The mathematical background of atomic swaps among blockchains

János Tapolcai
Department of Telecommunications and Media Informatics
Faculty of Electrical Engineering and Informatics

Blockchain research at TMIT

- Dr. Tapolcai János
  - BME-TMIT full professor
  - MTA Lendület 2012-2017
  - Applied mathematics in telecommunications

- Dr. Ladóczki Bence
  - PhD in Distributed Computing in Kobe, Japan
    - numerical methods on massively parallel architectures (quantum monte carlo simulations)
  - Since PhD: atomic swaps, consensus mechanisms, signature schemes, finite field arithmetics

Outline

- Motivation behind cryptocurrencies
- The list of key ideas built cryptocurrencies on
  - Transactions, blockchain, consensus algorithms
- Schnorr digital signatures
- A few word about the economics
- Application example: how to exchange cryptocurrencies with atomic swaps

Centralized vs distributed bank system

Centralized

- Trust a bank
  - In fact you trust the laws
- Efficient
  - It is expensive to change your bank
- Privacy issues
  - The bank may knows a lot about their customers

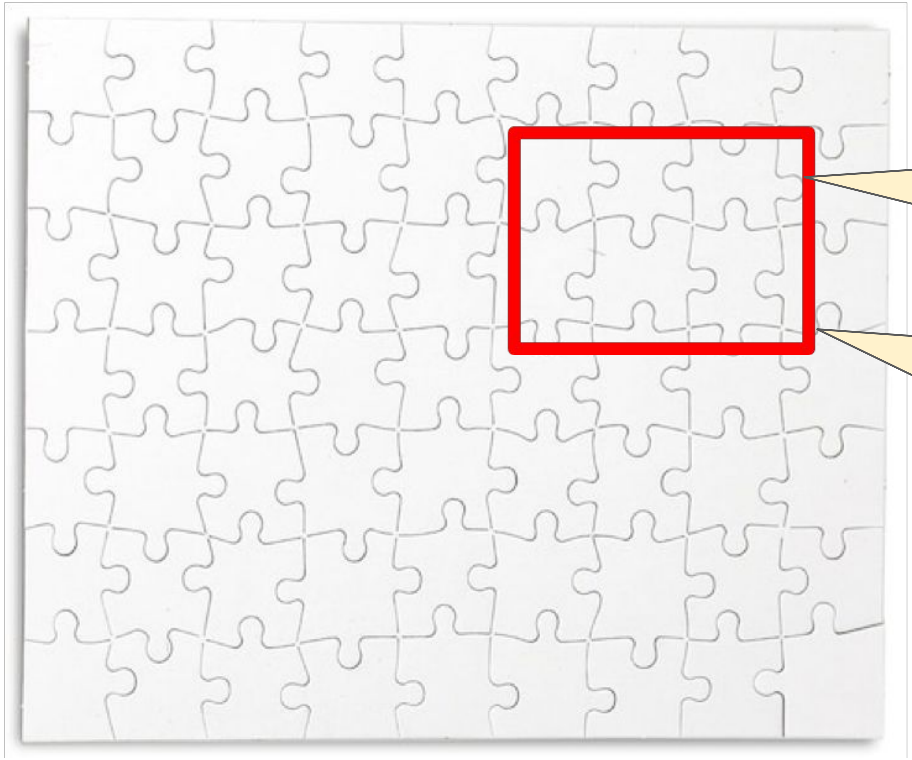
Distributed

- Do not need to trust a single entity
- There are no laws
  - Treat dishonesty as a part of the game
- Assume the majority is honest
  - A honest node follows the rule
- Expensive
  - because of the many dishonest nodes

Idea 1: Pay = show a solution of a puzzle

- Each crypto-coin is assigned with a puzzle
    - It is computationally hard to solve the puzzle
    - It is fast to verify a solution to the puzzle
    - You own the crypto-coin if you know the solution to the puzzle
    - The puzzle for each crypto–coin is stored in a ledger
- Payment
    - Show the solution to the puzzle and provide a new puzzle
    - You show only a “part” of the solution
        - the part depends on the new puzzle

Idea 2: Show the solution of a “part” of the puzzle defined by the new puzzle

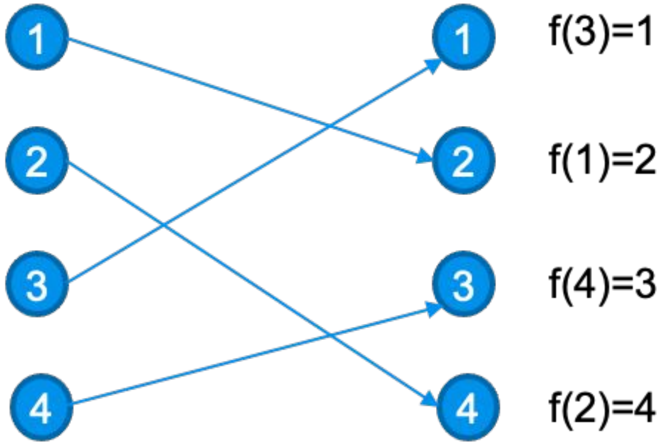


The part is defined by the new puzzle

The part is so small that basically you know nothing about the whole puzzle

What is a puzzle?

- We have n items (typically $n \leq 2^{256}$)
  - Finite field algebra
- One-way-function function $f(x)$
- Puzzle $f(x)$, the solution is x

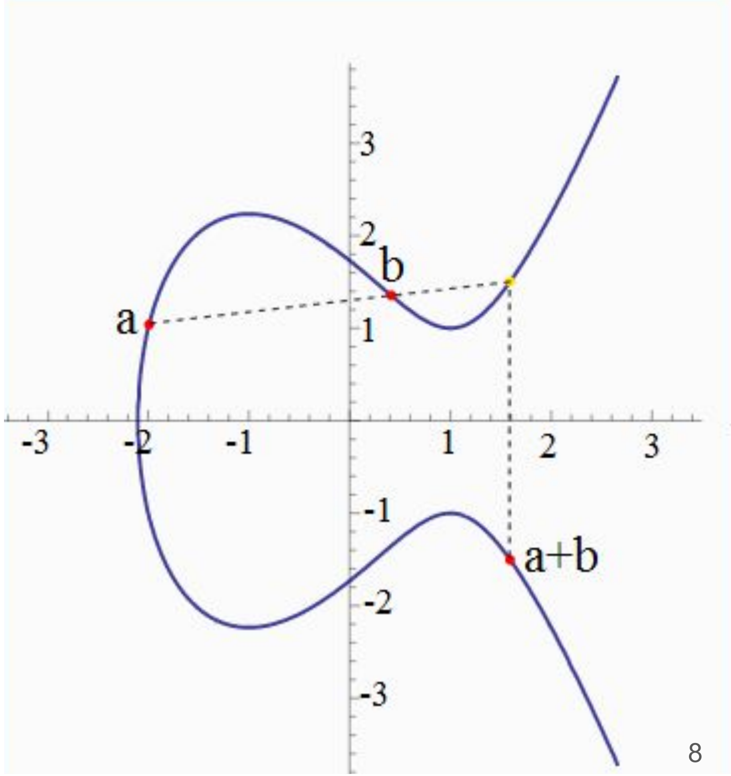


the domain is ideally all items

the co-domain is ideally all items

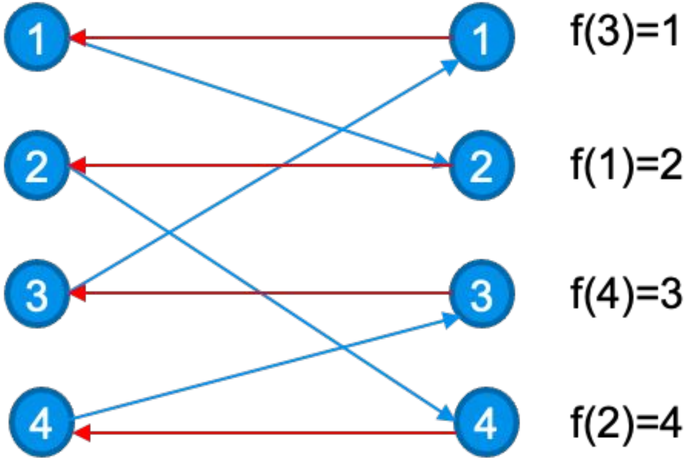
Elliptic curve secp256k1

- Most common
  - Bitcoin, Ethereum, Litecoin, Dogecoin
- Defined over the prime field $\mathbb{Z}_n$
  - $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$
- The items are (x,y) pairs on an elliptic curve
  - The curve is $y^2 = x^3+ax+b$ over F_p
    - a = 0, b=7
  - Any point on the curve can be reflected over the *x* axis and remain the same curve
  - Any non-vertical line will intersect the curve in at most three places
- Single operator
  - addition
  - There is a point *g*
  - compute $g, 2g, 3g, 4g, \ldots, ng, (n+1)g = g$
  - n is a prime



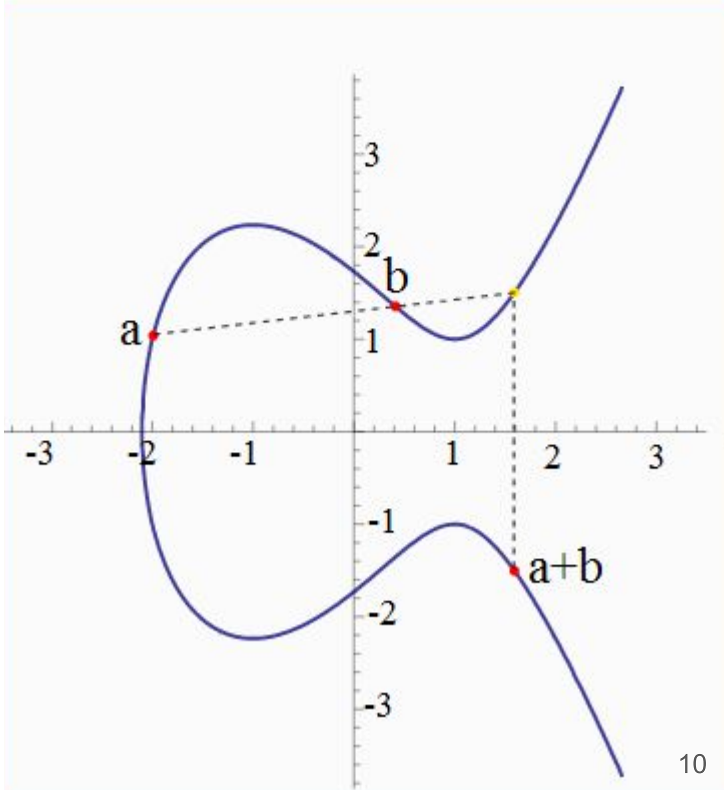
The one-way function in a finite field

- The one-way function
  - f(x)=xg
- Base point *g*
- The order of g is a bit smaller than p
  - *p* = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFC2F
  - *n* = FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE BAAEDCE6 AF48A03B BFD25E8C D0364141



Why is it a one-way function?

- The one-way function
  - *f(x)=xg*
- The algorithm to compute *f(x)*
  - exponentiation by squaring
  - compute $g, 2g, 4g, 8g, 16g, \ldots, 2^{256}g$
  - take the binary representation of *x* and multiply the corresponding powers
- The inverse function
  - discrete logarithm in this finite field is hard



Digital signature

- Given f(x), show that you know *x* without disclosing *x*
  - Disclose some information which can be verified without knowing *x*
  - However, you need to know *x* to generate it
- Furthermore it should also depend on the new puzzle
  - Digital data + digital signature
- Digital signatures used in blockchains:
  - ECDSA
    - Don Johnson, Alfred Menezes "The Elliptic Curve Digital Signature Algorithm (ECDSA)", Technical report, University of Waterloo, 1999.
  - Schnorr
    - Claus Schnorr "Efficient Identification and Signatures for Smart Cards", in Proc. CRYPTO, 1989.
    - U. S. Patent expired in 2008
  - EdDSA
    - Edwards-curve Digital Signature Algorithm (EdDSA)
  - RSA
    - Rivest–Shamir–Adleman

Prime fields

`edward25519` as

$$\begin{aligned}
q &: \text{a prime number; } q = 2^{255} - 19 \\
d &: \text{an element of } \mathbb{F}_q;\ d = -121665/121666 \\
\mathcal{E} &: \text{an elliptic curve equation; } -x^2 + y^2 = 1 + dx^2y^2 \\
G &: \text{a base point; } G = (x, -4/5) \\
l &: \text{the base point order; } l = 2^{252} + 27742317777372353535851937790883648493
\end{aligned} \tag{1}$$

`secp256k1` as

$$\begin{aligned}
p &: \text{a prime number; } p = 2^{256} - 2^{32} - 977 \\
a &: \text{an element of } \mathbb{F}_p;\ a = 0 \\
b &: \text{an element of } \mathbb{F}_p;\ b = 7 \\
\mathcal{E}' &: \text{an elliptic curve equation; } y^2 = x^3 + ax + b \\
H &: \text{a base point; } H = \\
&\big(\texttt{0x79BE667EF9DCBBAC55A06295CE870B07029BFCDB2DCE28D959F2815B16F81798}, \\
&\ \texttt{0x483ADA7726A3C4655DA4FBFC0E1108A8FD17B448A68554199C47D08FFB10D4B8}\big) \\
n &: \text{the base point order; } n = 2^{256} - 432420386565659656852420866394968145599
\end{aligned} \tag{2}$$

Schnorr signatures

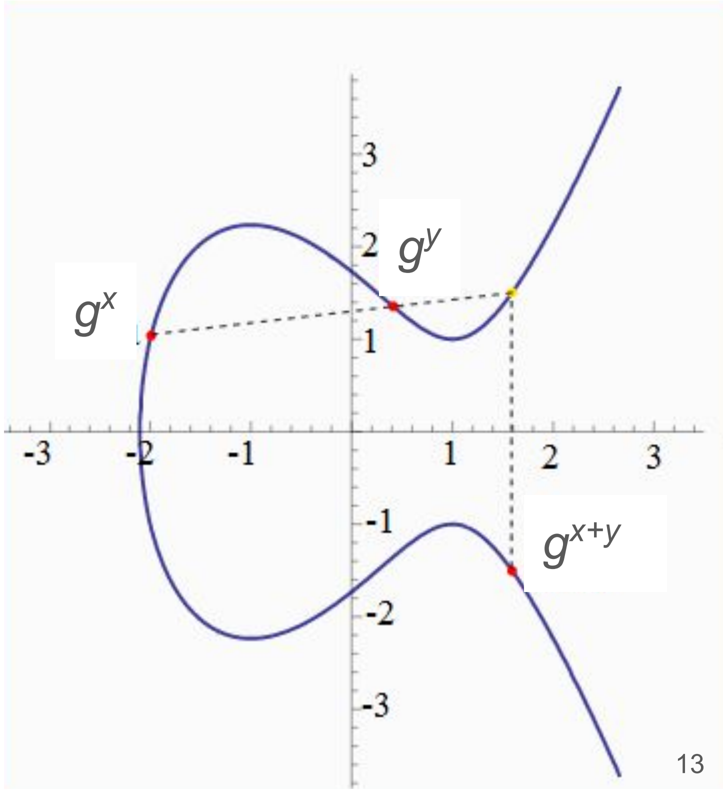
- secp256k1 elliptic curve
- We have

$$g^{x+y} = g^x * g^y$$

- In other words, it is linear:

$$f(x+y)=f(x) \oplus f(y)$$

here $\oplus$ denotes and algorithm



Multiply with a scalar

- Consequence of linearity:

$$f(cx)=c\otimes f(x)$$

  - here $\otimes$ denotes an algorithm

$$f(cx)=\underbrace{f(x)+f(x)+...+f(x)}_{c}$$

- take the binary representation of *c* and multiply the corresponding powers

The fundamental theorem of algebra:

- The following linear equations have single root:

$$s \equiv c*x+r \pmod{n}$$

  - If any 3 among s,c,x, r is given, there is only a single fourth, where $x \neq 0$, $c \neq 0$

- This also holds for

$$f(s) = c \otimes f(x) \oplus f(r)$$

The idea of Schnorr signature

First half of the digital signature

Solution to the previous puzzle (remains secret)

A random "nonce". It ensures that *x* cannot be decoded

Linear equality: $s = \boxed{c}\ x + r$ private keys

we also have: $f(s) = \boxed{c} \otimes f(x) \oplus f(r)$ public keys

The puzzle (transaction) hashed
c : = Hash(msg)

The previous puzzle

Second half of the digital signature

Schnorr signature

c = hash (f(r),f(x), and the msg)

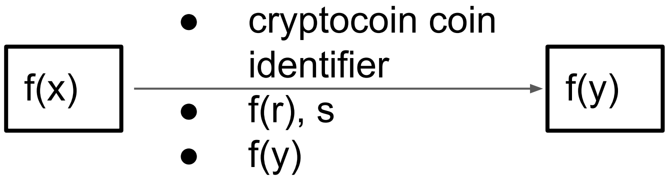
s = c x + r

f(s) = c ⊗ f(x) ⊕ f(r)

Gen/EncGen	Sign(sk_S, m)	Vrfy(pk_S, m, σ)
$x \leftarrow_\$ \mathbb{Z}_q; X \leftarrow g^x$	$(x, X) := sk_S;$	$X := pk_S; (R, s) := \sigma$
$sk := (x, X); pk := X$	$r \leftarrow_\$ \mathbb{Z}_q; R \leftarrow g^r$	$c := H(R\|\|X\|\|m)$
return (sk, pk)	$c := H(R\|\|X\|\|m)$	**return** $R = g^s X^{-c}$
	$s \leftarrow r + cx$	
	return $\sigma := (R, s)$	

Now we can define transaction

- A digital data which is
  - A cryptographic evidence that the buyer knows x
  - without disclosing x
  - the new puzzle is included
- A transaction is an evidence that a buyer gave the crypto-coin to the seller
  - It is distributed in a peer-to-peer network through public channels
  - It is registered by the network nodes
- The key problem that it allows double spending
  - The same crypto-coin is given to two seller
- The high level idea is that if sufficient network node registers the transaction the seller can be sure that the crypto-token was given to him



c = hash (f(r),f(x), and the msg)

$$s = c \quad x + r$$

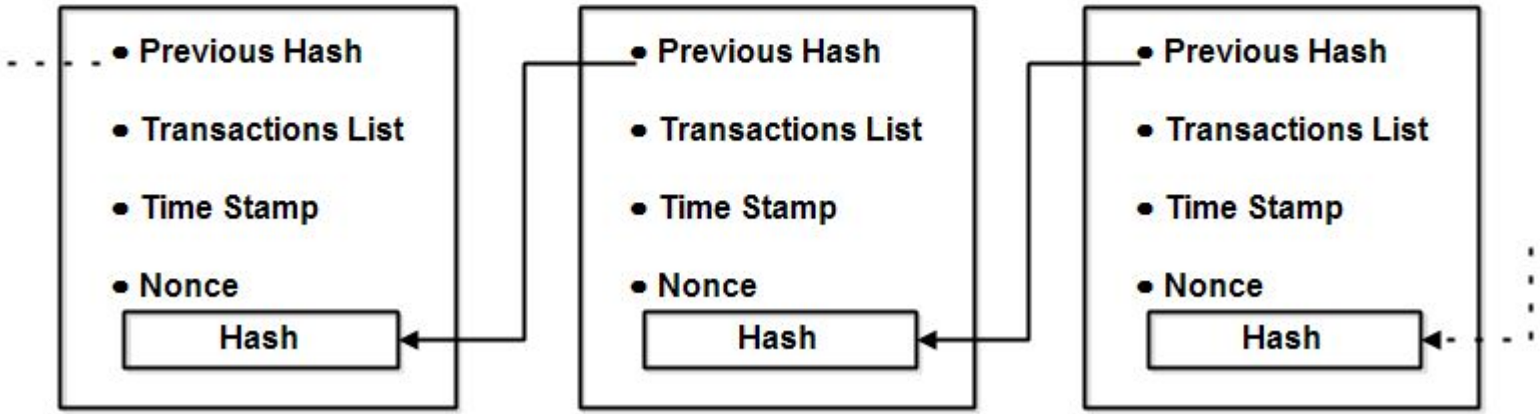
$$f(s) = c \otimes f(x) \oplus f(r)$$

How to avoid double spending

- The buyer gives the same crypto-coin to multiple sellers
- Which one is valid?
    - The first one, that is distributed in the network
- How to know which event was first among events in the past?
    - Proof of Work:
        - Nodes solve a giant puzzle, we measure the time as the size of the solved puzzle
        - The giant puzzle depends on the transaction
        - The puzzle is related to cryptographic hash function
- How to ensure that it is not possible to change the past?
    - Blockchain

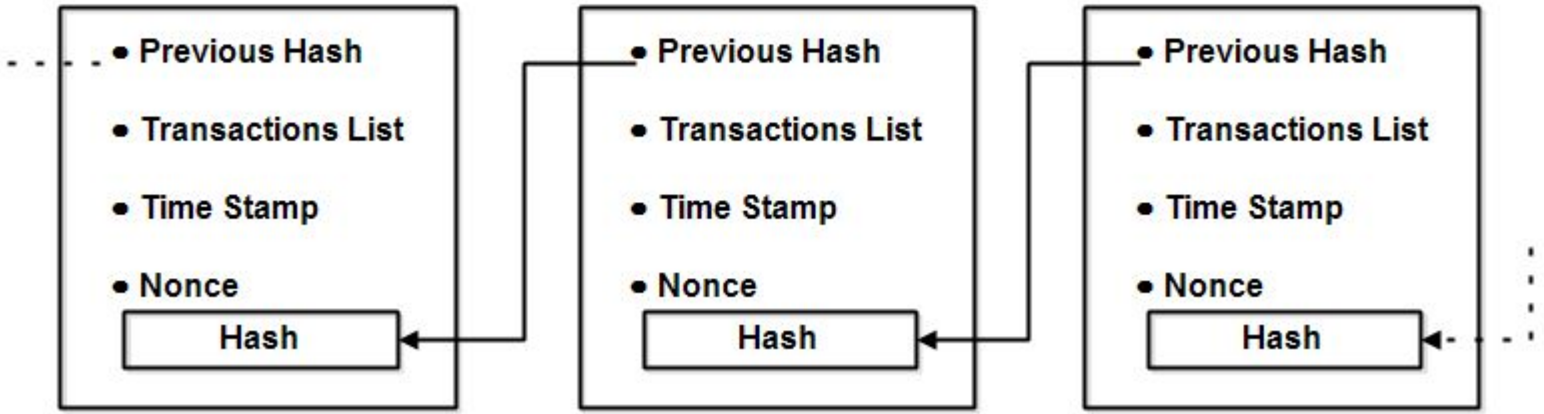
Idea 4: Blockchain

- In every time period publish a block
- You cannot change a transaction in the past keeping the same last hash



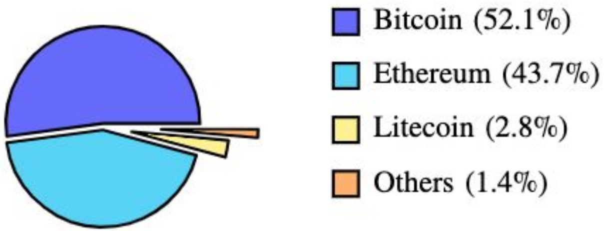
Idea 5: Consensus algorithm

- A single block chain is maintained.
- Proof of Work
  - In each iteration find a Nonce that provides hash <= difficulty
  - called mining

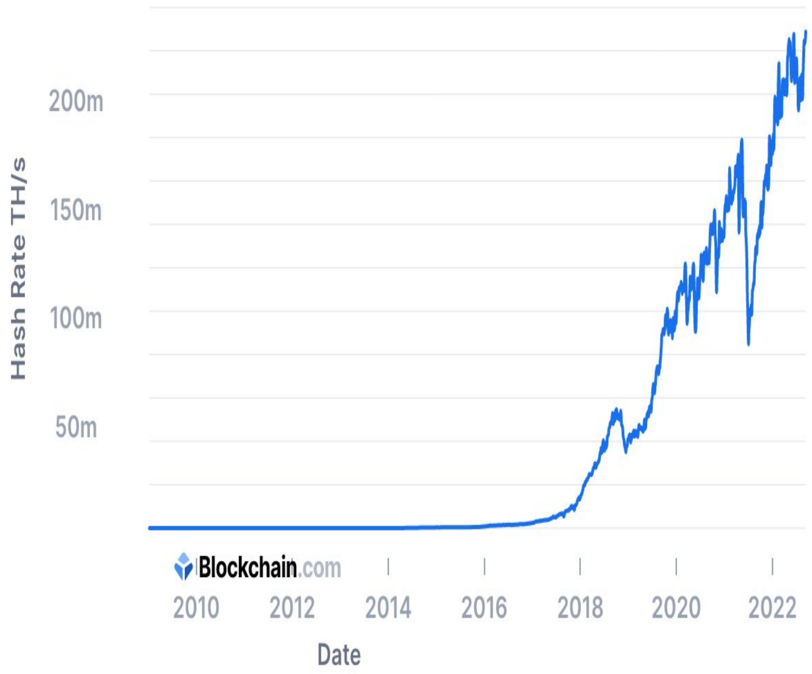


Proof of Work

- The nodes compete with each other
- The first node receives reward that finds a nonce with hash <= difficulty
    - Income for the miners
- The reward is a transaction in the block
- The other nodes verify the transactions and start mining the next block
- To change the past you need to redo the computations
- Consensus algorithm:
    - The majority of computation power
- Different cryptographic hash functions
    - ASIC: Bitcoin
    - ASIC-resistant (GPU-based): ETHash (Ethereum)



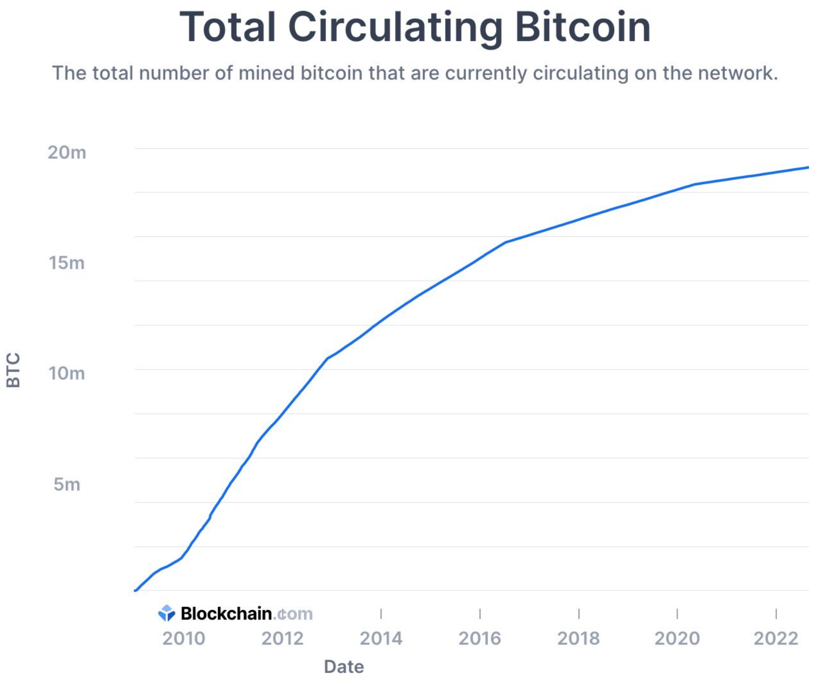
Hashrate


Hash Rate TH/s
200m
150m
100m
50m
Blockchain.com
2010
2012
2014
2016
2018
2020
2022
Date


ETH Difficulty: 12.06 P
Ethereum Block Height: 15,520,961
Zoom 1d 1w 1m 3m 6m 1y 3y All
Mar 21, 2016 → Sep 12, 2022
15 P
12.5 P
10 P
7.5 P
5 P
2.5 P
0
2017
2018
2019
2020
2021
2022

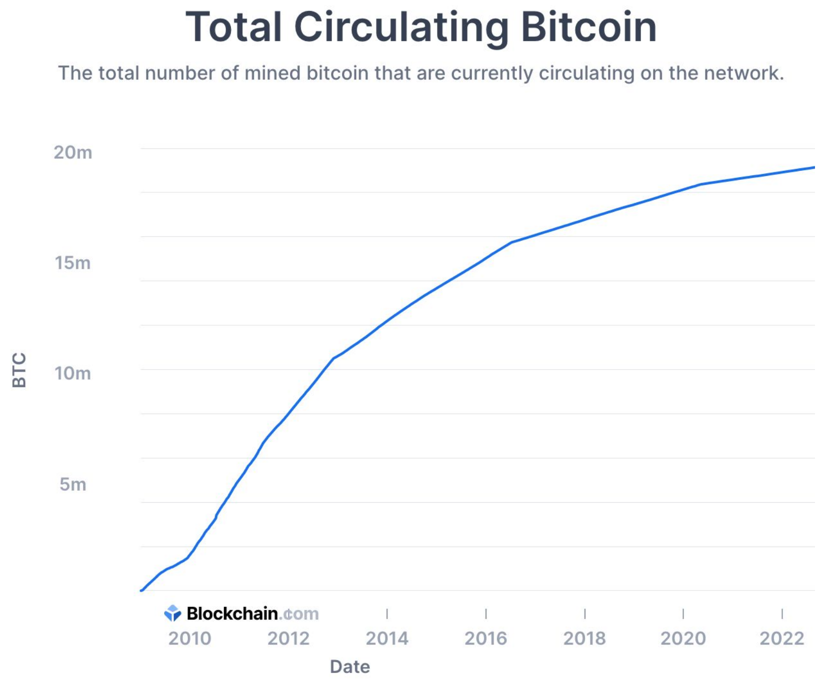
Idea 4: Turn it into a digital currency

- Limit the amount of crypto-coins
  - Extensive marketing
- Similar to diamonds
  - Adam Smith’s diamond-water paradox
  - In 1870 they relatively cheap
    - miners discovered huge deposits of diamonds in South Africa
  - Extensive marketing in 1940-80 by De Beers Consolidated Mines
    - a metaphor for eternal love
    - a sound investment
  - “A Diamond Is Forever”
    - sparkling pieces of carbon
    - incinerated to ash



Idea 4: Turn it into a digital currency

- Limit the amount of crypto-coins
  - Adam Smith's diamond-water paradox
    - Compared the high value of a diamond, which is unessential to human life, to the low value of water, without which humans would die
    - Diamonds are more expensive than water because they were more difficult to bring to market
    - Subjective prices drive costs.
- Marketing + limited amount

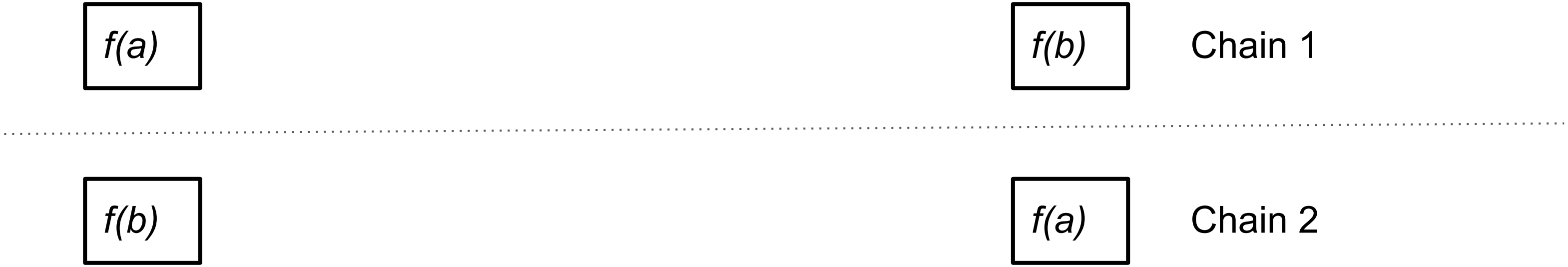


Cryptocurrencies

- Based on multiple ideas:
    - Digital signatures
    - Blockchain
    - Consensus Algorithms
        - Proof-of-Work
        - Proof-of-Stake
    - Marketing

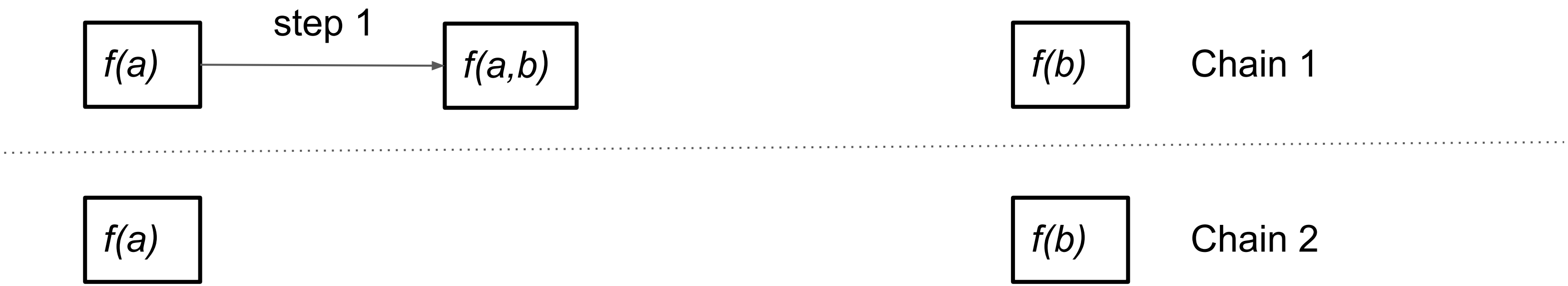
Exchange crypto-coins among blockchains

- How to exchange ETH to BTC
- Two different blockchains
  - Same elliptic curve (secp256k1)
- The two parties (Alice, Bob) do not need to trust
  - An alternative to the centralized exchange point



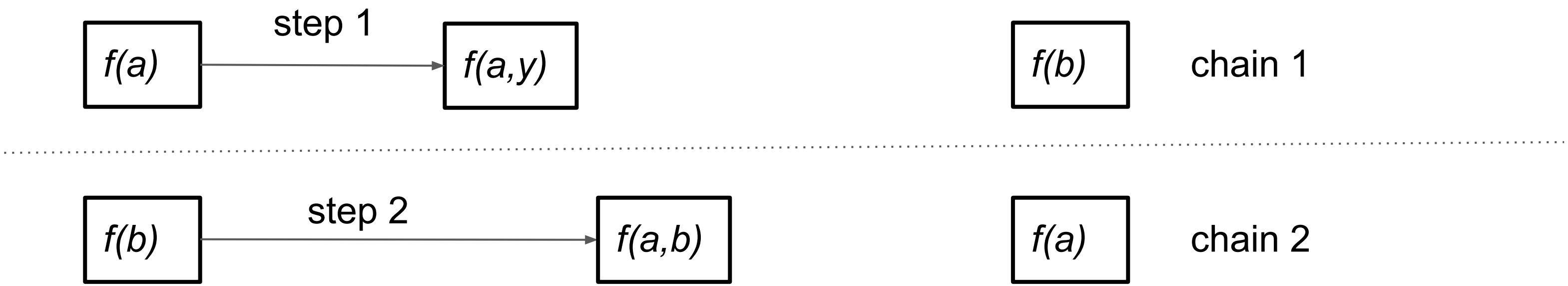
Atomic swap - Step 1

- Alice and Bob agree on the exchange rate
- Alice submits a transaction on chain 1 to transfer it's crypto-coin to a special address *f(a,b)*
    - A multisig address requires the knowledge of *a* and *b*
    - The two parties generate it through communication without disclosing *a* and *b*
    - There is a timeout, after which the coin is returned to f(*a*)



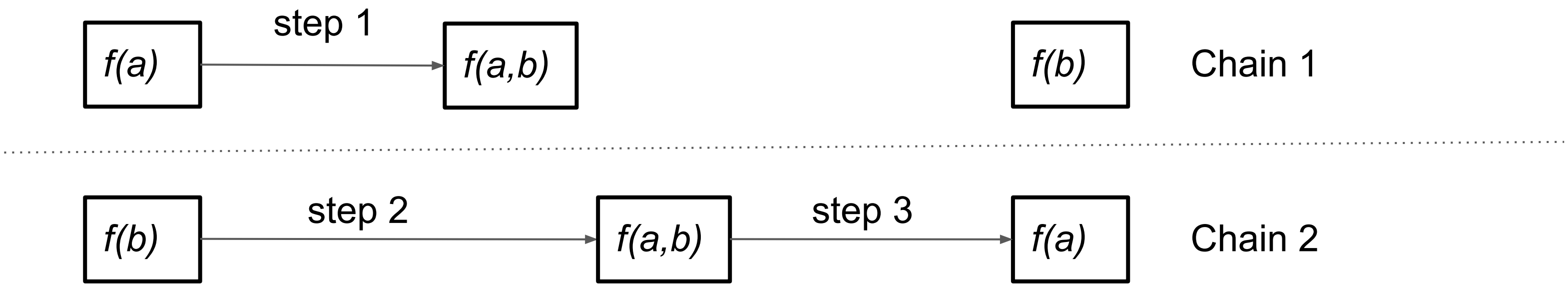
Atomic swap - Step 2

- Bob submits a transaction on chain 2 to transfer it's crypto-coin to a special address *f(a,b)*
  - There is a timeout, after which the coin is returned to *f(b)*
- At this point both coins are owned by Alice and Bob jointly
  - At least until the timeout



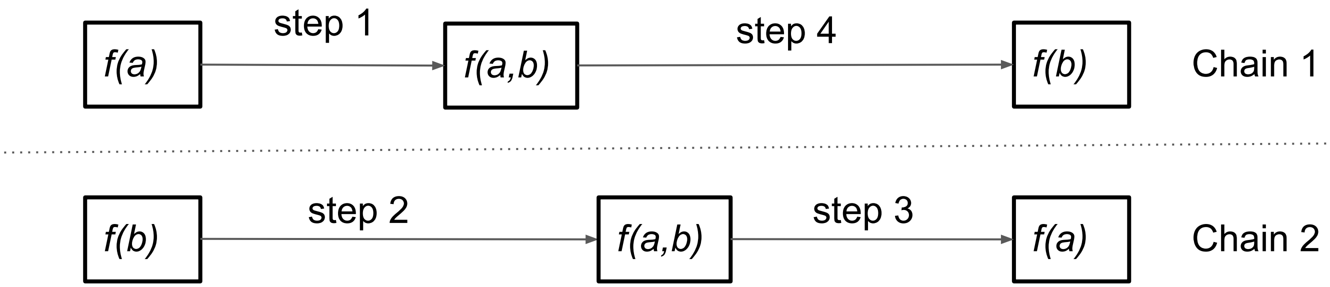
Atomic swap - Step 3

- Alice and Bob exchange sufficient information off-chain so that Alice can issue a transaction *f(a,b)* → *f(a)* on Chain 2
- The signature will reveal sufficient information for Bob to issue a transaction *f(a,b)* → *f(b)* on Chain 1
  - Not trivial, because transactions are designed not to reveal any information on the secret key



Atomic swap - Step 4

- Bob reads out the transaction *f(a,b)* → *f(a)* on Chain 2
- Bob issue the transaction *f(a,b)* → *f(b)* on Chain 1
- Atomic swap is completed
    - Otherwise the tokens return to their owner after the timeout



Multisig Signature

- Input: f(a) and f(b)
- Constraints:
  - It is infeasible to compute the private keys a and b
  - There is protocol that generates a valid *f(a,b)* signature with the two parties, Alice and Bob, such that only Alice knows *a*, and Bob knows *b*

$$f(a,b) = f(a) \oplus f(b)$$

c = hash ($f(r_1) \oplus f(r_2), f(a) \oplus f(b)$, and the msg)

aláírás: $s_1 + s_2$, $f(r_1) \oplus f(r_2)$

$$\boxed{s_1} = c \quad a + r_1$$

$$f(s_1) = c \otimes f(a) \oplus \boxed{f(r_1)}$$

$$\boxed{s_2} = c \quad b + r_2$$

$$f(s_2) = c \otimes f(b) \oplus \boxed{f(r_2)}$$

Adaptor signature

- A signature that becomes valid once *t* is known
- disclose f(t) so that it can be verified

$$\boxed{s} = c \quad a + r \quad + t$$

$$f(s) = c \otimes f(a) \oplus \boxed{f(r)} \oplus \boxed{f(t)}$$

Adaptor Signature with multisig

Alice:

$\boxed{s_a} = c \quad a + r_1 \quad + t$

$f(s_a) = c \otimes f(a) \oplus \boxed{f(r_1)} \oplus f(t)$

Bob:

$\boxed{s_b} = c \quad b + r_2$

$f(s_b) = c \otimes f(b) \oplus \boxed{f(r_2)}$

Alice, Bob:

$\boxed{s_{ab}} = \quad s_a + s_b = c \quad b + r_1 \quad + r_2 + t$

$f(s_a) \oplus f(s_b) = c \otimes f(b) \oplus f(r_1) \oplus f(r_2)$

Adaptor signature with multisig

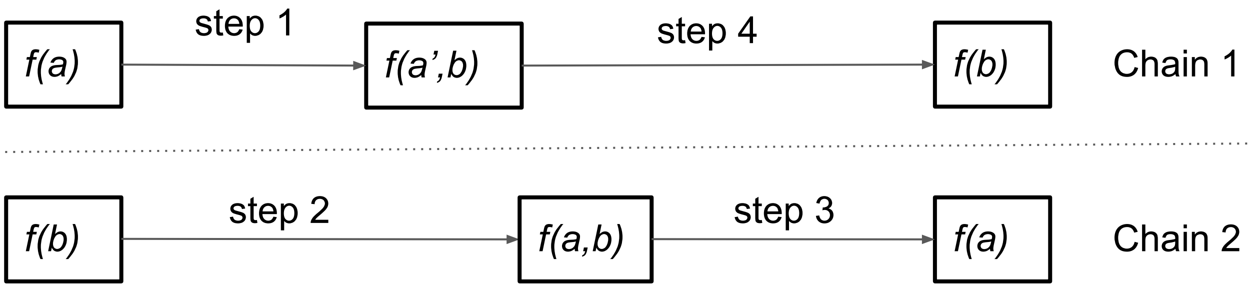
Step 1 Alice, Bob: $\boxed{s_{a'b}} = s_a + s_b = c\, b + r_1 + r_2 + t$

$f(s_a) \oplus f(s_b) = c \otimes f(b) \oplus f(r_1) \oplus f(r_2)$

Step 3 Alice, Bob: $\boxed{s_{ab}} = s_a + s_b = c\, b + r_1 + r_2$

$f(s_a) \oplus f(s_b) = c \otimes f(b) \oplus \boxed{f(r_1) \oplus f(r_2)}$

$t = s_{a'b} - s_{ab}$



Köszönöm a figyelmet!

tapolcai@tmit.bme.hu