Ethereum Blockchain

Prof. David (Duc) Trần, PhD

University of Massachusetts, Boston (USA)

Ethereum

The Ethereum Project's logo, first used in 2014

Original author(s)	Vitalik Buterin, Gavin Wood, Joseph Lubin
Initial release	30 July 2015
Repository	https://github.com/ethereum
Written in	Go, C++, Rust
Operating system	Clients available for Linux, Windows, macOS, POSIX, Raspbian
Platform	x86, AMD64, ARM
Type	Decentralized computing, Blockchain, Cryptocurrency
License	GPLv3, LGPLv3, MIT[1][2]

Vitalik Buterin

Vitalik Buterin, 2016

Native name	Виталий Дмитриевич Бутерин
Born	January 31, 1994 (age 24) Kolomna, Russia
Nationality	Russian-Canadian
Alma mater	University of Waterloo (dropped out)
Known for	Ethereum, *Bitcoin Magazine*

Vitalik Butarin

- Won a bronze medal in the International Olympiad in Informatics
- Attended the University of Waterloo
- Dropped out of university in 2014 when awarded with a a Thiel Fellowship ($100K grant) to work full-time on Ethereum

Vitalik Buterin

Виталий Дмитриевич Бутерин

Buterin in 2016

Born	31 January 1994 (age 27) Kolomna, Russia
Nationality	Canadian
Education	University of Waterloo (dropped out)
Known for	Ethereum, *Bitcoin Magazine*
Awards	Thiel Fellowship

What is Ethereum?

Ethereum is an open software platform based on blockchain technology that enables anyone to build and deploy decentralised applications (dapps).

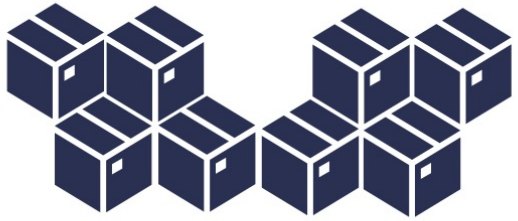
What is Dapp?

Decentralised application (Dapp)

is a new type of software program designed to exist on the internet in a way that is not controlled by any single entity.

Example

Bitcoin's Blockchain



Blockchain is a distributed database that is used to maintain a continuously growing list of records called blocks.

→ Public ledger showing all Bitcoin transactions

- ✓ Online based
- ✓ Decentralised
- ✓ Not controlled by any single entity

= Shared record of the entire transaction history

Binary.com

The birth of Ethereum

2009

Bitcoin was launched

2012

Ethereum idea was formed by **Vitalik Buterin**

2013

Ethereum **Whitepaper**

2014

Announcement of **Ethereum Project**

Ethereum Foundation was created.

Pre-sale of **60 million** digital tokens **(Ethers)**

2015

Ethereum Project launch

2016

DAO attack

Ethereum splits

Ethereum

Ethereum Classic

Gavin Wood
(Director)

Jeffrey Wilcke
(Director)

Vitalik Buterin
(Inventor, Founder)

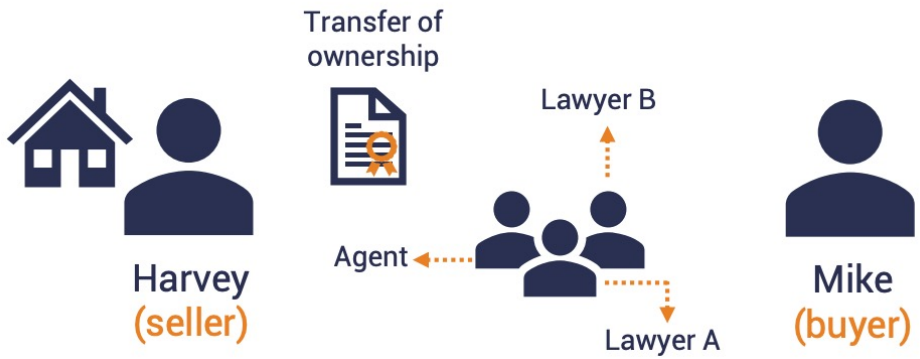
Binary.com

What is Ethereum?

Ethereum is powered by the Ethereum Virtual Machine which allows smart contracts to run on a decentralised blockchain. These contracts self-execute only when certain conditions are met.

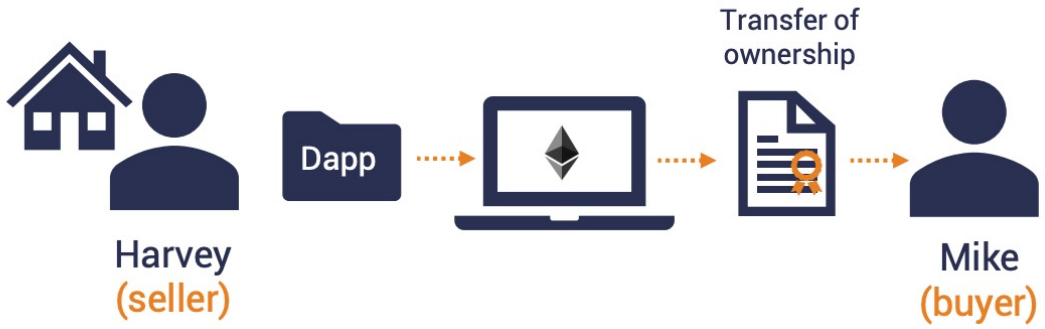
Normal contract

Mike wants to buy a house



Smart contract

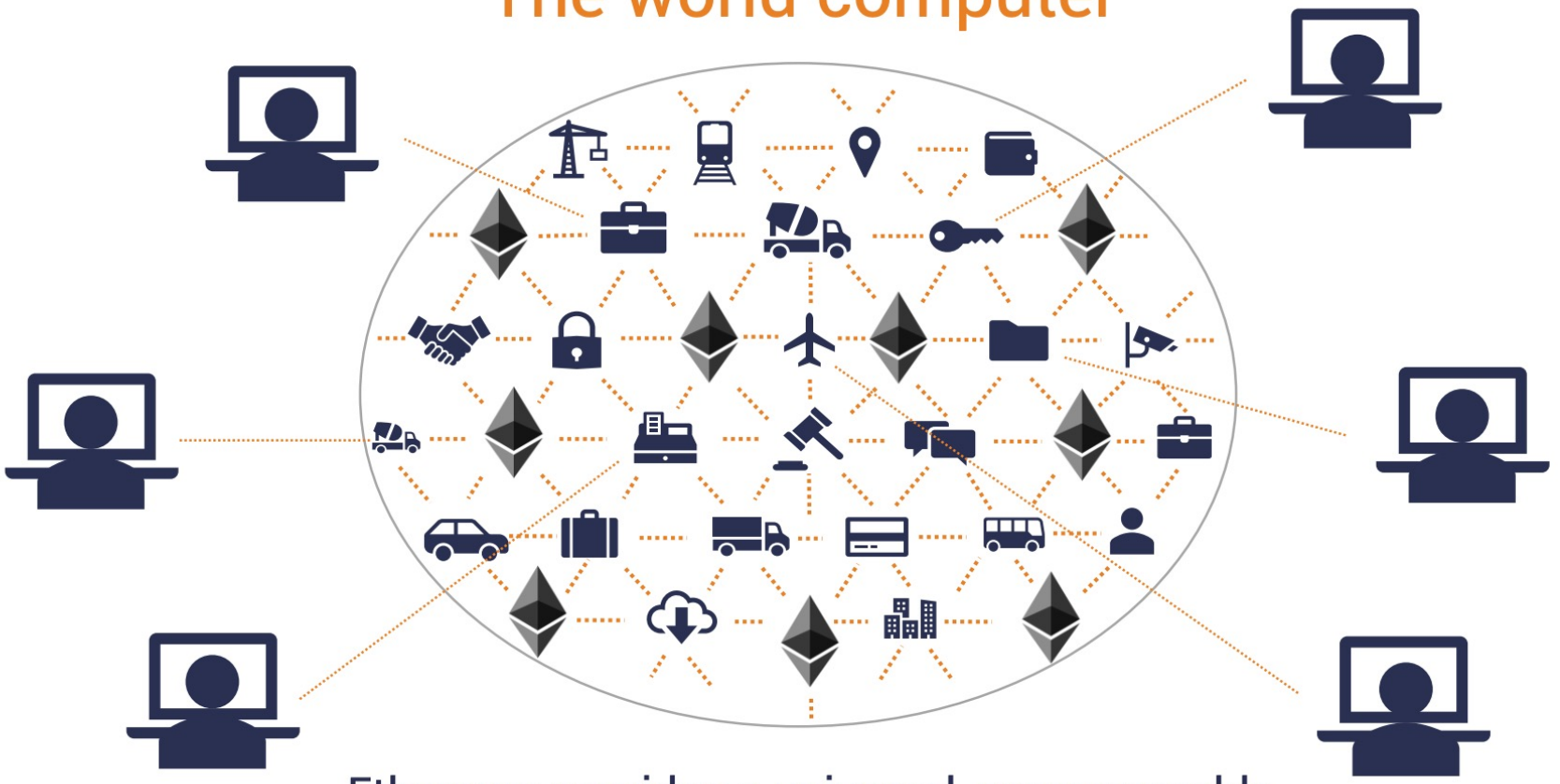
Mike wants to buy a house



Binary.com

What is Ethereum?

"The world computer"



Ethereum provides a universal, programmable blockchain which anyone can use.

Binary.com

Ethereum vs. Bitcoin

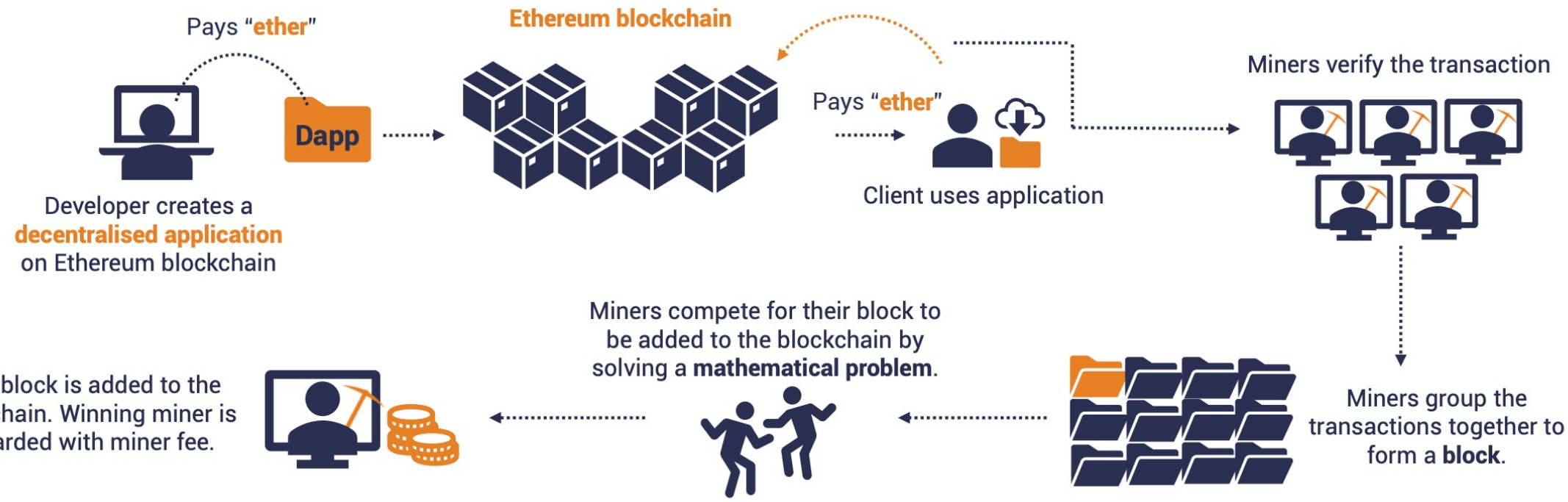
Ethereum	Bitcoin
"World computer"	"Digital gold"
Blockchain-based technology allowing smart contracts to run on decentralised applications	First digital currency which enabled online payment transactions using blockchain technology
Digital currency: Ether	Digital currency: Bitcoin
Coin supply: Unlimited	Coin supply: 21 million
Blockchain generation: every 15 seconds	Blockchain generation: every 10 minutes

Binary.com

Why Ethereum

- Bitcoin limitation
  - The script language too limited
  - Transaction processing too slow
- **Ethereum**
  - Also **proof-of-work** consensus, but
  - Built-in **Turing-complete programming language**
    - Anyone can write **smart contracts** with their own arbitrary rules for ownership, transaction formats and state-transition functions
  - **Faster**: 15 second/block vs. 10 minute/block of Bitcoin
- Whitepaper:github.com/ethereum/wiki/wiki/White-Paper

How does it work?



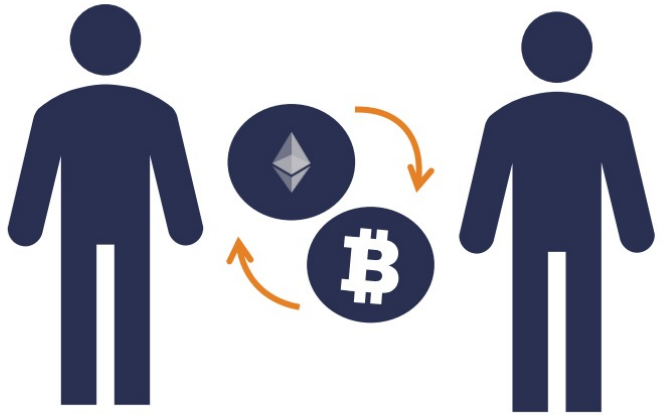
Ether is Ethereum's digital currency which fuels the Ethereum platform

Miners are people who help secure the Ethereum network and verify all transactions that take place in the blockchain.

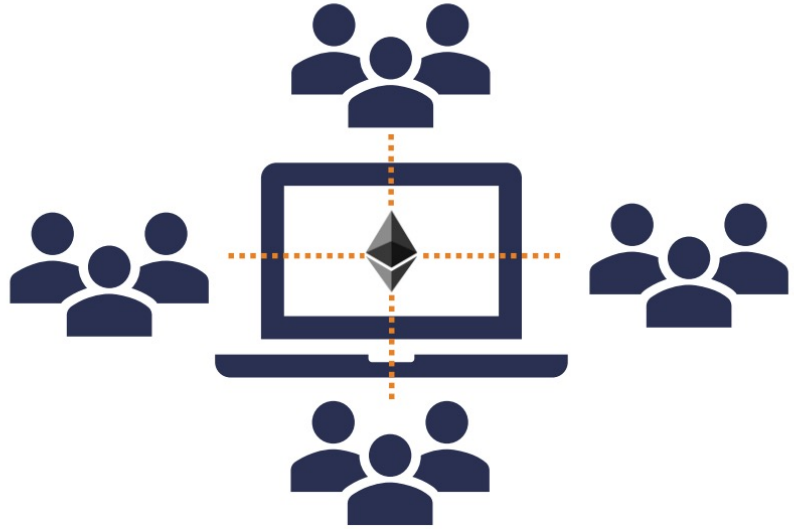
Binary.com

Who uses ether?

Developers who intend to build decentralised applications to run on the Ethereum blockchain



Traders and investors



Users who would like to access and interact with smart contracts on the Ethereum blockchain

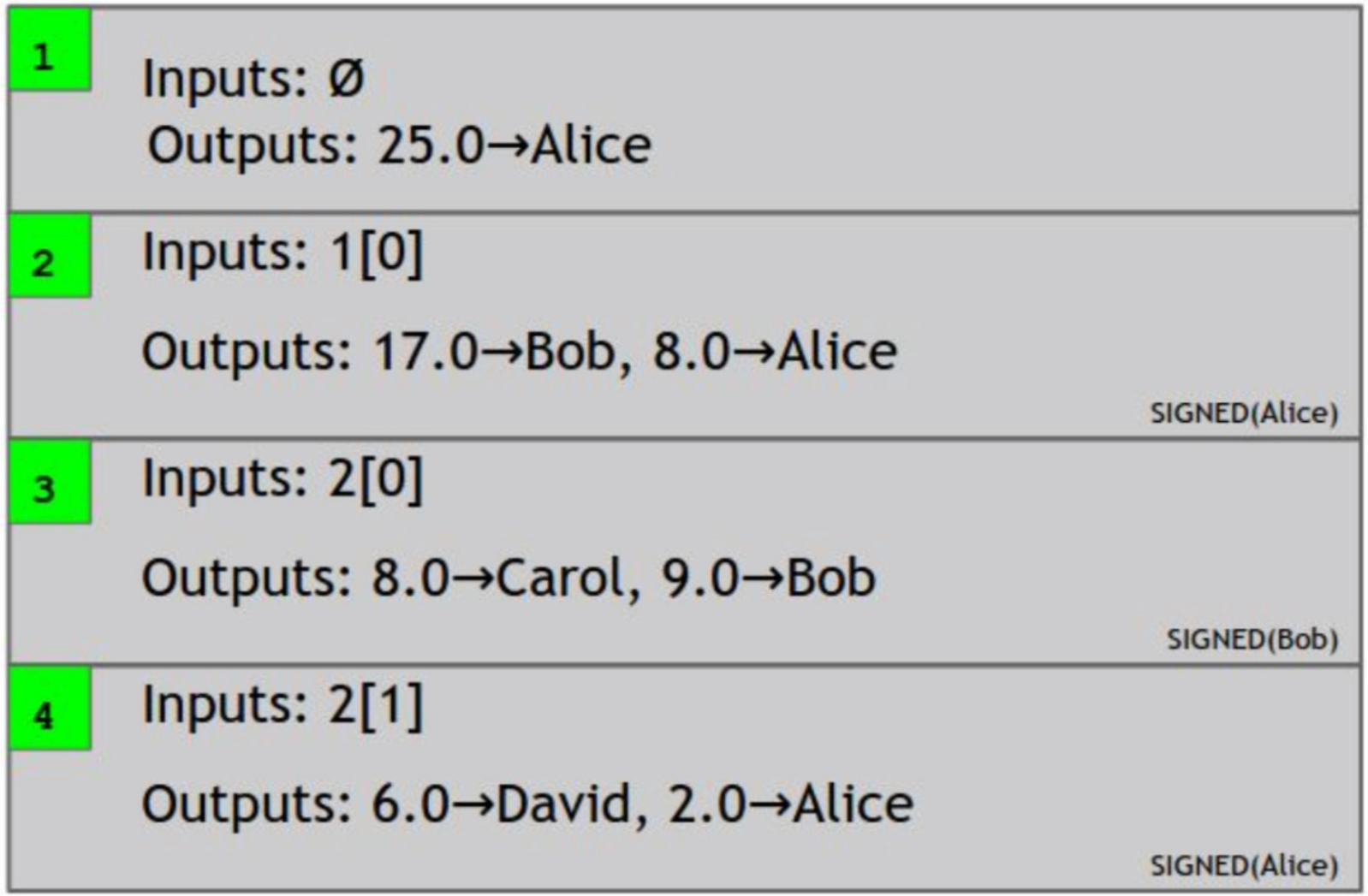
Binary.com

Ledger: **Account-based**

Create 25 coins and credit to Alice$_{\text{ASSERTED BY MINERS}}$
Transfer 17 coins from Alice to Bob$_{\text{SIGNED(Alice)}}$
Transfer 8 coins from Bob to Carol$_{\text{SIGNED(Bob)}}$
Transfer 5 coins from Carol to Alice$_{\text{SIGNED(Carol)}}$
Transfer 15 coins from Alice to David$_{\text{SIGNED(Alice)}}$

- **How to know if a transaction is valid?**
  - E.g., does Alice have the 15 coins to transfer to David?

Ledger: **Transaction-based**

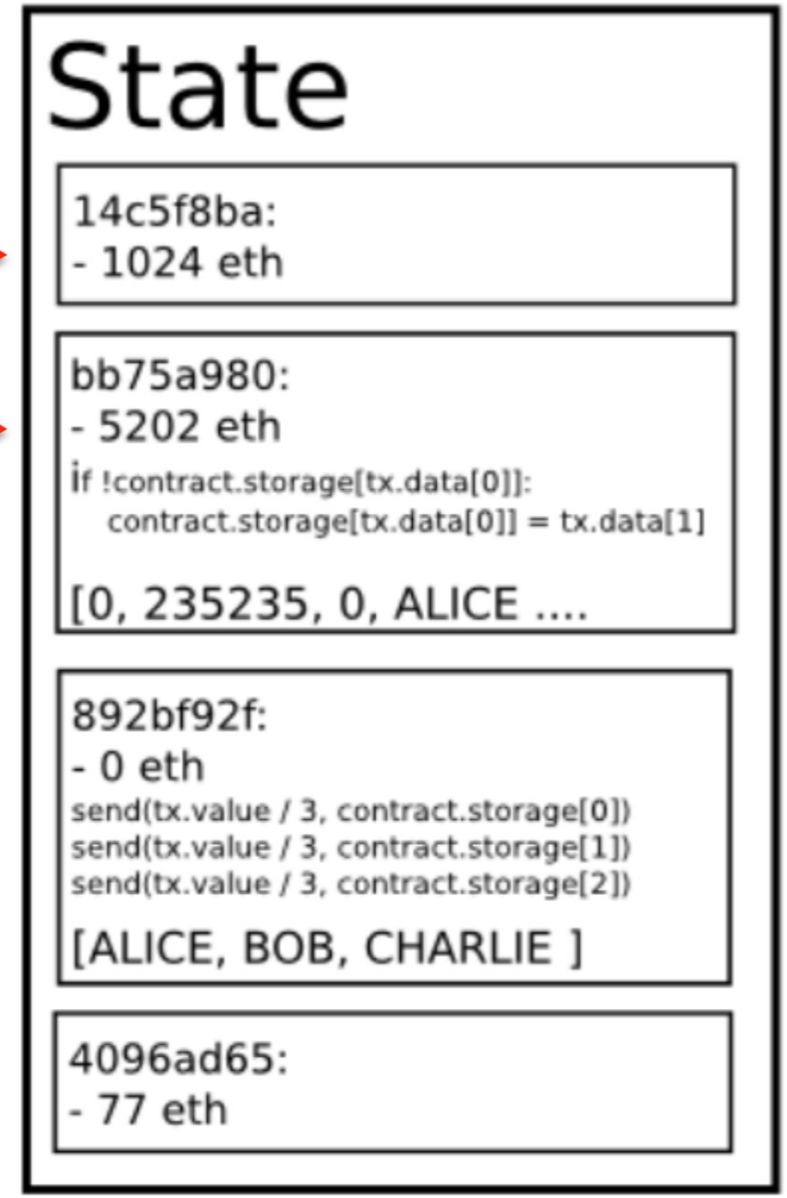


Verifying whether a transaction is valid is easy using "input" **pointers**

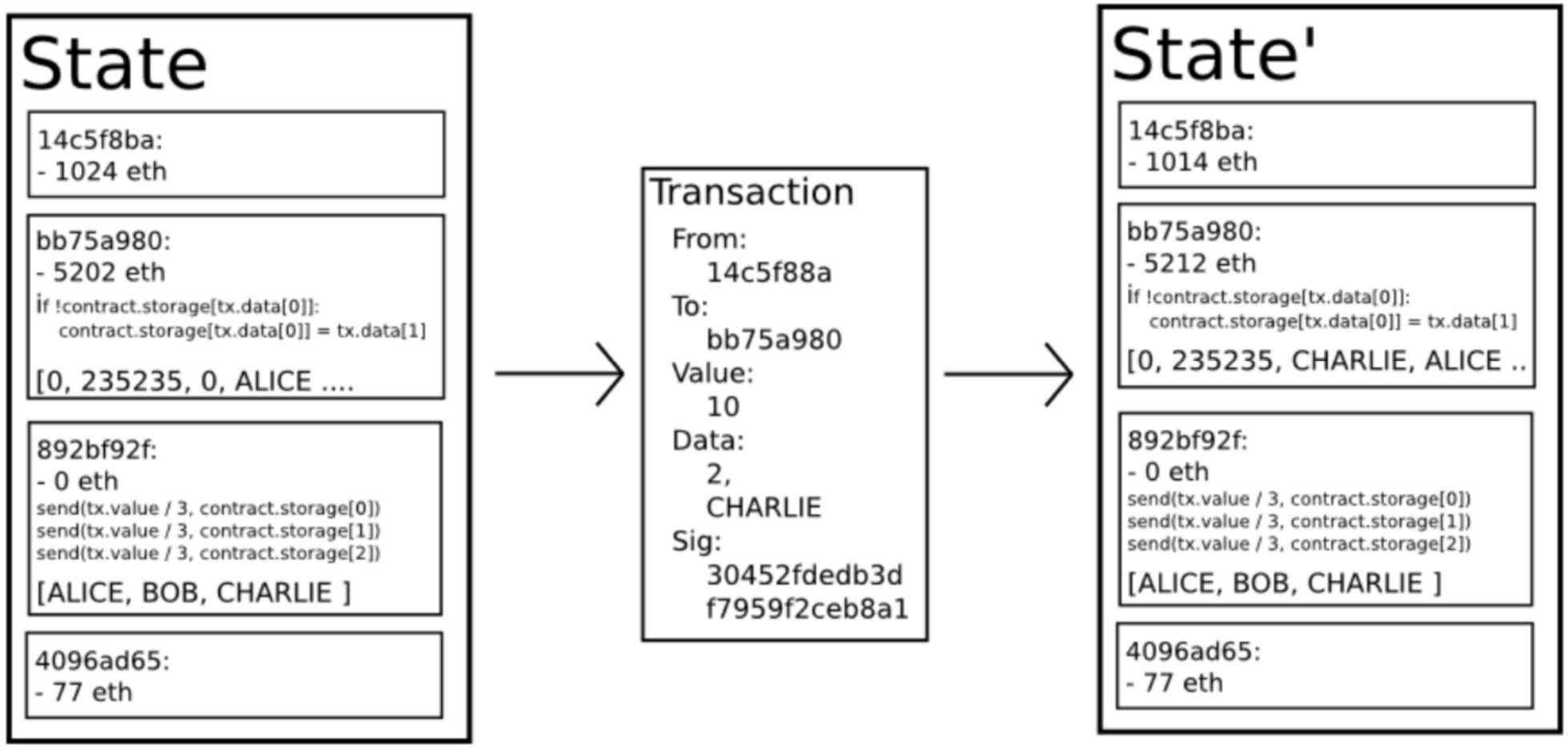
Accounts

- The state of the blockchain = made up of **accounts**
  - Externally owned accounts (like Bitcoin addresses)
  - Contract accounts (corresponding to each contract)

- **Account information:**
  - Nonce: a counter used to ensure each transaction can only be processed once
  - Current balance (in *ether*, currency of Ethereum)
  - Contract code (only for contract accounts)
  - Storage for data (empty by default)

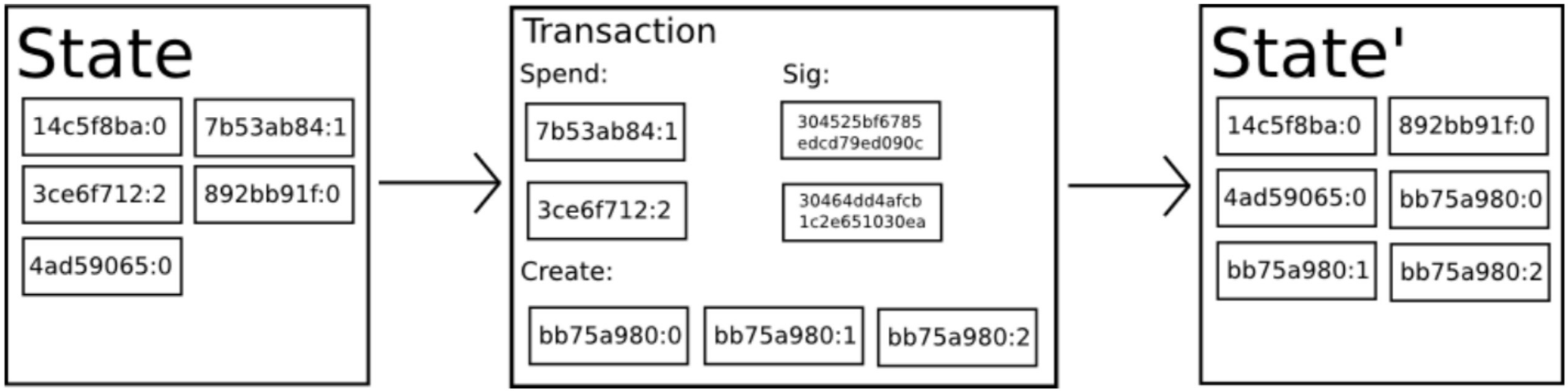


Transaction: Account → Account



- **State transition:** triggered by a transaction or message**,** for direct transfer of value+information between accounts
- If received by a contract account: execute contract code

In Contrast, Bitcoin States



- State of the Bitcoin blockchain = set of all existing bitcoins (and their histories)
- State transition function: message = transaction: bitcoin → address

Ethereum Transactions

- Transaction = sent from an Externally Owned Account
  - **Recipient** of the message
  - A signature identifying **sender**
  - **Amount** of ether to transfer from sender to recipient
  - An optional **data** field
  - A **startgas** value: representing max # of computational steps the transaction execution is allowed to take
  - A **gasprice** value: representing the fee the sender pays per computational step

Messages

- Message = sent from Contract to Contract
  - **Sender** of message (implicit)
  - **Recipient** of the message
  - **Amount** of ether to transfer with the message
  - An optional **data** field
  - A **startgas** value: representing max # of computational steps the transaction execution is allowed to take

State Transition Function

APPLY(state S, transaction TX) -> new state S’

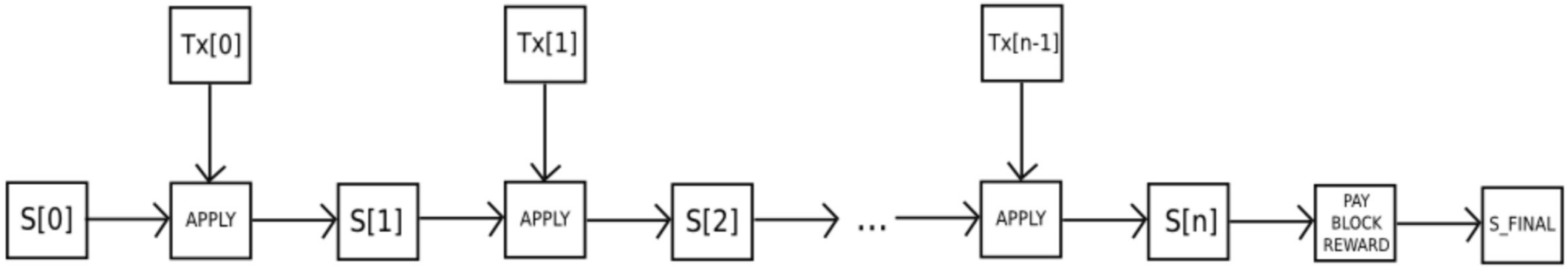
1. Check if TX is well-formed and valid. Else, ERROR
2. Transaction fee = STARTGAS * GASPRICE. Subtract this fee from the sender's balance and increment sender's nonce. If not enough balance, ERROR
3. Initialize GAS = STARTGAS, minus a certain quantity of gas per byte to pay for the bytes in the transaction
4. Transfer the transaction value from sender's account to receiving account
   - If the receiving account does not yet exist, create it.
   - If the receiving account is a contract, run the contract's code either to completion or until the execution runs out of gas
5. If this transfer fails (sender did not have enough money, or the code execution ran out of gas): revert all state changes except the payment of the fees, and add the fees to the miner's account
6. Else, refund the fees for all remaining gas to sender, and send the fees paid for gas consumed to the miner

Blockchain

- Ethereum block contains
  - List of transactions
  - The most recent state
- In contrast, a Bitcoin block contains only a TX list.
  - To get the state, must retrieve all the blocks

- Seem inefficient? Actually not!
  - State is stored as a tree, little changed from the prev state → store only the **difference** (by using Merkle Patricia tree)
  - A **node only needs to store the latest block** (not all the blocks), because this block has all the (latest) blockchain info

Checking if a Block is Valid

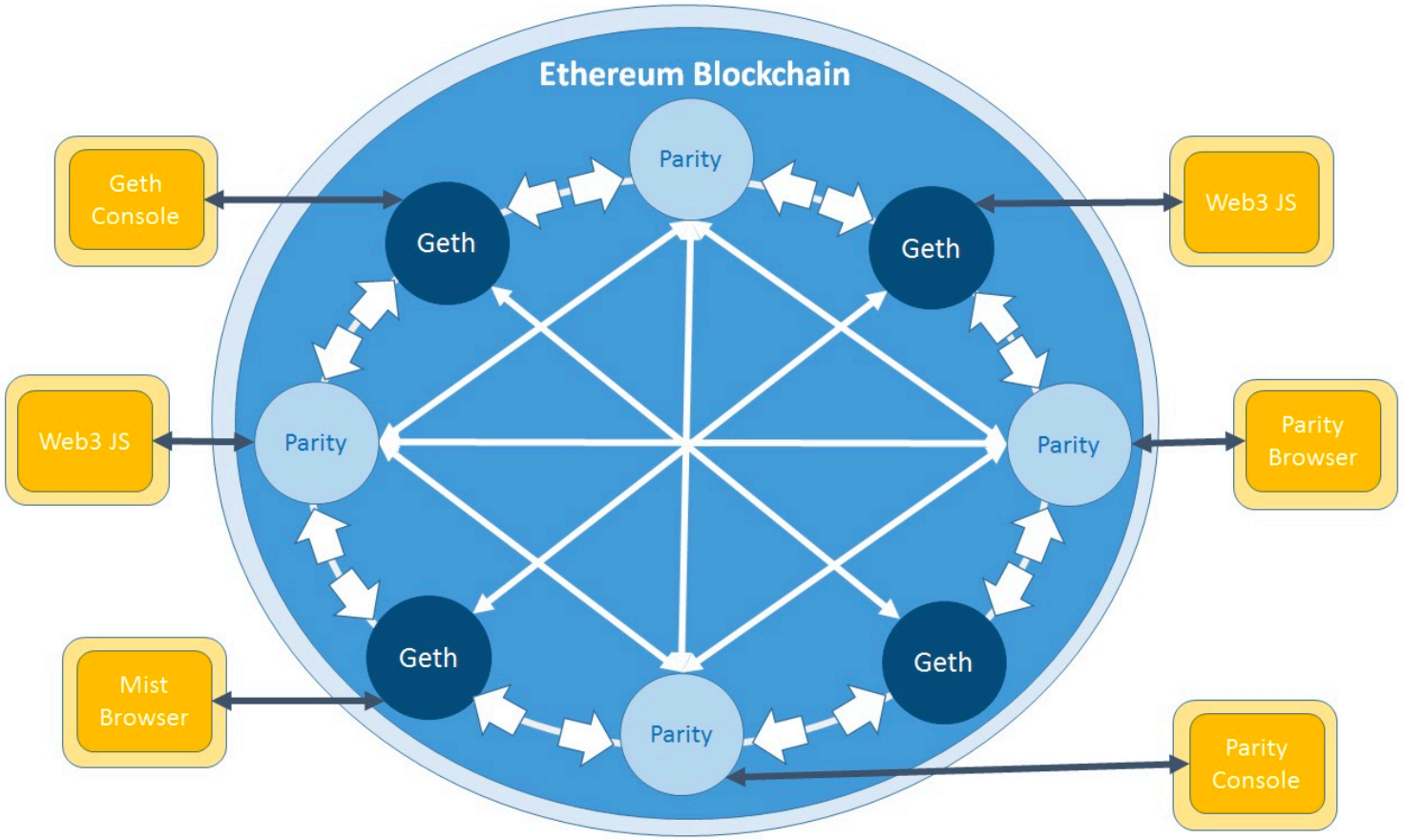


1. Prev block referenced exists and is valid
2. Timestamp is greater than that of referenced prev block and less than 15 minutes into the future
3. Block number, difficulty, transaction root, uncle root and gas limit (various low-level Ethereum-specific concepts) are valid
4. Proof of Work on the block is valid
5. Let S[0] = state at the end of the prev block, TX = block's transaction list, with n transactions.
   - For all i in 0...n-1, set S[i+1] = APPLY(S[i],TX[i])
   - If any APPLY returns error, or if the total gas consumed exceeds the GASLIMIT, return ERROR
6. Let S_FINAL = S[n], but adding the block reward paid to the miner.
7. If Merkle tree root of the state S_FINAL equals the final state root provided in the block header, then the block is valid; else, invalid.

Ether Cryptocurrency Denominations

Unit	Wei	Ether
Wei (wei)	1	10^{-18}
Kwei (babbage)	1,000	10^{-15}
Mwei (lovelace)	1,000,000	10^{-12}
Gwei (shannon)	1,000,000,000	10^{-9}
Twei (szabo)	1,000,000,000,000	10^{-6}
Pwei (finney)	1,000,000,000,000,000	10^{-3}
Ether (buterin)	1,000,000,000,000,000,000	1

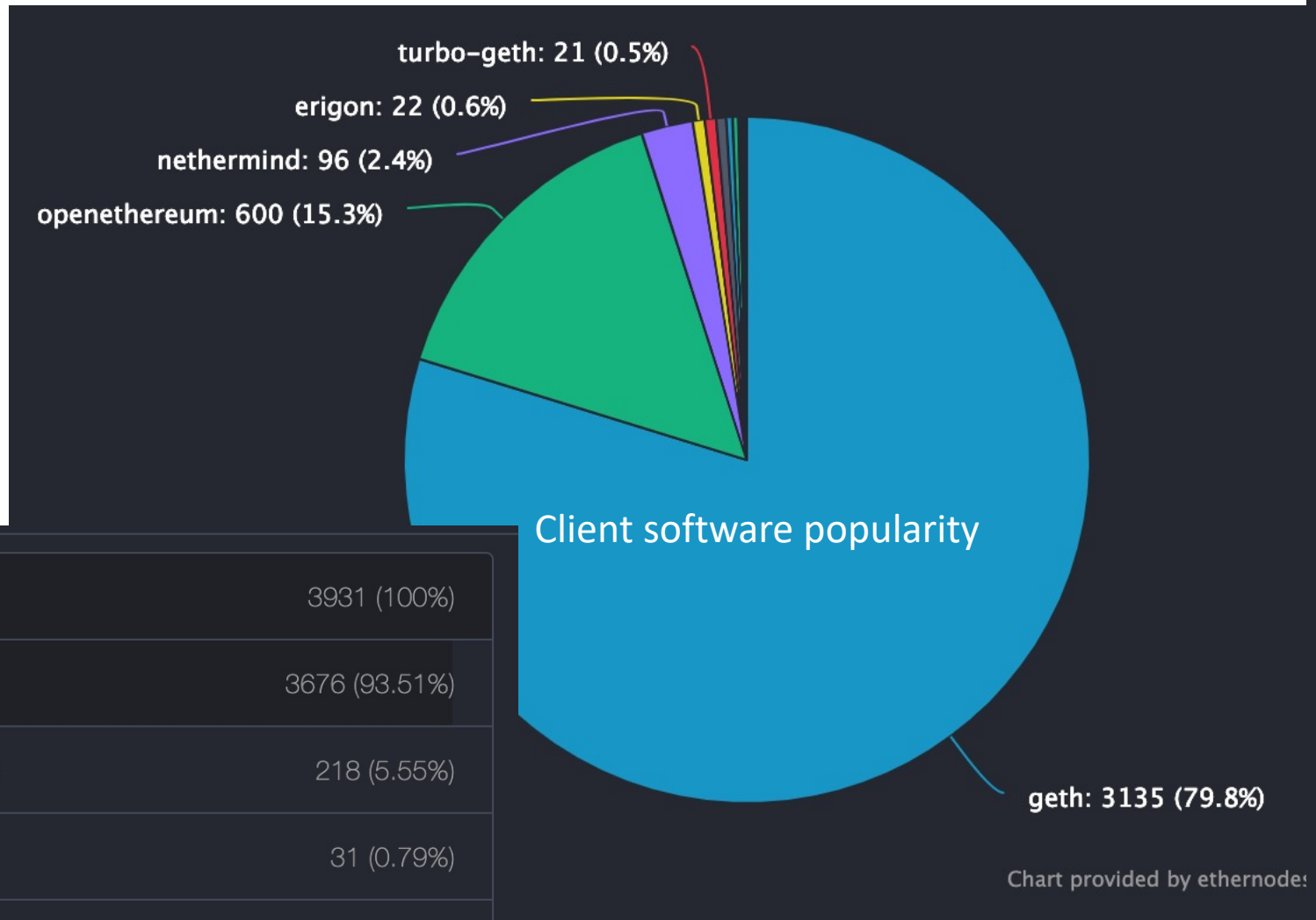
Running a node and how to interact



Client node software

- **Geth** ("**G**o **Eth**ereum"): provided by Ethereum Foundation, written in Go language
- **Parity** (written in Rust Language): provided by Parity Inc, written in Rust language
- **Web3js, Mist browser, Parity browser**: ways to interact with blockchain

Mainnet: ~4000 nodes



Client software popularity

Total	3931 (100%)
Linux	3676 (93.51%)
Windows	218 (5.55%)
MacOS	31 (0.79%)
Unknown	6 (0.15%)

Clients | **Countries** | Sync Status | OS | Netv

Total	3931 (100%)
United States	1503 (38.23%)
Germany	624 (15.87%)
China	293 (7.45%)
Singapore	155 (3.94%)
France	147 (3.74%)
United Kingdom	112 (2.85%)
Canada	108 (2.75%)
Japan	96 (2.44%)
Netherlands	92 (2.34%)
Finland	88 (2.24%)

Ethereum Virtual Machine (EVM)

- The **runtime environment** to run Ethereum smart contracts
- Operates on 256-bit integers (unlike most virtual machines)
- Helps in preventing **Denial-of-Service** attacks
- Each Ethereum node runs its own EVM implementation and has the capability to execute similar instructions
- To run the EVM, there is no centralized control

Ethereum Wallet

- UI-based **software** used to connect to the Ethereum blockchain, to **store**, **accept** and **send ether**
- Internally, it depends on the **Geth client** to seamlessly perform all the operations
- Create **accounts**, deploy **contracts**, **transfer** ether across accounts, and view **transaction details**.

Decentralized Applications (dApps)

- Backend code runs on a blockchain network, as opposed to typical applications where the backend code is running on centralized servers
- Frontend code and user interfaces can be written in any language that can make calls to its backend
- Frontend can be hosted on decentralized storage such as IPFS
- Typically open-source, decentralized, incentivized through providing tokens

Decentralized Finance (DeFi)

- **DeFi** = a kind of dApp providing financial services
- **MakerDAO:** the first DeFi app (2017) – Ethereum-based protocol
  - Allow users to issue a cryptocurrency (**Dai stablecoin**) that's pegged 1-1 to the USD by using digital assets (**Etherreum**) as collateral
  - Allow anyone to take out a loan without relying on a centralized entity
- **Compound Finance** (2018): a decentralized market for borrowers of collateralized loans and lenders who earn interests from borrowers
- **Uniswap** (2018): a decentralized exchange for users to swap between Ethereum tokens

Always keep in mind

Understanding Decentralisation

A system is decentralised if and only if it is:

- Distributed
- Trustless
- Permissionless

Automated Market Makers

- Traditional exchange: buyers and sellers offer up different prices for an asset. When other users find a listed price to be acceptable, they execute a trade and that price becomes the asset's **market price**
  - Due to lack of offers → trading is not always 24/7 and subject to volatility
  - Traders can see the order book and manipulate prices
  - Centralized → have to trust the owner of the exchange

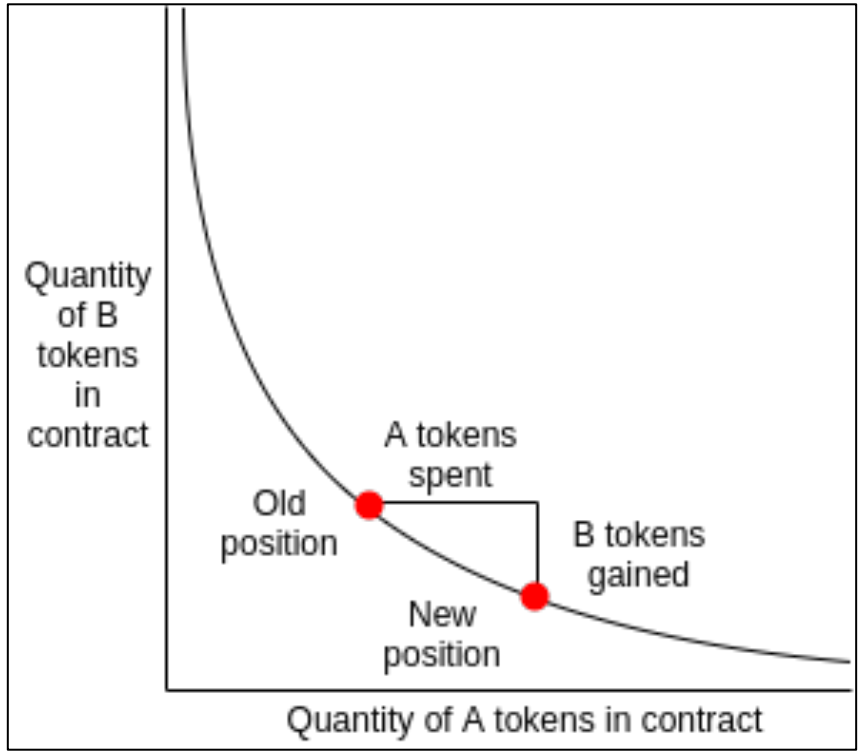
- Automated market makers (AMMs): allow trading 24/7 without permission and **automatically** by using liquidity pools instead of a traditional market of buyers and sellers.
  - AMMs are a DeFi tool unique to Ethereum

AMM: Liquidity Pools and Liquidity Providers

- Liquidity = how easily one asset can be converted into another asset, often a fiat currency, without affecting its market price.
- On decentralized exchanges (DEXs), which are still new, the number of buyers and sellers was **small→ difficult** to find enough people willing to trade on a regular basis
- **With AMMs**: creating **liquidity pools** and incentivizing liquidity providers (anybody can be a provider) to supply assets to these pools
  - Liquidity pool =a shared pot of tokens provided by liquidity providers
  - The price of the tokens in the pool is determined by a **mathematical formula**
  - Users trade against this pool of tokens (the liquidity pool)

AMM: Pricing Model

- **Model**: always keep **x * y** = **constant C**
  - **x, y** is the reserves (number of) of 2 assets swappable
- Consider swapping (ETH, USDC), fair price of **ETH = 2000 USDC**
- Size of the liquidity pool = **same** $amount of ETH and $amount of USDC; for example, x=**5** ETH and y=**10,000** USDC
  - they correspond to $10,000 worth of ETH and $10,000 worth of USDC, respectively
- So in the beginning C = 5 * 10,000 = 50,000.
- If 3 ETH is sold for USDC: need to compute **market price** 1 ETH = **p** USDC?
- We need (x-3)(y+3*p) = 50,000 (always constant)
- (5-3)(10,000+3p)=50,000
- So **p = 5000**

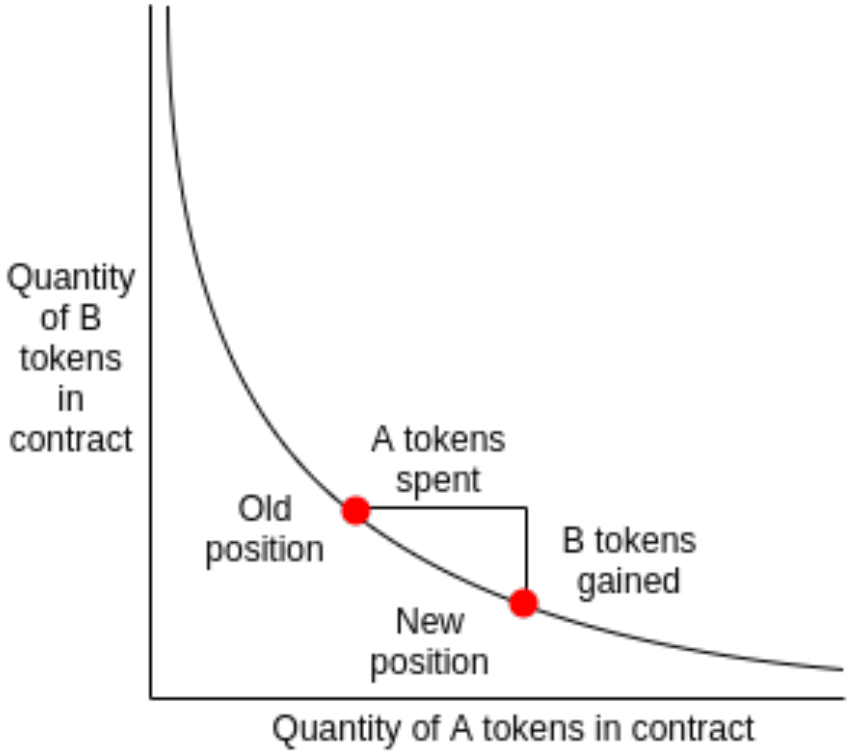


AMM: Front-Running Attack

- Current state (10, 10)
- I want to spend one unit of A, I would get 0.909091 B
- New state (11, 9.090909) (their product = 100)

Front-running attack: a miner can do the following:

- Starting state: (10, 10)
- Miner spends 1 A: (11, 9.090909), gets 0.909091 B
- I spend 1 A: (12, 8.333333); get 0.757576 B
- Miner spends 0.757576 B: (11, 9.090909), gets 1 A
- Miner earns 0.151515 B coins for free, all of which comes out of my pocket



Solution?

- 2 pools: pool (10, 10) if spending A and another pool (10, 10) if spending B
- Starting state: ((10, 10), (10, 10))
- Miner spends 1 A: ((11, 9.090909), (10, 10)), gets 0.909091 B
- I spend 1 A: ((12, 8.333333), (10, 10)); get 0.757576 B
- Miner spends 1.111111 B: ((12, 8.333333), (9, 11.111111)), gets 1 A
- You still lose 0.151515 coins, but the miner *loses* 1.111111 - 0.909091 = 0.202020 coins
- if the purchases were both infinitesimal in size, this is a 1:1 griefing attack,
- The larger the purchase, the larger relative loss the attacker gets

Fungible vs. Non-Fungible Tokens (NFT)

- **Fungible tokens**: tokens are identical. A token worth $1 is the same as another token worth $1
  - E.g., BTC, Ether: every one unit of BTC is identical to another unit of BTC
  - Fungibility is a fundamental property of traditional currencies, like the USD
- **Non-fungible tokens (NFT)**: a token is a digital representation of a unique asset
  - E.g., digital art, and in theory any real-world asset that wants to be traded digitally
  - used as digital proof-of-ownership of underlying assets.
- ERC-20 is Ethereum standard for Fungible Token
- ERC-721 is Ethereum standard for Non-Fungible Token

Co-founder Gavin Wood

- PhD in Computer Science
- Creator of **Solidity** (the smart contract language for Ethereum), the **EVM**, and Ethereum's first **testnet**
- Left Ethereum in 2016 to work on Web3 Foundation and its flagship product, **Polkadot**
- **Polkadot**: a framework for building an Internet of interoperable blockchains, based on **Proof of Stake**

Gavin Wood

Wood speaking in 2015

Born	Gavin James Wood April 1980 (age 41) Lancaster, Lancashire, England, United Kingdom
Nationality	British
Education	Lancaster Royal Grammar School
Alma mater	University of York

Co-founder Charles Hoskinson

- Studied **Math** in college
- **CEO** of the Ethereum startup in December 2013
- **Left** against Vitalik Buterin's view of making Ethereum non-profit
- He then created a programmable blockchain ecosystem called **Cardano**
- **Cardano** (a **Proof-of-Stake** blockchain): currently considered the biggest rival to Ethereum

Charles Hoskinson

Born	1987 or 1988 (age 33–34)[1] Hawaii, USA[*citation needed*]
Known for	Founder of Cardano, co-founder of Ethereum
Website	iohk.io

Ethereum 2.0

- Currently underway for a major upgrade to Ethereum 2.0 or Eth2
- **Transition** from Proof of Work to **Proof of Stake**
- **Purpose**: to scale up the blockchain
  - increase transaction throughput from currently 15 transactions/second to **tens of thousands of transactions/second**