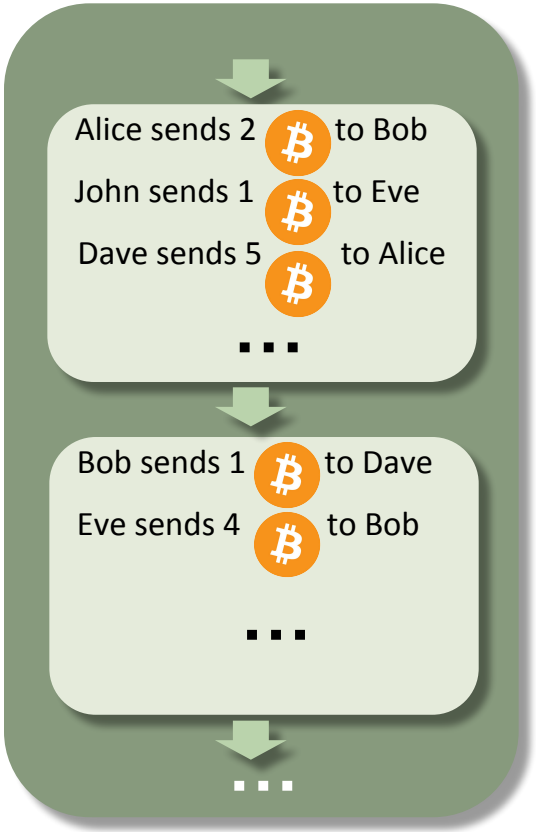
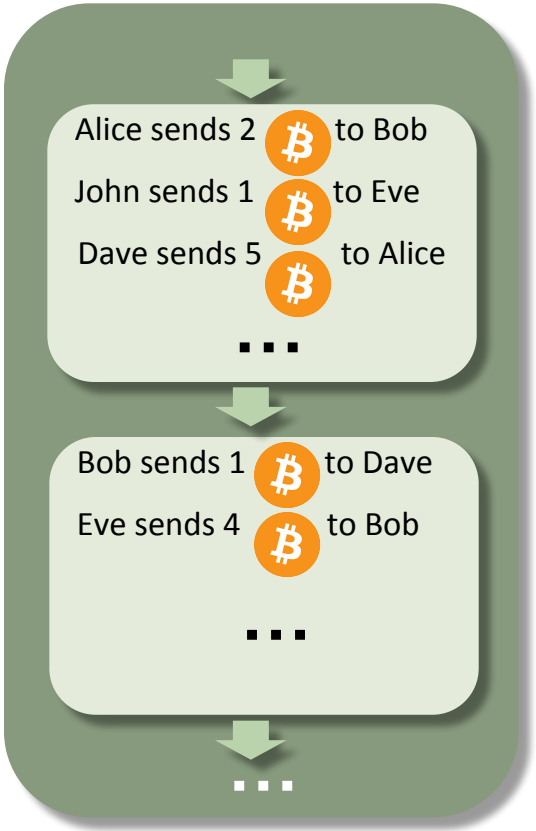
The power of Blockchain: Smart Contracts

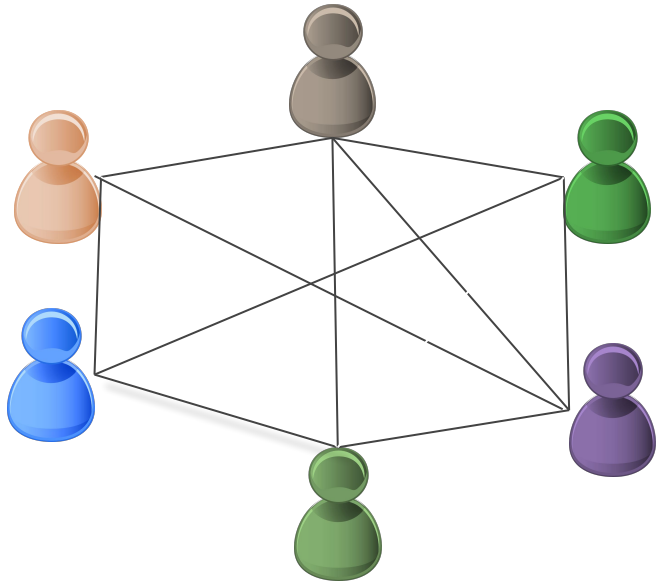
Foteini Baldimtsi

The Blockchain at the heart of a cryptocurrency

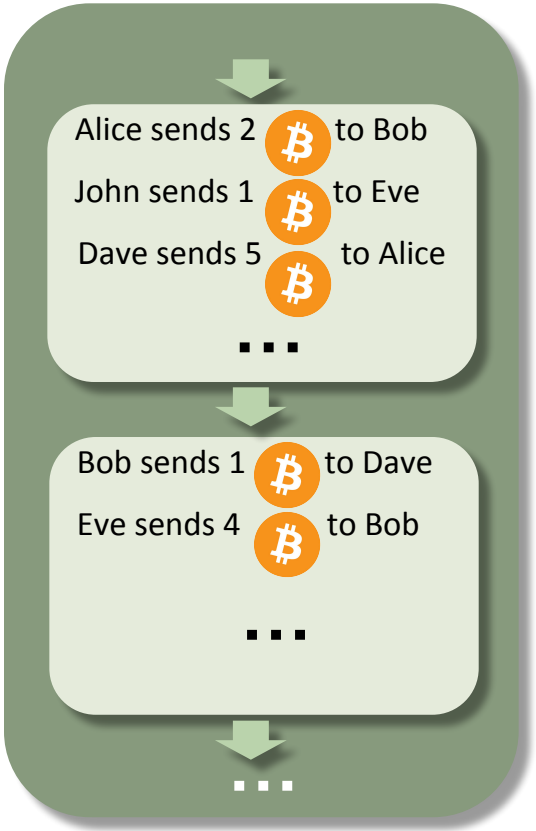


The Blockchain at the heart of a cryptocurrency





Abstracting the notion of Blockchain



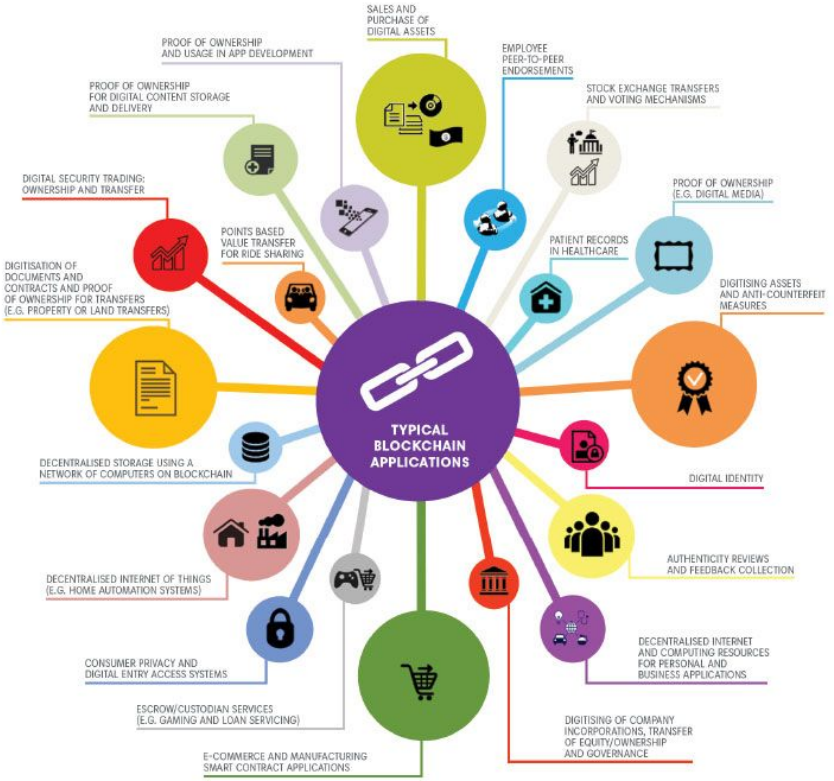
- Immutable data collection
  - Data added in an “append-only” manner
  - Nobody can modify an old transaction
- Controlled in a decentralized manner by multiple parties running a consensus protocol
  - Byzantine agreement
  - Proof-of-work
  - Proof-of-space

Where is Blockchain used?

Cryptocurrencies

Where is Blockchain used?

Cryptocurrencies



Why is Blockchain useful?

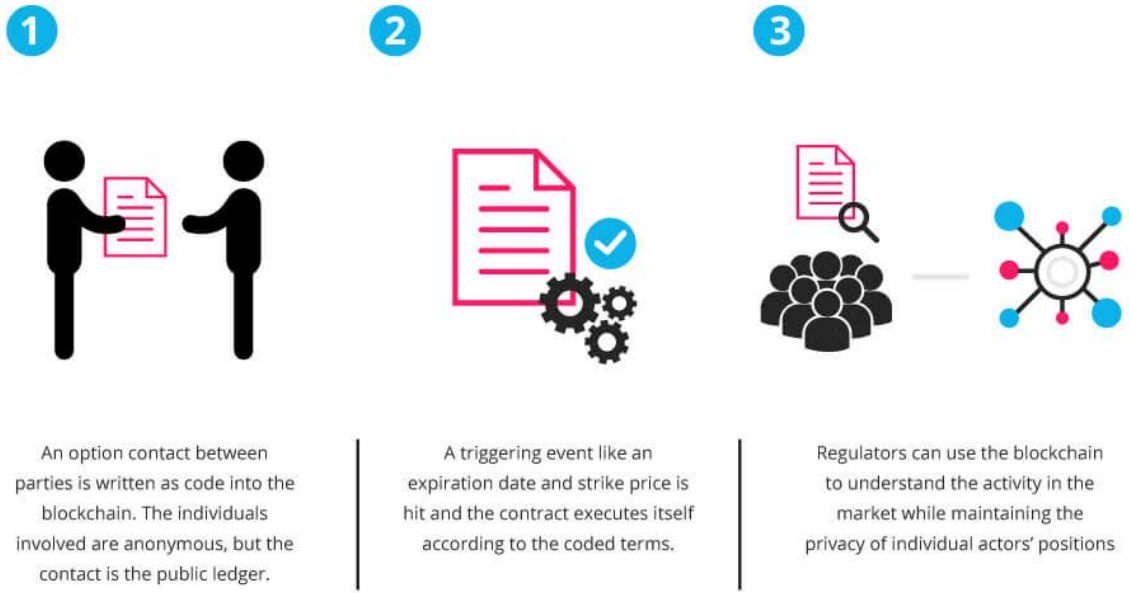
1. As a distributed, tamper-proof, **data structure**
   - No central trusted authority exists
   - Participating parties do not trust each other

2. As a mechanism for execution of **smart contracts**
   - Enforce the negotiation or performance of a contract
   - Allows for fair-exchange (blockchain is the mediator)
   - No direct interaction between parties

Smart Contracts

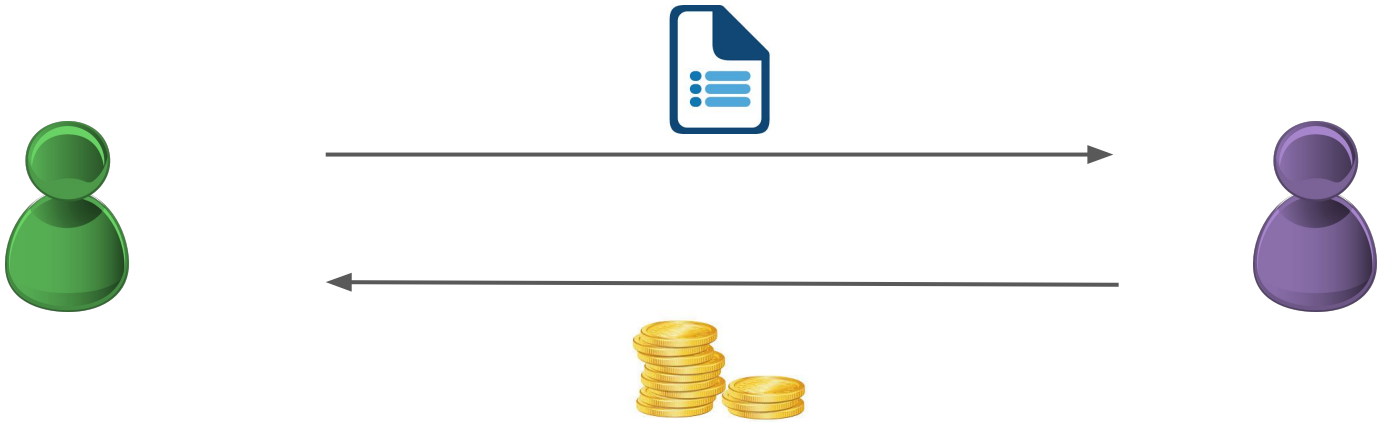
Introduced by Nick Szabo in 1994

Help you exchange money, property, shares, or anything of value in a transparent, conflict-free way while avoiding the services of a middleman

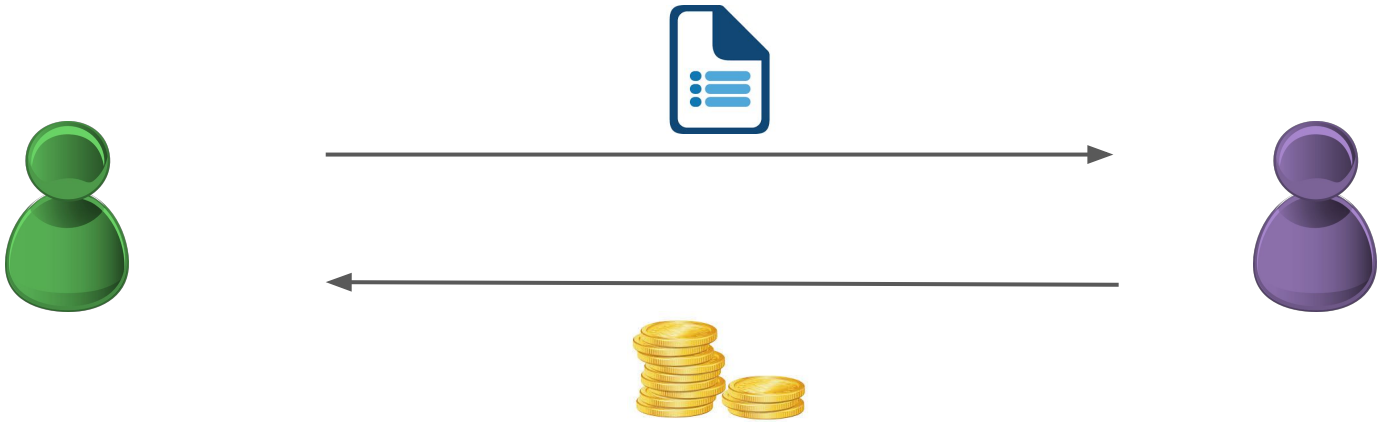


Smart Contract for Fair Exchange

Smart Contract for Fair Exchange



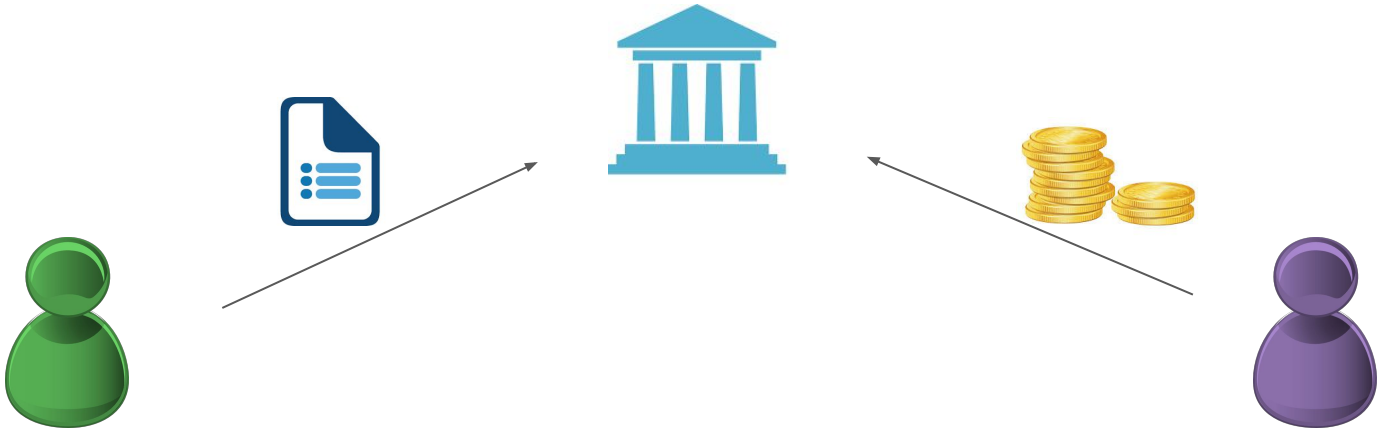
Smart Contract for Fair Exchange



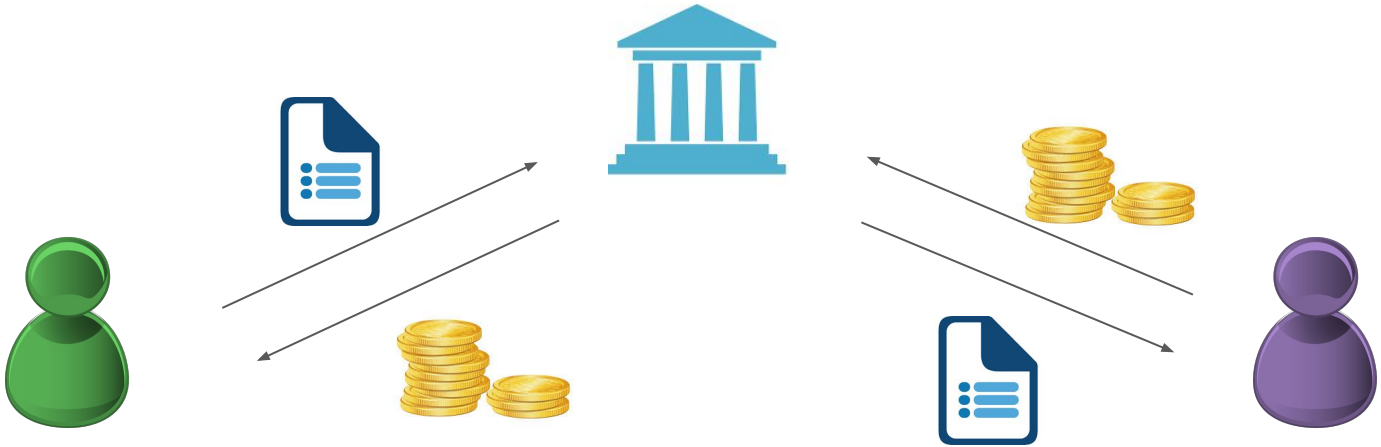
Impossible without the use of a trusted third party!

Needs to be an atomic operation!

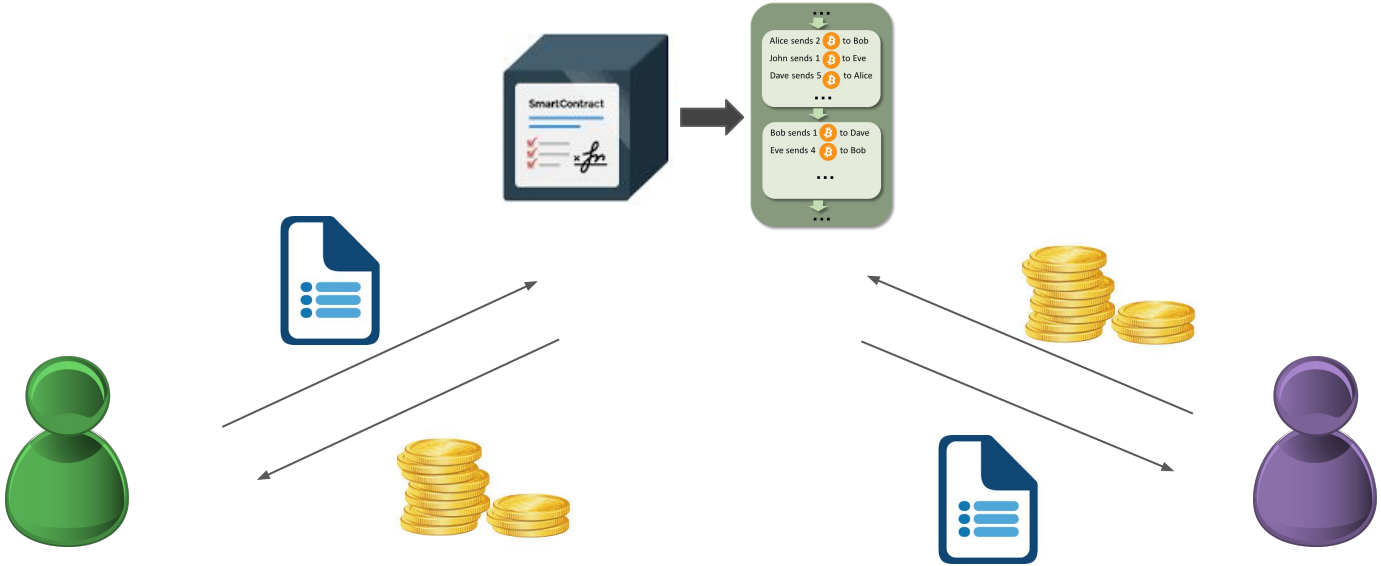
Smart Contract for Fair Exchange



Smart Contract for Fair Exchange



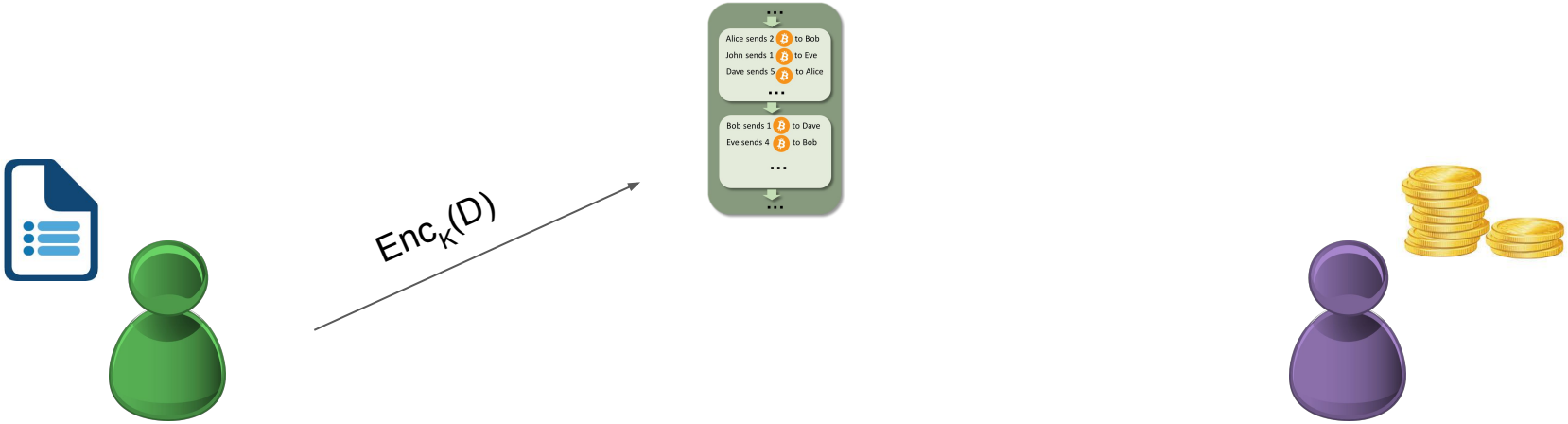
Smart Contract for Fair Exchange



Smart Contract for Fair Exchange

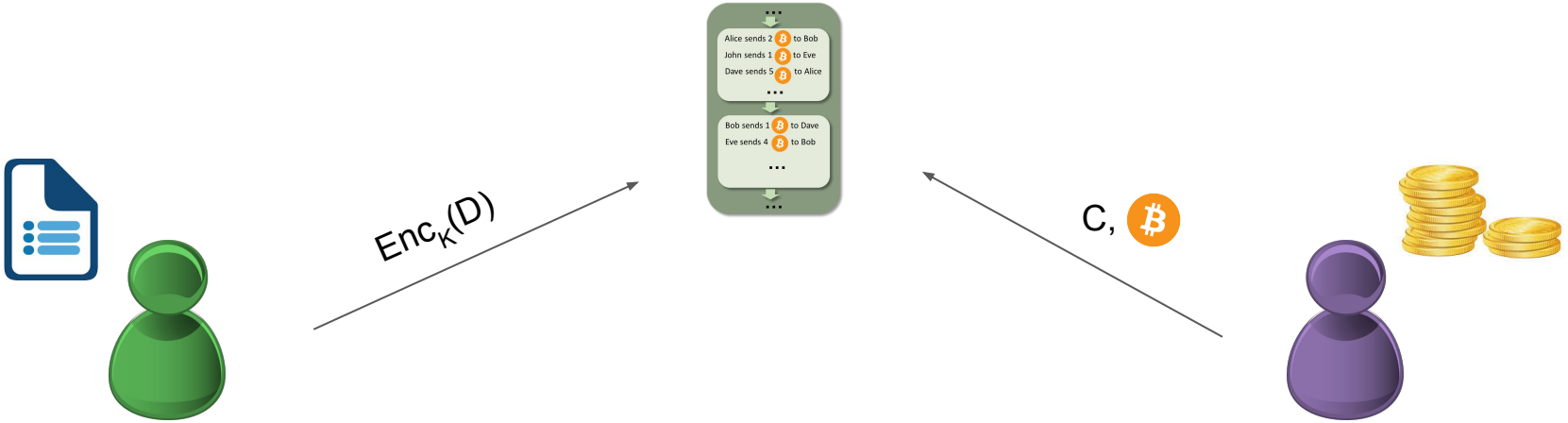
1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

Smart Contract for Fair Exchange



1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

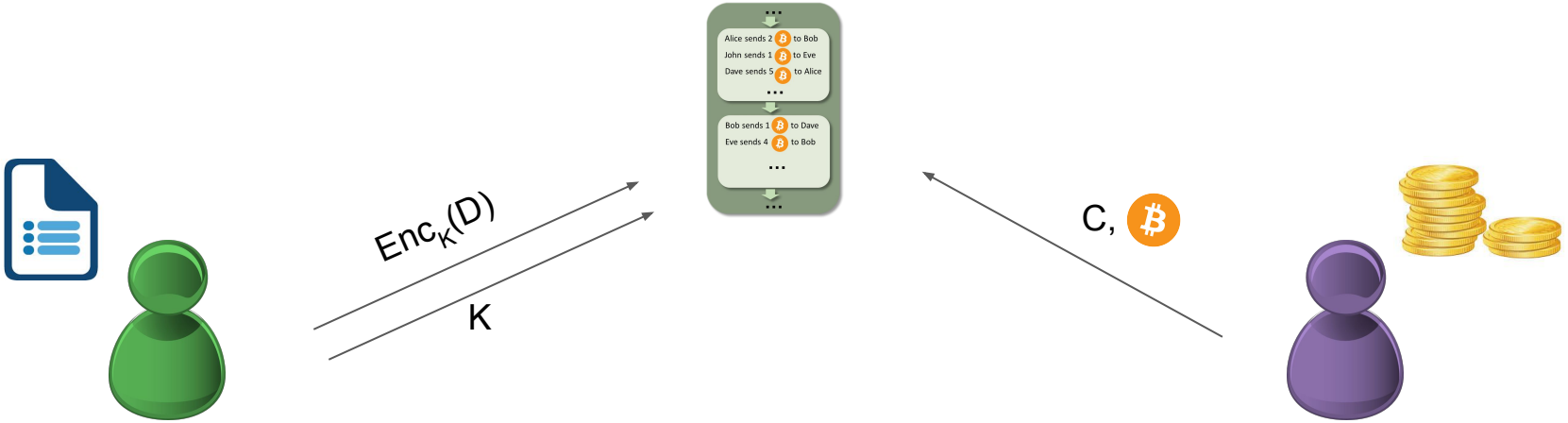
Smart Contract for Fair Exchange



1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

1. Prepare contract
   C = “ 1 ₿ for the secret key K”

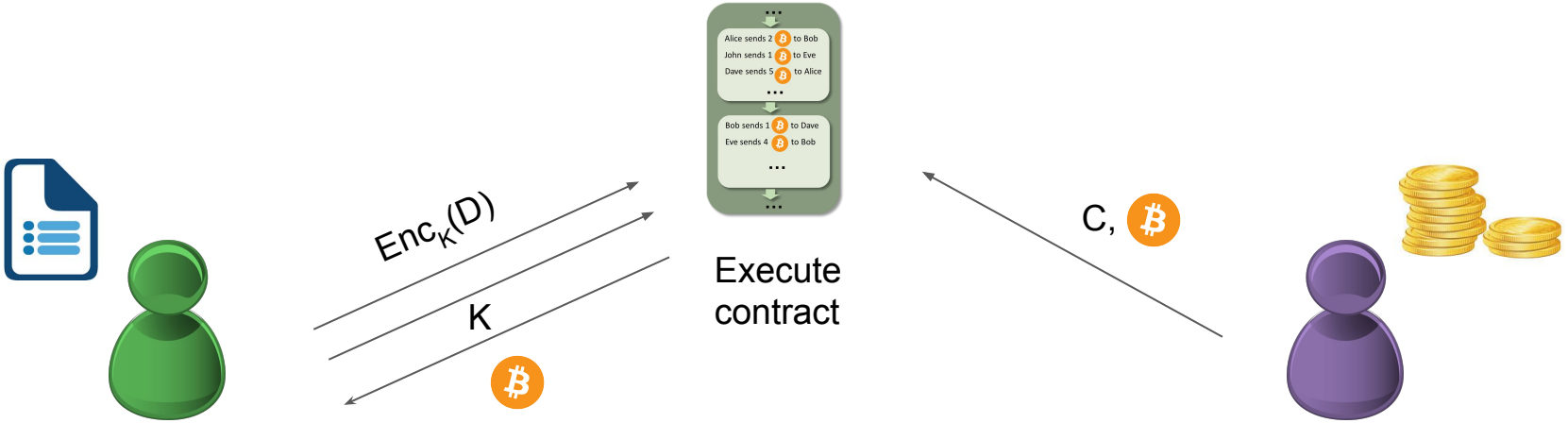
Smart Contract for Fair Exchange



1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

1. Prepare contract
   C = “ 1 ₿ for the secret key K”

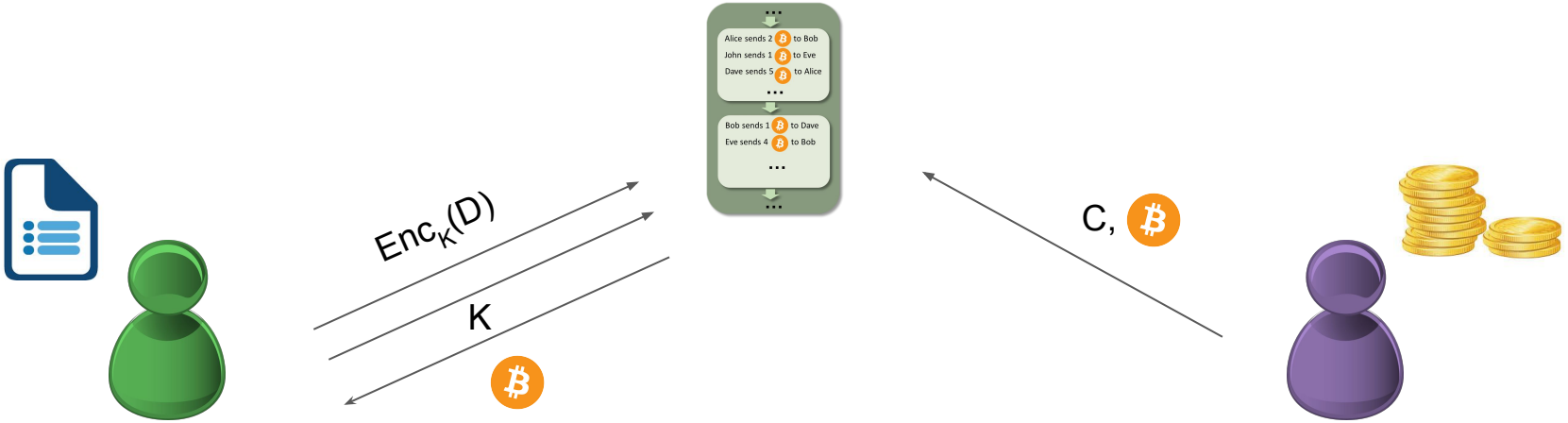
Smart Contract for Fair Exchange



1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

1. Prepare contract
   C = " 1 ₿ for the secret key K"

Smart Contract for Fair Exchange

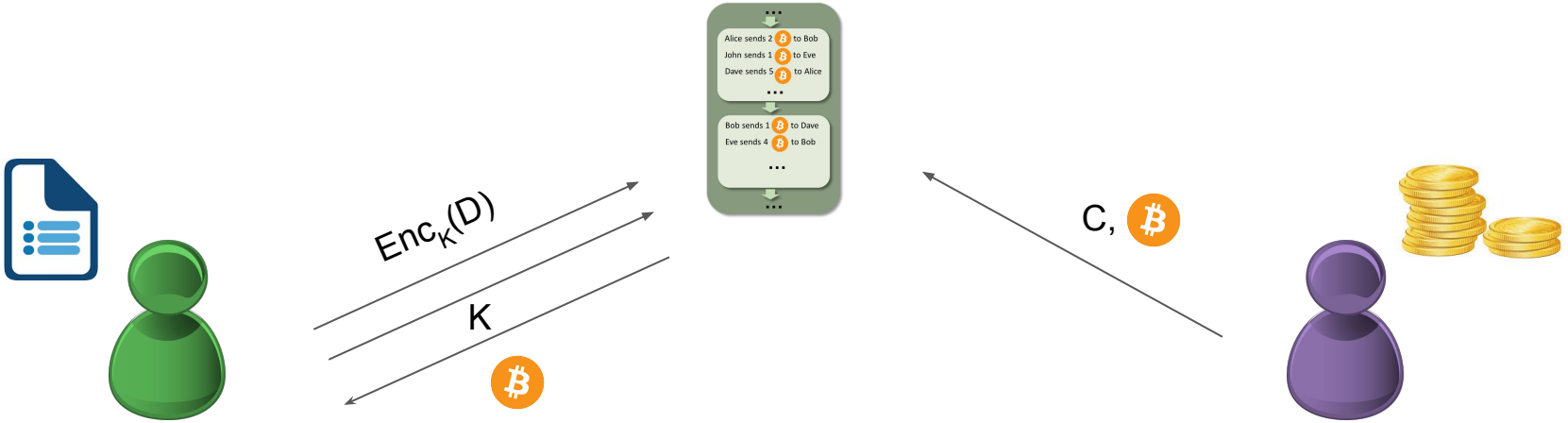


1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

1. Prepare contract
   C = " 1 ₿ for the secret key K"

Q: How can the buyer know that the seller has encrypted the correct document?

Smart Contract for Fair Exchange

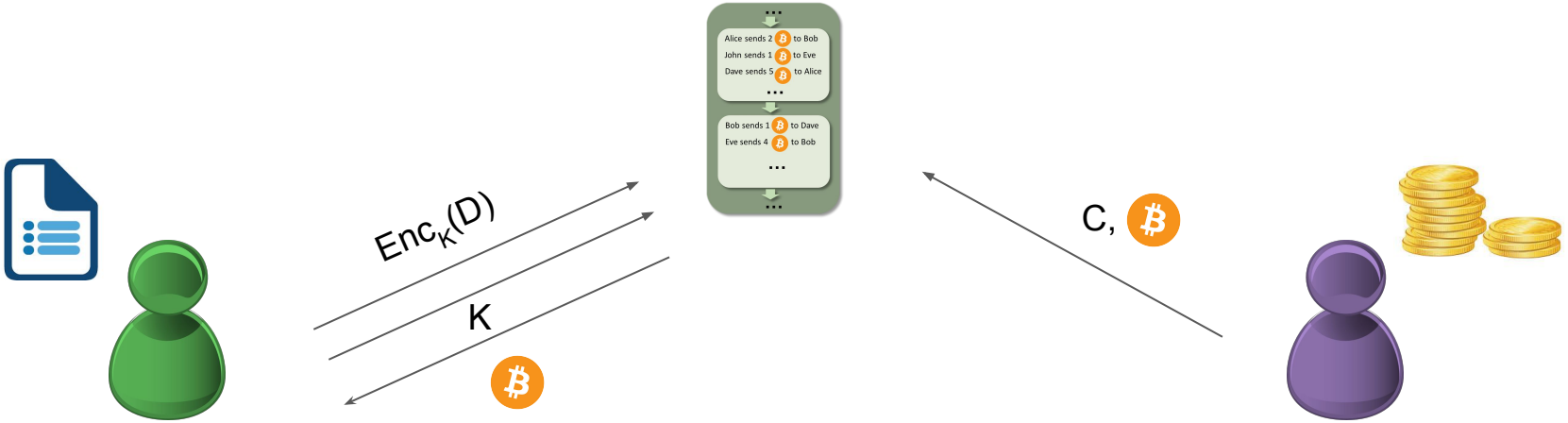


1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

1. Prepare contract
   C = " 1 ₿ for the secret key K"

Q: How can the buyer know that the seller has encrypted the correct document?
A: Break $D = D_1...D_n$, encrypt and post all shares and reveal some random ones before selling.

Smart Contract for Fair Exchange

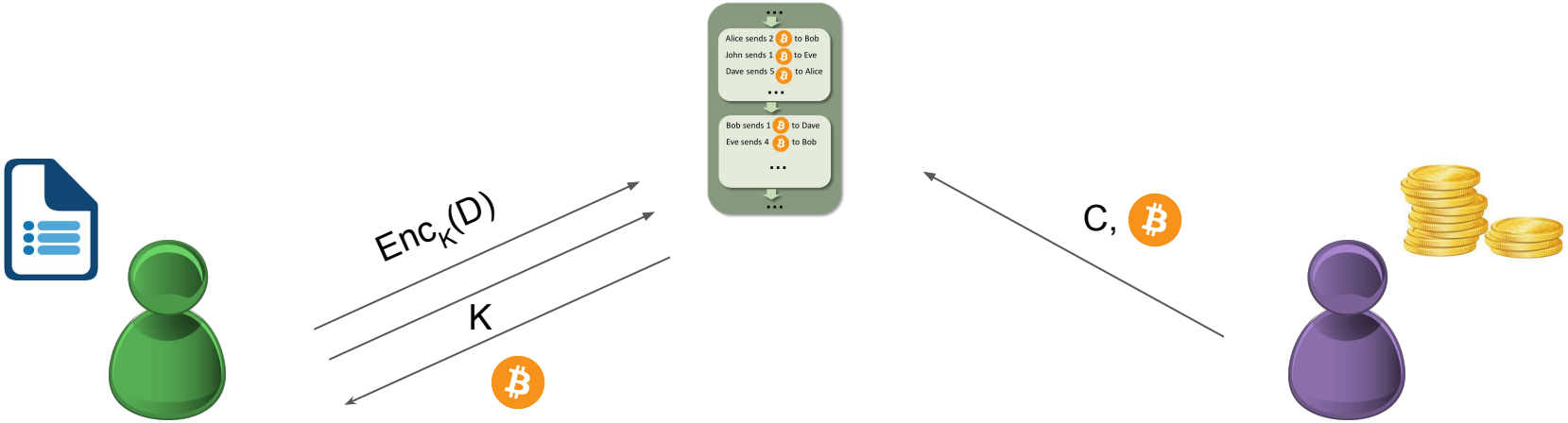


1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

1. Prepare contract
   C = " 1 ₿ for the secret key K"

Q: What if the seller never reveals K? What happens to the buyers coin?

Smart Contract for Fair Exchange

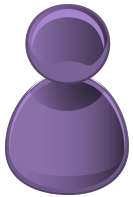


1. Pick a secret key K
2. Encrypt document into $Enc_K(D)$

1. Prepare contract
   C = " 1 ₿ for the secret key K for tw=10 hours"

Q: What if the seller never reveals K? What happens to the buyers coin?
A: Timelocked transactions: funds are returned after a specific time window.

Bitcoin Smart Contracts

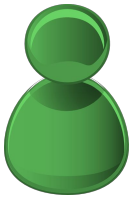


C = “ 1 ₿ for the secret key K for tw=10 hours”

Such a contract cannot be implemented in Bitcoin!

Bitcoin has been designed to only check two conditions:

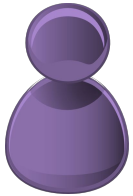
- Verification of an ECDSA signature (under specific parameters)
- Preimage of a hash function output



Post: $Enc_K(D)$, Hash(K) and ZK proof that the hashed key is the correct one

After verifying the proof, post

C = “ 1 ₿ for the preimage of Hash(K) for tw=10 hours”



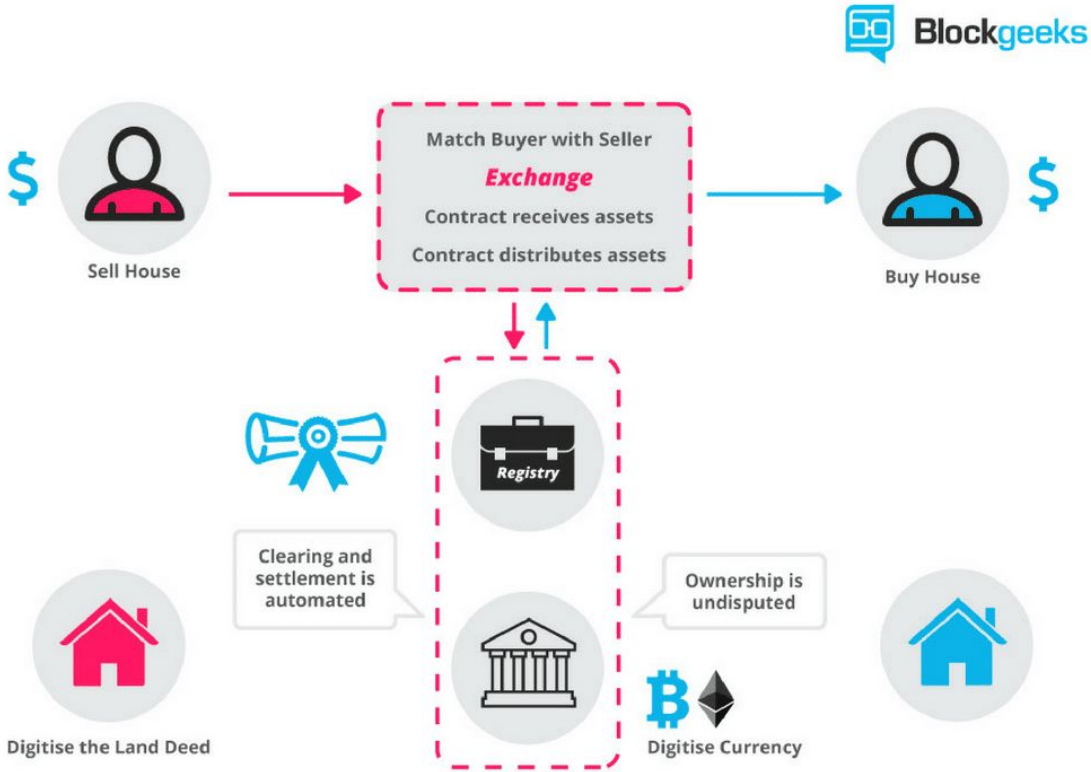
Ethereum smart contracts

Provides a “Turing-complete” scripting language: supports a broader set of computational instructions.

- Contracts can use data outside of the blockchain (i.e. weather, stock prices etc)
- We can build chains of contracts (the 2nd settles on information from the output of the 1st contract)

Smart contracts run on the **Ethereum Virtual Machine** (works at a level too low to be convenient to directly program).
Solidity is the most popular language for writing contracts (JavaScript-like)

Smart contracts interacting with each other



https://blockgeeks.com/guides/smart-contracts/

Ethereum smart contracts  need **gas** to run

Task	Gas Required	Cost (ETH)	Cost (USD)	Ops per ETH	Ops per USD	Ops per Block	Blocks to complete Op
Add or subtract two integers	3	0.00000009	0.00002655	11111111.11	37664.78343	1566666.667	0.0000006382978723
Add or subtract two integers 1 million times	3000000	0.09	26.55	11.11111111	0.03766478343	1.566666667	0.6382978723

Task	Gas Required	Cost (ETH)	Cost (USD)	Ops per ETH	Ops per USD	Ops per Block	Blocks to complete Op
Save a 256-bit word to storage	20000	0.0006	0.177	1666.666667	5.649717514	235	0.004255319149
Save 1 MB to storage (31250 256-bit words)	625000000	18.75	5531.25	0.05333333333	0.000180790960	0.00752	132.9787234
Save 1 GB to storage (1000 MB)	625000000000	18750	5531250	0.000053333333	0.000000180790	0.00000752	132978.7234

There is a need for efficient smart contract protocols!

https://hackernoon.com/ether-purchase-power-df40a38c5a2f

A note on privacy

The blockchain is **public** and all posted data:

- **Code** of the contract
- **PKs** of the participants
- **Data** send to the contract
- **Payments** (amounts and PKs)

- Generic smart contract anonymity compilers for ethereum [Hawk]
- Specialized tools for Bitcoin contracts anonymity [Tumblebit]

Advantages of smart contracts

Applications: Government: e-voting, e-IDs (decentralized PKIs), Auctions, Real estate, Healthcare, Supply chain

- Autonomy: no need for trusted third party
- Trust: data stored in a shared ledger (cannot be lost)
- Backup: data duplication
- Speed: automating various processes

thank you!

foteini@gmu.edu