Blockchain Tech

UNSW COMP9243 18s1
Michael Sproul

bitcoin

ethereum

Warning

Blockchain is all the rage, but…

- It's **very easy** to lose money (volatility, scams)
- The technology is still immature
- You may become obsessed ;)

Overview

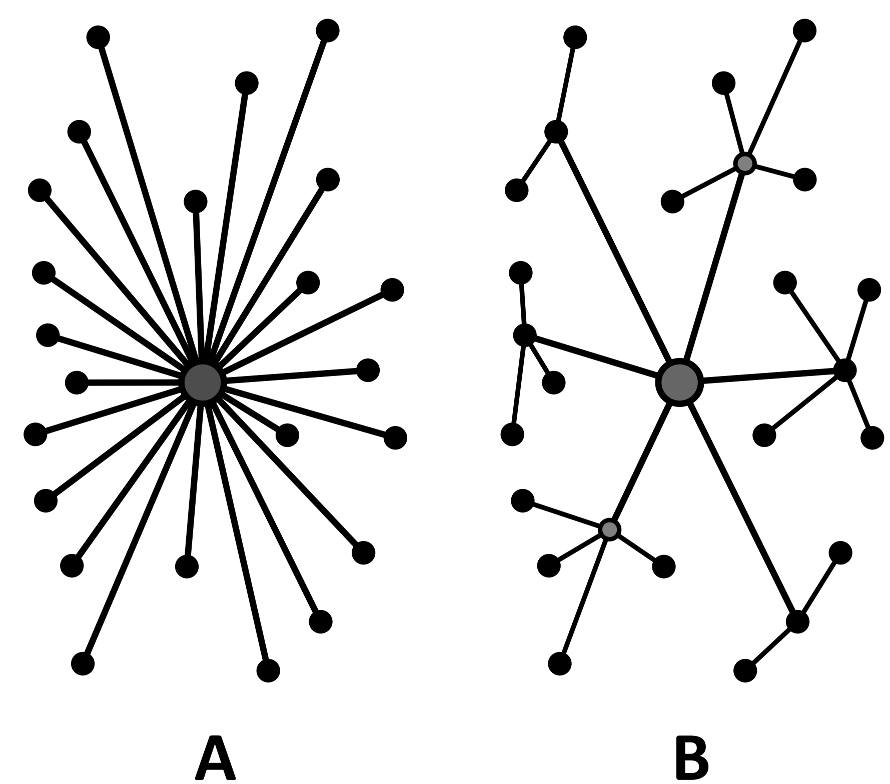
- Bitcoin
- Proof of Work
- Hard Forks vs Soft Forks
- Ethereum and Smart Contracts
- Proof of Stake

Bitcoin – Motivation

- Electronic cash
- Anonymous
- No middlemen or *centralised* control
- Deflationary monetary policy
- Satoshi Nakamoto 2008: *A Peer-to-Peer Electronic Cash System*

Bitcoin – Implementation

- Shared **ledger** storing account balances
- Accounts identified by **public keys**
- **Peer-to-peer** network (decentralised)
- Finite supply: 21 million BTC total
- Open source software

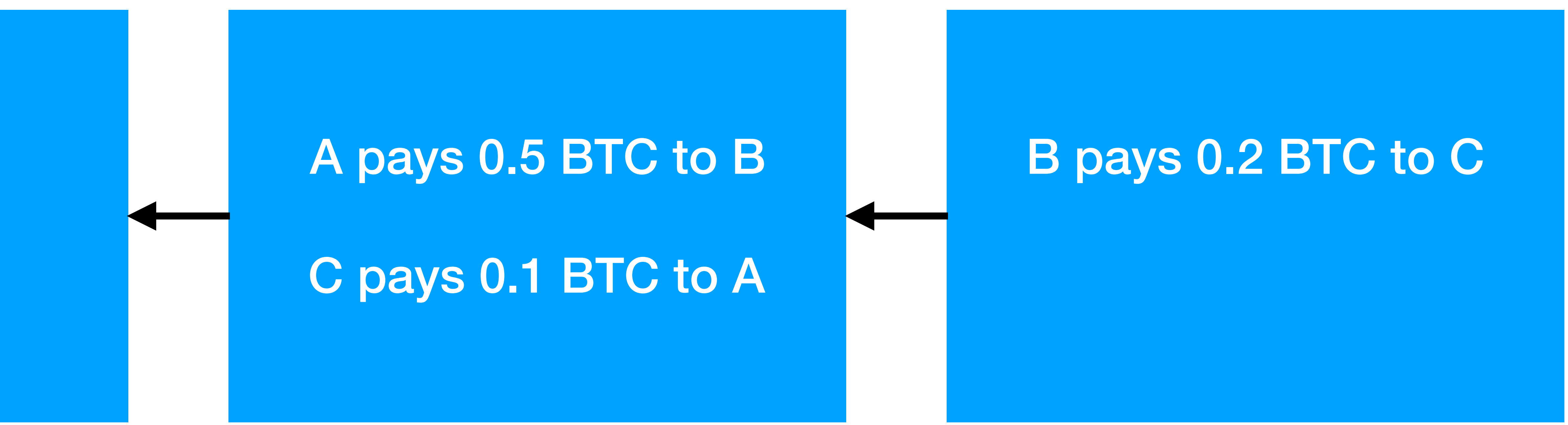


Blockchain Basics

Account Balances

A: 1.0 BTC **B**: 0.0 BTC **C**: 5.1 BTC	**A:** 0.6 BTC **B:** 0.5 BTC **C:** 5.0 BTC	**A:** 0.6 BTC **B:** 0.3 BTC **C:** 5.3 BTC

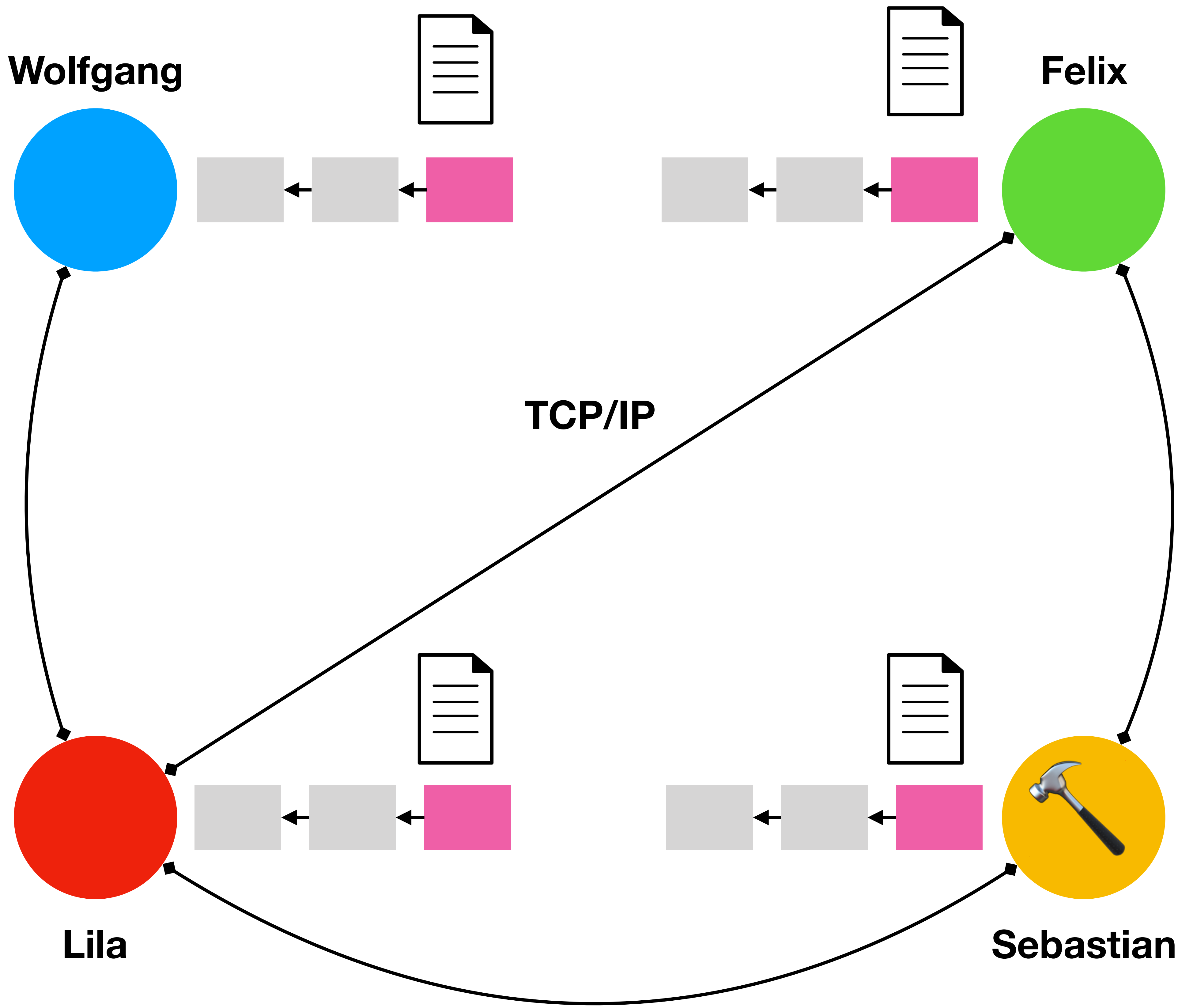
Blockchain State



Blockchain Basics

- Block = bundle of transactions + metadata
- Blockchain = list of blocks, chained together by hash pointers
- Each block includes the SHA256 hash of the previous block – impossible to change an earlier block without invalidating all subsequent blocks

The Bitcoin Network



The Bitcoin Network

- Obtain initial peers manually or via DNS
- Exchange knowledge of peers with other peers
- Maintain TCP/IP connections to reliable peers
- Broadcast transactions and blocks to all connected peers

Bitcoin Transactions

- New coins are created in special **coinbase** transactions that are paid to miners
- All other transactions refer to previous transactions as *inputs*, and specify recipients as *outputs*
- Transactions specify the conditions under which their outputs can be spent using a simple programming language: **Bitcoin Script**

Example Transaction

Wolfgang pays 0.5 BTC to Felix

Inputs

0:
Previous txn: <hash of previous txn where Wolfgang received 0.5 BTC>
Index: <which of the previous outputs Wolfgang wants to spend, e.g. #0>
scriptSig: <Wolfgang's signature> <Wolfgang's public key>

Outputs

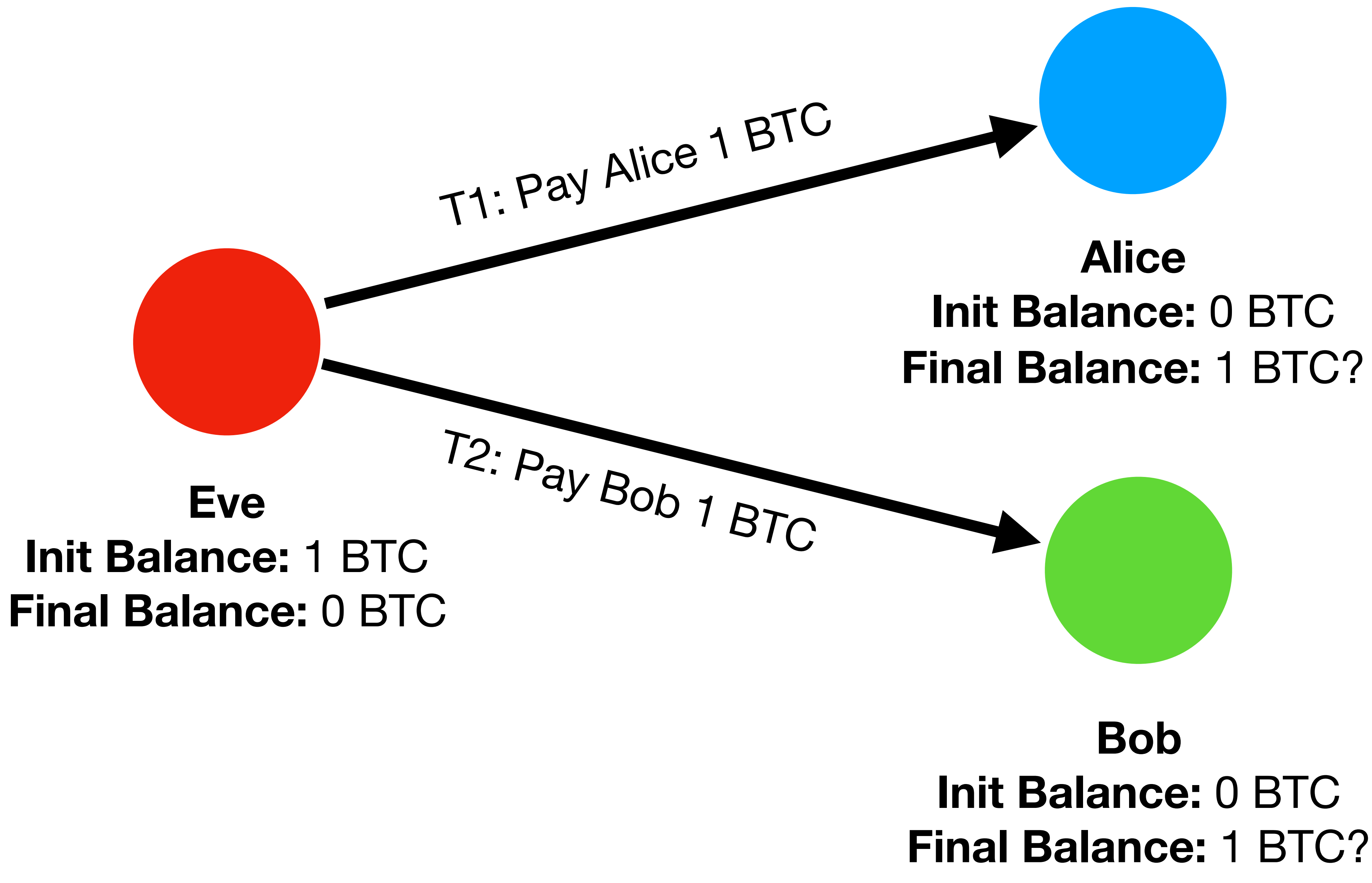
0:
Value: 50,000,000 (0.5 x 10^8 satoshis)
scriptPubKey: OP_DUP OP_HASH160 <Felix's Bitcoin address> OP_EQUALVERIFY OP_CHECKSIG

Signatures prevent anyone except the owner of the coins from spending them

Consensus

- A blockchain is only secure so long as *everyone* agrees on the same chain (and thus transactions)
- Without consensus, malicious users could **double-spend** their coins by sending different transactions to different users

Double-spending



Both transactions cannot be accepted!

Consensus

- Consensus is achieved by all nodes in the network running the same **consensus algorithm** to decide on the state of the blockchain

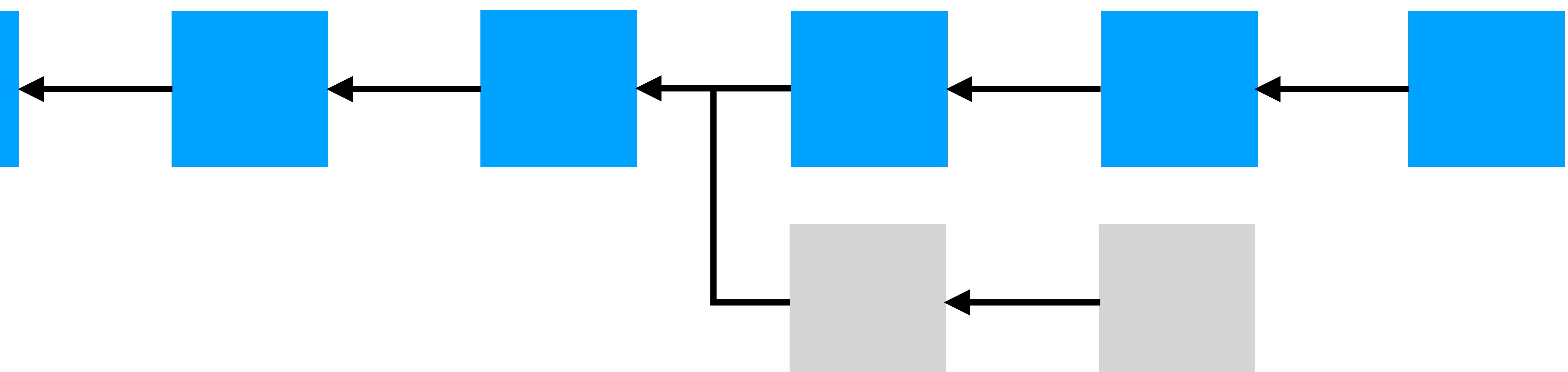
Proof of Work Consensus

- **Miners** solve computational puzzles
- Solving a puzzle grants a miner the right to add a new block to the head of the chain (and claim a **block reward**)
- The **difficulty** of the puzzles automatically adjusts to the rate at which puzzles are solved by the miners, so that the **average time between puzzle solutions remains constant**

Proof of Work Consensus

- Miners must **compete** with each other to solve the puzzles, and this competition prevents any single entity from controlling the chain

- In any case where two chains conflict, the **longest chain** is taken to be valid



Bitcoin's PoW

- Puzzle: find an arbitrary **nonce** to include in your block so that the double SHA256 hash of the block is *less than* a **target** value

  SHA256(SHA256(block)) = 0000e12a… < 0000fffff…

- Difficulty: adjust the **target** every 2016 blocks so that the average block time moves closer to 10 minutes

Economic Security

- Once a transaction is buried several blocks deep, it becomes **very expensive** to revert. An attacker would have to mine a new chain starting from before the transaction occurred, and **outrun** the main chain ("51% attack")
- Miners spend (and are paid) **millions of dollars** per day to secure the network ($10-50M USD/day)
- 6 confirmations ≈ $825,000 (6 * 12.5 * $11000)

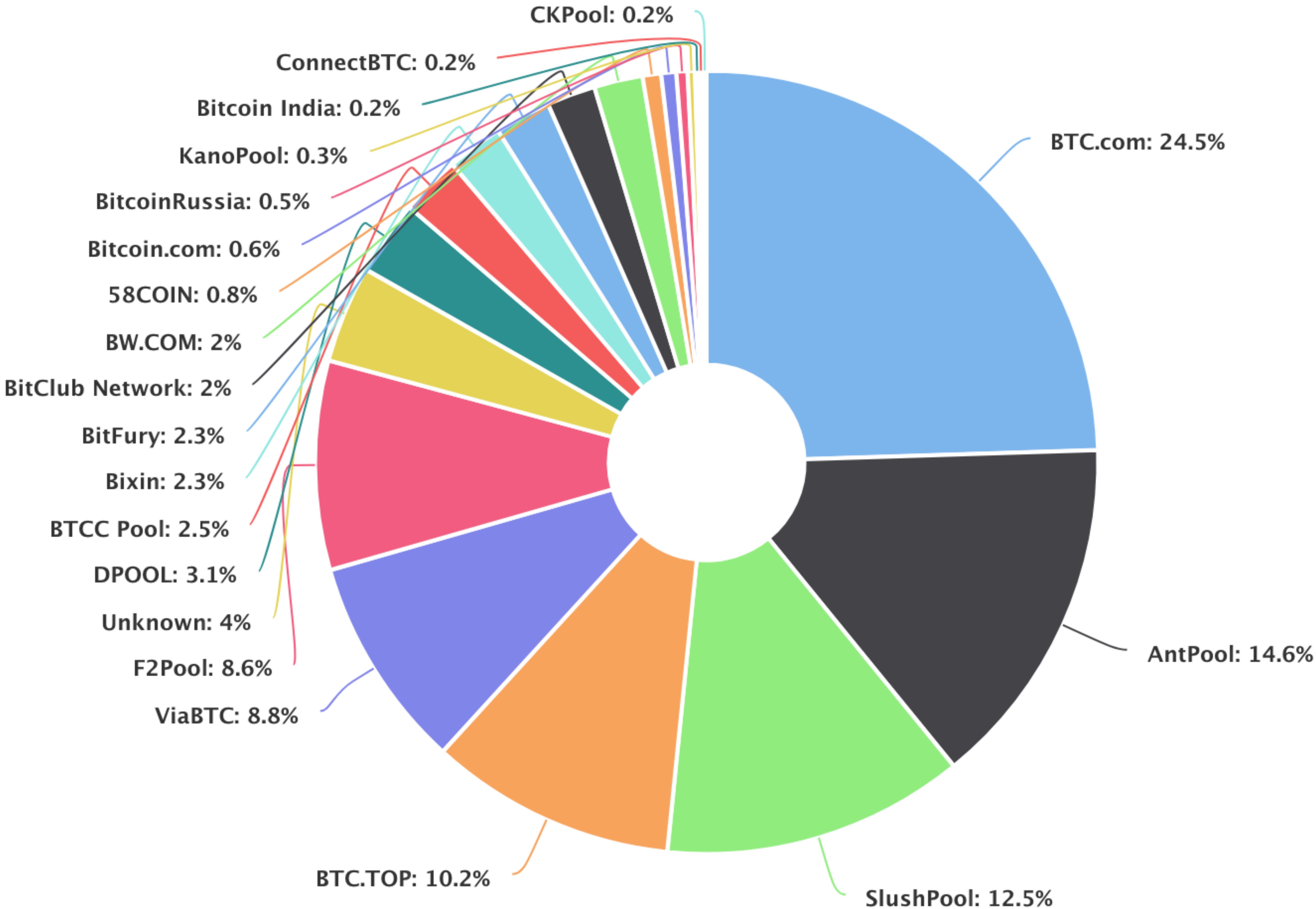
Attacks on PoW

Several (difficult) ways to **double-spend**:

- 51% attack: mine an alternative history.
    - Requires a huge **budget**, but **cost** is minimal if the attack is successful (paid via block rewards)
- Network attack: **partition** your target from the rest of the network and present them with an alternative history, or censor their transactions

Mining

- Economies of scale make **large-scale** mining enterprises much more profitable than small ones
- Lifecycle for a PoW coin: CPUs -> GPUs -> ASICs
- Result is mining **centralisation** (!) around a handful of companies



Percentage of Bitcoin blocks mined by different pools around 28/05/2018, source: blockchain.info

Energy Efficiency

- Miners can (must) spend a large portion of their **mining revenue** on electricity to run their ASICs
- Therefore, energy usage follows USD price (!!)
- Bitcoin uses **6-60 TWh** of electricity per year, which is somewhere between **Ethiopia** (6.7 TWh) and **Switzerland** (58 TWh)
- Energy use mitigated by time and money required to manufacture and deploy new ASICs

What About Upgrades?

- Recall: all nodes in the network run the same validation and consensus logic
- Botched upgrades can cause failures, so use a **flag day** – release a new version of the software that runs the upgraded code once an agreed upon block number is reached 🏴
- What if some node operators refuse to upgrade?

Hard Forks vs Soft Forks

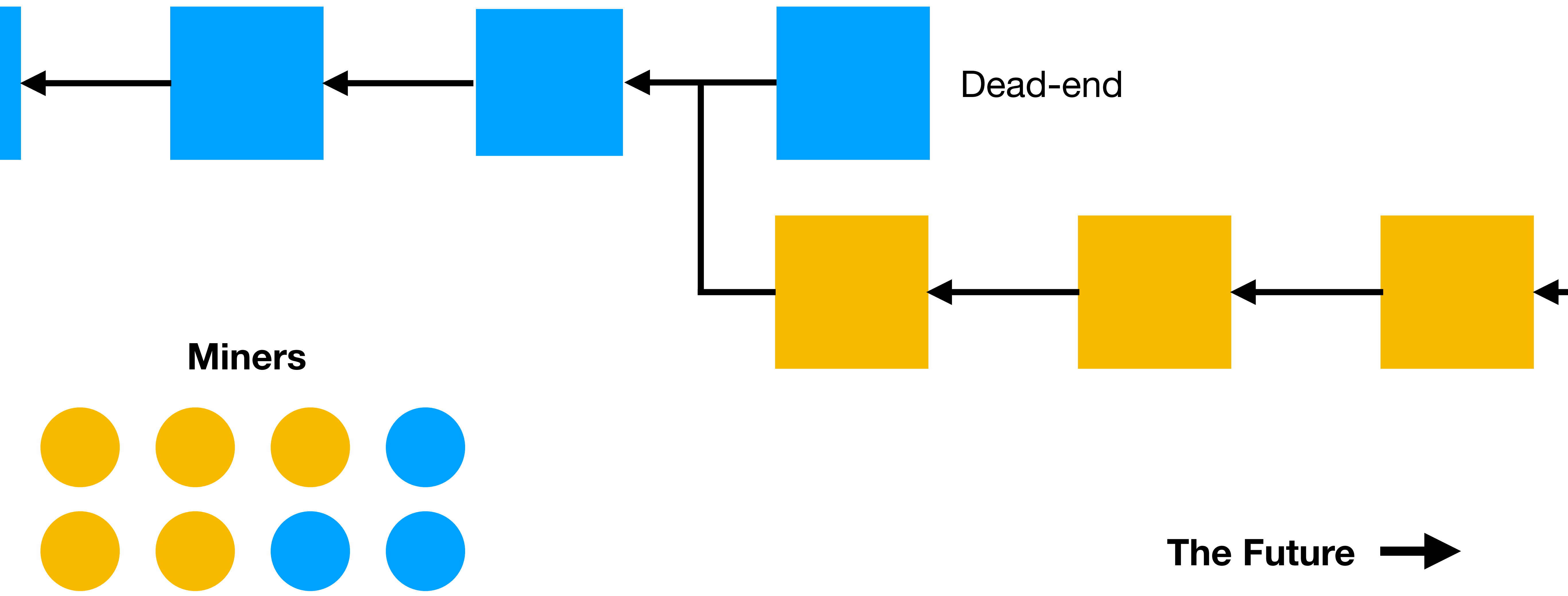
- **Soft fork:** *Restricts* the set of valid blocks. Backwards-compatible with old nodes that don't upgrade. Example: *decreasing* max block size

- **Hard fork:** *Expands* the set of valid blocks. *Not* backwards-compatible with old nodes. Example: *increasing* max block size

- **Chain split:** When the chain splits permanently, which can happen with either a minority (<50%) soft fork, or non-unanimous (<100%) hard fork

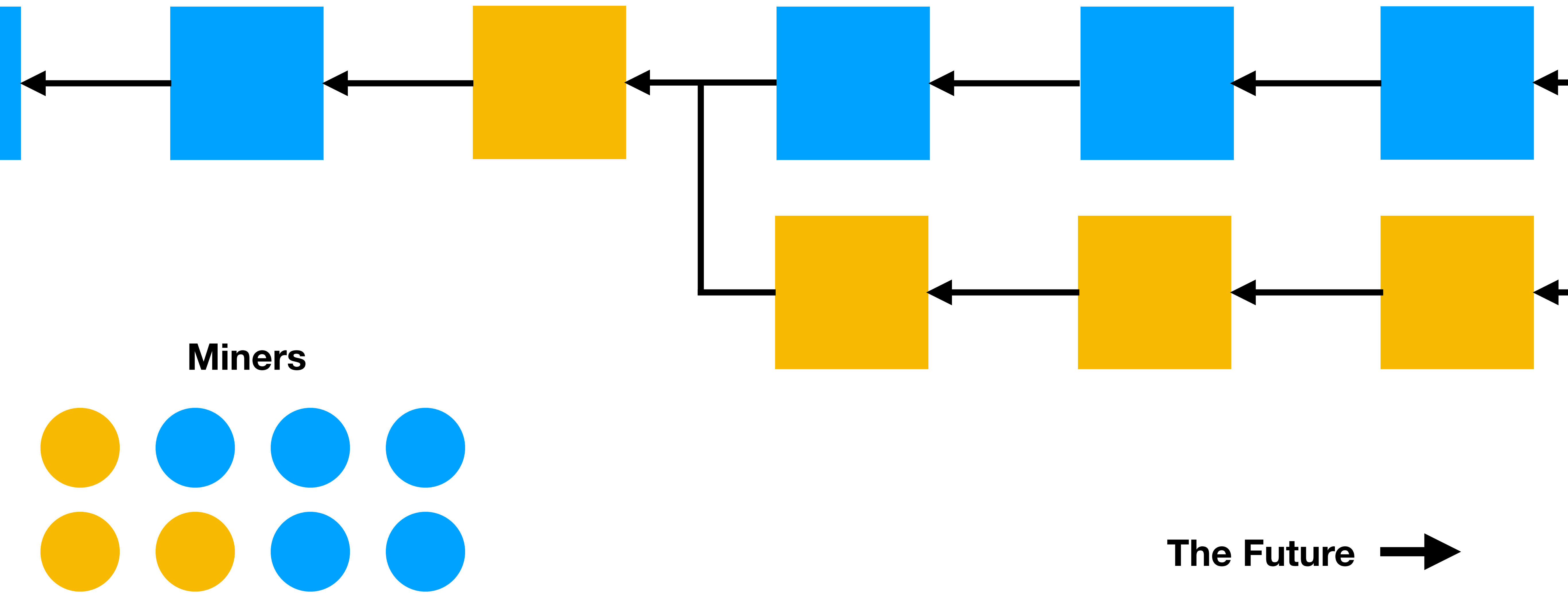
Soft Fork, No Split

Majority of miners follow the new rules, so the new chain wins



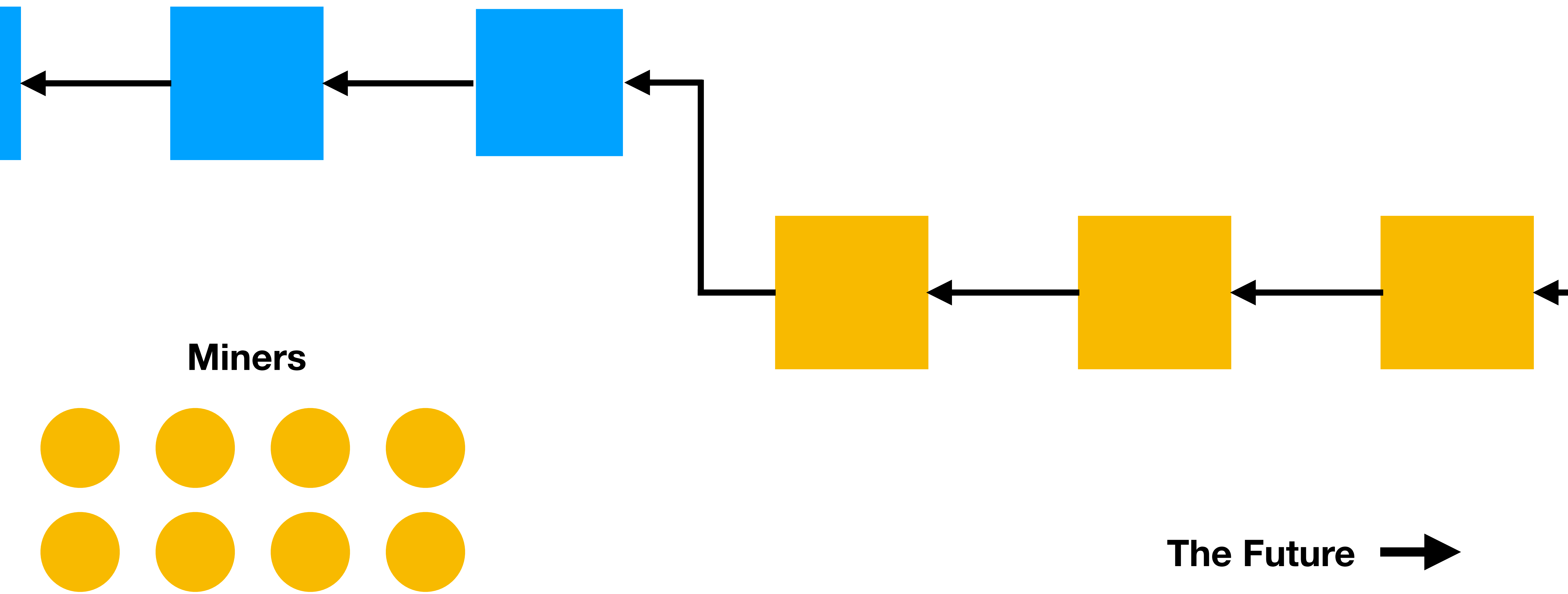
Soft Fork, Split

Majority of miners don't upgrade, upgraded nodes split onto a weaker chain



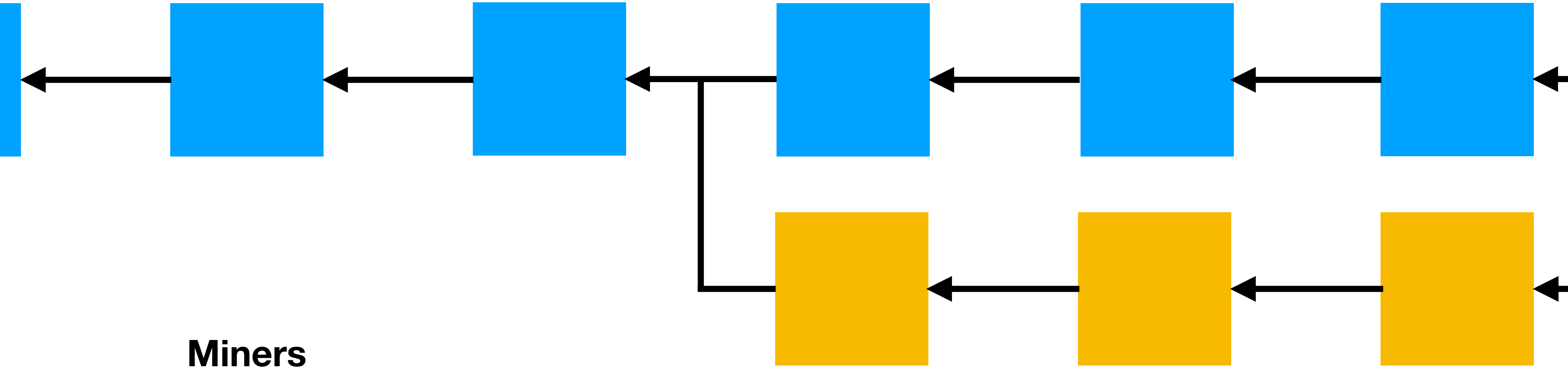
Hard Fork, No Split

All nodes and miners upgrade

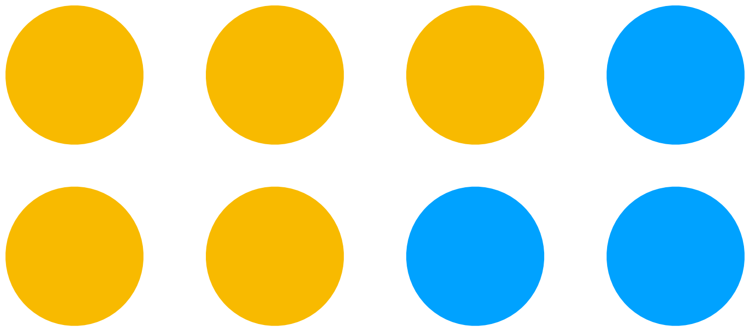


Hard Fork, Split

Some nodes don't upgrade
and continue the original chain



Miners



Health of each chain depends
on its % support

The Future →

Fork Politics

- Bitcoin is quite “fork-conservative”. Bitcoin has only ever upgraded via soft forks (e.g. SegWit)
- Bitcoin hard forks have happened, but mostly for the creation of new derivative coins, e.g. Bitcoin Cash, Bitcoin Gold
- Ethereum has experienced hard forks that split the chain (Ethereum Classic), and hard forks that don’t (Byzantium)
- Forks are POLITICAL. Different people have different ideas about how blockchain networks should operate

Bitcoin’s Strengths

- Fault-Tolerant (no single point of failure)
- Censorship Resistant (infeasible to stop a txn)
- Simple and Stable (compared to competition)

Bitcoin’s Weaknesses

- Generality (blockchains can do more than money)
- Energy Inefficiency (PoW is wasteful)
- Privacy (all data public)
- Scalability (not many transactions/second)

Other Cryptocurrencies

- **Ethereum**: improves generality by adding *smart contracts* which can express more complex applications than Bitcoin Script
- **Proof of Stake coins**: improve energy efficiency by replacing mining with in-protocol rewards and punishments
- **Zcash/Monero**: improve privacy by hiding the details of transactions using sophisticated cryptography

Ethereum

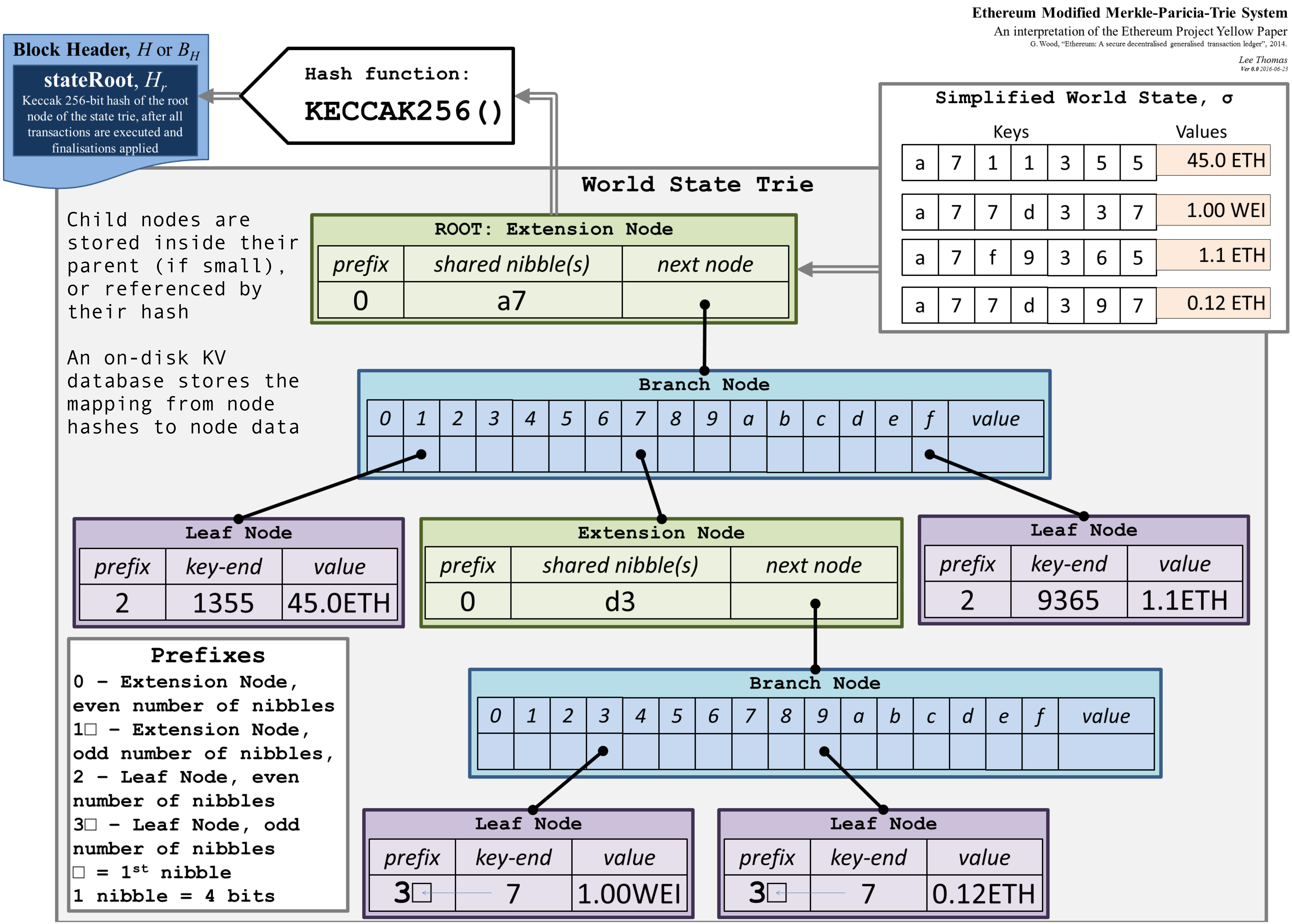
- Uses a blockchain to agree on the state of programmable “world computer”, the **Ethereum Virtual Machine (EVM)**

- **Smart Contracts** are programs written in EVM bytecode that are stored on the blockchain and executed by all the nodes on the network

- Users can send transactions that create new smart contracts, execute existing ones, or transfer funds

Ethereum World State

- World state: 160 bit Ethereum address => Account
- Account (or Contract)
  - **balance:** number of Wei owned by this account (1 Ether = 10^{18} Wei)
  - **storageRoot:** hash of the root of a *Merkle Patricia Trie* for this contract's storage. Storage is itself a map from 256-bit VM addresses to 256-bit values
  - **codeHash:** hash of the VM bytecode for this contract
- World state is also stored in a *Merkle Patricia Trie*


Ethereum Modified Merkle-Paricia-Trie System
An interpretation of the Ethereum Project Yellow Paper
G. Wood, "Ethereum: A secure decentralised generalised transaction ledger", 2014.
Lee Thomas
Ver 0.0 2016-06-23
Block Header, H or B_H
stateRoot, H_r
Keccak 256-bit hash of the root node of the state trie, after all transactions are executed and finalisations applied
Hash function:
KECCAK256()
Simplified World State, σ
Keys
Values
a 7 1 1 3 5 5
45.0 ETH
a 7 7 d 3 3 7
1.00 WEI
a 7 f 9 3 6 5
1.1 ETH
a 7 7 d 3 9 7
0.12 ETH
World State Trie
Child nodes are stored inside their parent (if small), or referenced by their hash
An on-disk KV database stores the mapping from node hashes to node data
ROOT: Extension Node
prefix
shared nibble(s)
next node
0
a7
Branch Node
0 1 2 3 4 5 6 7 8 9 a b c d e f value
Leaf Node
prefix
key-end
value
2
1355
45.0ETH
Extension Node
prefix
shared nibble(s)
next node
0
d3
Leaf Node
prefix
key-end
value
2
9365
1.1ETH
Prefixes
0 – Extension Node, even number of nibbles
1□ – Extension Node, odd number of nibbles,
2 – Leaf Node, even number of nibbles
3□ – Leaf Node, odd number of nibbles
□ = 1st nibble
1 nibble = 4 bits
Branch Node
0 1 2 3 4 5 6 7 8 9 a b c d e f value
Leaf Node
prefix
key-end
value
3□
7
1.00WEI
Leaf Node
prefix
key-end
value
3□
7
0.12ETH

Ethereum Blocks

- Hash of previous block (just like Bitcoin)
- Root hash of the Merkle trie for all transactions in the block
- Root hash of the Merkle trie for the world state after all transactions have been applied to it

Ethereum Smart Contracts

- Each contract has access to its own address space mapping 256-bit addresses to 256-bit values
- EVM assembly is a low-level stack-based language. Instructions include: PUSH, POP, JUMP, MLOAD, MSTORE, ADD, SSTORE, SLOAD
- Each instruction has a **gas cost** that must be paid to execute it. This compensates miners, and ensures that every transaction terminates
- Most contracts are written in high-level languages that compile down to EVM assembly, e.g. **Solidity**, **Vyper**

```
contract MyToken {
    // Map data-structure to store balances
    mapping (address => uint256) public balanceOf;

    // Constructor that gives all the tokens to the contract creator
    constructor(uint256 initialSupply) public {
        balanceOf[msg.sender] = initialSupply;
    }

    // User-invocable function to transfer tokens
    function transfer(address _to, uint256 _value) public {
        // Check that the sender has sufficient balance
        require(balanceOf[msg.sender] >= _value);
        // Check that the receiver's balance won't overflow
        require(balanceOf[_to] + _value >= balanceOf[_to]);
        // Update the balanceOf data structure
        balanceOf[msg.sender] -= _value;
        balanceOf[_to] += _value;
    }
}
```

Ethereum DApps

- **Ethereum Name Service**: decentralised DNS replacement. I registered `comp9243.eth`
- Lots of **Initial Coin Offerings** (ICOs/tokens)
- Decentralised Exchanges (e.g. EtherDelta)
- Identity Management (e.g. uPort)
- Games (e.g. CryptoKitties)

Proof of Stake

- Recall: proof of work mining consumes massive amounts of electricity
- What if we could achieve similar security by using some game theory instead?
- Ethereum is planning to switch to Proof of Stake as soon as possible

Proof of Stake Basics

- Miners replaced by **stakers**
- Instead of buying ASICs, stakers lock funds in **security deposits**
- Stakers **vote** on blocks to be added to the chain, as in a traditional Byzantine Fault Tolerant consensus algorithm
- Stakers are rewarded with new coins for casting valid votes
- Stakers **have their deposit seized** if they misbehave

Slashing Conditions

- We only want one block at a given height, so we **slash a staker's deposit** if they vote on two conflicting blocks at the same height
- We require >2/3 of stakers by weight to vote for a block for it to be considered valid
- Therefore, if two conflicting blocks become valid we know that at least 1/3 of the stakers misbehaved and will lose their deposits

Liveness

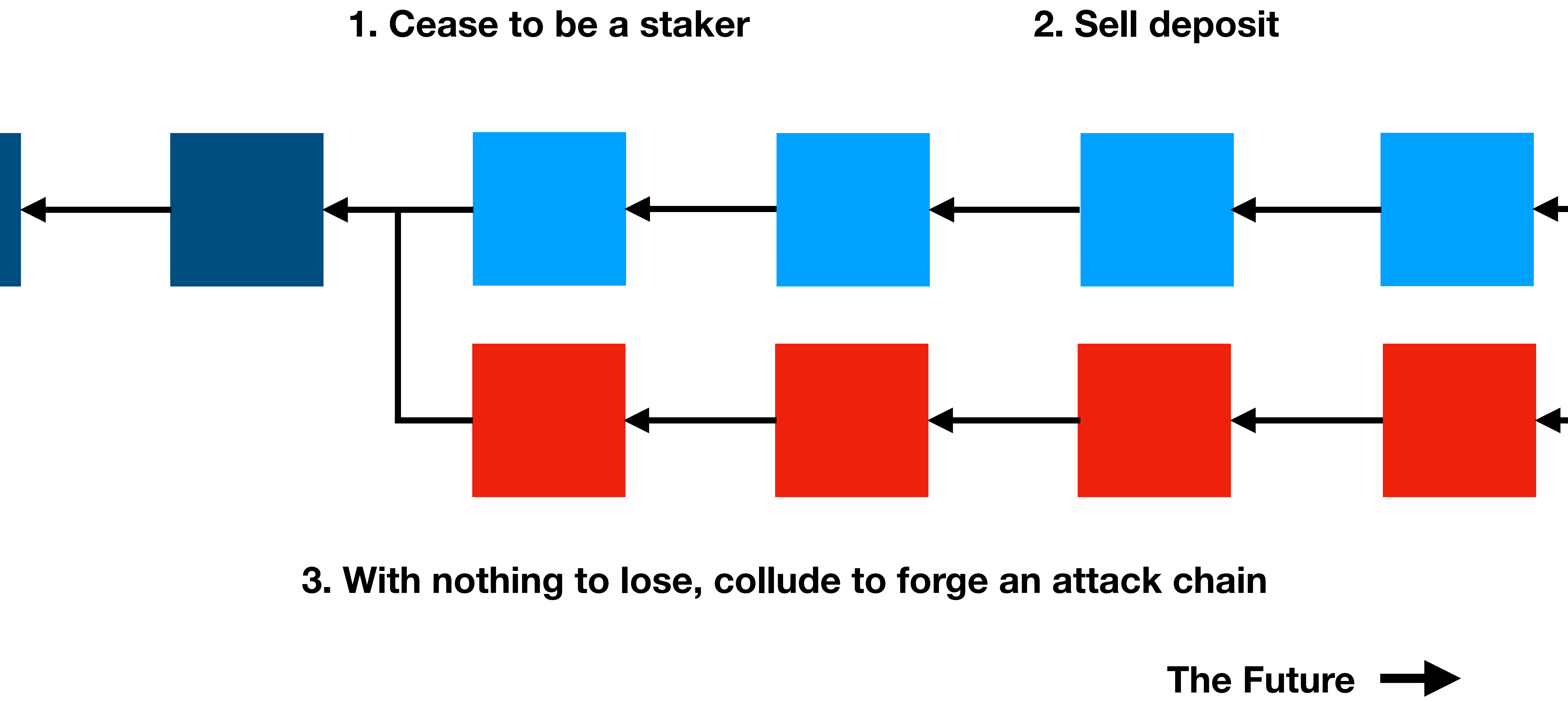
- What happens if there are two blocks at the same height with 50% of the vote weight each?
- Most protocols use some kind of **timeout** and allow the stakers to retry
  - Vitalik's Casper uses an **epoch** number and allows the validators to retry in the next epoch if no block is finalised
  - Still an area of <u>open research</u>! Join the fun!

Economic Finality

- Achieve similar security guarantee to a PoW blockchain by requiring 1/3 of total deposits to be some large amount (e.g. **$50M USD**)

- Recall that the *cost* of a 51% attack on PoW is quite low: PoS security can be **stronger** than PoW because malicious actors are **actually punished**

Short-range Attacks



Short-range Attacks

- Once a staker stops staking, their security deposit is still **locked** and can't be withdrawn for a reasonably long time, e.g. a month
- Prevents **short-range attacks** where an attacker sells all their coins and then creates an alternative history in the recent past that they can't be punished for

Long-range Attacks

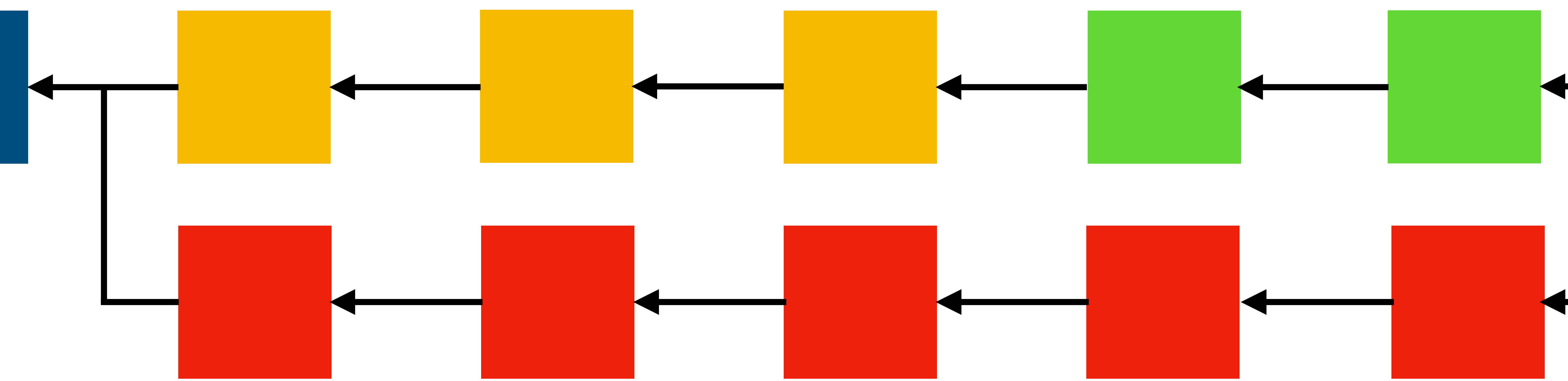
- What if an attacker just attacks as soon as their deposit is unlocked?

Long-range Attacks

1. Cease to be a staker

2. Wait a month for your deposit to unlock

3. Sell your deposit



4. Forge attack chain from >1 month ago, with nothing to lose

Long-range Attacks

- What if an attacker just attacks as soon as their deposit is unlocked?
- Idea: **never** revert blocks older than some large duration of time (e.g. 1 week)
- Intuition: the network should have plenty of time to propagate messages and come to consensus over a week – any changes after that are likely to be malicious long-range attacks

Weak Subjectivity

- Objectivity: PoW chain contains intrinsic proof of the work required to create it
- Weak Subjectivity: PoS chains are trivial to fabricate with the right keys, need to rely on external knowledge to choose a canonical version
- In practice: PoS clients that log on infrequently need to get a checkpoint from a trusted source

Summary

- Blockchains solve the decentralised consensus problem, allowing parties who don't trust each other to agree on a single transaction history (no **double-spends**)
- Proof of work is a blockchain consensus mechanism that relies on the difficulty of obtaining more computational power than the rest of the network
- Proof of stake attempts to improve the energy-efficiency of PoW by replacing mining with **in-protocol rewards and penalties**

Summary

- Upgrading a blockchain network is hard! Soft forks are slightly easier to pull off than hard forks, both can cause permanent chain splits
- **Smart contracts** are programs that run on blockchain networks, and use the blockchain to store their state
- There's lots of work to do! What could you build?

Questions?

Learn More

- Bitcoin: https://bitcoin.org/bitcoin.pdf
- Bitcoin Block Explorer: https://blockchain.info
- Bitcoin Wiki: https://en.bitcoin.it/wiki/Main_Page
- Bitcoin Source: https://github.com/bitcoin/bitcoin
- BIPs: https://github.com/bitcoin/bips
- Bitcoin Energy Consumption: https://digiconomist.net/bitcoin-energy-consumption
- Ethereum: https://ethereum.github.io/yellowpaper/paper.pdf
- Ethereum Block Explorer: https://etherscan.io/

Learn More

- Ethereum Research Forum: https://ethresear.ch/
- Go Ethereum Client: https://github.com/ethereum/go-ethereum
- Solidity Contracts: https://solidity.readthedocs.io/
- Solidity Gas Golfing Competition: https://g.solidity.cc/
- Crypto Twitter: https://twitter.com/search?q=cryptocurrency
- Vitalik's Casper PoS: https://arxiv.org/abs/1710.09437
- Correct-by-construction Casper PoS: https://github.com/ethereum/research/blob/master/papers/CasperTFG/CasperTFG.pdf
- Ouroboros PoS: https://eprint.iacr.org/2016/889

Image Credits

- Bitcoin Logo: https://en.bitcoin.it/wiki/Promotional_graphics
- Ethereum Logo: https://www.ethereum.org/assets
- Decentralised vs centralised: https://en.wikipedia.org/wiki/File:Decentralization_diagram.svg
- Txn icon: https://commons.wikimedia.org/wiki/File:Document_icon_(the_Noun_Project_27904).svg
- Mining distribution: https://blockchain.info/pools
- Merkle trie: https://ethereum.stackexchange.com/questions/6415/eli5-how-does-a-merkle-patricia-trie-tree-work

License