CSE446: Blockchain & Cryptocurrencies

Lecture – 13: Tendermint & Ethereum - 1

Inspiring Excellence

Agenda

- Tendermint
- Ethereum
- Motivations behind Ethereum
- Ethereum History
- Ethereum Components

Tendermint

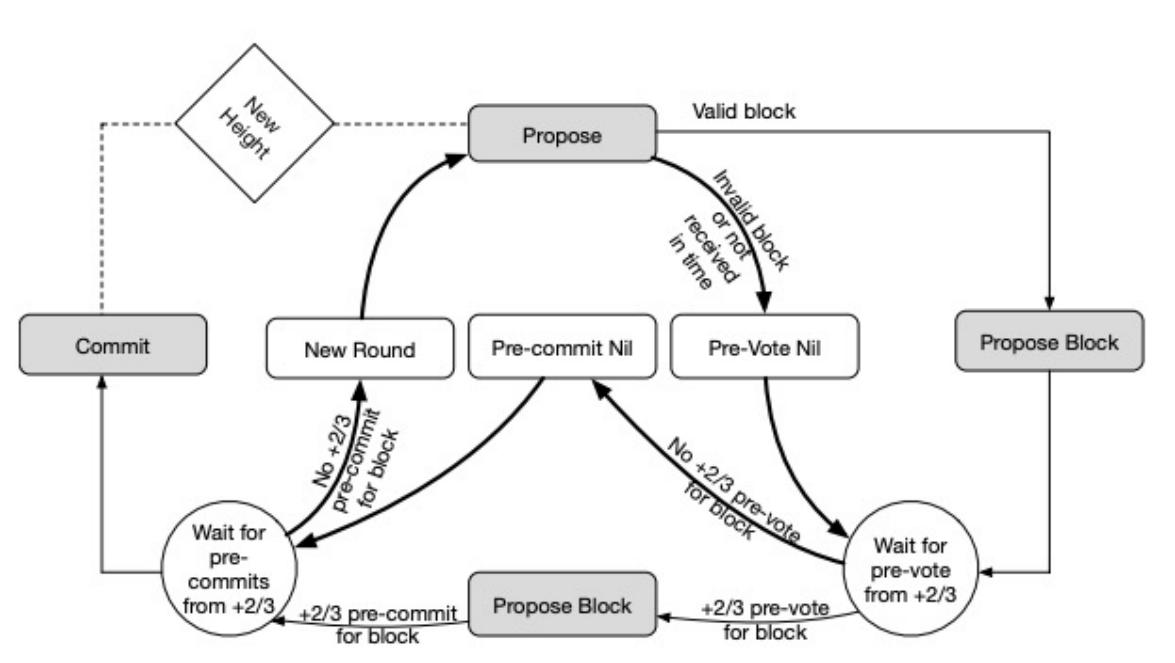
- Tendermint is the first PoS based BFT consensus algorithm
- It consists of two major components:
  - a consensus engine known as Tendermint Core and
  - its underlying application interface, called the Application BlockChain Interface (ABCI)
- The Tendermint core is responsible for deploying the consensus algorithm
- The ABCI can be utilised to deploy any blockchain application using any programming language

Tendermint

- The consensus algorithm is a round-based algorithm where a proposer is chosen from a set of validators
- The proposer itself is selected using a deterministic round-robin algorithm
  - The algorithm relies on the voting power of the validators
- The voting power, on the other hand, is proportional to the security deposit of the validators
- In each round , the proposer proposes a new block for the blockchain at the latest height

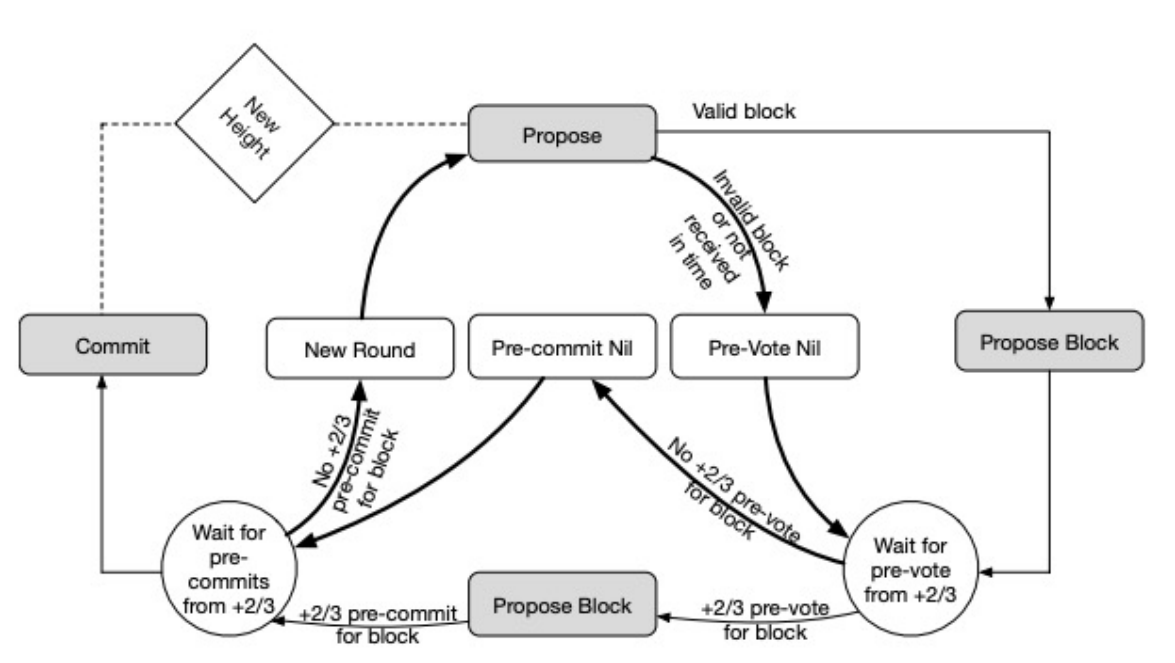
Tendermint

- The consensus algorithm consists of three steps: propose, pre-vote, and pre-commit
- Each round is equally divided into three slots for these three steps
- These steps signify the transition of states in each validator



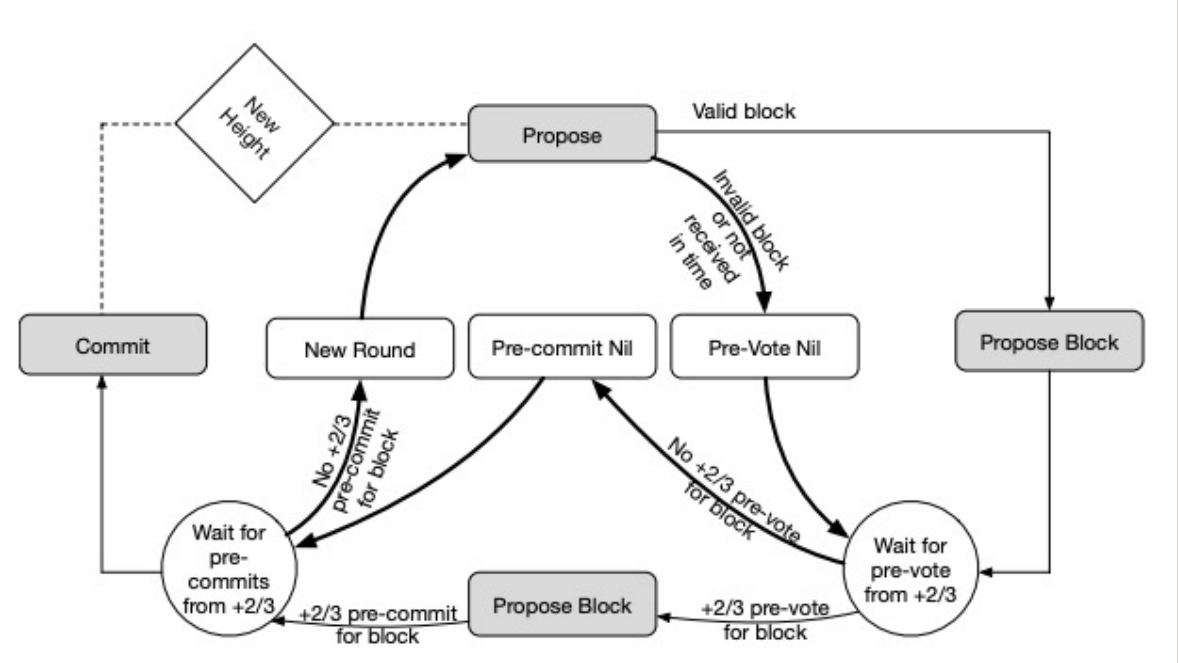
Tendermint

- At the beginning of each round, a new proposer is chosen to propose a new block
- The proposed block needs to go through a two-stage voting mechanism before it is committed to the blockchain



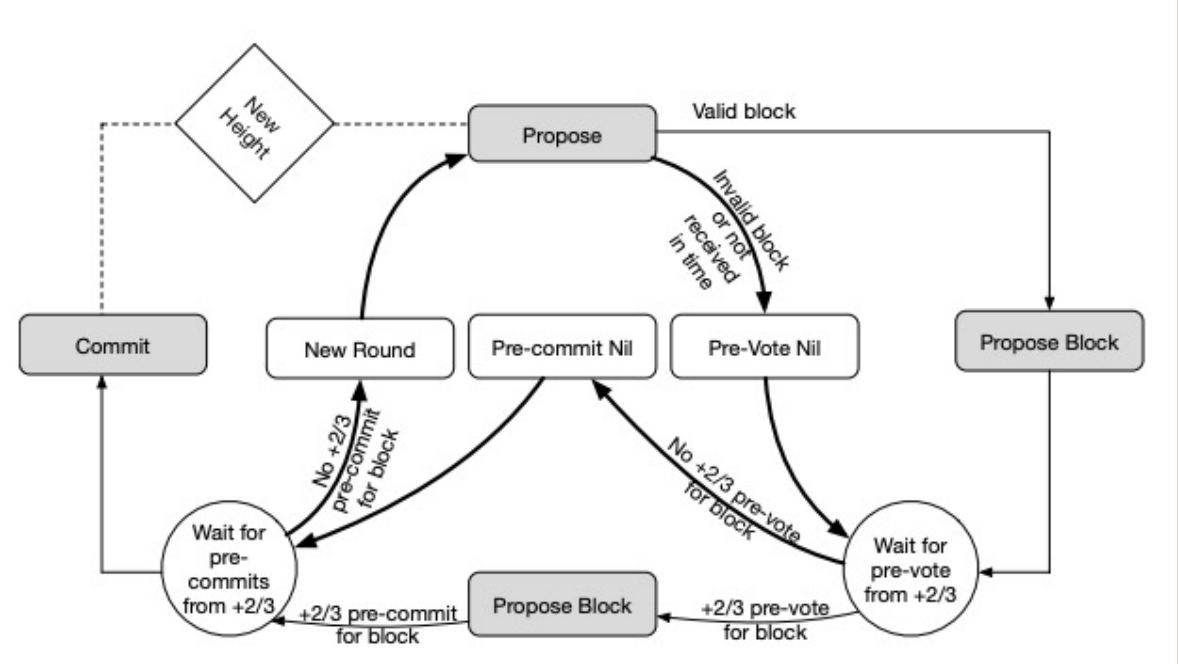
Tendermint

- When a validator receives the proposed block, it validates the block at first, and pre-votes it
- If the block is not received within the propose timer or the block is invalid, the validator submits a special vote called Pre-vote nil
- Then, the validator waits for the pre-vote interval to receive pre-votes from the supermajority (denoted as +2/3) of the validators



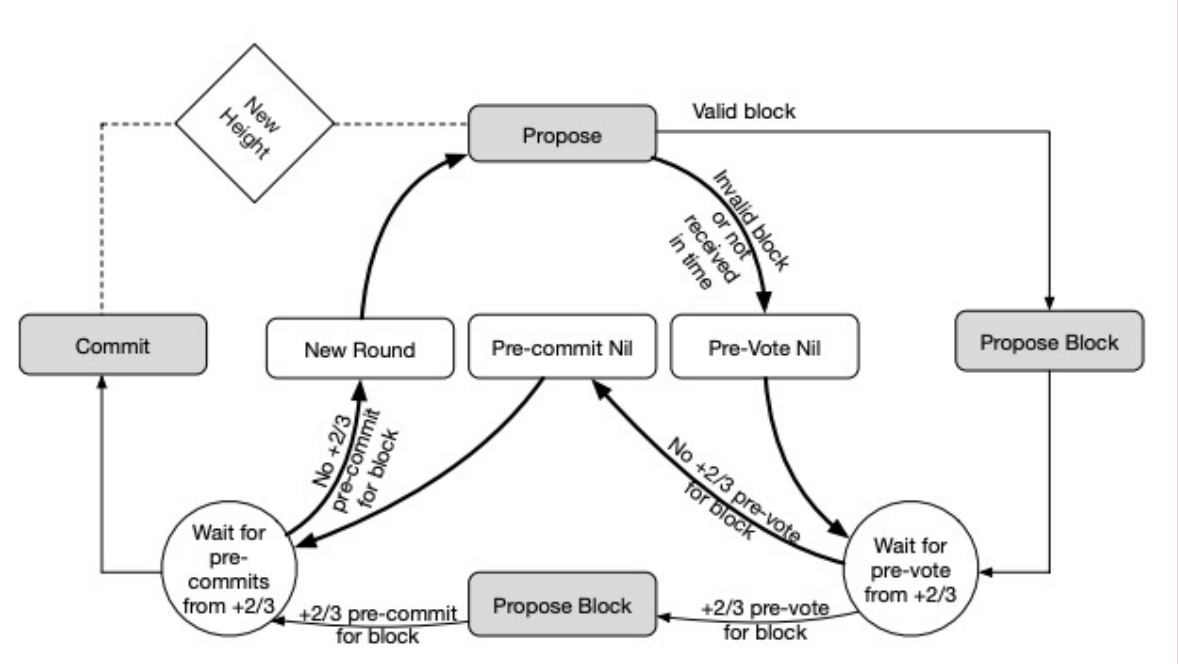
Tendermint

- A +2/3 pre-votes signifies that the super-majority validators have voted for the proposed block
- This implies their confidence on the proposed block and is denoted as a *Polka* in Tendermint
- At this stage, the validator pre-commits the block



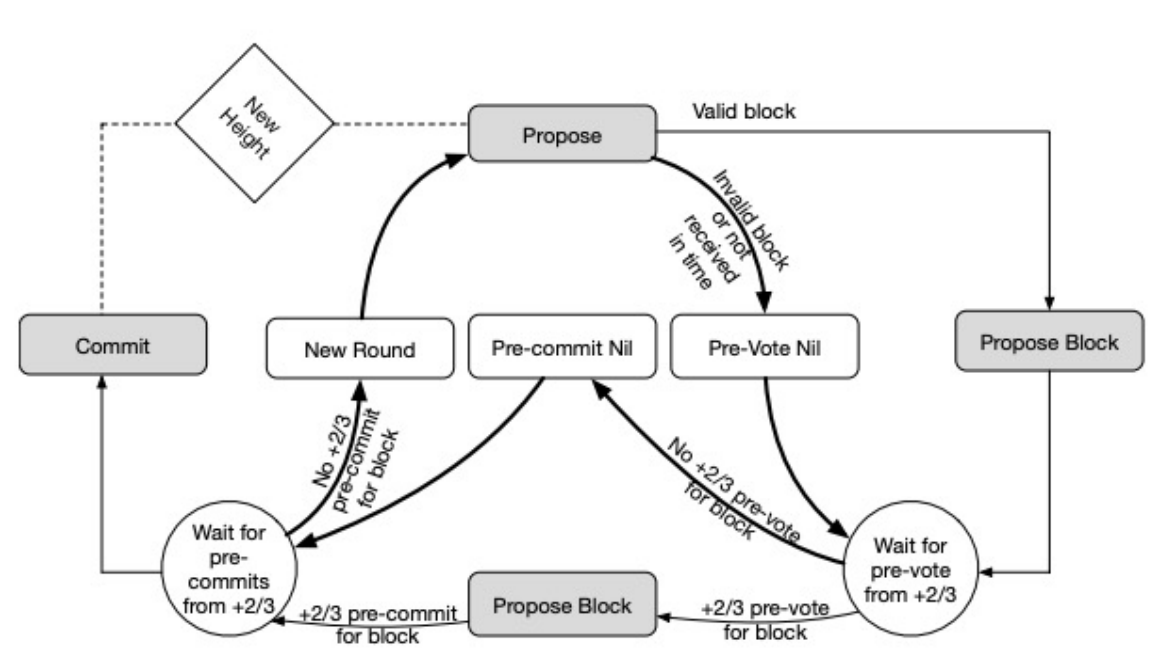
Tendermint

- If the validator does not receive enough pre-votes for the proposed block, it submits another special vote called Pre-commit nil
- The validator waits for the pre-commit time-period to receive +2/3 pre-commits from the supermajority of the validators



Tendermint

- Once received, it commits the block to the blockchain
- If +2/3 pre-commits not received within the pre-commit time-period
  - the next round is initiated where a new proposer is selected, and the steps are repeated



Tendermint

- To ensure the safety guarantee of the algorithm, Tendermint is also coupled with locking rules
- Once a validator pre-commits a block after a polka is achieved, it must lock itself onto that block
- Then, it must obey the following two rules:
  - it must pre-vote for the same block in the next round for the same blockchain height
  - the unlocking is possible only when a newer block receives a polka in a later round for the same blockchain height

Tendermint

- Tendermint inherits the properties of any BFT consensus
  - It guarantees that the consensus is secure when f $< n/3$
  - f is the number of byzantine nodes and n are total nodes
- This implies that conflicting blocks will never be committed at the same blockchain height
- In other words, Tendermint guarantees that no fork will occur under this assumption

Tendermint

- Tendermint has one particular weakness
- It requires 100% uptime of its +2/3 (super-majority) validators
- If more than one-third (+1/3) are validators are offline or partitioned, the system will stop functioning

Ethereum

Bitcoin review

- Each block is a list of transactions
  - Each transaction consumes one or more inputs;
  - Each transaction includes a set of outputs (amount + destination)
  - Input consumption has conditions (e.g., valid script, typically enforcing valid signature)

Bitcoin review

- Each block is a list of transactions
  - Each transaction consumes one or more inputs;
  - Each transaction includes a set of outputs (amount + destination)
  - Input consumption has conditions (e.g., valid script, typically enforcing valid signature)

- In practice, each input/output has script:
  - ScriptPubKey (outputs), ScriptSig (inputs)

Bitcoin review

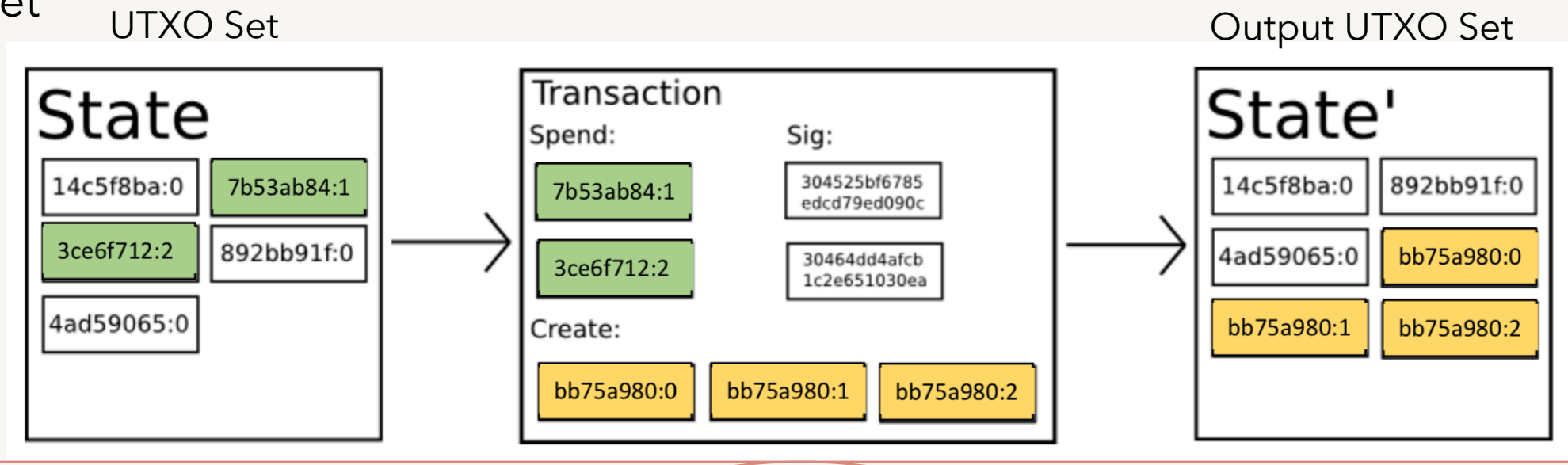
- Bitcoin script allows us to attach conditions to payments
  - However, script is deliberately limited
  - Highly limited access to global state data (chain state)
- Let's consider each Bitcoin transaction as a state transition function
- What is a state transition function?
  - A machine remains in a state (input state)
  - An instruction (input) causes the machine to do something (some operations)
  - The operations generates an output (output state)

Bitcoin review

- For bitcoin
  - What is the input state?
  - What is the output state?
  - What does a Transaction (an input) do to the state?

Bitcoin review

- Input state: list of coins available for spending (UTXO set)
- Transaction: set of instructions for updating the UTXO set
- Output state: Transfer of Bitcoin to one or more accounts updating the UTXO set



Smart-contracts

- Smart contracts were first proposed in the early 1990s by Nick Szabo, who coined the term
- Smart-contrasts are referred to as
  - "a set of promises, specified in digital form, including protocols within which the parties perform on these promises”

Smart-contracts

- A smart-contract is a computer program or a transaction protocol that is intended to automatically execute, control or document legally-relevant events and actions according to the terms of a contract or an agreement
- The objectives of smart contracts are the reduction of
  - need for trusted intermediaries
  - arbitration costs
  - fraud losses
  - malicious and accidental exceptions

Smart-contracts

- Can Bitcoin be used as a smart-contract platform?
- Yes, Bitcoin script is a ‘contract’ in the sense that it provides programmable conditions for redeeming a coin
- However, conditions are highly limited
- Can we use Bitcoin to
  - pay out a coin iff (if and only if) a user has a signing key?
  - implement a second currency/asset?
  - pay out a coin iff a candidate wins the US election?
  - pay out a coin iff a majority of users vote to invest in a service?

Smart-contracts

- Can Bitcoin be used as a smart-contract platform?
- Yes, Bitcoin script is a ‘contract’ in the sense that it provides programmable conditions for redeeming a coin
- However, conditions are highly limited
- Can we use Bitcoin to
  - pay out a coin iff (if and only if) a user has a signing key? ✅
  - implement a second currency/asset? ✅
  - pay out a coin iff a candidate wins the US election? ❌
  - pay out a coin iff a majority of users vote to invest in a service? ❌

Ethereum

- Ethereum concept was proposed by Vitalik Buterin in 2013
- Basic idea: extend Bitcoin by adding Turing-complete scripting languages, with full access to chain state
- Ethereum is a computing platform on top of a blockchain
  - Equipped with a virtual machine called EVM (Ethereum Virtual Machine)
  - Based on a stack-based architecture with RAM, ROM and arbitrary storage

https://imageio.forbes.com/specials-images/imageserve/609034cec99cb743ece612fc/0x0.jpg?format=jpg&width=1200

Ethereum

- Ethereum supports several new Turing-complete programming scripts/languages:
  - Solidity, Vyper and LLL
- Scripts run inside of the EVM, can call other scripts & each other
- Includes a native token (Ether/ETH) to pay for transactions
- Using these languages
  - Programs (smart-contracts) can be written to be executed in the blockchain
  - Data can be stored in the blockchain

https://imageio.forbes.com/specials-images/imageserve/609034cec99cb743ece612fc/0x0.jpg?format=jpg&width=1200

Ethereum history

- In November 2013, Vitalik Buterin started working on the first version of the Ethereum whitepaper
- Buterin made the first public announcement of Ethereum on the 24th January of 2014 at the Bitcoin conference in Miami
- On the 7th July of 2014, Buterin announced the start of the public crowd sale (Crowd-funding)

https://imageio.forbes.com/specials-images/imageserve/609034cec99cb743ece612fc/0x0.jpg?format=jpg&width=1200

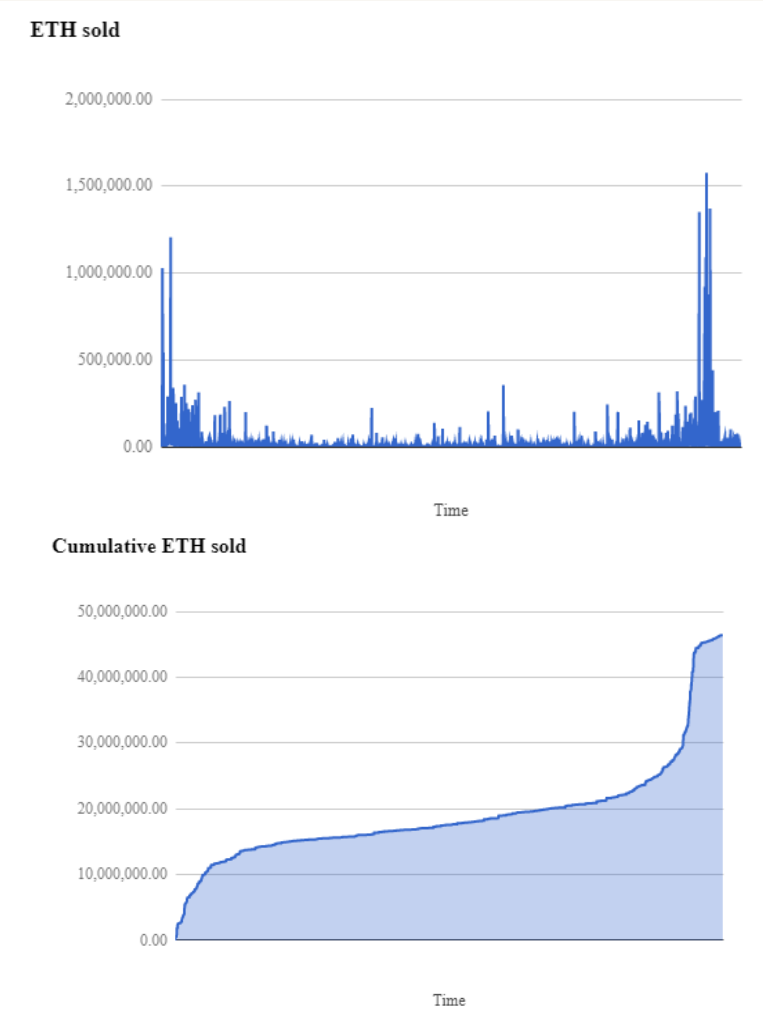
Ethereum history

- The sale lasted 42 days until 02/09
  - For the first 14 days the price was 1 BTC for 2000 ETH
  - After that period the price went up to 1 BTC for 1337 ETH
- In total, ~60 million Ether were sold in exchange for 31,591 Bitcoins
  - Worth around 18.5 million USD at that time
  - Used by the Ethereum foundation

https://imageio.forbes.com/specials-images/imageserve/609034cec99cb743ece612fc/0x0.jpg?format=jpg&width=1200

Ethereum history

- Most ETH were sold at the beginning and the end of the period
- Biggest single purchase during the first period was 500 BTC which equals 1,000,000 ETH
- Smallest purchase was 0.01 BTC
- 43.6% bought 2000 or more ETH
- 0.8% bought 200,000 or more ETH
- 11,901,464.23948 ETH to the development team (https://etherscan.io/address/0x5abfec25f74cd88437631a7731906932776356f9)



Question?