Blockchain-based Storage

Blockchain Properties

At the most basic level, a blockchain provides an alternative way for storing data in a database.

A major application of blockchain is in the creation of decentralized digital ledgers [去中心化数字账本].

Blockchain-based distributed ledgers maintain a ledger cooperatively among several parties

- Each transaction is digitally signed as proof of authenticity
- Once entries are added, they cannot be deleted or modified by one party, without detection by others.

It can provide a secure data-storage and data-processing foundation for business applications, without requiring complete trust in any one party [不需要完全信任任何一方].

Cryptocurrencies:

- Purely online
- Maintained by a decentralized distributed ledger [由去中心化的分布式账本维护]

Types of Blockchain:

Public:

Anyone can download the needed software and create a blockchain node

No trust assumed among participating nodes

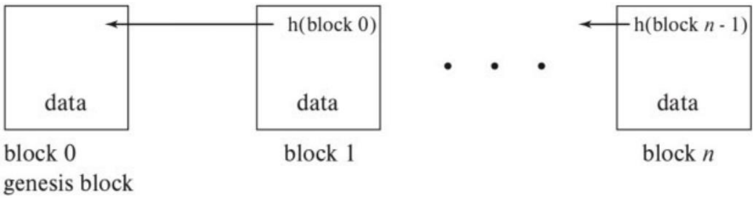
Permissioned:

Permission to run a blockchain node is granted by a permissioning authority

Some degree of relaxation of the assumptions of trustlessness and autonomy

The type of blockchain influences the choice of algorithms used by which nodes agree on the next block to be added to the blockchain

Structure of Blockchain:



（层层嵌套 hash 函数，从第一个 genesis 块开始，非常安全）

- Each block contains a pointer to the previous block plus a hash of the previous block (except the first block, called the genesis block)
- Node types:

  Full node – maintains copy of blockchain and participates in the consensus process

  [全节点-维护区块链的副本，并参与协商一致的过程]

  Light node – submits updates to the blockchain but does not participate in the consensus process

  [轻节点-向区块链提交更新，但不参与协商一致的过程]
- Consensus algorithms to choose node to add the next block: Proof of work, Proof of stake, Byzantine consensus, proof of activity, proof of burn, proof of capacity, proof of elapsed time
- Digital signature[数字签名]:

  Public-key encryption is used to allow users to sign their transactions.

  Ensures that users cannot deny submitting the transaction, a property called irrefutability.
- Anonymity[匿名]:

Users can remain anonymous unless there is a way to tie the user's ID to a real-world entity

▲Summary of blockchain properties:

Decentralization[去中心化] – majority consensus with no central authority.
Tamper resistance[抗干扰] – infeasibility of changing the contents of blocks on the blockchain.
Irrefutability[不可反驳性] – user cannot deny having submitted a transaction.
Anonymity[匿名] – IDs not directly tied to any real-world entity

▲Achieving Blockchain Properties via Cryptographic Hash Functions [通过加密哈希函数实现区块链属性]

▲Cryptographic Hash Functions

Let h denote a cryptographic hash function. Then h must satisfy the following properties:

Collision resistant[耐碰撞] – It is infeasible to find two distinct values x and y such that h(x) = h(y)
Irreversible[不可逆的] – Given h(x), it is infeasible to find x (use mathematical evidence to proof).

▲Blockchain Transactions

Exact transaction model is specific to each blockchain. [确切的事务模型是特定于每个区块链的]

Bitcoin(比特币):

No account balances stored directly.
A transaction specifies:
Input transactions (whose output is to be spent by this transaction)
A set of outputs, each specifying the recipient [容器] and the amount [数额]
A digital signature from the user submitting the transaction
Additionally, a Bitcoin transaction may:
Store a small amount of data on the blockchain
Specify a slightly more complex transaction using the Bitcoin scripting language

Ethereum(以太坊):

Maintains account balances, which are modified by transactions
Has a more sophisticated [更精致的], Turing-complete scripting language

▲Consensus [一致性]

All nodes must agree on additions to the blockchain

In a decentralized system like a blockchain, there is no central coordinator (unlike the case for 2PC and 3PC)

Categorization of consensus algorithms:

Proof of Work [工作量证明] (public blockchain)

- Node needs to solve a cryptographic puzzle in order to add a block

Proof of Stake [权益证明] (public blockchain)

- Node is chosen to add next block based on amount of currency held with proportionate to stake
- Not only of overall stake, but also the total time a stake has been held
- Probability of mining success is made higher for nodes in proportion to their stake

Byzantine Consensus [拜占庭共识] (permissioned blockchain)

- Node is chosen to add next block based on a message passing-based consensus algorithm
- Byzantine failure [拜占庭错误]: a failed node can behave in an arbitrarily bad manner, including taking the exactly correct set of steps to sabotage the system
- Practical Byzantine Fault Tolerance [实用拜占庭容错算法]: achieving consensus with Byzantine failure that at most (n-1)/3 nodes fail, where n is the total number of nodes

Sybil Attacks:

A Sybil attack is an attempt to overwhelm the consensus algorithm by adding a large number of nodes.

Protection against Sybil attack:

•Proof of work: hard for an attacker to control a majority of the computing power in the network Thus, making it hard to dominate success in solving the cryptographic [加密的] puzzle.

•Proof of stake: costly to acquire a majority of all outstanding currency.

•Byzantine consensus: vulnerable to attack unless there is a permissioning mechanism for new nodes:

♦Trusted permission-granting agent.

♦A decentralized trust-based feature in the protocol itself.

Data Management in a Blockchain

Efficient Lookup in a Blockchain

Without a good data structure, this step would be prohibitively costly.

Bitcoin: Look up input transactions to ensure that their output has not already been spent (“double-spend”).

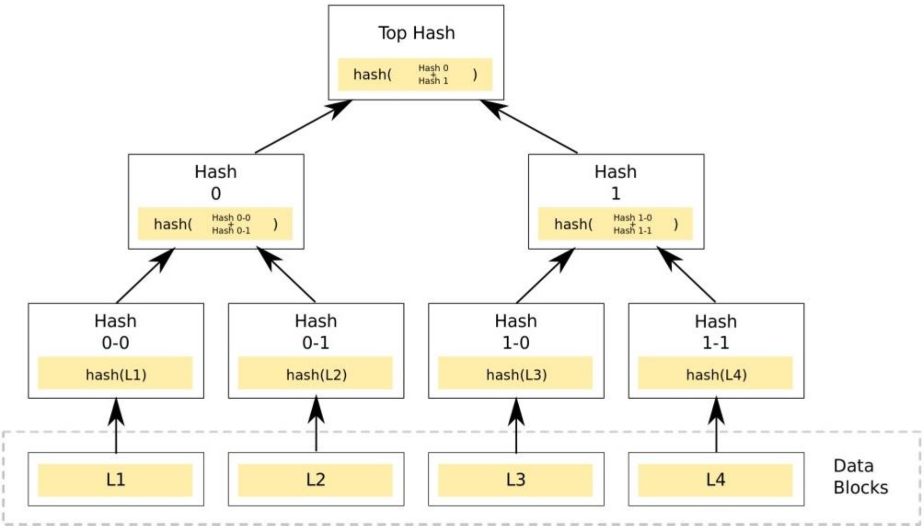
Ethereum: Lookup account balances.

Blockchains use the Merkle-tree data structure:

•Allows a node to store just the root-hash of the Merkle tree for verification purposes, rather than the entire blockchain.

•Particularly useful for light nodes since they need to retain only the root hash of the tree for verification.

•A full node can provide any needed data to the light node, i.e. any data plus the hashes needed for verification.



Maintaining Blockchain State:

Ethereum maintains a state that holds the balance in each account.

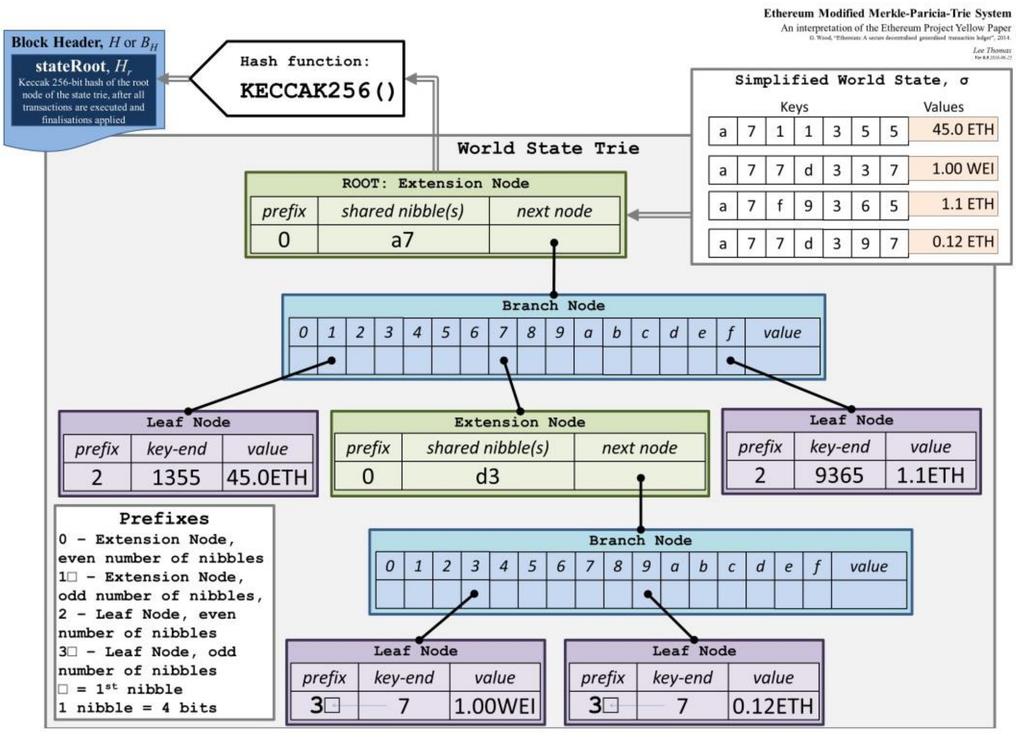
Transactions move currency units (ether in Ethereum) among accounts.

A variant of the Merkle-tree data structure, called a Merkle-Patricia-tree, is used for this purpose

Merkle-Patricia-tree structure:

Patricia-tree structure allows efficient key-based search.

Insertion and deletion: updates performed by creating a new root that points to unchanged parts of the data structure. [通过创建指向数据结构中未更改部分的新根来执行的更新]


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
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()
World State Trie
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
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 Merkle Patricia Trie (Extension node) - Ethereum Stack Exchange

Smart Contracts [智能合约]

Smart contracts are programs stored on a blockchain.

Run when predetermined conditions are met.

Automate the execution of an agreement.

Transaction execution is specified by code

Bitcoin: uses a relatively simple stack-based scripting language for fund transfer

m of n users must approve to enable escrow transactions.

Infinite loops not possible.

Ethereum: uses a scripting Turing-complete language

Based on Ethereum virtual machine (EVM)

Solidity: high-level language compiled to EVM code

Greater expressive power but risk of infinite loops

A smart contract may be defined in terms of external events:

Messages from other contracts;

Input from trusted sources called oracles.

Autonomy - smart contracts can be deployed as independent entities:

A smart contract may run indefinitely by receiving Ether from external sources to fund its continued operation, called distributed autonomous organizations (DAOs).

A smart contract may be used to create a separate currency or token on top of the Ethereum blockchain.

Smart contracts may be used in the implementation of cross-chain transactions, allowing transactions between separate blockchain systems.

Performance Enhancement

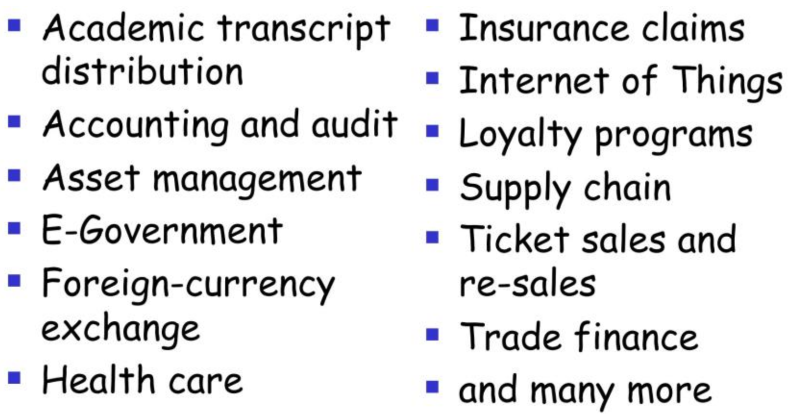
The consensus mechanism is an important factor in blockchain performance.

Other ways to enhance performance include:

•Sharding: parallelizing the mining of new blocks;

- Off-chain transaction processing: creation of a separate channel for users;
  (1) Channel funded with funds from the underlying blockchain;
  (2) Routine transactions avoid mining overhead;
  (3) Users can terminate the agreement to process transactions off-chain at any point, with current channel funds balances refunded on the underlying blockchain.
- Database-style blockchain data structures.

▲ Emerging Applications [高级和新兴应用]



- Academic transcript distribution
- Accounting and audit
- Asset management
- E-Government
- Foreign-currency exchange
- Health care
- Insurance claims
- Internet of Things
- Loyalty programs
- Supply chain
- Ticket sales and re-sales
- Trade finance
- and many more