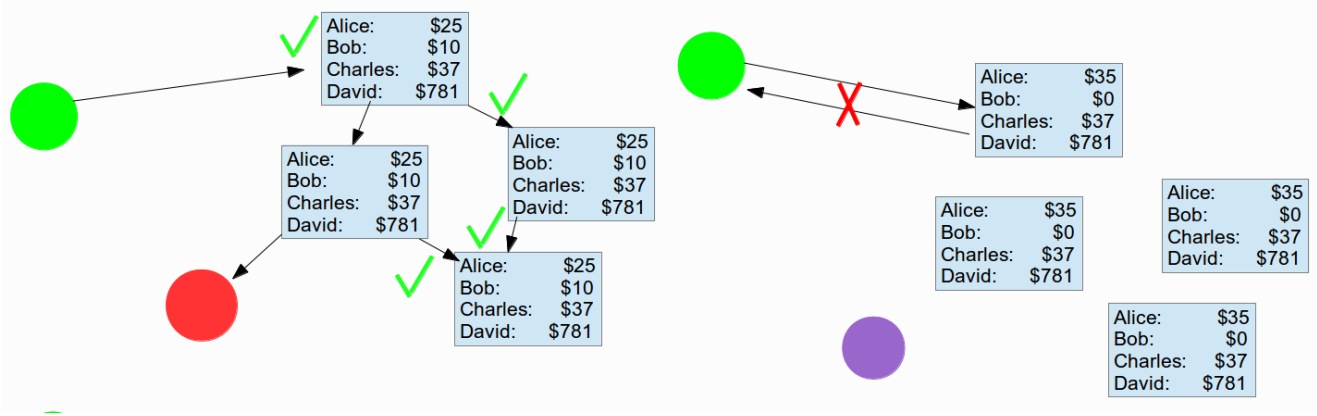


One key problem: trust

- Trust the database to respect users' privacy
- Trust the database not to manipulate account balances (inflation risk and theft risk)
- Trust the database not to shut down (operator boredom risk and regulatory risk)

Solution II: A Peer-to-Peer Database

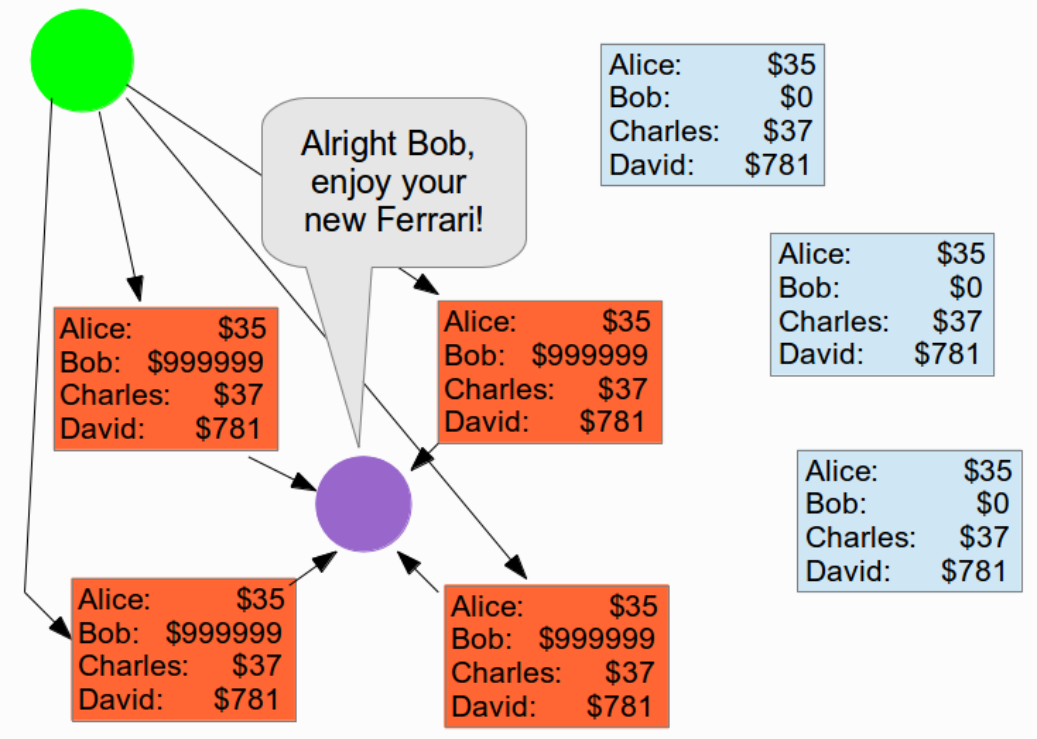
So, the obvious solution is: make the database a peer-to-peer network.



Question: can we?

Sybil attacks

Making a network of computers that maintain a synchronized database is actually not that difficult. However, the naive solution is vulnerable to an exploit known as a Sybil attack.



Solution III: Proof of Work

No matter what we do, it's impossible to completely prevent an attacker from pretending to be the entire community. However, we can make it really hard to do so.

> "Pricing via Processing or Combating Junk Mail" - Cynthia Maork and Mino Naor, 1992: make it computationally difficult to sne an email to combat spam
>
> HashCash - Adam Back, 1997: A practical implementation of this proposal

A proof of work function must have two properties:

> It must be moderately hard to compute, preferably with easily adjustable difficulty
>
> It must be easy to verify

Hash functions

Hash functions are functions which take any input and deliver a fixed size output in a certain format. Hash functions are designed to be pseudorandom and irreversible.

For all secure hash functions, the fastest algorithm for making a value (or "preimage") that maps to a given hash is brute force. For SHA256, that's 2^{256} computations!

Preimage		Hash
The quick brown fox	$\rightarrow$	5cac4f980fed...
The quick brown fax	$\rightarrow$	4a9b37b88b27...
The quick brown fox.	$\rightarrow$	42e25dd386eb...

How Proof of Work Works

But we can relax the hash preimage problem a little to make it just hard enough to be a perfect candidate for proof of work. How? Make it only look at the first few bits.

Preimage		Hash
"Hi Susan, want to go out for lunch? | 1"	$\rightarrow$	c0716b77...
"Hi Susan, want to go out for lunch? | 2"	$\rightarrow$	e8789a77...
"Hi Susan, want to go out for lunch? | 3"	$\rightarrow$	961a730c...
"Hi Susan, want to go out for lunch? | 3188"	$\rightarrow$	facd3288...
"Hi Susan, want to go out for lunch? | 3189"	$\rightarrow$	0000f7b9...

The Bitcoin Blockchain: Intuition

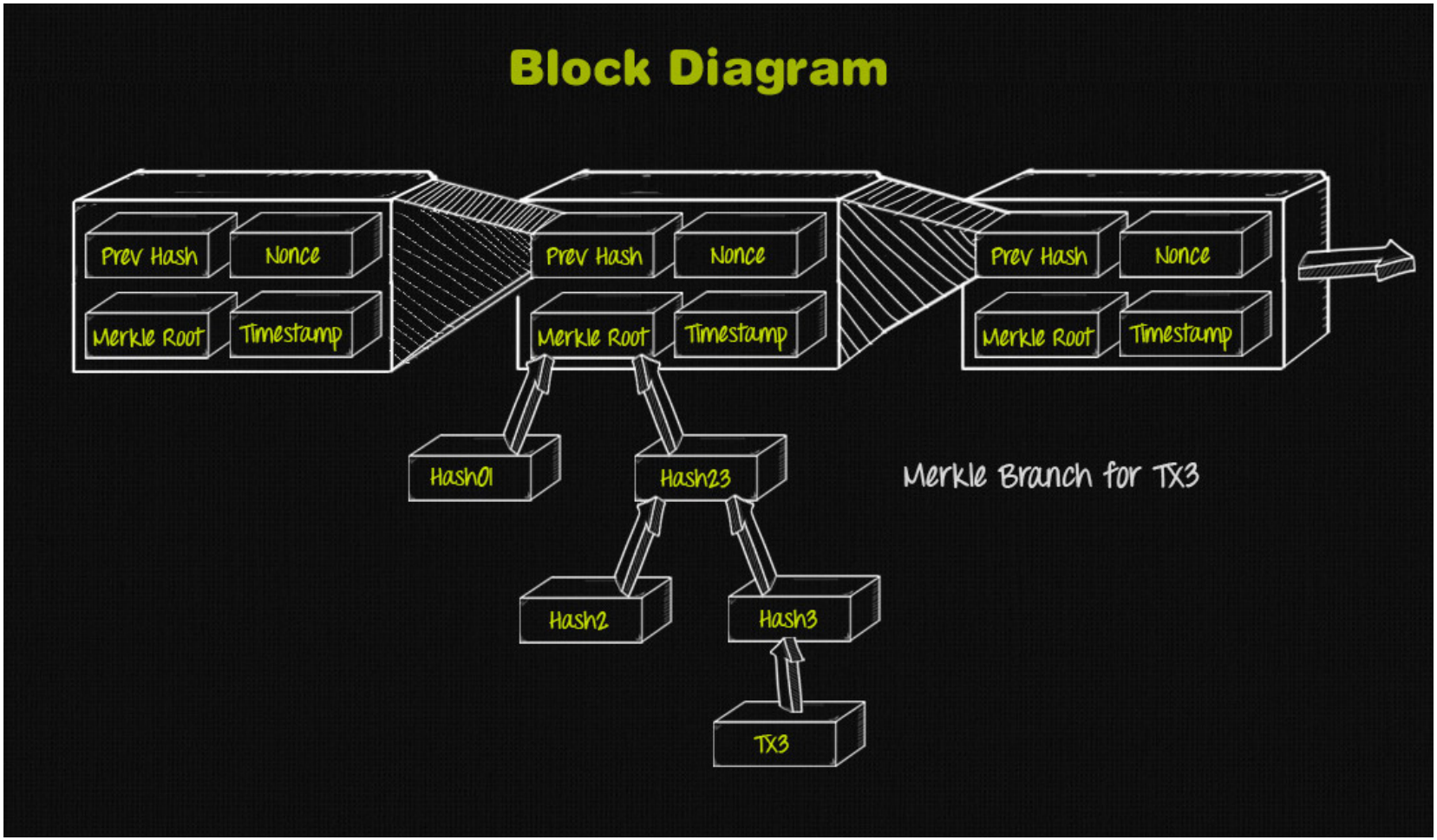
Bitcoin is a P2P network just like any other. However, instead of the number connections determining what constitutes a community consensus, it's CPU power.

The Bitcoin database takes the form of a chain of blocks, where each block takes a moderately difficult amount of work to create, and each block is cryptographically linked to the previous. The process of creating these blocks is called mining.

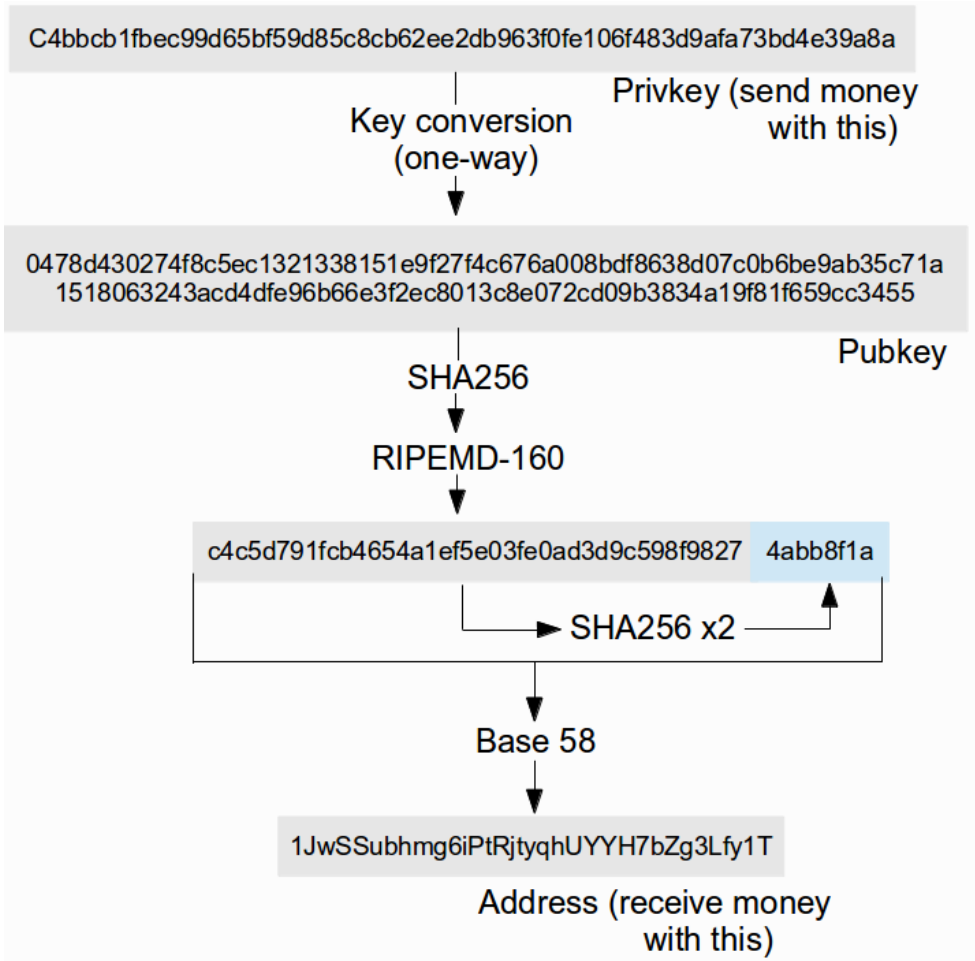
Nodes deem the longest (ie. hardest to compute in total) blockchain to be valid.

Therefore, an attacker needs to have more CPU power than the rest of the Bitcoin network combined to be able to pass off a fraudulent transaction history.

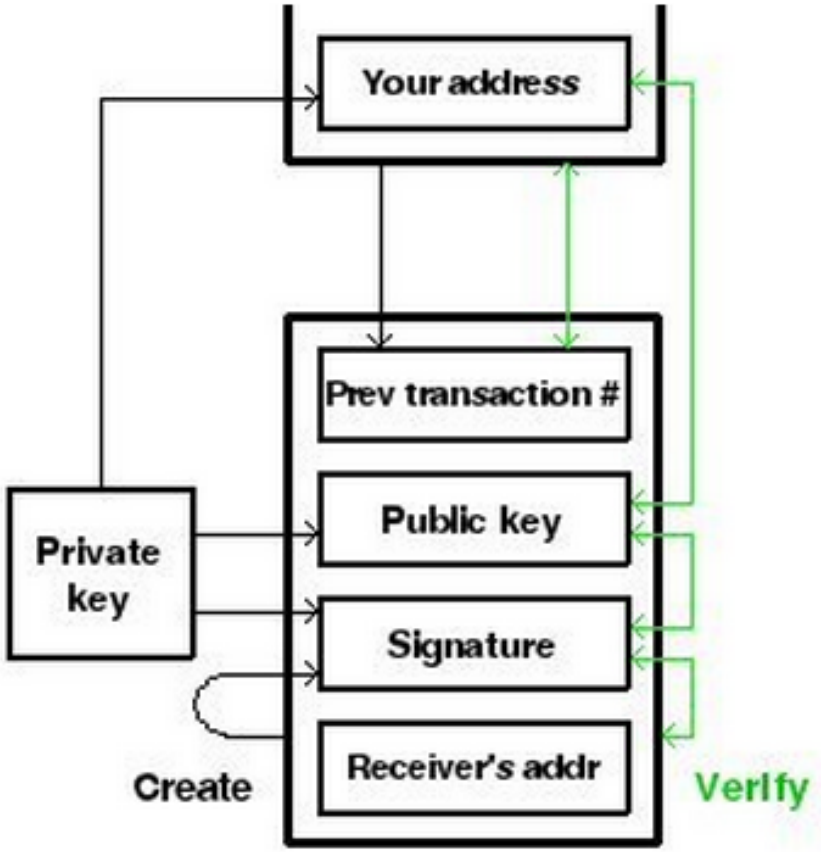
What's In A Block?



What's In A Bitcoin Address



Transactions: Putting It All Together



Namecoin

- Intuition: blockchain-based DNS
  - Can register .bit domains, or names in general
  - Impossible to censor domain names
- Problems:
  - Bootstrap: getting people to use the DNS
  - Separate blockchain - vulnerable to 51% attacks
  - Very hard to balance pricing (too high, no one buys domains, too low, squatters take over)

Proof of Stake

Idea: create a distributed consensus algorithm that does not rely on wasting energy

Algorithm:

$$SHA256(prevtx + output + timestamp) \leq \frac{value * 2^{256}}{D}$$

Every unspent transaction output has one chance per second to become the seed of a new block, proportional to the value of the output

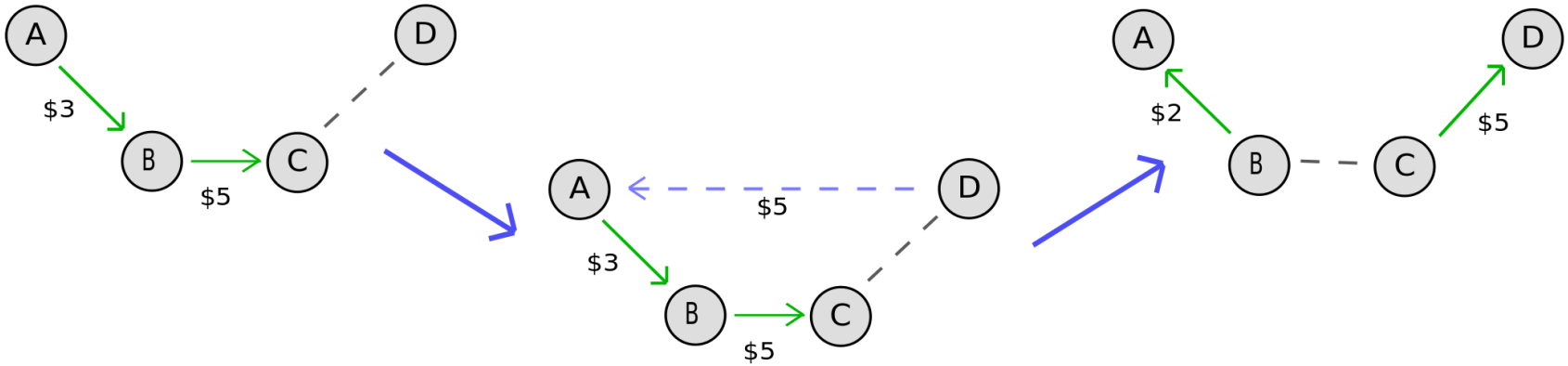
Useful Proof of Work

Primecoin: make cryptocurrency not waste electricity by making the mining algorithm useful to society (finding sequences of prime numbers)

- Are prime number sequences useful? Not really, but the research to create more efficient miners for them has side benefits
- Marginal social benefit will decrease over time
- Roadblock in making optimally useful PoW: must be easy to verify

Ripple

All money (except XRP) is debt, and all transactions are debt transfer chains:



Key advantage: store any currency on the blockchain if someone is willing to act as a backer

Mastercoin

Alternative on-blockchain protocol:

```
{
    inputs: [
        { address: '1LQBddrjjUaMLHcd4cG9XnN4cCZbHfREJF' , value: 1445759 }
    ],
    outputs: [
        { address: '1EXoDusjGwvnjZUyKkxZ4UHEf77z6A5S4P', value: 6000 },
        { address: '12ARS3euPbdQ9S68xXhmq4ySzSADfMaR1a', value: 6000 },
        { address: '1D3tBJ6b3htSaMhEV3EtTAPLvTHwLBrQPH', value: 1417759 },
        { address: '121AS7PVawbgo7f4zbkZisYEC4yhJCoUEN', value: 6000 }
    ]
}
```

`0b0000000000000001000000004042cd1d000000`

- `0b` – sequence number
- `00000000` – transaction type (regular send)
- `00000001` – currency ID (Mastercoin)
- `000000004042cd1d` – value (1078119709)

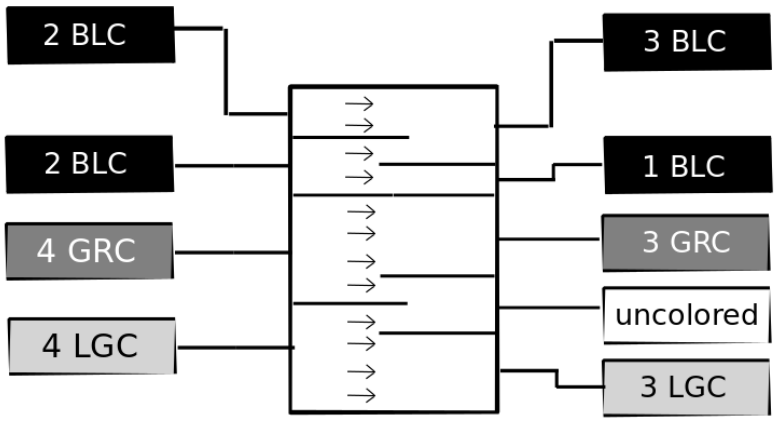
User-defined currencies

On-blockchain futures and bets

Built-in exchange functionality

Colored Coins

Label specific bitcoins as having a particular color and trace them down the blockchain



Pro: simpler to implement, more secure

Con: less powerful

Use case: Issuing stocks

Get investors from around the world

Very low friction, no commitment requires

Regulatorily easier (no central stock exchange)

Issue tradable coupons as a funding round

Provably Fair Gambling

Dice rolling algorithm:

- Every day, generate a fresh *secret*
- At the start of the day, publish $h = SHA256(secret)$
- To calculate the result of a bet, run $SHA256(txid + secret) \mod n$
- At the end of the day, publish *secret*

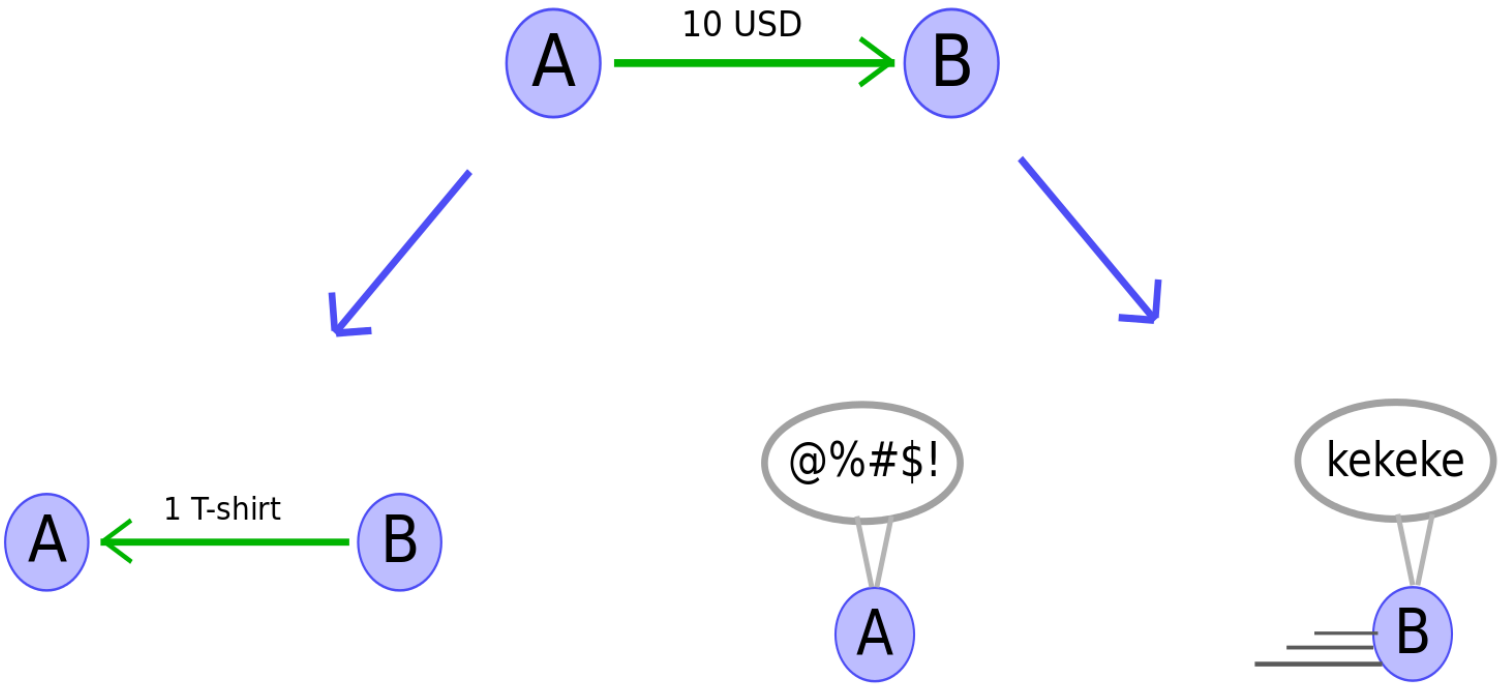
Bet verification algorithm:

- At the start of the day, download h
- At the end of the day, download *secret*
- Verify that $SHA256(secret) = h$ and $SHA256(txid + secret) \mod n =$ result of bet

Used by nearly every BTC gambling site in the world.

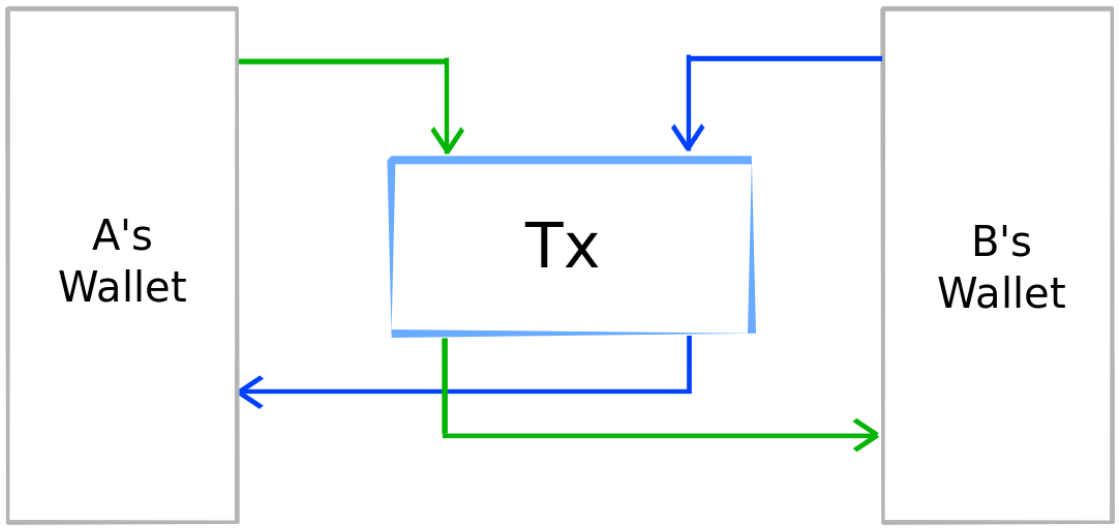
Decentralized exchange, Part 1: Atomic swaps

Exchange between any two ordinary currencies or assets requires trust:

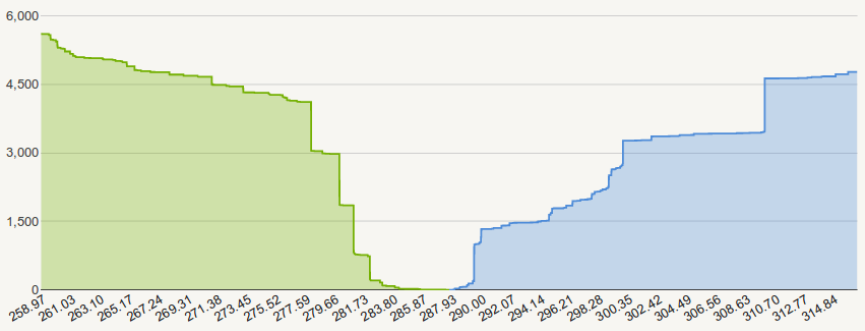


Decentralized exchange, Part 1: Atomic swaps

Exchange between two colored coins can be done "atomically" (no trust):



Decentralized exchange, Part 2: The Order Book



Weakness of colored coins protocol: if I want to sell, how do I find a willing buyer?

Typical centralized solution: order book

Solution 1: make an order book on a P2P channel. Problem: orders not enforceable, so subject to spam

Partial solution: Type 3 exchange

Solution 2: explicitly add "make order" and "accept order" transaction types (Ripple and Mastercoin)

Self-stabilizing Mastercoin currencies

Example: I register a self-stabilizing currency, call it XUSD, based on eToro's USD/MSC price feed (hypothetical)

Creating the self-stabilizing currency fund automatically creates a virtual central bank which issues it

Say 40 USD = 1 MSC. Then, I can buy 1 XUSD for 0.026 MSC from the central bank

I can also sell 1 XUSD for 0.024 MSC, as long as the central bank has MSC left

Say MSC rises to 50 USD. Central bank now buys at 0.0192 and sells at 0.0208.

Market anticipates central bank action, so central bank does very little. Currency mostly self-stabilizes around 1 USD.

Decentralizing the Price Feed

Proof of work voting: block miner has the right to publish a price reading

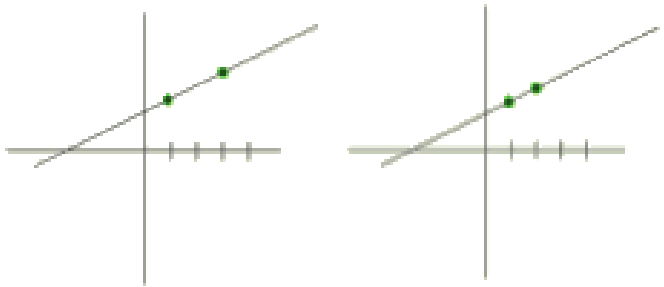
Proof of stake voting: every block, a random MSC holder has the right to publish a price reading

Take median of last 10 to avoid attacks

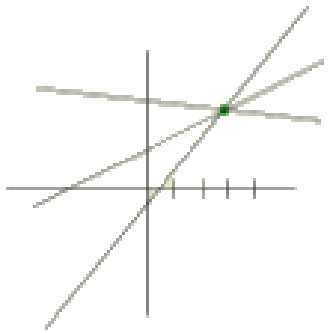
Unsolved problem: are there perverse incentives?

Secret Sharing, Part 1

Any two points on a line can be used to reconstruct the entire line:

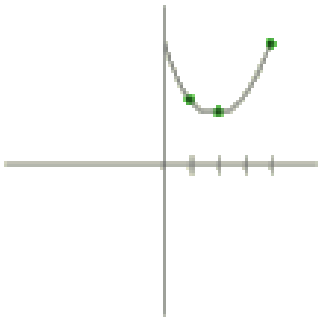


But one point can't:



Secret Sharing, Part 2

Any three points can recover a parabola:



General idea: encode a secret between k of n parties as n points on a random k-1 degree polynomial, where the secret is the y-intercept.

Reconstruction

Reconstruction equation (2-of-n):

$$L(x) = \frac{x - x_2}{x_1 - x_2} * y_1 + \frac{x - x_1}{x_2 - x_1} * y_2$$

Reconstruction equation (3-of-n):

$$Q(x) = \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} * y_1 + \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} * y_2 + \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)} * y_3$$

General idea: Lagrange interpolation

Decentralized Dropbox

Naive implementation:

- Split data across multiple servers with k-of-n secret sharing
- Each server charges for storage
- Problem: how do you prove that the server is actually storing any data?

Decentralized Dropbox 2

Stateful implementation:

- Servers have persistent identities
- Decentralized system automaticaly audits servers and pays them every N days
- System may take some fee from every new server to block short-term cheating attacks
- Problem: decentralized system needs a decentralized wallet

Secure Multiparty Computation

- Notice: Secret sharing is additive (ie. share k of a + share k of b = share k of a+b)
- More complex protocols exist for multiplication and even comparison
- Arbitrary circuits can be evaluated

Idea: we can decentralize the entire auditor code across N servers

- Decentralized private key ownership (easy)
- Decentralized transaction signing (harder, but only due to a bad BTC protocol choice)
- Decentralized auditing (very hard to do in a non-gameable way)

Difficult problem: how to make this both long-lasting and safe against Sybil attacks? PoW, PoS

Far future of Decentralized Corporations

Can decentralized corporations...

- Hire employees?
- Make sure employees are not cheating?
- Withstand hostile takeover attempts?
- Detect and use third-party APIs?
- Command physical resources
- Become Skynet or Daemon?

Prelude: Zero-knowledge proofs

ZKP example: prove you know x such that $g^x \mod p = y$

1. Prover generates random r
2. Prover hands verifier $C = g^r$
3. Verifier returns randomly 0 or 1
4. If 0, prover returns r, if 1 prover returns $x + r$
5. If 0, verifier checks $C = g^r$, if 1 verifier checks $yC = g^{x+r}$

Idea: repeat the above 128 rounds. Verifier knows that prover knows x, but learns nothing else

SCIP

Proving step:

- Express any function as a circuit
- Run circuit, store circuit execution trace
- Create a linear function to check if state T+1 is a legitimate successor to state T (return 0 if success, something else otherwise)
- Run secret sharing on (T,T+1) pairs (for obfuscation), return 80 pseudorandom outputs based on hash of function

Verification:

- checker on all 80 outputs should return 0 (by linearity)

Conclusion: we have a compiler that can make a zero-knowledge proof that you have the result of any computation

Applications

Compress the blockchain validation data to a few kilobytes

Add perfect anonymity to Bitcoin

Useful PoW: mine cryptocurrencies using Folding@home

Banks can prove their own solvency

Anti-cheating for games

Provable cloud computing


Welcome to the
financial singularity