区块链与数字货币

浙江大学 杨小虎

2020年12月11日

04 以太坊技术分析

以太坊是什么？

- 以太坊是一个为去中心化应用程序而生的全球开源平台。
- 以太坊是互联网新时代的基础：
  - 内建货币与支付。
  - 用户拥有个人数据主权，且不会被各类应用监听或窃取数据。
  - 人人都有权使用开放金融系统。
  - 基于中立且开源的基础架构，不受任何组织或个人控制。

以太坊是什么？

ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER

PETERSBURG VERSION 3e2c089 – 2020-09-05

DR. GAVIN WOOD
FOUNDER, ETHEREUM & PARITY
GAVIN@PARITY.IO

ABSTRACT. The blockchain paradigm when coupled with cryptographically-secured transactions has demonstrated its utility through a number of projects, with Bitcoin being one of the most notable ones. Each such project can be seen as a simple application on a decentralised, but singleton, compute resource. We can call this paradigm a transactional singleton machine with shared-state.

Ethereum implements this paradigm in a generalised manner. Furthermore it provides a plurality of such resources, each with a distinct state and operating code but able to interact through a message-passing framework with others. We discuss its design, implementation issues, the opportunities it provides and the future hurdles we envisage.

以太坊最主要的创始人和贡献者：Vitalik Buterin 和 Gavin Wood

以太坊发展历史

2013年末，初版白皮书发布，项目启动。2014年3月开始陆续发布了3款测试网络。2014年4月发布了以太坊黄皮书。2014年7月24日起，以太坊进行了为期42天的以太币预售。

是以太坊的最初版本，于2015年7月30日发布，不是一个完全可靠和安全的网络。

2016年3月14日，Homestead版本发布，表明以太坊网络已经平稳运行。在此阶段，以太坊提供了图形界面的以太坊钱包。

2017年10月。以太坊将这一阶段分为两个版本，分别为Byzantium和Constantinople。该阶段旨在使以太坊更轻量、快速和安全。

发布日期尚未确定。在该阶段，以太坊将共识机制从PoW转换到PoS。前三个阶段所需要的挖矿将被终止，新发行的以太币也将大为降低，甚至不再增发新币。

以太坊的组成

- **P2P 网络**

  以太坊以P2P方式进行网络通信，通过TCP端口30303访问。

- **交易 Transactions**

  以太坊交易是网络消息，包括转账交易、合约交易等。

- **状态机 State Machine**

  以太坊的状态转移由 以太坊虚拟机（EVM） 处理，这是一个执行 bytecode（机器语言指令）基于栈的虚拟机。称为“智能合约”的EVM程序以高级语言（如Solidity）编写，并编译为字节码以便在EVM上执行。

- **区块链账本**

  以太坊的区块链账本存储在每个节点上，该区块链在Merkle Patricia Tree 的序列化哈希数据结构中包含交易和系统状态。

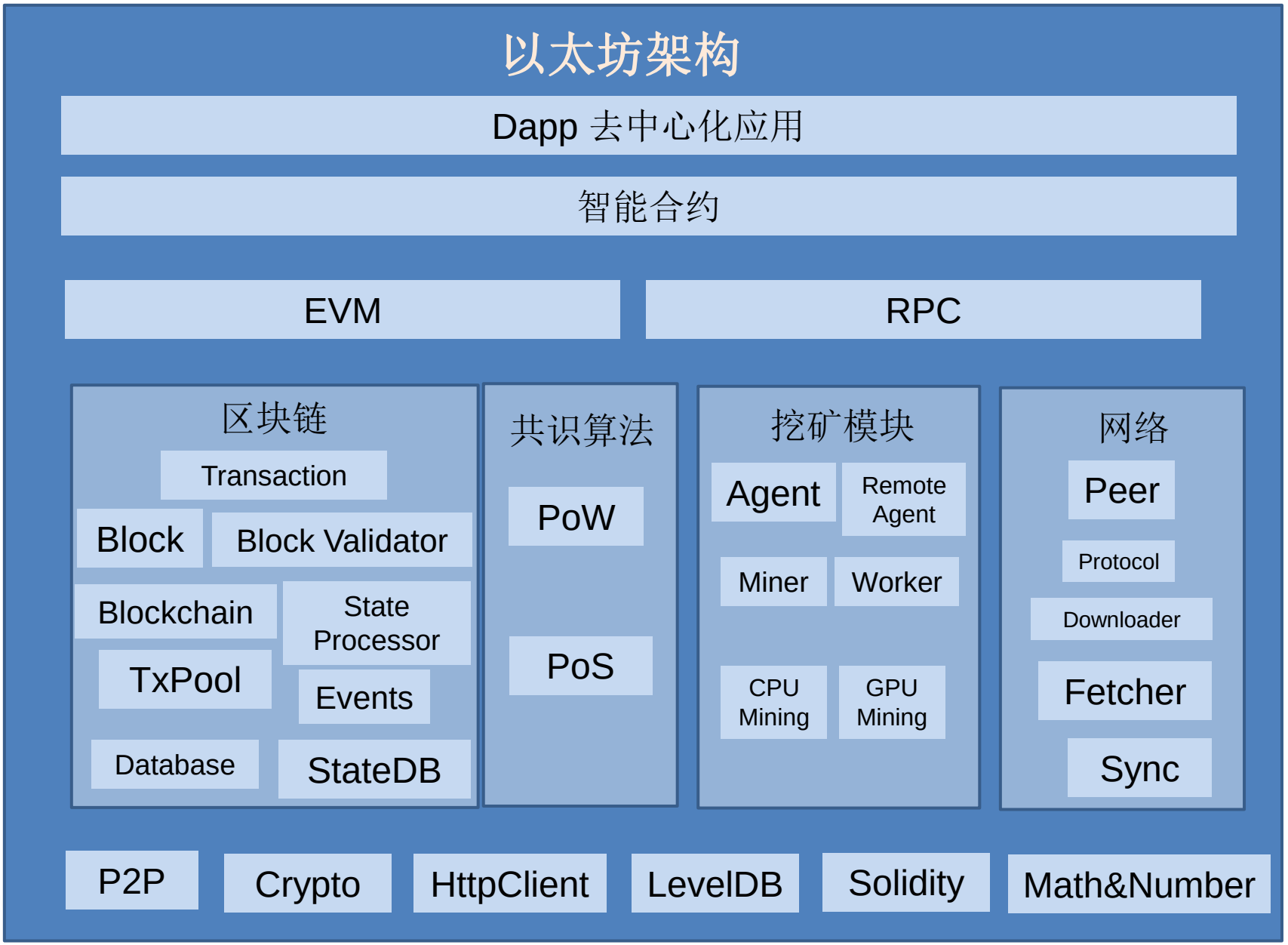
- **共识算法**

  以太坊目前使用名为Ethash的工作量证明算法，计划在不久的将来将过渡到称为Casper的权益证明机制（Proof-of-Stake）。

- **客户端**

  以太坊有几个可互操作的客户端软件实现，最著名的是 Go-Ethereum（Geth）和Parity

以太坊架构


Dapp 去中心化应用
智能合约
EVM
RPC
区块链
Transaction
Block
Block Validator
Blockchain
State Processor
TxPool
Events
Database
StateDB
共识算法
PoW
PoS
挖矿模块
Agent
Remote Agent
Miner
Worker
CPU Mining
GPU Mining
网络
Peer
Protocol
Downloader
Fetcher
Sync
P2P
Crypto
HttpClient
LevelDB
Solidity
Math&Number

以太坊基本概念

以太坊由大量的节点组成，节点有账户与之对应，两个账户之间通过发送消息进行一笔交易。交易里携带的信息和实现特定功能的代码叫做智能合约，运行智能合约的环境是以太坊虚拟机（EVM），EVM类似于Java虚拟机JVM，编译后基于字节码运行，开发时则可以使用高级语言实现，编译器会自动转化为字节码。基于以上的智能合约代码和以太坊平台的应用叫做去中心化应用（DApp，Decentralized Application）。

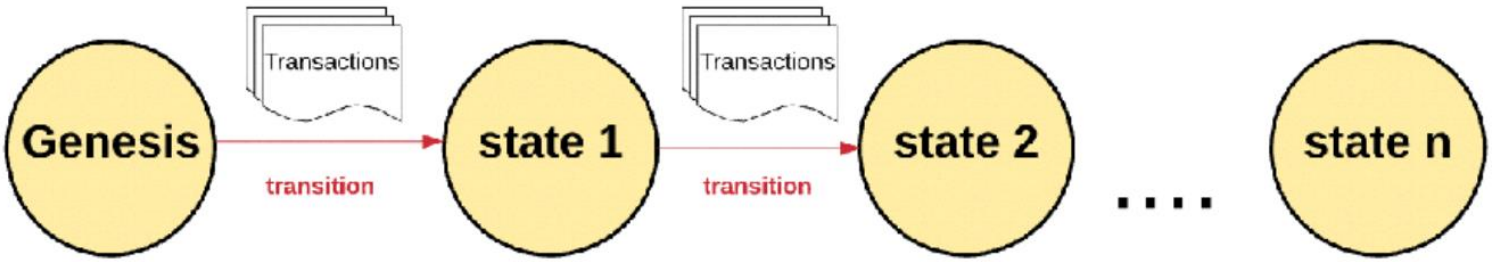
- **节点**　以太坊是由网络上的许多节点组成的，每一个节点都运行着以太坊虚拟机，通过节点可以进行区块链数据的读写。节点之间使用共识机制来确保数据交互的可靠性和正确性。
- **账户**　以太坊中包含两类账户，包括外部账户和合约账户。外部账户由公私钥对控制，合约账户则在区块链上唯一的标识了某个智能合约。
- **交易**　交易包括转账交易和合约交易，通过状态转移来标记。状态由账户之间的转移价值和信息状态转换沟通。

以太坊基本概念（续）

- **智能合约**　合约是代码和数据的集合，存在于以太坊区块链的指定地址。合约方法支持回滚操作，如果在执行某个方法时发生异常（如gas消耗完），则该方法已经执行的操作都会被回滚。但是如果交易一旦执行完毕，是没有办法篡改的
- **EVM**　以太坊虚拟机是以太坊中智能合约的运行环境，并且是一个沙盒，与外界隔离。智能合约代码在EVM内部运行时，是不能进行网络操作、文件I/O或执行其它进程的。智能合约之间也只能进行有限的调用。
- **矿工与挖矿**　矿工是指通过不断重复哈希运算来产生工作量的网络节点。在以太坊中，发行以太币的唯一途径是挖矿。
- **gas**　以太坊上的每一笔交易都有矿工的参与，且都需要支付一定的费用，这个费用在以太坊中称为gas。gas的目的是限制执行交易所需的工作量，同时为执行交易支付费用。

世界状态

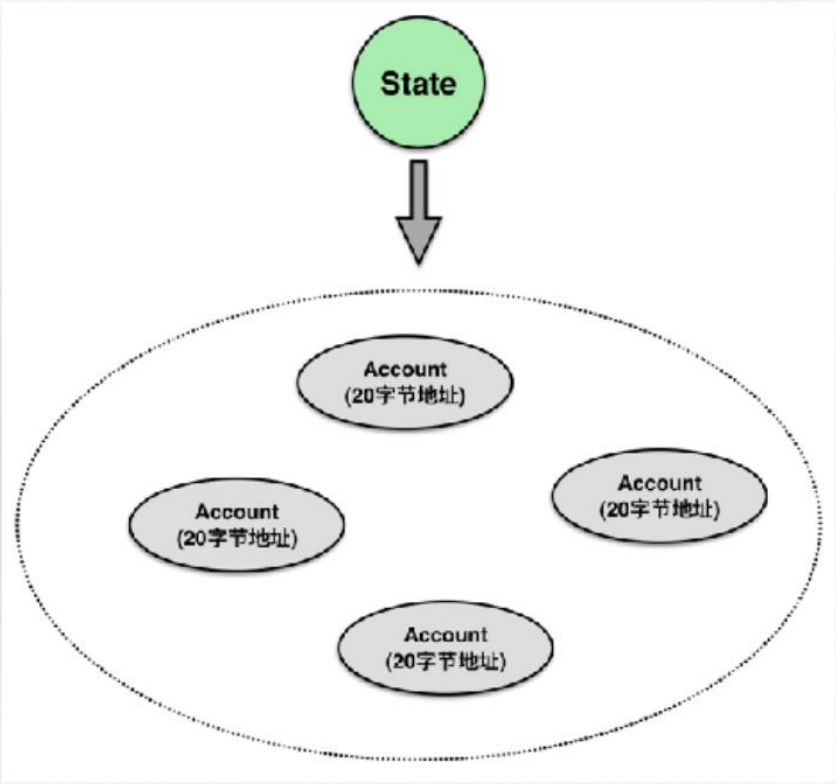
- 世界状态 world state：所有账户（包括外部账户和合约账户）的状态合集。



- 以太坊本质上是一个基于交易的状态机
- 以太坊有一个初始状态我们称为Genesis
- 状态转换的最小单元是交易(原子性、一致性)
- 每次执行一条或者多条交易后发生状态转换

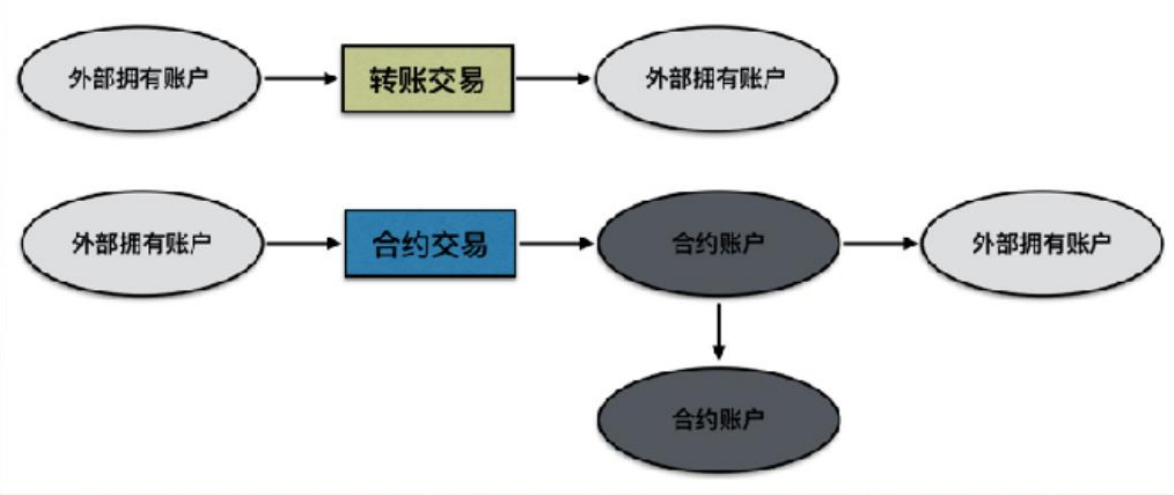
世界状态

- “世界状态”是由一系列“账户”所组成
- 每一个账户都有一个唯一的地址
- 地址的产生规则分为两类
  - 由账户对应的公钥计算所得
  - 由部署者的信息计算所得

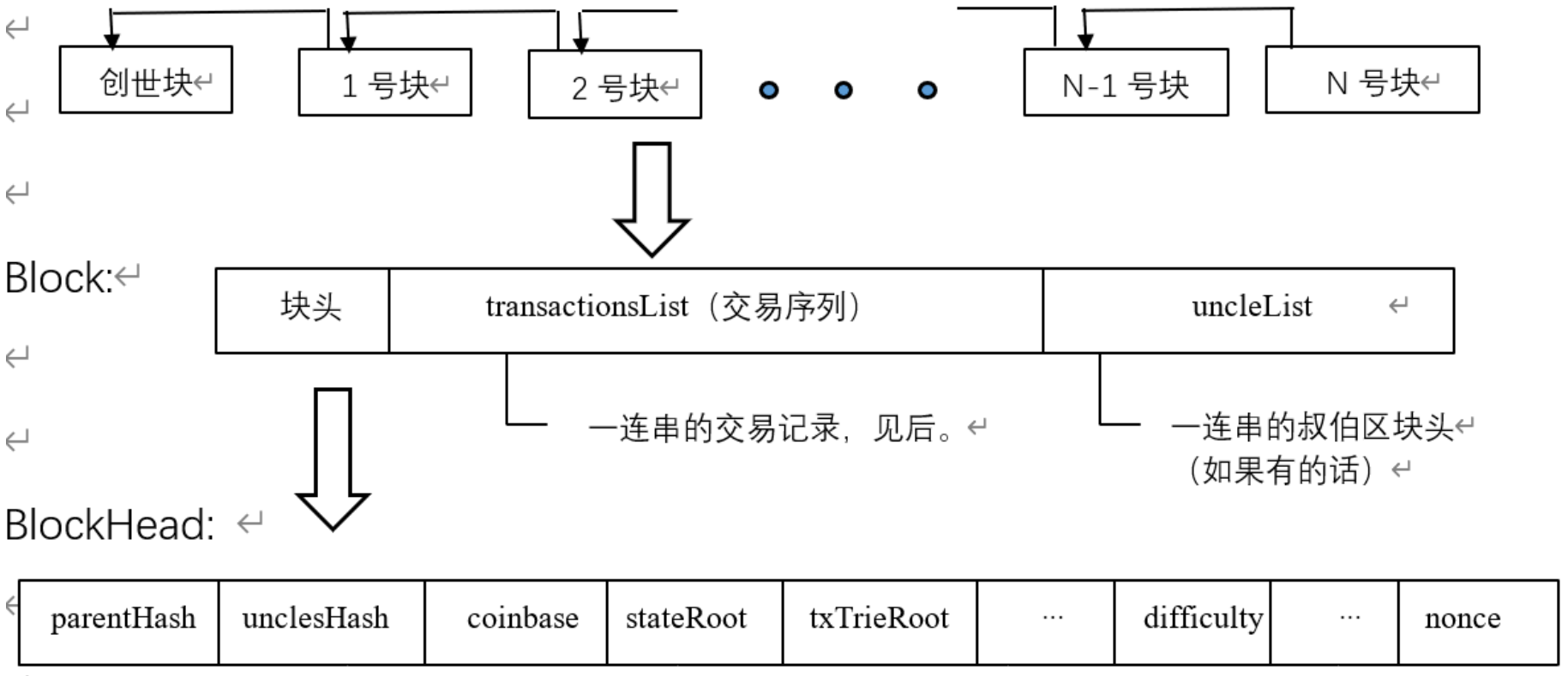


账户

- 外部账户可以主动发起交易：转账交易或合约交易
  - 转账交易完成内置数字资产Ether的转移
  - 合约交易触发接收账户的运行码进行运行
- 合约账户只能通过“合约交易”或“消息调用”的方式被触发运行代码
- 合约账户在代码运行期间可以完成任意复杂度的操作：修改存储空间数据项，发送“消息调用”给其他账户等



区块头


创世块
1 号块
2 号块
N-1 号块
N 号块
Block:
块头
transactionsList（交易序列）
uncleList
一连串的交易记录，见后。
一连串的叔伯区块头
（如果有的话）
BlockHead:
parentHash
unclesHash
coinbase
stateRoot
txTrieRoot
…
difficulty
…
nonce

区块数据结构

class **Block** {}　　//Block类是一个数据结构，内含下面这些结构成分：
] BlockHeader **header**;　　//块头
] List<Transaction> **transactionsList**　//交易记录序列
] List<BlockHeader> uncleList　　//叔伯块的块头序列

class **BlockHeader** {}　　//最长可达800字节
] byte[] parentHash;　　//前导块（父块）的块头Hash值
] byte[] unclesHash;　　//块身中uncleList的Hash值
] byte[] coinbase;　　//表示本区块的Coinbase和手续费应该给谁，其160位地址。
] byte[] stateRoot;　　//执行完本块所含全部交易后的状态树Hash值。
] byte[] txTrieRoot;　//块身中所有交易记录的树根Hash值。
] byte[] receiptTrieRoot;　//各个交易收据所构成树根的Hash值。
] byte[] difficulty;　//挖矿难度，用以调整发块周期长度
] long timestamp;　//时戳
] long number;　　//本块在区块链中的高度，即区块链中处于本块之前的区块个数。
] byte[] gasLimit;　//本块所含所有交易所提供“汽油”量即手续费的上限总和。
] long gasUsed;　　//本块所含所有交易实际消耗“汽油”量的总和。
] byte[] extraData;　//有关本区块的任意额外数据，不超过32字节。
] byte[] nonce;　　//

回执

属性	描述
Status	交易执行状态
CumulativeGasUsed	累积使用的Gas值
Bloom	交易日志的布隆过滤器信息
Logs	交易执行过程中所产生的日志集

日志

- 对于以太坊上部署的智能合约来说，外部拥有账户所发起的交易，是链下世界对链上世界的输入
- 智能合约必须也需要某种途径把链上世界的信息传递出去 - 日志
- 合约编码者可以在智能合约中定义Event
- 当智能合约运行过程中执行该语句，便会产生一个虚拟机日志，并且将其存储在回执中

交易的数据结构

class **Transaction** {}

] byte[] hash; //注意这是对RLP编码之后的交易请求Tx的Hash值。

] byte[] nonce; //实质上是Tx的序号，用以防止对同一交易请求的重复处理。

] byte[] value; //支付的币值，Wei是以太币的最小单元。

] byte[] receiveAddress; //对方地址，这地址可以是账户地址或合约地址。

] byte[] gasPrice; //油价

] byte[] gasLimit; //可以付出的最大油量

] byte[] data; //见下页解释

] Integer chainId; //子网/子链ID

] ECDSASignature signature; //签名

] byte[] sendAddress; //发出方地址

交易的数据结构：data字段作用

- 在支付交易中，可以用这个字段作付款说明。
- 如果把支付额设置成0（收款方地址不能为0），就可以用来存证。
- 在合约部署交易中（以0作为对方地址），这个字段的内容就是所欲部署到以太坊网络中的合约。
- 在合约调用交易中，用这个字段传递调用函数名和参数。

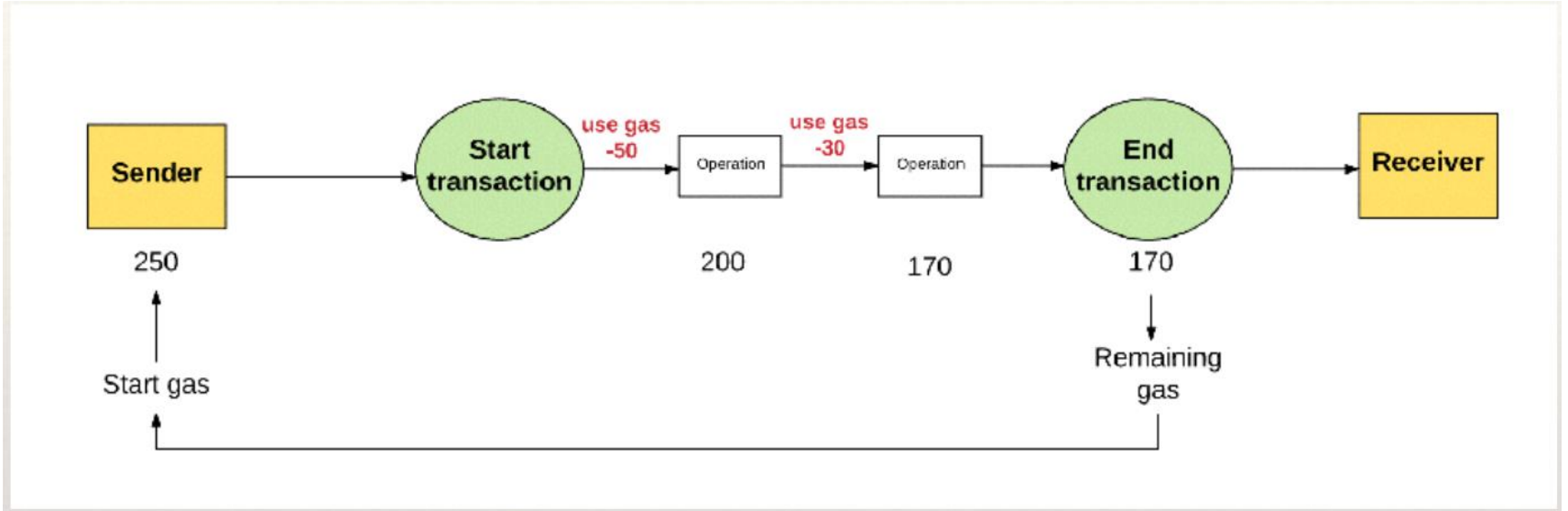
交易的类型

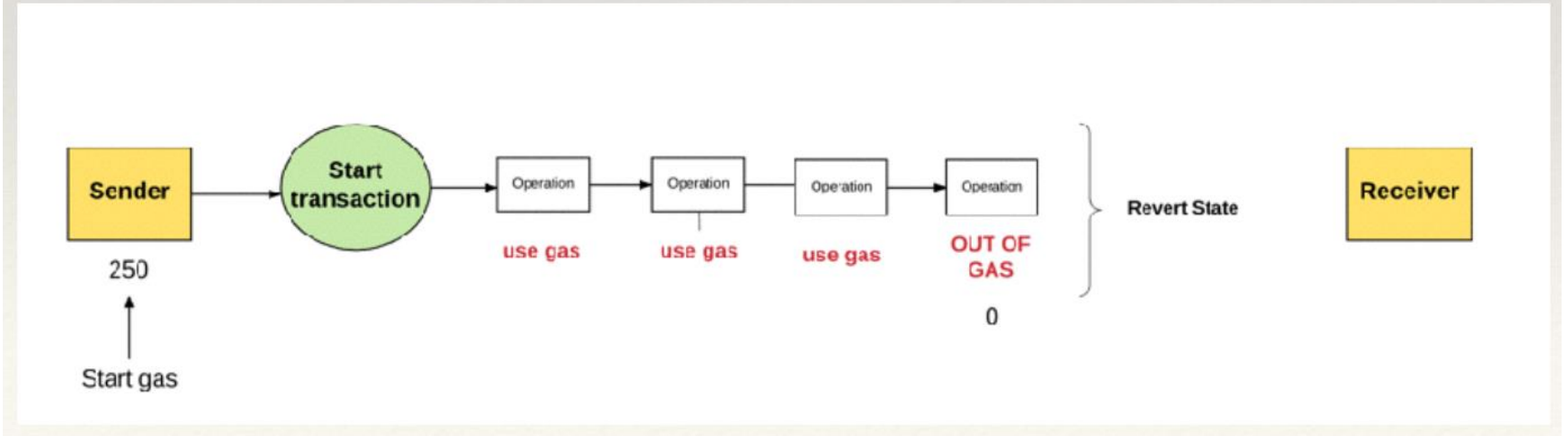
- **简单支付交易** - 以太币的账户间转账，从知己账户转到对方账户，不涉及智能合约，无需动用以太坊虚拟机，“耗油”也最少。
- **存证交易** – 在简单支付交易中把支付额设置成0，需要存证的内容写在data字段中，就成了存证交易。当然，也可以为存证机制专门部署一个智能合约，以后要存证时就调用这个合约，这样可以为存证增添一些附加的操作。因为是0支付，对方账户就无关紧要，但不能是0。
- **合约部署交易** - 将对方地址设置成0，就可以将智能合约的程序部署到以太坊网络中，实际上是所有的验证节点上。一旦交易记录进入区块链即部署生效，以太坊网络会在交易“收据”中返回新建合约账户的地址。
- **合约调用交易** - 对智能合约的调用，依具体智能合约的不同，又可以分为以下几种：
  - 复杂支付交易，更确切地说是数字资产转移交易，这是对智能合约中代表着数字资产的Token的转移。如果用Token实现各种虚拟货币，这样的转移就成了采用某种自定义虚拟货币的支付。
  - 查询交易，查询是区块链网络中常用的操作。以太坊网络中提供了若干“系统合约”，即由系统提供、无需用户部署的合约，其中之一就是用于查询。
  - 其它（智能合约调用）交易，如在电子商务，电子政务等领域的应用。

Gas

- 在以太坊上，任何引起状态转移的操作都是需要收费的
  - 数学运算
  - 状态存储
- Gas是用来计量以太坊系统资源使用情况的最小计量单位
  - 状态转移中所有的动作都有一个复杂度的衡量公式
  - Add操作花费3个Gas，SStore操作花费20000个Gas
- 一次交易执行过程，累积消耗Gas超过发送者预付的总量，交易执行失败

Gas


Sender
Start transaction
use gas -50
Operation
use gas -30
Operation
End transaction
Receiver
250
200
170
170
Start gas
Remaining gas


Sender
Start transaction
Operation
Operation
Operation
Operation
Revert State
Receiver
250
use gas
use gas
use gas
OUT OF GAS
0
Start gas

Gas

- GasPrice表示发送者预付的Gas价格
- Fee = Gas * GasPrice
- 发送者必须有足够多Ether余额来支付交易费用
- 交易所产生的手续费作为Block Producer的经济激励

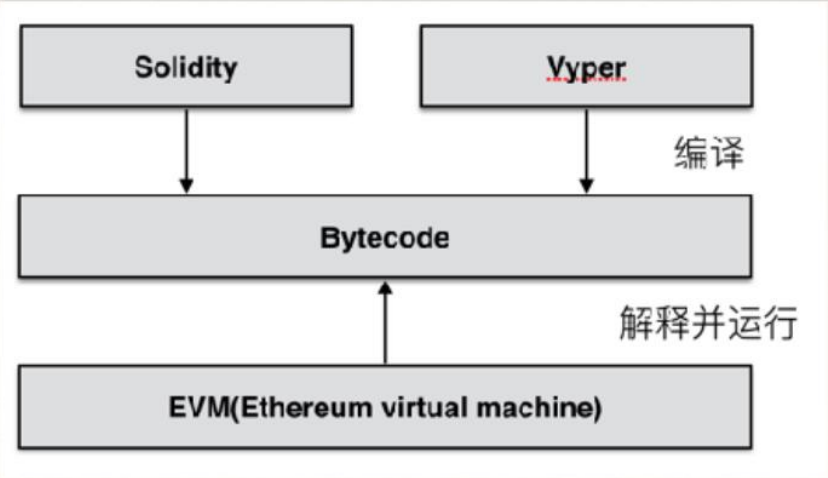
以太坊的网络

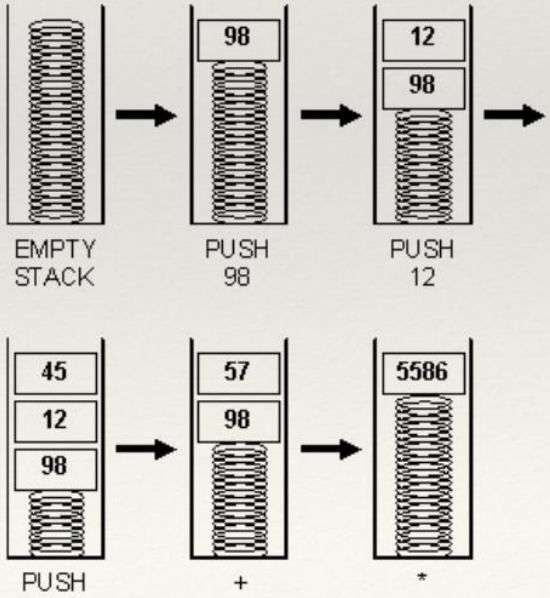
- 常见的两个
  - 主网 main net
  - 测试网 test net
- 其他
  - 以太坊经典主网
  - 以太坊经典测试网
  - ……

```
public class ChainId {              /* Ethereum chain ids. */
  public static final byte NONE = -1;
  public static final byte MAIN_NET = 1;
  public static final byte TEST_NET = 3;
}
```

以太坊虚拟机EVM

- EVM(Ethereum Virtual Machine) 是指用来解释跟执行合约账户字节码的解释器
- EVM基于栈的解释器
- EE(Execution environment) 是EVM的执行环境，包含环境参数以及存储空间读写函数
- EVM是拥有完全隔离的执行环境，合约无法访问宿主机的“网络”，“文件系统”等系统资源





以太坊虚拟机EVM

- EVM有一个固定的指令集
- 指令集包含：算数运算，比特运算，逻辑运算，跳转指令，状态读取、存储指令等
- 所有指令的运算必须是确定性的（例如高精度的浮点运算是不支持的）

以太坊虚拟机：图灵完备

- 图灵完备（Turing Completeness ）是针对一套数据操作规则而言的概念。数据操作规则可以是一门编程语言，也可以是计算机里具体实现的指令集。当这套规则可以实现图灵机模型里的全部功能时，就称它具有图灵完备。
- 高级语言具备了if/else语句、循环语句等，可以实现图灵完备。
- EVM是图灵完备的，但为了防止出现死循环，设置gas机制。

以太坊虚拟机（EVM）

运行智能合约的环境，运行在每一个节点上，类似于一个独立的沙盒，严格控制了访问权限，合约代码在EVM中运行时，是不能接触网络、文件或者其他进程的。

EVM模块主要分为以下三大模块：

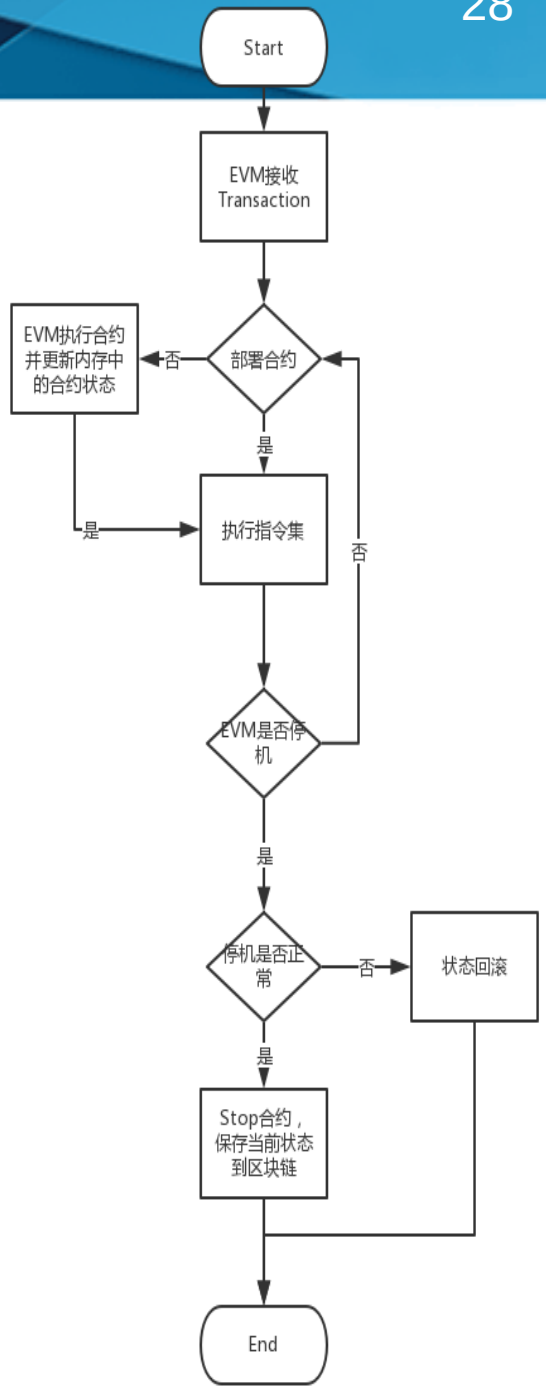
- **编译合约模块**： 主要是对底层Solc编译器进行一层封装，提供RPC接口给外部服务，对用Solidity编写的智能合约进行编译。编译后将会返回二进制码和相应的合约abi，abi可以理解为合约的手册，通过ABI可以知道合约的方法名、参数、返回值等信息。
- **Ledger模块**：主要是对区块链账户系统进行修改和更新，账户一共分为两种，分别是普通账户和智能合约账户，调用方如果知道合约账户地址则可以调用该合约，账户的每一次修改都会被持久化到区块链中。
- **EVM执行模块**（核心模块）：主要功能是对交易中的智能合约代码进行解析和执行，一般分为创建合约和调用合约两部分。同时为了提高效率，EVM执行模块除了支持普通的字节码执行外还支持JIT模式的指令执行，普通的字节码执行主要是对编译后的二进制码直接执行其指令，而JIT模式会对执行过程中的指令进行优化，如把连续的push指令打包成一个切片，方便程序高效执行。

EVM执行模块的大概流程

1. EVM接收到Transaction信息，然后判断Transaction类型是部署合约还是执行合约，如果是部署合约，则新建一个账户来存储合约地址和编译后的代码；如果是执行合约或是调用合约，则使用EVM来执行输入指令集。

2. 执行上一条指令集之后，判断EVM是否停机，如果停机则判断是否正常停机，正常停机则更新合约状态到区块链，否则回滚合约状态。如果不停机则继续执行下一条指令集，重复2；

3. 执行完的合约会返回一个执行结果，EVM会将结果存储在Receipt回执中，调用者可以通过Transaction的hash来查询其结果。

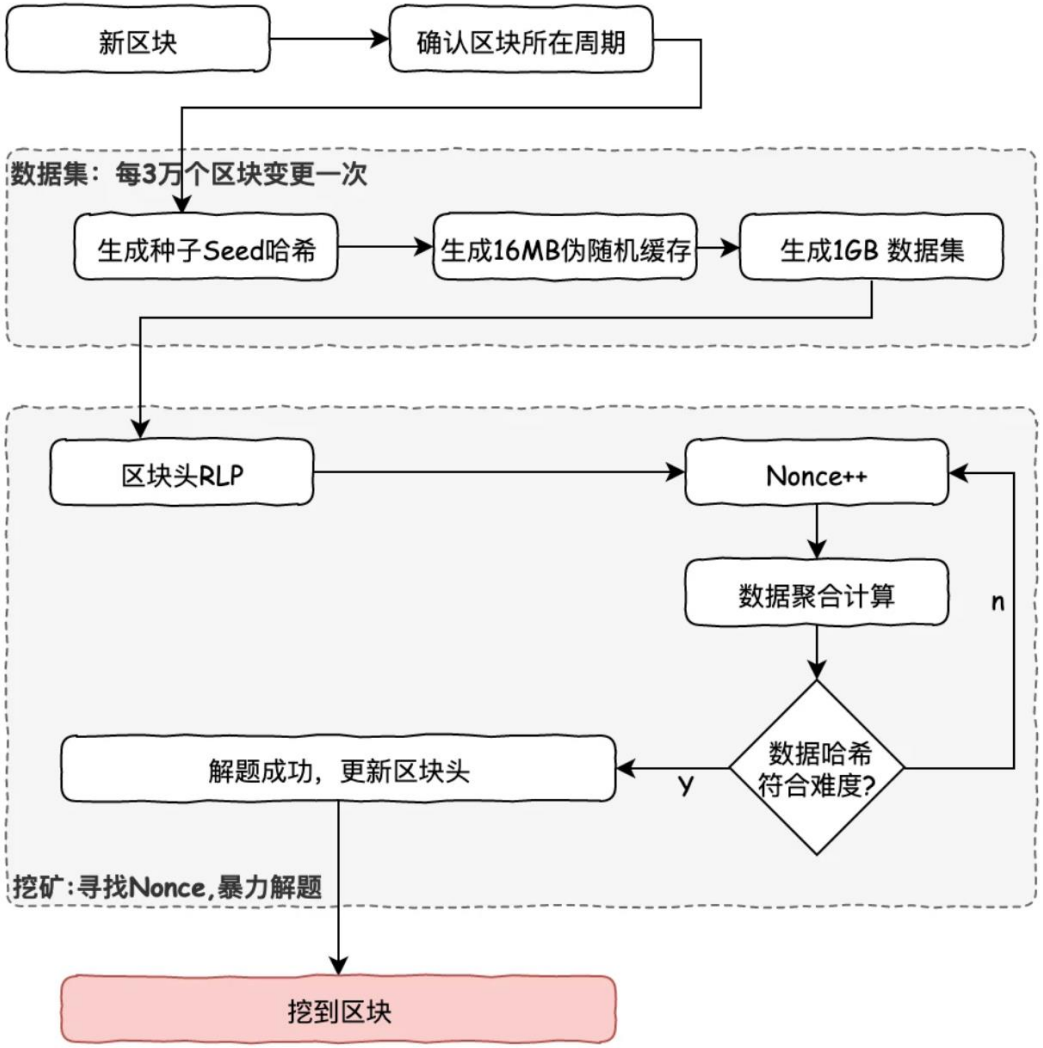


以太坊的共识机制

- 以太坊前三个阶段采用PoW，第四个阶段转移到PoS。

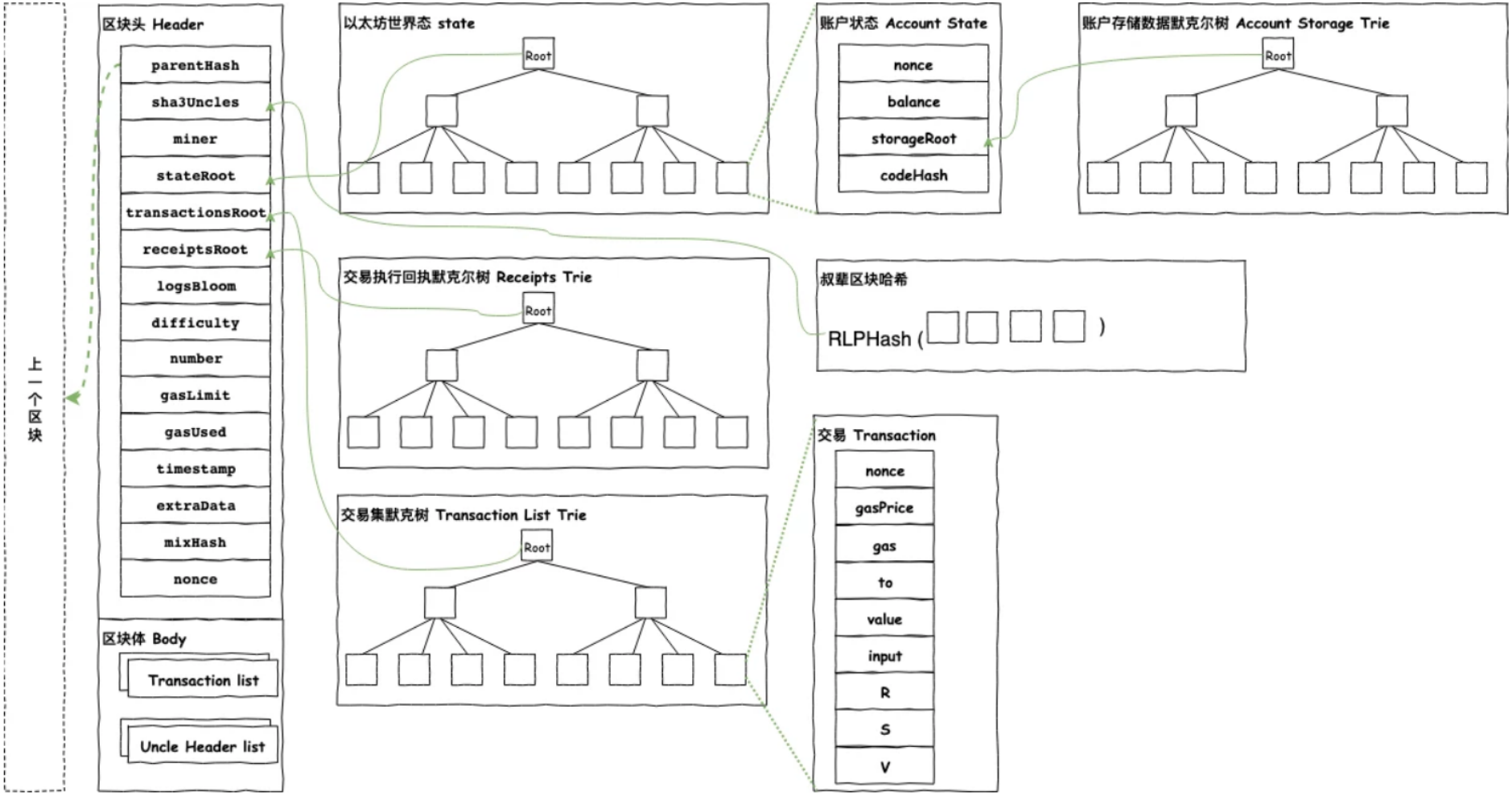
- 常见共识机制：
  - **PoW**：进行运算来获取记账权，资源消耗高，可监管性弱。每次达成共识都需要全网共同参与运算（性能低）。
  - **PoS**：根据每个节点所拥有代币的比例和时间来投票决定记账权，性能提高，但公平性存在问题。
  - **DPoS**：DPoS大幅缩小参与验证和记账节点的数量，但是整个共识机制还是依赖于代币。
  - **PBFT**：实用拜占庭容错。该共识机制允许系统存在作恶节点，性能更高，耗能更低，容错性为33%。

以太坊PoW算法 ehash

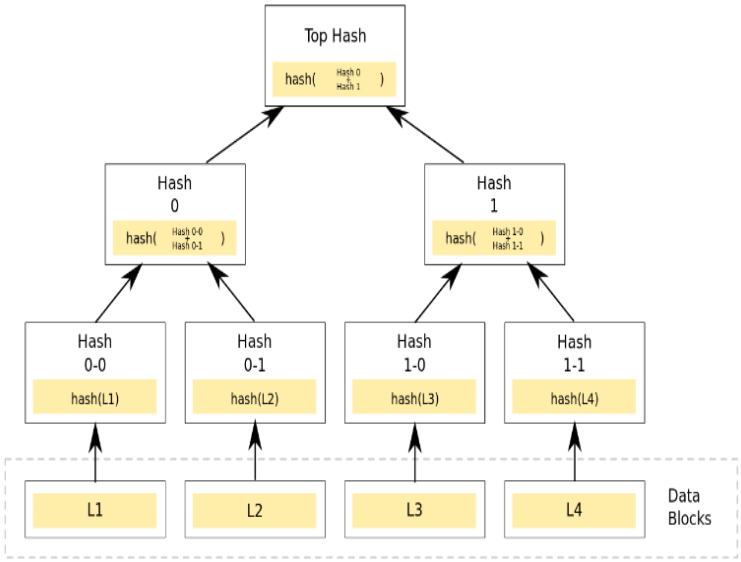


以太坊中采用的hash算法是Keccak256，Keccak256后来被NIST采用，改造为SHA3,SHA3不是SHA2的替代品，而是一种有别于SHA2的全新哈希设计方案，2015年8月被NIST正式批准。

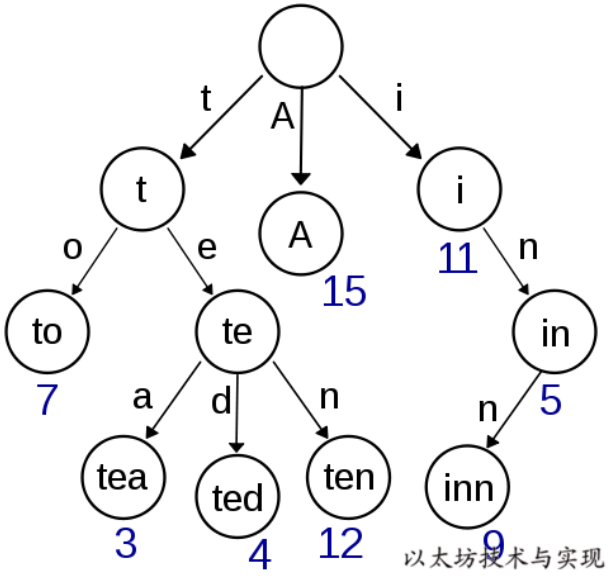
以太坊的数据存储


上一个区块
区块头 Header
parentHash
sha3Uncles
miner
stateRoot
transactionsRoot
receiptsRoot
logsBloom
difficulty
number
gasLimit
gasUsed
timestamp
extraData
mixHash
nonce
区块体 Body
Transaction list
Uncle Header list
以太坊世界态 state
Root
账户状态 Account State
nonce
balance
storageRoot
codeHash
账户存储数据默克尔树 Account Storage Trie
Root
交易执行回执默克尔树 Receipts Trie
Root
叔辈区块哈希
RLPHash ( )
交易集默克树 Transaction List Trie
Root
交易 Transaction
nonce
gasPrice
gas
to
value
input
R
S
V

以太坊的数据存储：Merkle Patricia Trie

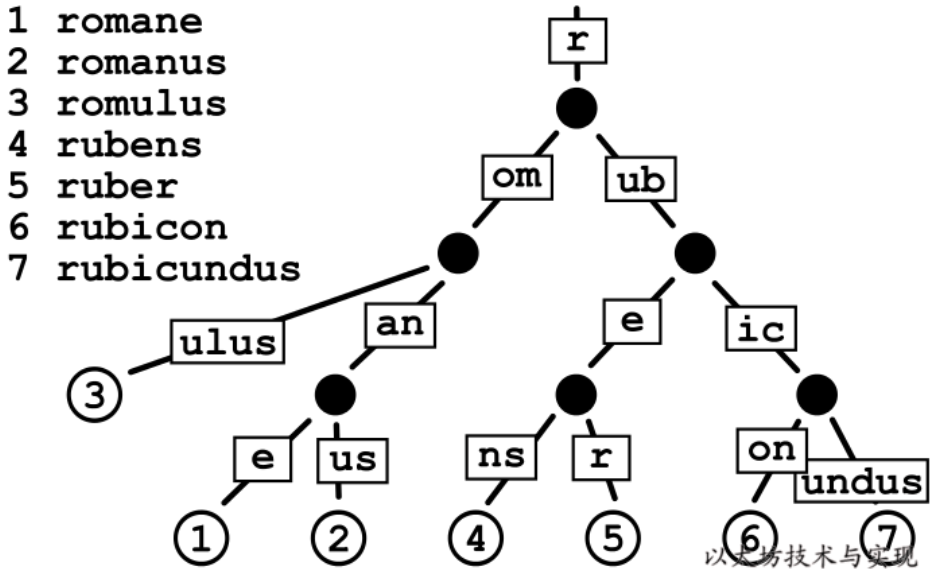


Merkle树



前缀树

以太坊的数据存储：Merkle Patricia Trie

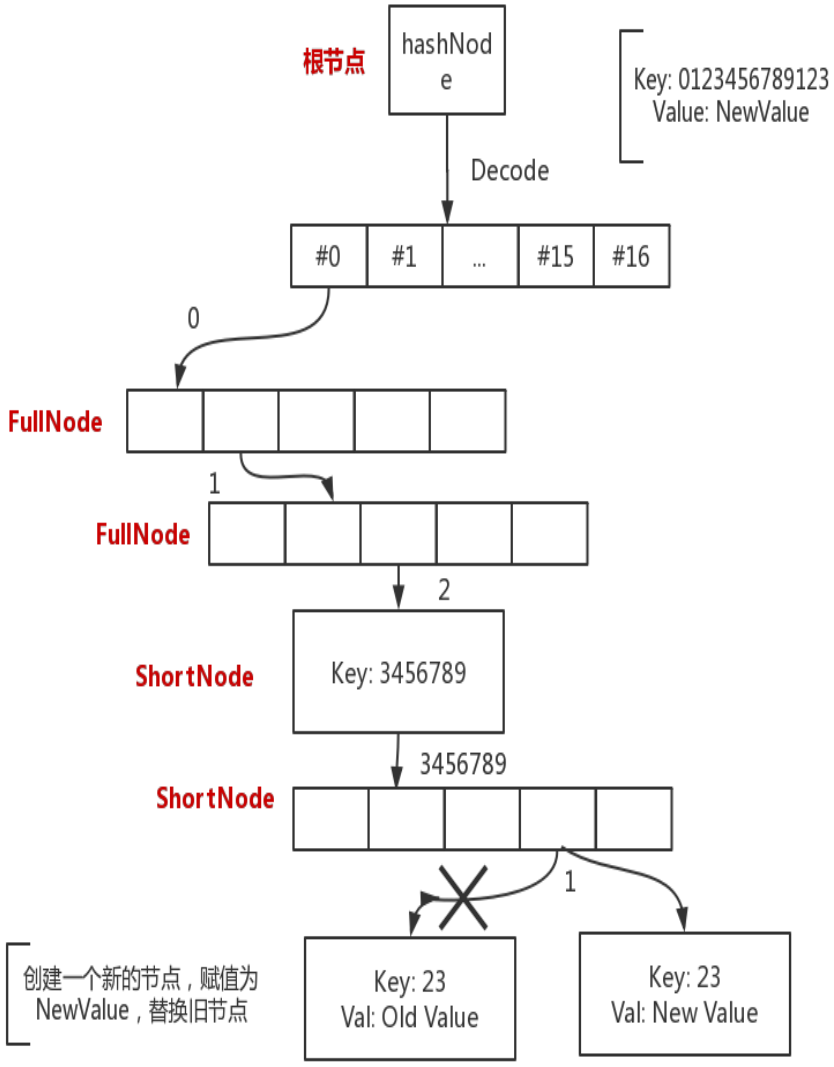


Patricia树

以太坊的数据存储：Merkle Patricia Trie

Patricia 树会存储每个账户的状态。这个树的建立是通过从每个节点开始，将节点分成多达16个组，然后散列每个组，再对散列结果继续散列，直到整个树有一个最后的“根散列”。

Patricia树很容易进行更新、添加、或者删除树节点，以及生成新的根散列。

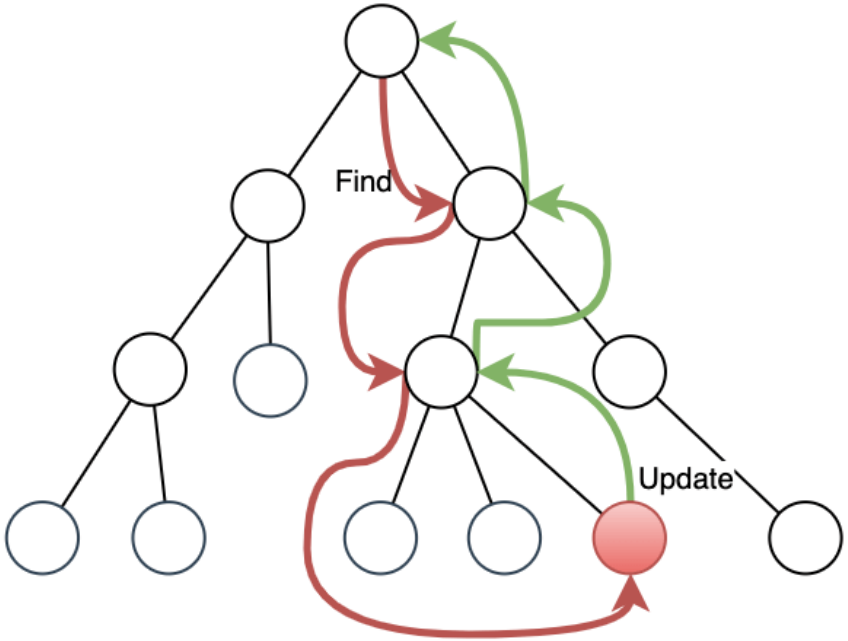


以太坊的数据存储：Merkle Patricia Trie

MPT树更新

（1）将根节点传入作为当前处理节点，传入目标节点的Key作为路径path；

（2）传入新的Value值，若Value值为空，则找到该节点并删除，反之，创建一个新节点替换旧节点。



The DAO事件

2016年4月30日开始，一个名为“The DAO”的初创团队，在以太坊上通过智能合约进行ICO众筹。 28天时间，筹得1.5亿美元，成为历史上最大的众筹项目。

THE DAO创始人之一Stephan TualTual在6月12日宣布，他们发现了软件中存在的“递归调用漏洞”问题。 不幸的是，在程序员修复这一漏洞及其他问题的期间，一个不知名的黑客开始利用这一途径收集THE DAO代币销售中所得的以太币。6月18日，黑客成功挖到超过360万个以太币，并投入到一个DAO子组织中，这个组织和THE DAO有着同样的结构。

THE DAO持有近15%的以太币总数，因此THE DAO这次的问题对以太坊网络及其加密币都产生了负面影响。

The DAO事件

针对The DAO攻击事件，当时有两种提议：

- 软分叉：进行一次软分叉，不会有回滚，不会有任何交易或者区块被撤销。软分叉将从块高度1760000开始把任何与The DAO和child DAO相关的交易认做无效交易，以此阻止攻击者在27天之后提走被盗的以太币。
- 硬分叉：要求矿工彻底解除盗窃并且归还The DAO所有以太币，这样就能自动归还给代币持有人，从而结束The DAO项目。

2016年7月20日，在块1,920,000 以太坊实施了DAO硬分叉，因此创建了两个以太坊：

- 以太坊 Ethereum
- 以太坊经典 Ethereum Classic

以太坊是什么？

- In the Ethereum universe, there is a single, canonical computer (called the Ethereum Virtual Machine, or EVM) whose state everyone on the Ethereum network agrees on. Everyone who participates in the Ethereum network (every Ethereum node) keeps a copy of the state of this computer. Additionally, any participant can broadcast a request for this computer to perform arbitrary computation. Whenever such a request is broadcast, other participants on the network verify, validate, and carry out ("execute") the computation. This causes a state change in the EVM, which is committed and propagated throughout the entire network.

以太坊是什么？

- Requests for computation are called transaction requests; the record of all transactions as well as the EVM's present state is stored in the blockchain, which in turn is stored and agreed upon by all nodes.
- Cryptographic mechanisms ensure that once transactions are verified as valid and added to the blockchain, they can't be tampered with later; the same mechanisms also ensure that all transactions are signed and executed with appropriate "permissions".

作业2

- 分析以太坊源代码，说明以太坊中Merkle Patricia Trie的数据结构、操作流程和作用。

- 提交格式：word文件
- 文件名：姓名+学号+作业2.docx
- 提交邮箱：sherlin@zju.edu.cn
- 提交截止日期：2020年12月20日12点

深入阅读

- 以太坊黄皮书
- 以太坊技术文档

谢谢！