



北京航空航天大学

Beijing University of Aeronautics and Astronautics

区块链基本原理

胡 凯

北航计算机学院

010-82339460

hukai@buaa.edu.cn

计算机学院



1. 培养接受新技术原理能力：掌握区块链的基本原理和基本技术体系；

2. 培养学科交叉思维和创新能力：掌握分布式、区块链的思维方式、结合数字经济宽广的领域，创新技术赋能模式；

3. 培养多种技术融合的工程应用设计能力：掌握分布式、区块链等技术和数字经济应用场景结合的设计方法；

□毕业要求1：能跟踪学习新技术带来的思维和理论知识，提高毕业后职业适应能力和创新。

□毕业要求2：能融合运用交叉技术，提高技术融合和工程应用能力



参考书目：

- ✓ 互链网：将来世界连接方式：蔡维德等，2020年，东方出版社
- ✓ 智能合约：重构社会契约：蔡维德等，2020年，法律出版社
- ✓ 分布式计算系统导论-原理与组成：作者 北航 胡建平，胡凯，清华大学出版社 2014；
- ✓ 区块链革命：作者 [加]唐塔普斯科特（Don Tapscott），[加]亚力克斯·塔普斯科特著；凯尔，孙铭，周沁园 译；
- ✓ 数字经济影响未来的新技术、新模式、新产业：作者 汤潇 人民邮电出版社 2019.5

课程要求：

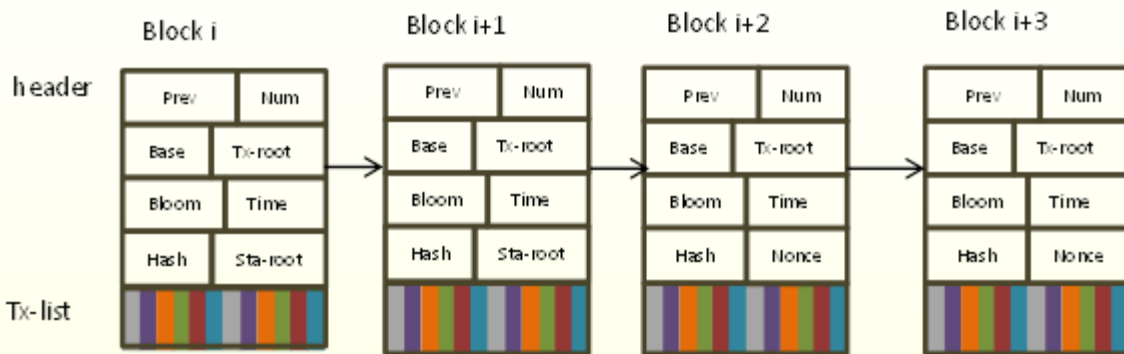
1~9周，无故旷课2次者没有成绩

课程考核：

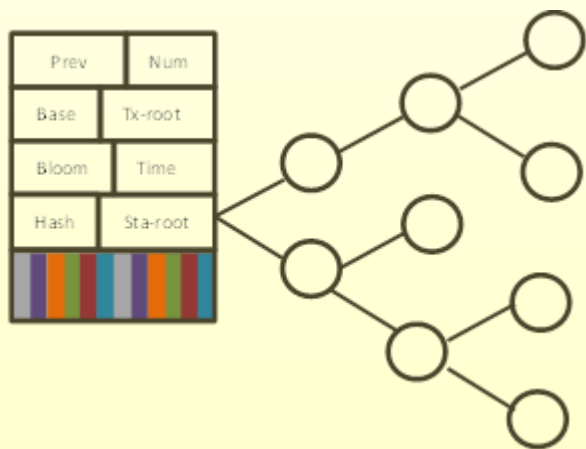
基于所学习的课程内容，针对数字经济某个应用场景，调研完成一个区块链应用设计报告。

成绩判定方法：

1. 设计目标清楚，并能结合区块链特点设计应用模式为合格；
2. 有详细应用设计，设计合理可应用性较强为良好；
3. 具有技术融合或一定的创新设计、或有一定的原型实现为优秀。



Prev: 表示前一个区块的哈希值,
 Num: 表示块号,
 Base: 表示制作区块的节点公钥地址
 Tx-root: 表示交易树根哈希值
 Bloom: 每个交易的哈希过滤器 (算法)
 Time: 表示区块构造的时间
 Hash: 表示这个块的Hash (算法)
 Sta-root: 状态树的根哈希值。



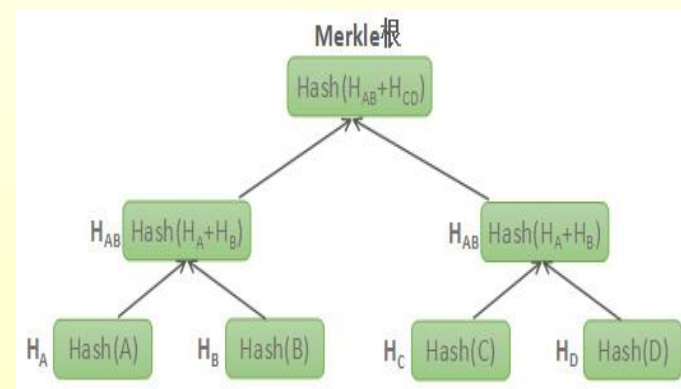
MPT{key, value}

状态树的节点:

- Key是账户的公钥地址,
- Value: 为账户的值。

交易树的节点:

- Key: 为被处理的序列号,
 - Value: 为交易的信息。
- 树存在数据库中。





北京航空航天大学

Beijing University of Aeronautics and Astronautics

区块链发展史



- 2008年11月1日，一个自称中本聪(Satoshi Nakamoto)的人在一个隐秘的**密码学**讨论组上贴出了一篇研究报告，报告阐述了他对电子货币的新构想，比特币就此问世！
- 2010年12月12日6点22分，就在他发帖争辩给维基解密捐赠比特币7天后，中本聪在论坛发了最后一个帖子。
- 2009年1月3日，中本聪制作了比特币世界的第一个区块“**创世区块**”并挖出了第一批比特币50个。
- 2010年5月21日，佛罗里达程序员用**1万比特币购买了价值25美元的披萨优惠券**，随着这笔交易诞生了比特币第一个公允汇率。
- 2010年7月，第一个比特币平台成立，新用户暴增，价格暴涨。
- **2011年2月**，比特币价格首次达到1美元，此后与英镑、巴西雷亚尔、波兰兹罗提汇兑交易平台开张。
- 2012年，瑞波(Ripple)发布，其作为数字货币，利用区块链转移各国外汇。
- 2013年，比特币暴涨。美国财政部发布了虚拟货币个人管理条例，阐明虚拟货币释义。
- 2014年，以**中国为代表的矿机**产业链日益成熟，同年，美国IT界认识到了区块链对于数字领域的跨时代创新意义。
- 2015年，美国纳斯达克证券交易所推出基于区块链的数字分类账技术Linq进行股票的记录交易与发行。

实现了数据传输中对数据的自我证明，降低了全球“信用”的建立成本



- 2015.7 以太坊发布Frontier版，开创区块链2.0时代
- 2015.10. 《经济学人》封面文章，制造信任的机器—比特币后面的技术将如何改变世界
- 2015英国宣布区块链为**国家战略**
- 证券交易所：2015年12月，纳斯达克首次在个股交易商使用区块链技术，基于区块链技术的交易平台Linq
- 会计审计机构：主要会计事务所都宣布进军区块链，德勤推出软件平台Rubix
- 银行体系：区块链联盟R3 CEV近期宣布，50家银行，R3分布式账本：CORDA
- 高盛、摩根大通、德银等机构以及Blythe Masters（华尔街CDS女皇）普遍认为区块链将会成为互联网金融的下一站
- 7家华尔街公司成功测试区块链用于信贷违约掉期交易
- 英格兰银行准备发行中心化数字货币RSCoin
- 美国商品期货交易委员会尝试将区块链应用于票据交换
- 科技企业：IBM开放式账本项目（Open Ledger Project）
- IBM和三星展示了应用区块链技术控制互联设备的概念平台ADEPT
- 区块链技术公司BitSE：区块链防伪平台Vechain业务
- 万云区块链云平台提供的BaaS
- 井通的“企业级钱包”，服务海航的供应链融资



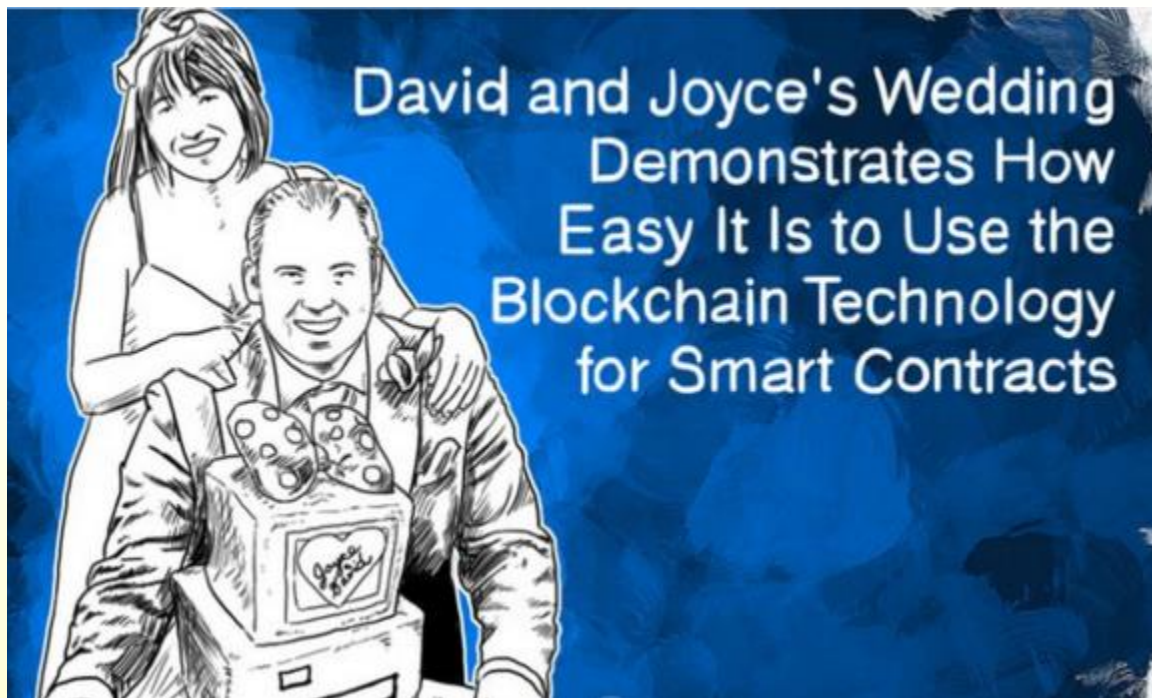
北京航空航天大学

Beijing University of Aeronautics and Astronautics

区块链思维



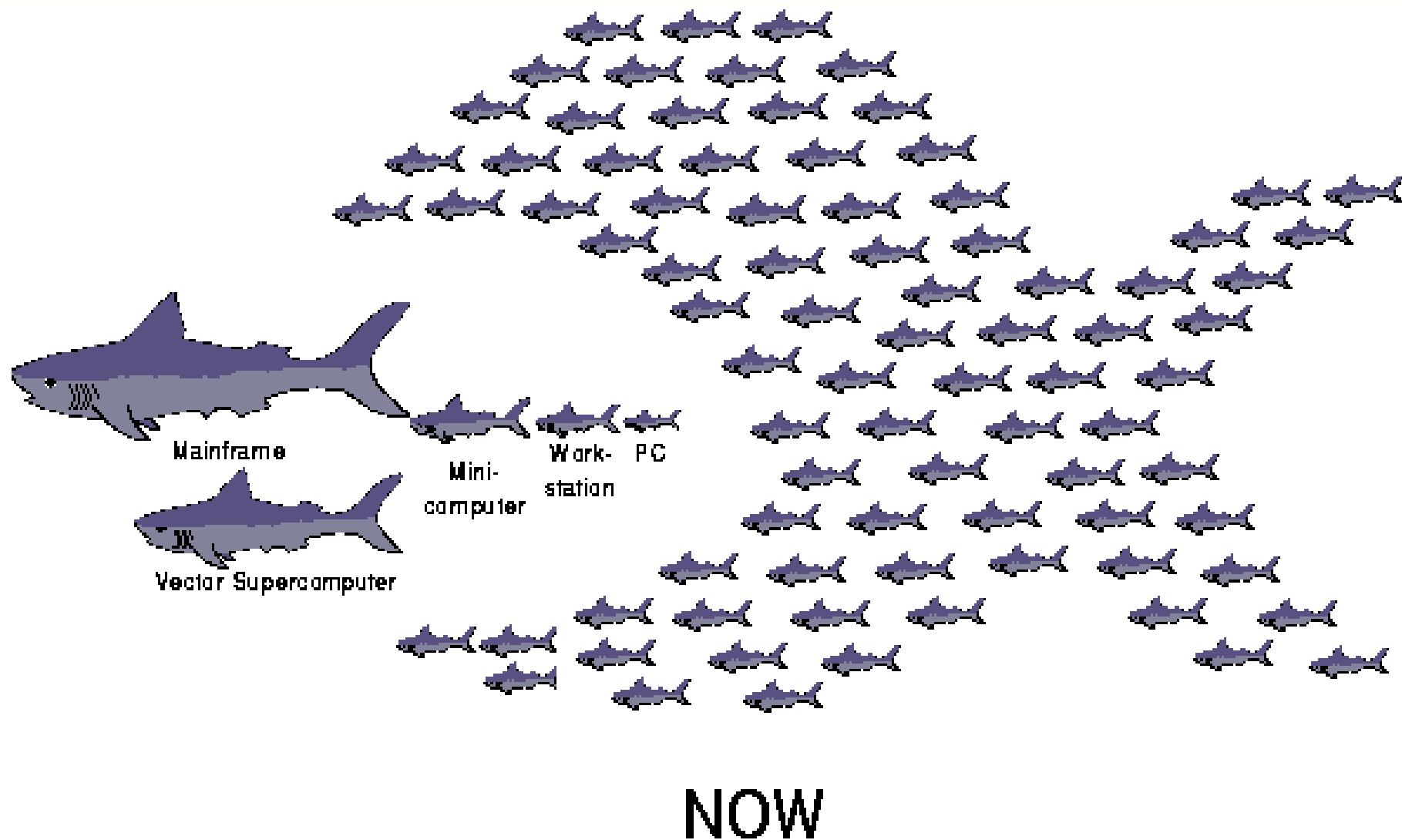
第一场区块链婚姻



- 2015. 10. 5，第一场区块链婚礼上，两位新人将他们的结婚誓言记录在了区块链上，整个过程都没有政府或是宗教的介入。区块链里将会有伴侣之间的真正合约。



- 浏览器的创始人马克. 安德森：“区块链是互联网上一直被需要却有一直没有实现的**分布式可信网络**，这是我们一直在等待的重大突破，它把一切问题都解决了，应该得诺贝尔奖”
 - 区块链是一种分布式账本，价值传递互联网和货币互联网
 - 区块链是一种数据库革命，它是时间轴数据库！
 - 区块链是下一代互联网！重塑商业世界
 - 37%**的受访者认为，2016年互联网金融颠覆的最大机遇所在
 - 据Accenture预测，“引爆点”预计将在**2018**年发生，增长期到**2025**年。
 - Gartner公布的2017年十大技术趋势之一
- 区块链技术有潜力引领所有行业的潮流！**





- **低带宽通讯**：竖尾、表情转达攻击信息
- **层次管理**：头狼领导，通过等级传递不断队形
- **动态重组**：组成不同的战斗队来攻击不同的猎物，包括不同的策略（如攻击小鹿或驼鹿）
- **协同分工**：一旦攻击，各狼会分工合作攻击不同部位
- **冗余结构**：一只狼受伤或退出，另一只将接替



复杂的集体行为能够通过简单的个体行动的自同步来实现！



- 1982年 Lamport提出拜占庭将军问题

David Chaum提出密码学支付系统，具有不可追踪的特性，是比特币区块链在隐私安全面的雏形

- 1985年 提出椭圆曲线密码学
- 1990年 提出Paxos
- 1991年 使用时间戳确保数字文件安全(被比特币采用)
- 1992年 提出椭圆曲线数字签名算法ECDSA
- 1997年 Adam Back发明Hashcash技术
一种工作量证明演算法，应用于阻挡垃圾邮件
- 1998年Wei Dai发表匿名的分散式电子现金系统B-money
许多设计之后被比特币区块链所采用
- Nick Szabo发表Bit Gold：参与者可贡献运算能力来解出加密谜题
- 2005年 Hal Finney可重复使用的工作量证明机制（RPOW）出现
- 2008年比特币区块链



- 1974诞生的TCP/IP协议：决定了区块链在互联网技术生态的位置
- 1984年诞生的思科路由器技术：是区块链技术的模仿对象
每台路由都保存完成的互联网设备地址表，一旦发生变化，会同步到其他路由器上，确保每台路由器都能计算最短最快的路径。
- B/S（C/S）架构：区块链的对手和企图颠覆的对象
1989年欧洲物理学家蒂姆·伯纳斯·李发明万维网并放弃申请专利。此后近30年中，包括谷歌，亚马逊，facebook，阿里巴巴，百度，腾讯等公司利用万维网B/S（C/S）结构，成长为互联网的巨头。
- 对等网络（P2P）是区块链的父亲和技术基础
- 哈希算法：产生链和加密的关键
- 共识算法是区块链核心



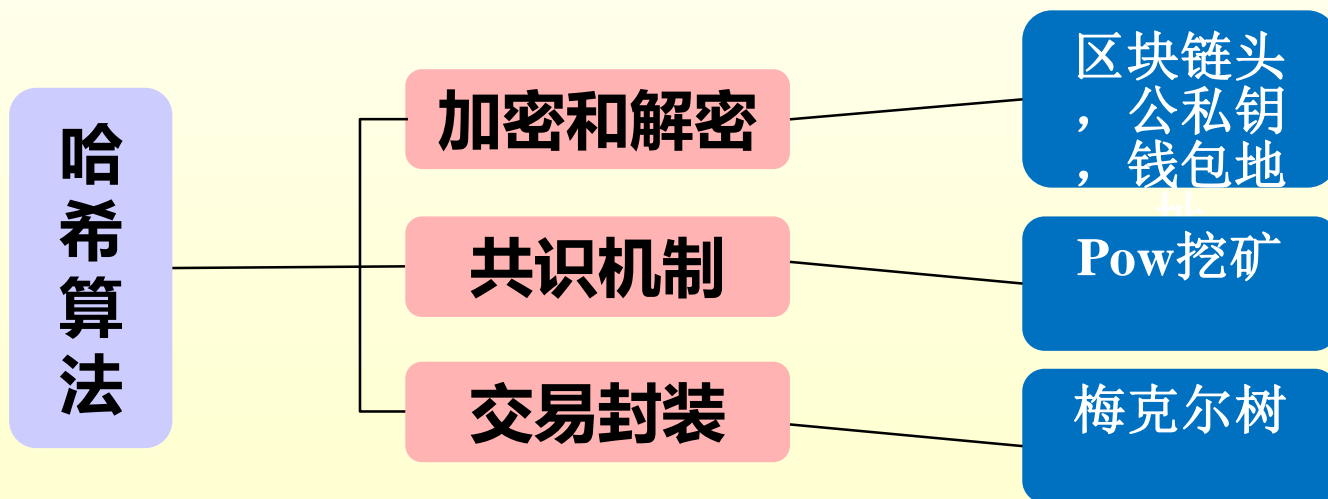
区块链基础知识



哈希算法的核心作用

SHA256算法为例，无论输入是什么数据文件，输出就是256bit，16进制是64位，如：

00740f40257a13bf03b40f54a9fe398c79a664bb21cfa2870ab07888b21eeba8





特性:

- 碰撞阻力: 如果无法找到两个值 x 和 y , $x \neq y$, 但 $H(x) = H(y)$, 则称 H 具有碰撞阻力

应用: 文件信息摘要, 用户上传文件

- 隐秘性: 知道 $H(x)$ 不能反推出 x

应用: 承诺 投标书

$com := commit(msg, nonce)$

已知 com , 没法找到 msg

没有两组不同的 $(msg, nonce)$, 有相同的 com

- 谜题友好: 如果输入有一些非常随机, 将很难求得哈希的输入
- 易压缩, 易计算



主要包括MD4, MD5, SHA-1, SHA-2

MD4: RFC1320定义, MIT的Ronald 1990 设计, 128位, 已不安全

MD5: RFC1321定义, MD4的改进版, 128位, 不具备强抗碰撞性

SHA系列, 美国国家安全局设计, 美国国家标准

SHA-0: 1993年发布, 很快被撤回

SHA-1: 1995年发布, 应用广泛, 2017年荷兰和谷歌称破解了它

SHA-2: 2001年发布, 包括了SHA-224, SHA-256, SHA-384...等, 至今安全

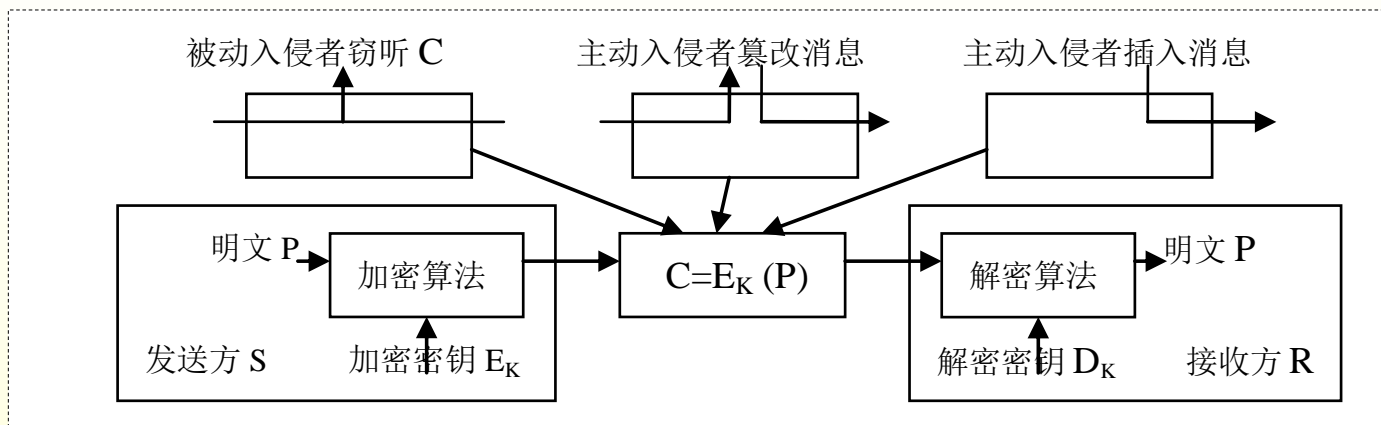
SHA-3: 2015年发布, 新的算法

RIPEMD160算法, RACE原始完整性校验消息摘要, 比利时COSIC小组开发, 1996年发布, 改进了MD系列。

2004年中国王小云团队破解MD系列, 2005年破解SHA-1



- 古典密码学：1949年前，基于加密算法的保密性，古老的算法。
- 现代密码学：1949年，香农“保密系统的通信理论”，对称密码学，DES和AES，基于密钥安全，复杂计算密码。如通过AES对钱包私钥进行加密，保护私钥。
- 公钥密码学：1976年 Whitfield Diffie 和Martin Hellman提出基于数学难题的公钥密码机制。1978，RSA公钥密码机制提出。
- 椭圆曲线密码体制ECC：1985年 Neal Koblitz和Victor Miller分别提出的，比RSA更安全、更小的密钥，私钥可通过伪随机数生成。**证明数据是签名者发出的，不可抵赖的。**
- 椭圆曲线数字签名算法ECDSA：1992年 Scot和Vanstone提出，成为国际标准
- Base58编码：一种可读的编码方式，如用于产生钱包地址



防止三种不同的安全威胁：窃听、篡改消息、插入信息。

加密密钥和解密密钥是否相同？

- 对称密码体制： $P = Dk(Ek(P))$
- 非对称密码体制： $P = Dk1(Ek2(P))$

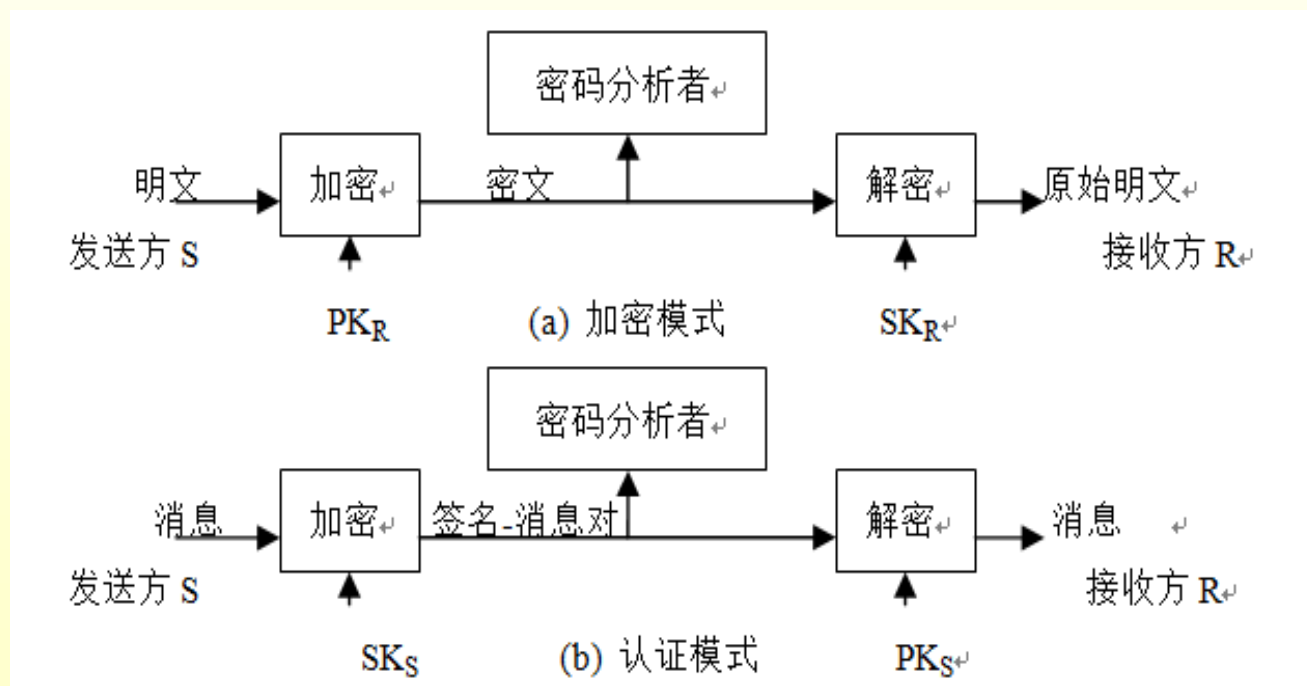


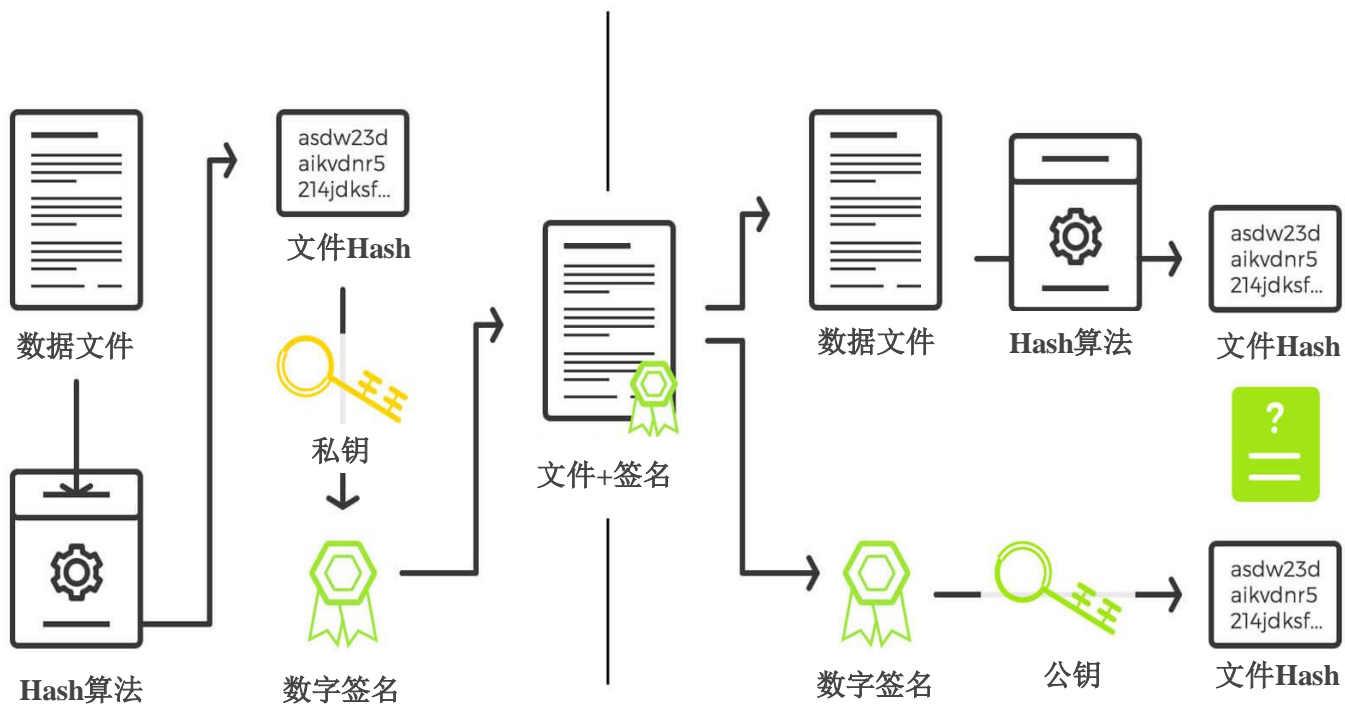
非对称密码体制

- 基于大数因子分解困难的RSA（发明者缩写）公钥密码系统
- 基于椭圆曲线离散对数求解困难的ECC(Elliptic Curves Cryptosystem)公钥密码系统

加密模式：发送方用接受方公钥加密，接受方用自己私钥解密

认证模式：发送方用自己私钥加密，接受方用发送方公钥解密



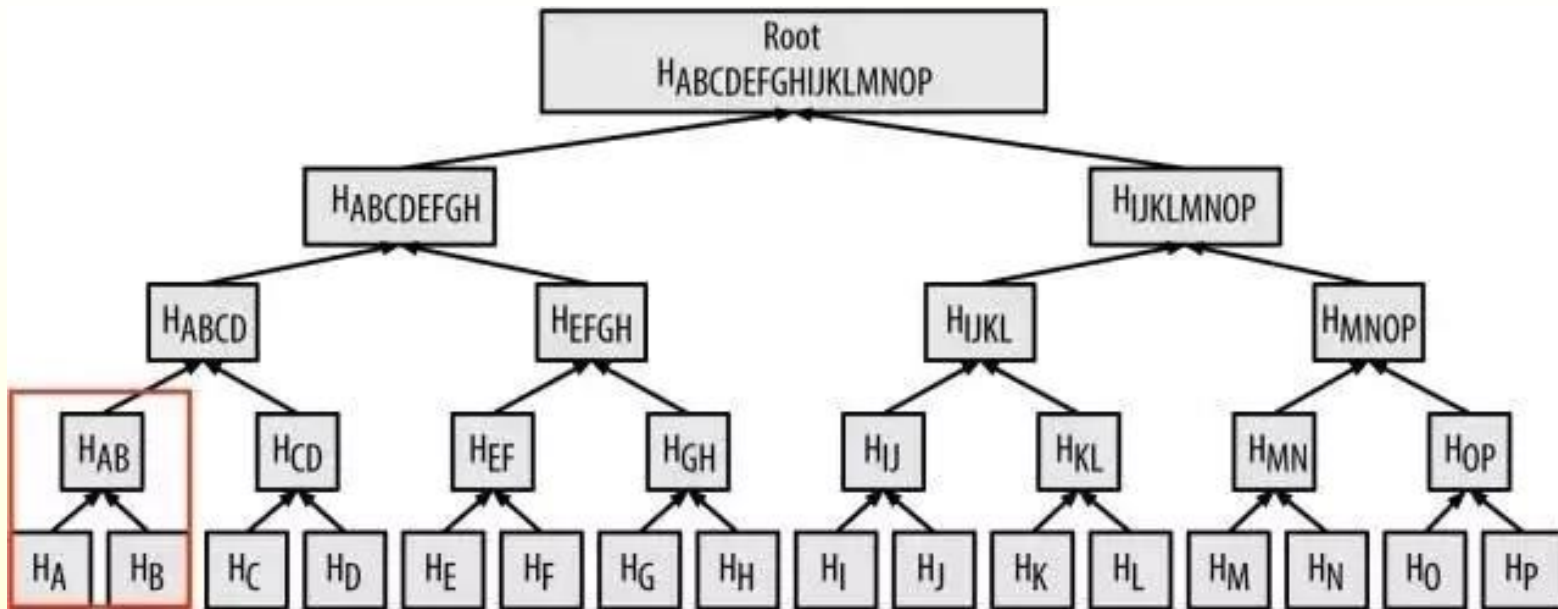




1989年著名密码学家Merkle提出

- 2003年 Michael Szydło采用尽量减少冗余哈希函数值，对树的遍历算法和计算量进行了改进
- 2005年 Glen Nuckolls提出了混合树的思想，降低了对树高的依赖
- 2006年 W. E. Hall提出平行认证树的思想，借助外部设备，实现对多个叶子节点同时进行认证。
- Johannes Buchmann提出CMSS方案，提高数的签名数量，后又提出了新一代时空大数据平台，使签名数量进一步增加。

如何进一步提高Merkel树认证的效率仍然有待于进一步研究

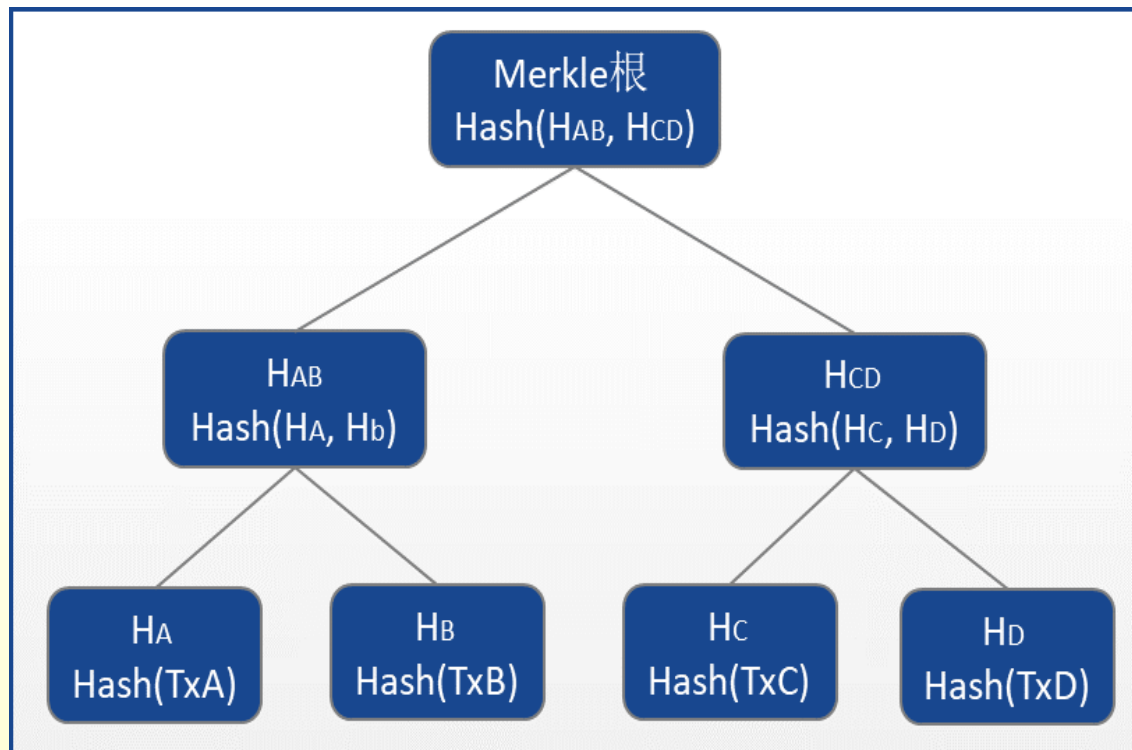


2048个交易,进行11次打包, 得到1个哈希值
65536个交易,进行16次打包, 得到1个哈希值
1048576个交易,进行20次打包, 得到1个哈希值

每次哈希计算大约需要0.01秒, 计算20次需要0.2秒



Merkle 树快速验证



防篡改

树种任意叶子节点发生改变，都会导致最终树根的该表，因此可以通过树根唯一对应一组叶子节点

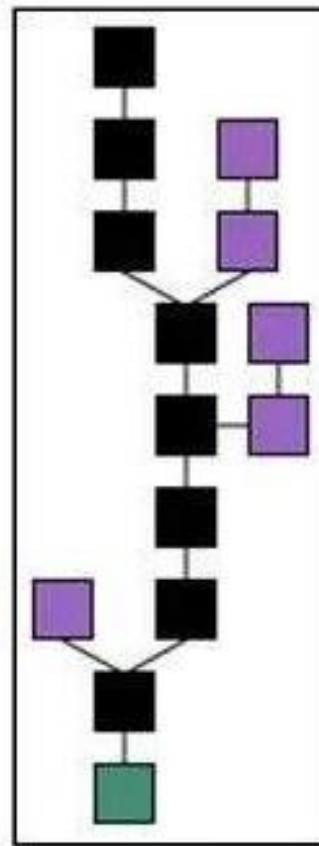
存在性证明

可以使用少量数据快速证明一个节点的存在性，比如证明 H_D 存在于树中，只需要提供 H_C 、 H_{AB} 、 Root 其他人便可以通过计算快速验证

什么是区块链

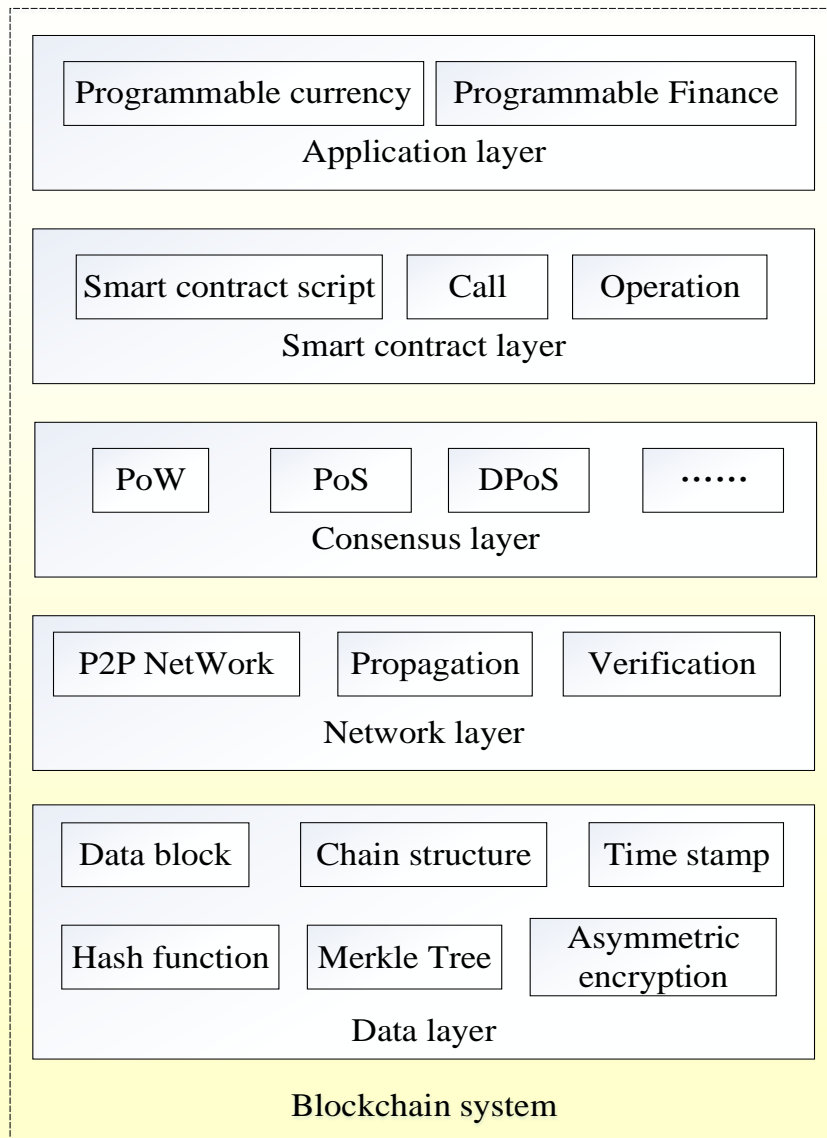
- **区块链**是一种“去中心化”的数据库，包括一张被称作“区块”（Block）的列表，其中每个区块都含有一个“时间戳”（Timestamp）、一条与前一个区块的“链接”（Link）和交易数据。
- **区块链**是一个基于比特币协议的不需要许可的分布式数据库，他维护了一个持续增长不可篡改的数据记录，对运营者也是如此。

如图所示，黑色的区块主链，而紫色的孤立区块在主链之外存在





区块链分层结构



四大特征：
Distributed,
Autonomous,
Contractual,
Trackable

- 1. 公有区块链 (Public blockchains)

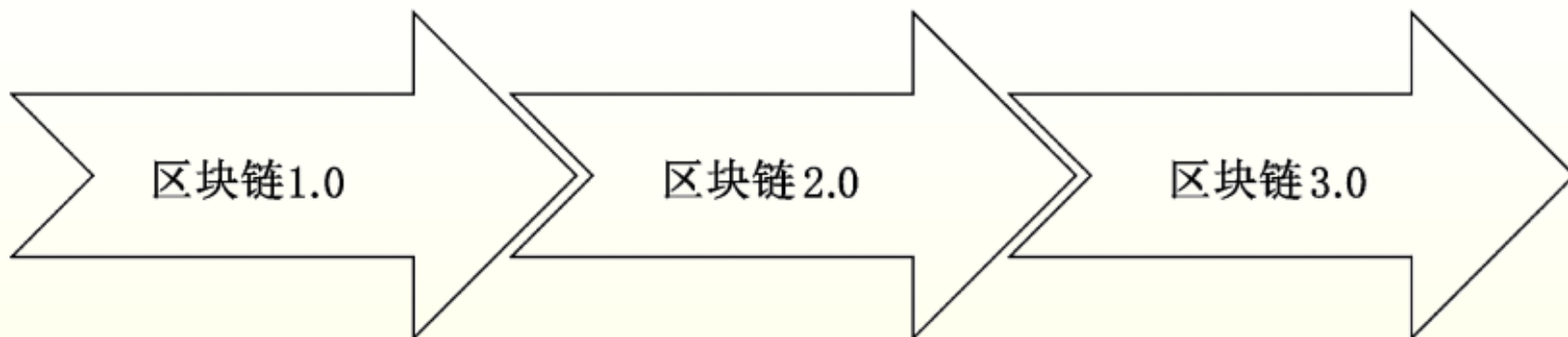
公共区块链是指全世界任何人都可读取的、任何人都能发送交易且交易能获得有效确认的、任何人都能参与其中共识过程的区块链——共识过程决定哪个区块可被添加到区块链中和明确当前状态。作为中心化或者准中心化信任的替代物，公共区块链的安全由“加密数字经济”维护——“加密数字经济”采取工作量证明机制或权益证明机制等方式，将经济奖励和加密数字验证结合了起来，并遵循着一般原则：每个人从中可获得的经济奖励，与对共识过程作出的贡献成正比。这些区块链通常被认为是“完全去中心化”的。

- 2. 联盟区块链：(Consortium blockchains)

联盟区块链是指其共识过程受到预选节点控制的区块链；例如，不妨想象一个有15个金融机构组成的共同体，每个机构都运行着一个节点，而且为了使每个区块生效需要获得其中10个机构的确认（2/3确认）。区块链或许允许每个人都可读取，或者只受限于参与者，或走混合型路线，例如区块的根哈希及其API（应用程序接口）对外公开，API可允许外界用来作有限次数的查询和获取区块链状态的信息。这些区块链可视为“部分去中心化”。

- 3. 完全私有区块链 (Fully private blockchains)

完全私有的区块链是指其写入权限仅在一个组织手里的区块链。读取权限或者对外开放，或者被任意程度地进行了限制。相关的应用囊括数据库管理、审计、甚至一个公司，尽管在有些情况下希望它能有公共的可审计性，但在很多的情形下，公共的可读性并非必须是必须的。



可编程货币

区块链 1.0 的主要功能是数字货币，是加密货币的应用，它构建了去中心化的数字支付系统，实现了快捷的货币交易、跨国支付等多样化的金融服务，它的主要代表是比特币

可编程金融

区块链 2.0 时代主要是智能合约的应用，主要领域扩展到金融领域，是智能资产、智能合约市场的去中心化，可作货币之外的数字资产转移，区块链在金融和市场的应用中更加广泛

可编程社会

区块链技术以去中心化的方式配置全球资源，进一步延拓到货币、经济和市场以外的领域，其潜在的应用领域包括选举、医疗、公证、版权，以及网络安全、汽车租赁、学历鉴定等



北京航空航天大学

Beijing University of Aeronautics and Astronautics

区块链1.0



区块链1.0： 比特币

提出分布式账本概念：

- 存放在互联网的分布式节点上，每个节点有完整的备份
- 记录着自诞生以来的所有转账交易记录
- 分区块链存储，每块包括部分记录，每一个区块有前一块的ID，形成一个链状的结构
- 发起交易时，把交易信息广播到P2P网络中，纪录到一个区块中，新区块链接到区块链上。

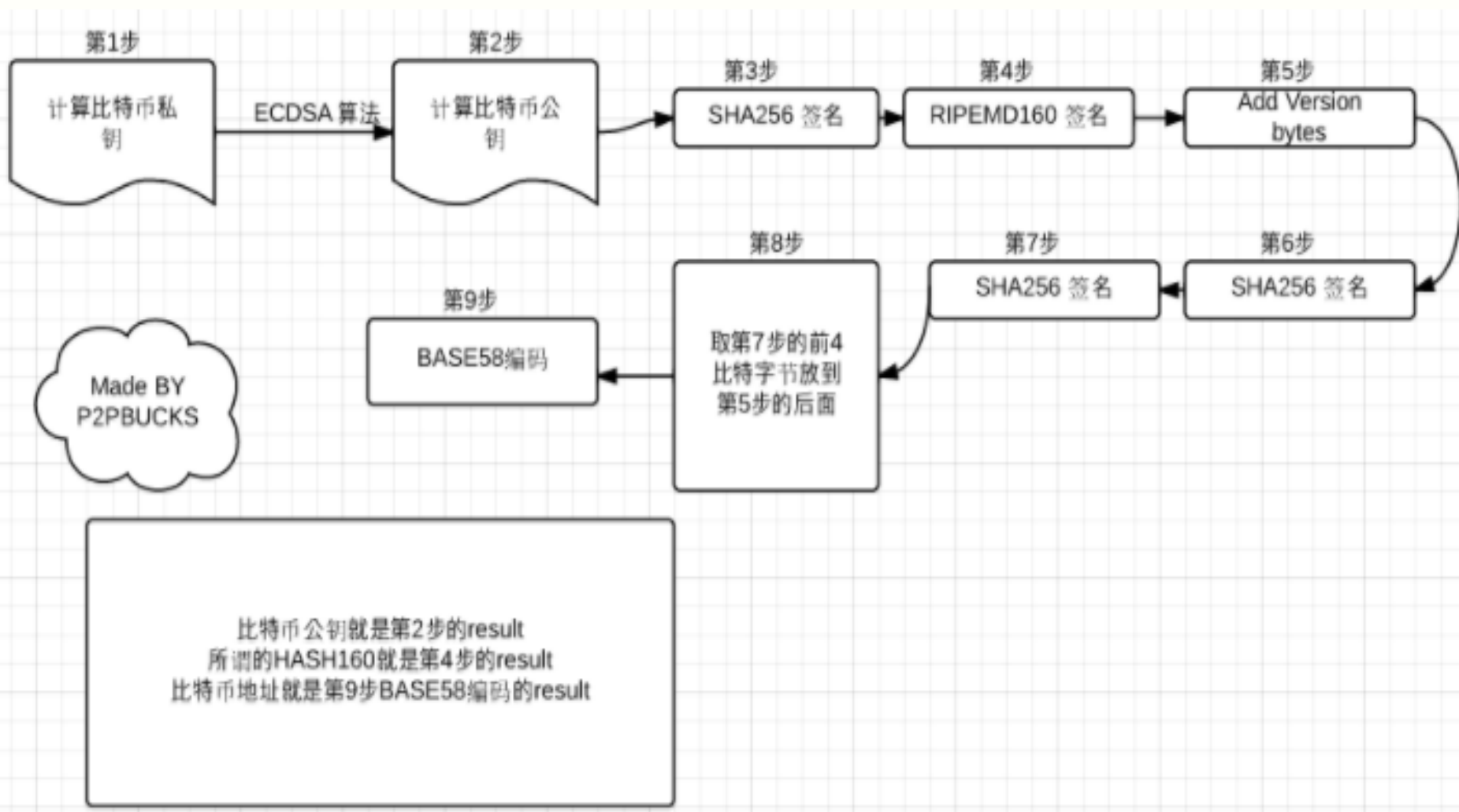


- **比特币**：意在解决“法币滥发”经济难题，是不可逆的发明，第一次在数字世界创造了一种不可复制又不是中心化的路径，“论子、发动机、复印件、互联网、移动电话、传感器、社交软件”等，没有时，不觉得少什么，一旦应用，则不可或缺，世界无法回到没有它的状态。
- **激励机制**：将一定数量新货币持续添加到货币系统中的方法，非常类似于耗费资源去挖掘金矿，并将黄金注入流通领域。CPU的时间和电力消耗就是资源消耗。
- **利益驱动诚实**：如果有人能调动相当的计算力，他要面临一个选择，将其用于产生新的挖币，还是将其应用于双花攻击？他会发现，诚实工作更有利所图，不是破坏规则 and 系统，使其利益受损。
- **工作量证明机制**：解决了集体投票表决时谁是大多数的问題。一个IP一票易被破坏，按CPU算力投票，最长的链包括最大的工作量。

攻击者试图赶上随后的区块，其成功概率将呈指数化递减。



比特币地址生成





Account: 易于理解、节省空间、易于实现、模式成熟

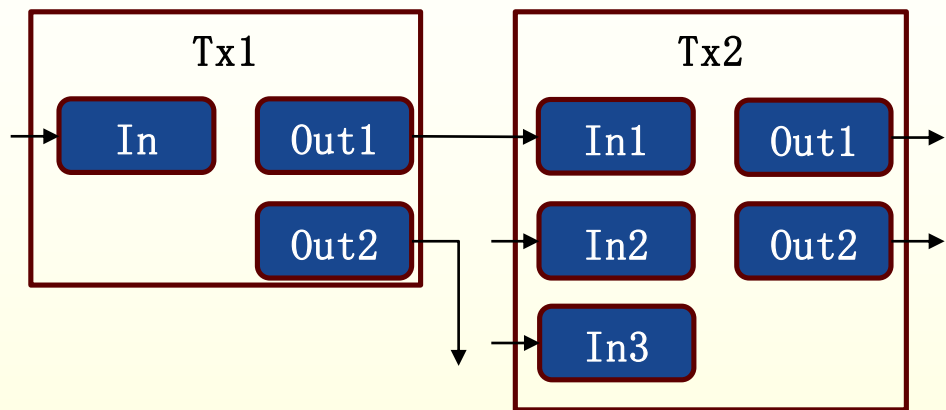
UTXO: 比特记账方式，原理简单、易于扩展、高度并行、隐匿性强



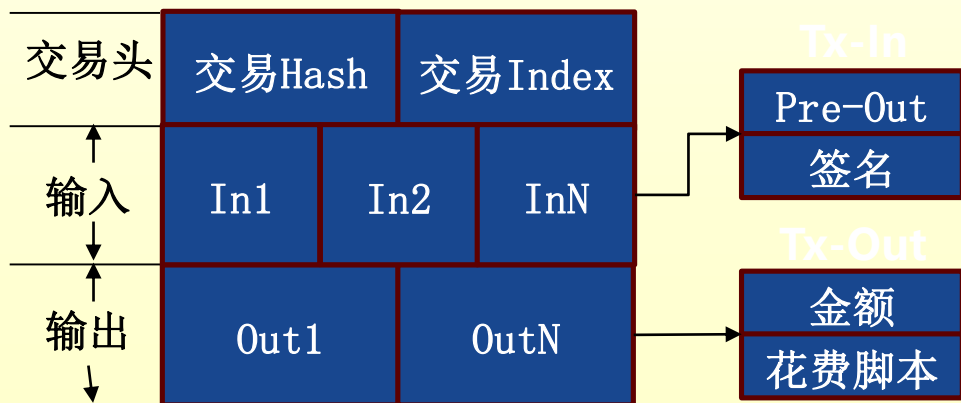


UTXO (unspend transaction output)

- 所有交易花费的都是前置交易未被花费的输出
- 所有正常的合法交易的输入都必须有一个对应的未花费的前置交易的输出
- 使用一个用类Forth脚本语言编写的脚本去验证比特币的交易



交易结构





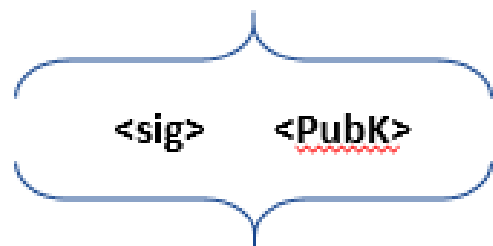
相对于传统的账户模型，具有以下好处：

1. 账户数据库会不断膨胀，因为账户不会被删除，而UTXO数据库体积会小很多。
2. 由于只有未花费的输出会被保留，所以每一个比特币用户可以拥有几乎无限多的地址，提高了匿名性。
3. UTXO为高并发的交易带来可能，试想传统的账户模型，每个人的账户交易必须是线性的，无法并发。而现在每个人可能拥有多个UTXO，可同时发起多笔交易，实现了并发



脚本语言包含许多操作，但都故意限定为一种重要的方式——没有循环或者复杂流控制功能以外的其他条件的流控制。这样就保证了脚本语言的图灵非完备性，这意味着脚本的复杂性有限。

交易签名



+

花费脚本



- 交易输入 TxIn 携带的签名信息
- 主要包括：数字签名，公钥信息

- 验证公钥是否是收款账户对应的公钥
- 验证数字签名是否合法

1. 交易费是一种激励，同时也是一种安全机制，使经济上不利于攻击者通过大量交易来淹没网络。
2. 交易费是通过字节大小来计算，而不是花费比特币的多少。
3. 目前交易均使用动态交易费，交易费的高低会影响交易被矿工处理（加入区块链）的优先级，交易费过低或为0的交易极少会被处理，甚至不会在网络上广播。
4. 交易费不会存储在交易信息中，而是通过输出(output)与输入(input)的差值来替代（如果你要自己创建交易必须千万注意输出的金额大小，因为交易的差值无论大小全部会被矿工获得）

$$\text{交易费} = \text{SUM（所有输入）} - \text{SUM（所有输出）}$$

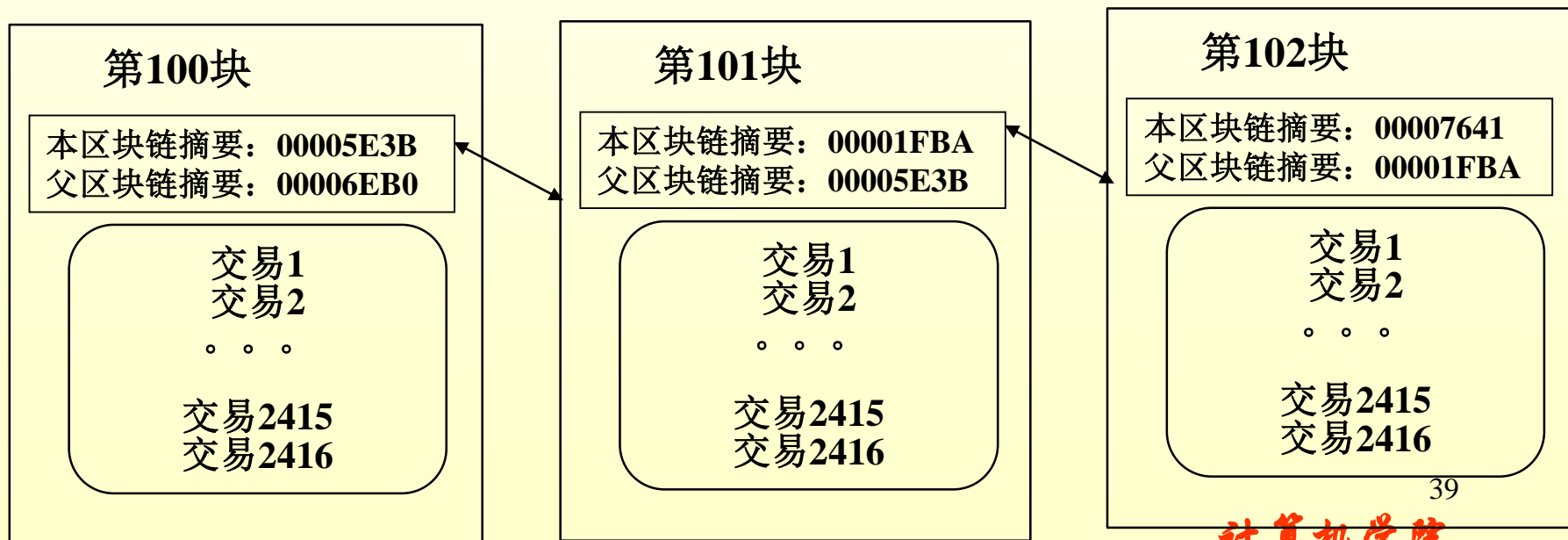
5. 预估当前合适的交易费



区块链结构

区块链有如下特点：

- 点对点非中心化
- 时序数据
- 集体维护
- 安全可信
- 可编程





区块的数据结构

大小	字段	描述
4字节	区块的大小	表示该字段之后的区块大小
80字节	输入计数器	区块头的组成字段
1-9字节	交易计数器	交易的数量
可变的	交易列表	许多交易



区块链头的数据结构

大小	字段	描述
4字节	版本	跟踪软件协议的更新
32字节	父区块哈希值	引用上一块的哈希值
32字节	Merkle树根	交易的Merkle根的哈希值
4字节	时间戳	区块产生的近似时间（UNIX）
4字节	难度目标	工作量算法的难度目标
4字节	随机数Nonce	工作量证明的计数器

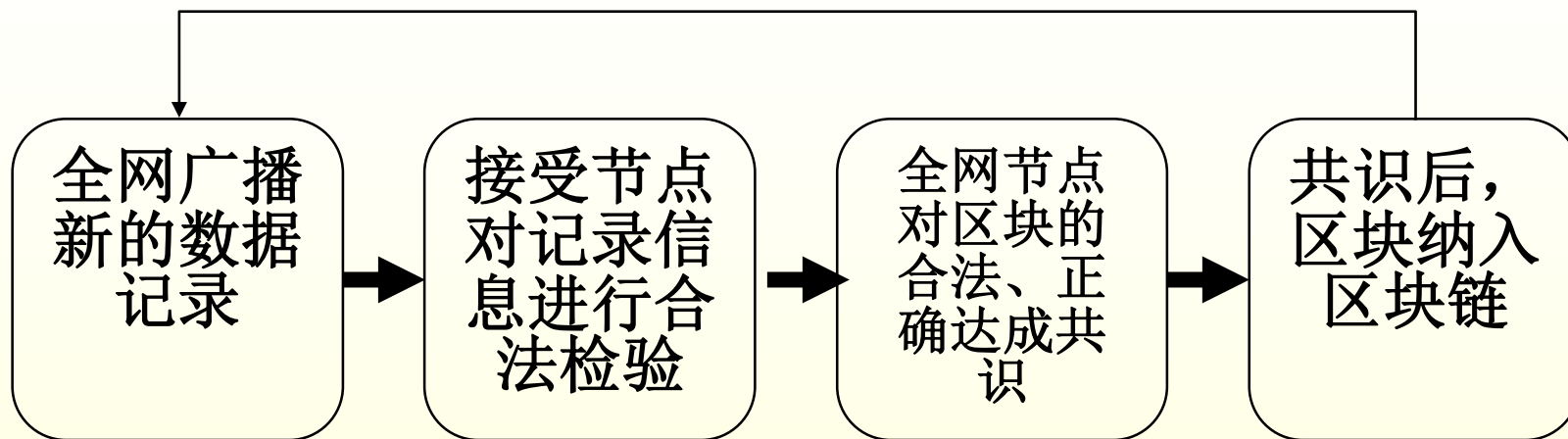


大小	字段	描述
4字节	版本	明确这笔交易参照的规则
1-9字节	输入计数器	被包含的输入的数量
不定	输入	一个或多个交易输入
1-9字节	输出计数器	被包含的输入的数量
不定	输出	一个或多个交易输出
4字节	时钟时间	一个UNIX时间戳或区块号



区块构造:

- 交易广播：每一笔交易需要全网广播，取得全网节点的验证后，才能进入区块链，这是信任的第一步
- 最先完成区块封装的节点把结果广播出去，其他节点停止构造，进行验证：
 - 区块链的数据结构在语法上有效
 - 区块头的哈希值小于目标难度，包含足够的工作量证明
 - 区块链时间戳早于验证时刻两个小时，允许时间错误
 - 区块大小在长度限制之内
 - 第一个交易是Coinbase交易
 - 验证区块内的交易是有效的
- 保证区块链的唯一性：6个区块链的缓冲，马太效应，全网节点都会选择最长链继续做块。



- 平均每10分钟竞争产生一个区块，每个区块产生时间不一样
- 每个区块大小1M，每笔交易需要 250B 来存储数据，约可记录 4194笔交易
- 因此大约TPS是每秒7笔交易
- 需等待6个后续块最终写入，需要60分
- 每2016个区块链会调整一次难度值，变难或变易，各节点按统一公式调整，差不多2周
- 总量2100万个，给挖矿人奖励一定的币，交易费约为0.00001个



区块链如何是可信的

- 区块链网络是公开的，节点知道过去发生的所有数据记录，之前的区块真实性是严格保证的
- 修改数据几乎不可能：要重头构造一条长度比之前的还要长的链，才能制造双重支付等虚假数据
- 多节点存储设置，各节点保存了大量数据完整数据
- 加密机制，采用哈希机制的多重加密手段
- 共识算法，各种共识算法对交易和构造新块进行确认和验证
- 经济规律，让破坏付出的代价远远大于能获得的收益



- 拥有超过全网一半算力下，有可能推翻已经确认过的交易，产生分叉，完成双花获利。
 - 修改自己的交易记录，如双重支付；
 - 阻止确认部分或全部交易。
- 经济学角度，攻击获取的收益要大于成本，51%攻击成本巨大，收益却很小，实际上会选择诚实的工作和理性的选择

如Ghash社区，接近51%算力，造成社区恐慌，矿工撤离，从而算力急剧下降。



- **交易：**交易被创建并签名之后将会被广播入比特币网络，然后每一个节点会**验证并同时传播**该交易。节点验证交易后，该节点可以选择验证完成的交易加入他创建的区块中。一旦挖矿成功，区块将上链，其中的交易将永远成为账本（区块链）的一部分。一段时间的交易，打包为一个区块，依次相连形成区块链
- **验证：**
 - 历史上发生过的所有交易信息分为两类，一类为“验证过”的交易信息，即已经被验证过的交易信息，它保存在一连串的blocks”里面。另外一类指那些还“未验证”的交易信息。
 - 由于该节点保存了历史上**所有的**交易信息，它可以推算中在当时每个地址的账面余额，从而可以推算出该交易信息是否有效，即付款的账户里是否有足够余额。
 - 交易可能有效也可能无效。就如同签支票一样，人们可能签订一张有效支票，也可能是**“空头支票”**，但是只有当金融系统承认该支票的有效性之后人们才能使用它。交易只有验证完比特币的所有权之后，才能进入“交易池”，等待节点收入并进行挖矿。



- **同步**：以区块为单位同步交易数据，如A节点块高度是200，B节点是210，则**A向B请求同步**201~210这10个区块。
- **打包**：打包交易的节点（矿工）将获得币作为酬劳。因此，有大量节点想打包，为了区块链的唯一性，节点使用“**竞争方式**”争夺打包权。**即挖矿**，计算哈希值算法（不是预置的谜语）
- **出块**：一旦有个节点算出规定值，则广播区块，其他节点收到后验证无误，就会停止竞争改区块，而改为竞争下一区块。
- **奖励**：生产block的速度：平均2016个每两个星期，大约10分钟生产一个。生产block的奖励数量：开始每生产一个**block奖励** 50BTC，每四年减半，2013年开始奖励25BTC，2017年开始奖励额为12.5BTC。另一部分来自交易手续费，每KB交易100~1000聪的手续费
- **意义**：公平地分配初始币，虽然消耗了大量电力等资源，但与消耗资源**去挖金矿洞**一样。

■ 所有权：

所有权是通过数字密钥、比特币地址和数字签名三者确立的。比特币地址就好比银行卡号，数字密钥似银行卡密码，而数字签名就似个人签名。

- 和银行账户不一样的地方在于，银行会保存所有的交易记录和维护各个账户的账面余额，而bitcoin的交易记录则由整个P2P网络通过事先约定的协议共同维护。
- 没有一个地方维护每个地址的账面余额。它只能通过所有历史交易记录去实时推算账户余额。

最长链原则：

矿工可以在任意区块基础上计算下一个区块，但只有最长区块链上的区块才能获得系统的承认并得到挖矿奖励，这个奖励在该区块上增加**99个新区块**之后才能使用，是不分裂的重要机制。

拜占庭容错问题

由Lamport教授在1982年提出： 解决错误节点模型

- 包围敌国的四周，依靠通信兵相互通信来协商进攻意向及进攻时间。困扰这些将军的问题是，他们不确定他们中是否有叛徒，叛徒可能擅自变更进攻意向或者进攻时间。在这种状态下，拜占庭将军们能否找到一种分布式的协议来让他们能够远程协商，从而赢取战斗？

其他共识机制：

- 工作量证明机制(Proof of Work, POW)
- 股权证明机制(Proof of Stake, POS)
- 瑞波共识机制(Ripple Consensus)
- 授权股权证明机制(DPOS)
- 基于交易的股权证明机制(TaPOS)



PoW协议：1993年 C. Dwork 和M. Naor在1993年论文中提出
1999年 M. Jakobsson 和 Ari Juels文章提出
应对经服务攻击和其他服务滥用的经济对策。

PoW原理：

不对称性，客户端有一定难度工作，验证方很容易检查客户端是否做了相应的工作。

PoW过程：

- ✓ 构建区块，组成交易列表，生产Merkle根
- ✓ 把Merkle根和其他字段组装为区块头，将区块头80字节作为哈希
- ✓ 不停变换区块头中随机数nonce, 做双重SHA-256运算
- ✓ 与难度值对比，如果小于则工作量证明完成
前导0个数越多，则代表难度越大。



- **工作量证明**：让计算节点计算一组数学公式，这个过程被称为“挖矿”，其他参与节点可以用相关数学公式去验证这个值是否有效。
- **比特币的挖矿**：找到一个随机数Nonce，使得这个区块头的哈希值小于设定的难度值。
- **挖矿成功**：矿工不断变化数据数计算哈希值，与网络目标值比较，如果小其最接近难度值，则向全网广播这一随机数Nonce，其他矿工验证通过，则区块成为新区块，该矿工获得一定数量的比特币。

所有矿工一起寻找答案的过程，每个矿工都有找到答案的可能

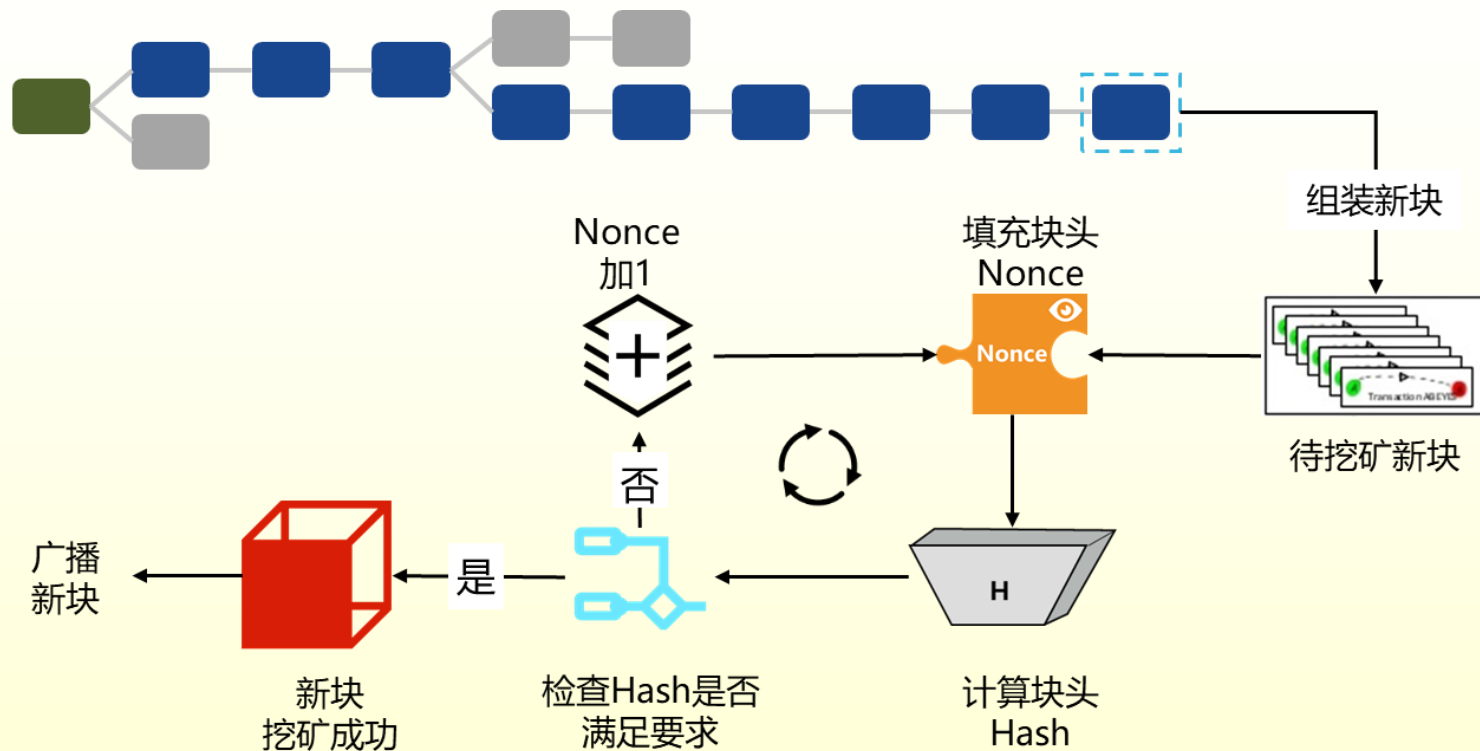
- **难度值**根据平均10分钟的比值，调整变难或易，每产生2016个块调整一次



- 构建区块链，生产Coinbase交易，并与其他所有准备打包进区块的交易组成交易列表，通过Merkle树算法生产Merkle根。
- 组装成区块头，把区块头80字节数据作为工作量证明的输入
- 不停的变更区块头中的随机数，即nonce的数值，并对每次变更后的区块头做双重的SHA256哈希：
$$\text{SHA256}[\text{SHA256}(\text{区块头})]$$
- 将结果值与当前网络的目标难度值做对比，小于则解题成功，工作量证明完成。

安全性高，非中心化！

挖矿过程



概率共识, Nonce计算(挖矿), 最长链原则, 6块确认
难度调整

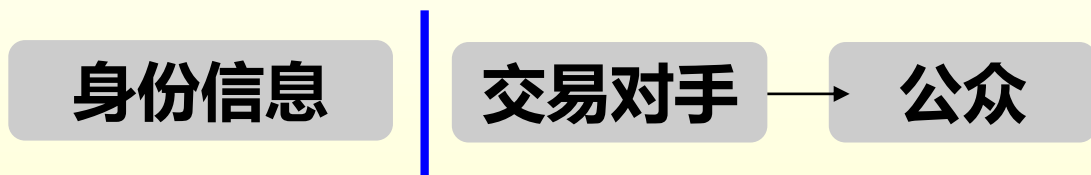
难度要求: 每一块块头的256位Hash值前置0的个数, 可以根据历史块的出块时间计算所得。



传统隐私保护



区块链隐私保护



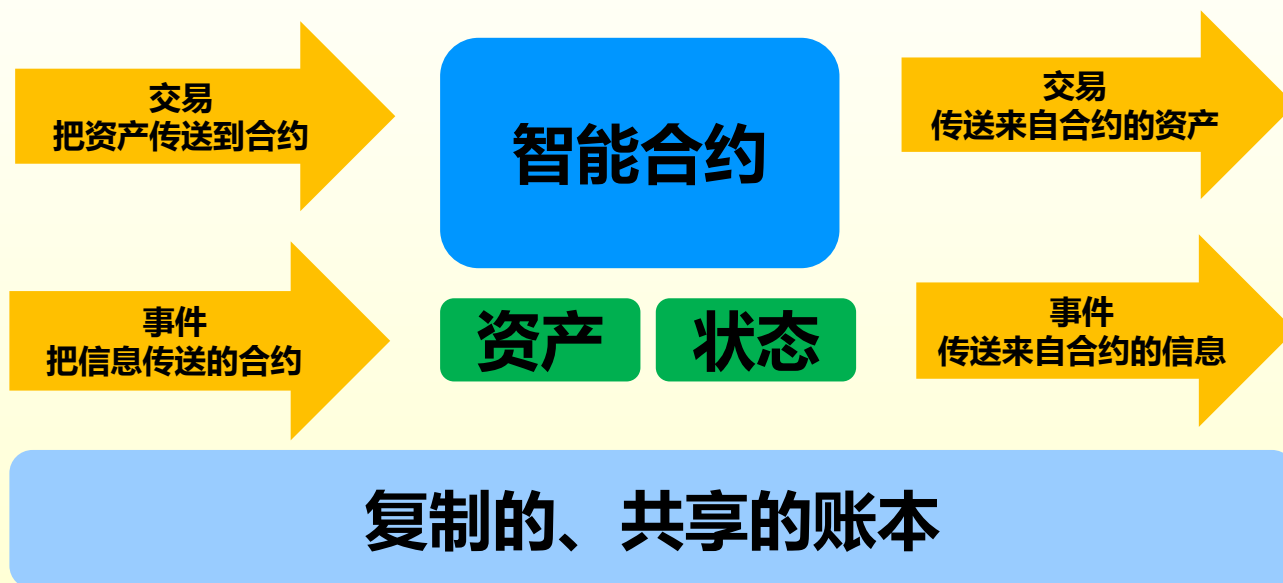
交易向全网公开，但公钥保持为匿名，
可以让每次交易都生产一个新地址。难以被追溯



区块链2.0



区块链2.0

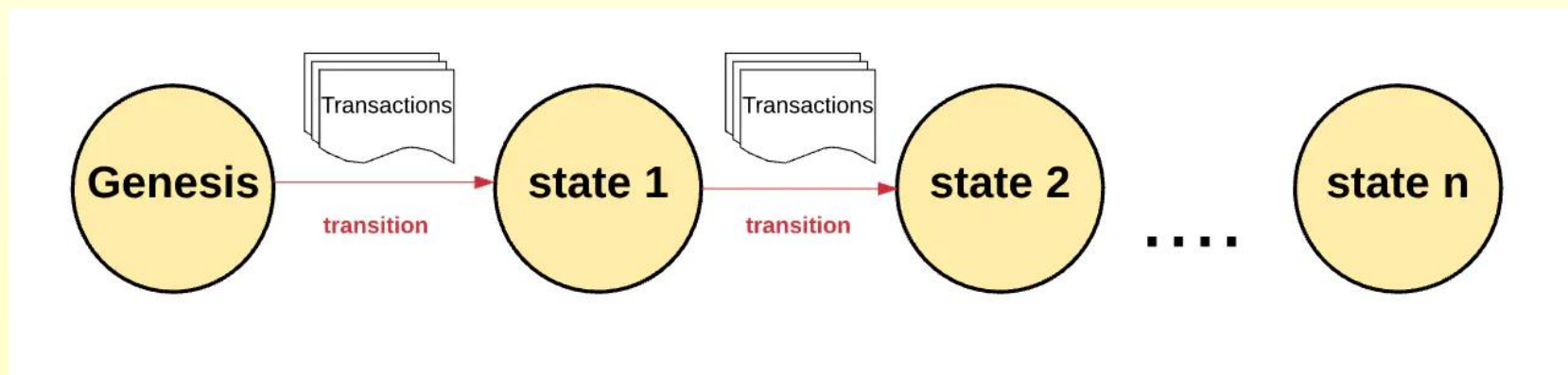


定义： 区块链就是一个具有共享状态的密码性安全交易的单机 (cryptographically secure transactional singleton machine with shared-state)



区块链1.0的问题

- 第一代区块链存在非图灵完备
- 没有保存状态的账户
- Pow挖矿带来的资源和效率问题。
- 更公平的激励问题
- ASIC矿机出现带来的中心化问题
- 共识算法比较单一





2.0的代表以太坊

- 2013年比特币社交群里 Vitalik Buterin 发表以太坊白皮书
- 2014年募集到1843万美元开发资金
- 2015. 7. 30 以太坊正式启用

主要特点:

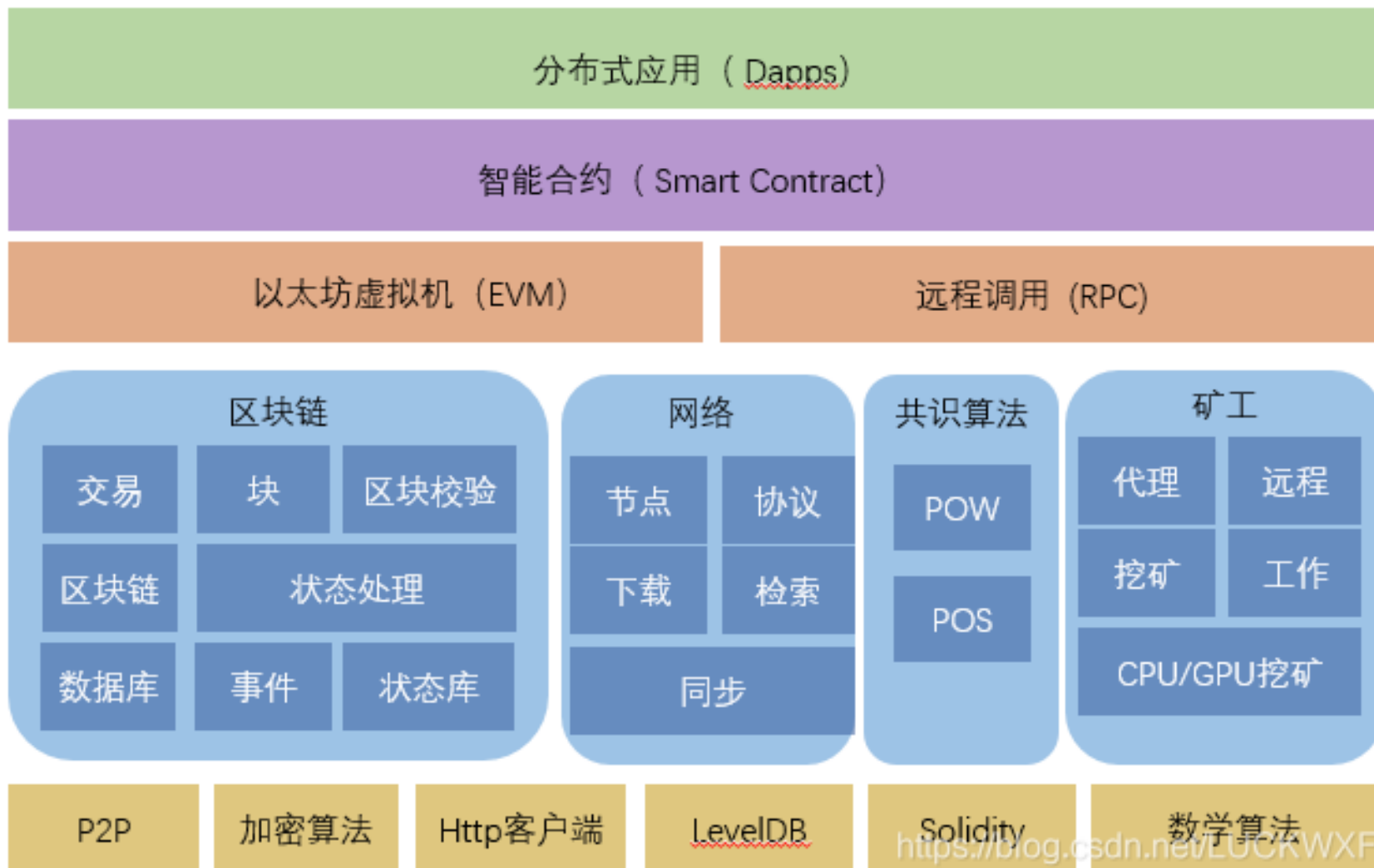
- 最大特色是结合智能合约机制，制定Solidity编程语言
- 使用内存需求较高的哈希函数，避免出现算力矿机
- 叔块激励机制，降低区块产生间隔为15秒
- 难度调整算法，一定的自反馈机制
- GAS机制，限制代码执行指令数，避免循环攻击
- 记录当前状态的哈希树的根哈希值到区块链，实现轻量级客户端
- 设计了虚拟机EVM
- 设计了MPT树



以太坊六层结构



<https://blog.csdn.net/LUCKWXF>



<https://blog.csdn.net/LUCKWXF>



与区块链1.0的不同点

以太坊有更小的区块，比特币为1M

- 以太坊的区块大小没有限制，但是gas有个上限的，最大区块大小大约为1500000Gas，区块大小在22KB，从一个账户到另一个账户交易大约消耗21000GAS，每个区块大概包含70个交易。可调整，25~40TPS。1M的数据全网广播需要1.5s左右。

以太坊的区块时间更短

- 以太坊区块间隔时间大约14秒，比特币是10分钟

以太坊虚拟机上可以运行智能合约

- 图灵完备语言，预先设定规则，交易发生时，作为输入参数到智能合约，按预先设定处理后输出结果。

以太坊有账户

- 以太坊账户有：序号，余额（拥有Wei的数量），Merkle根的哈希值，代码哈希值

以太坊有GAS机制

- 限制交易执行的工作量，EVM执行时，按照规则被消耗，执行结束由交易创建者设置，剩余GAS返回账户。



以太坊把**用户账户**与**智能合约**均称为账户，每个账户有一对密钥，地址取自其公钥的最后20字节。由私钥和地址形成的密钥对均被编码和存放在一个密钥文件中，密钥文件采用JSON文本文件格式

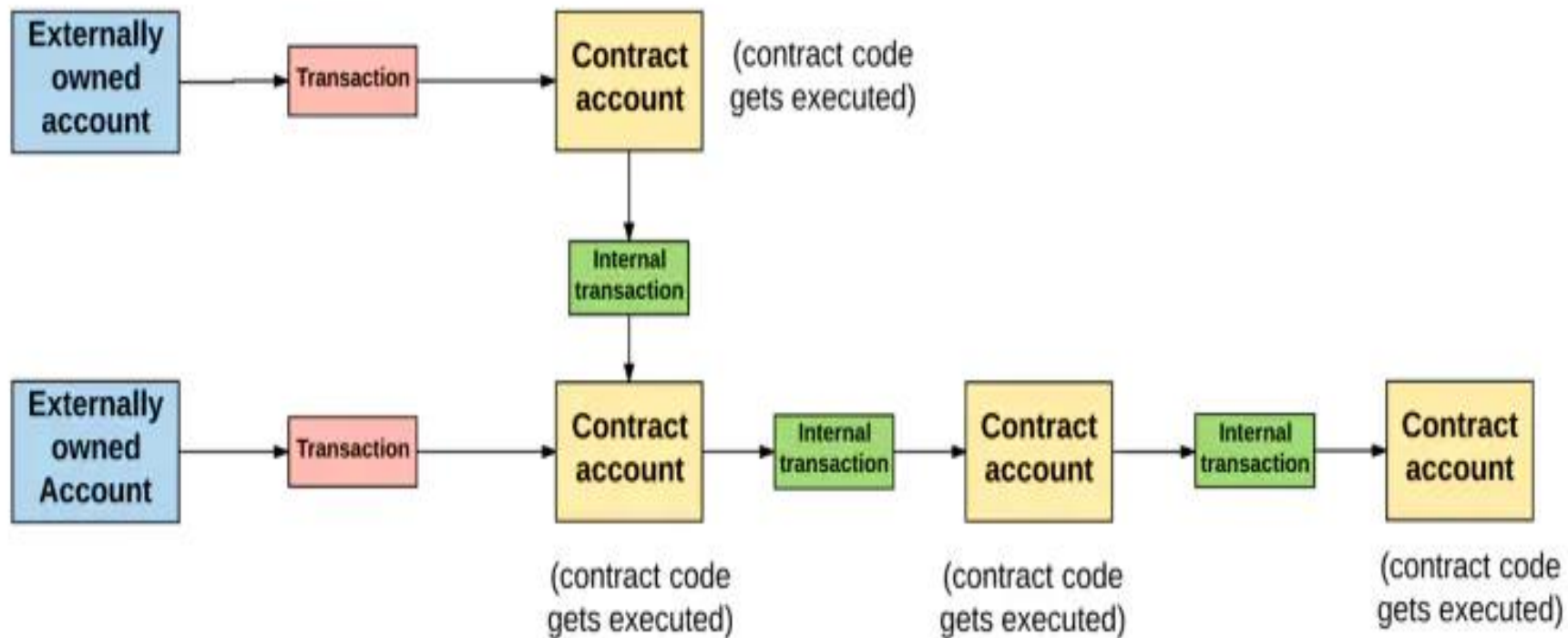
- 外部账号：简称 EOA，是由私钥来控制的；
- 合约帐户：由合约代码来控制，且只能由一个 EOA 账号来操作；

合约账户由其合约代码控制，从一个外部帐户到一个合约账户的消息会**激活**合约账户的代码。

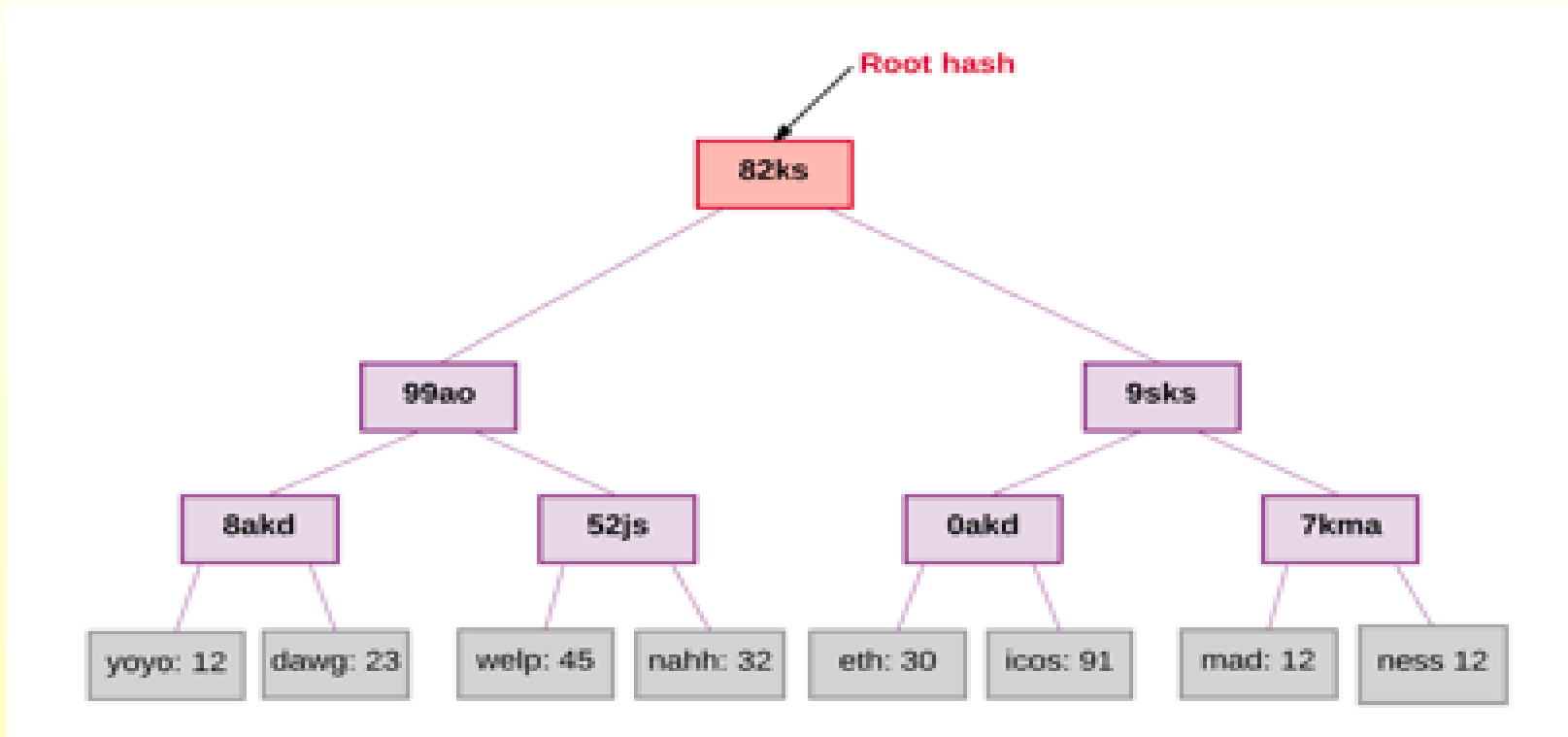


帐户状态:

- nonce: 如果帐户是一个外部帐户, 这个数字代表从帐户地址发送的交易数量。如果帐户是一个合约帐户, nonce是帐户创建的合约数量。
- balance: 这个地址拥有的Wei (以太坊货币单位) 数量, 每个以太币有 $1e+18$ Wei。
- storageRoot : 一个Merkle Patricia树根节点的哈希, 它对帐户的存储内容的哈希值进行编码, 并默认为空。
- codeHash: EVM (以太坊虚拟机) 的哈希值代码。对于合约帐户, 这是一个被哈希后并存储为codeHash的代码。对于外部帐户, codeHash字段是空字符串的哈希。

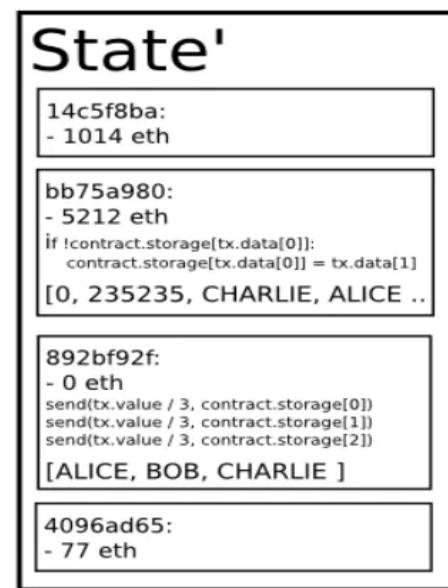
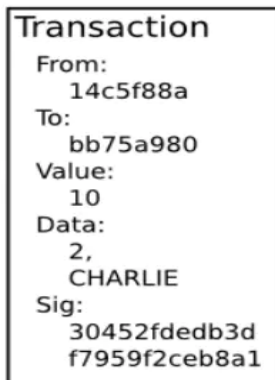
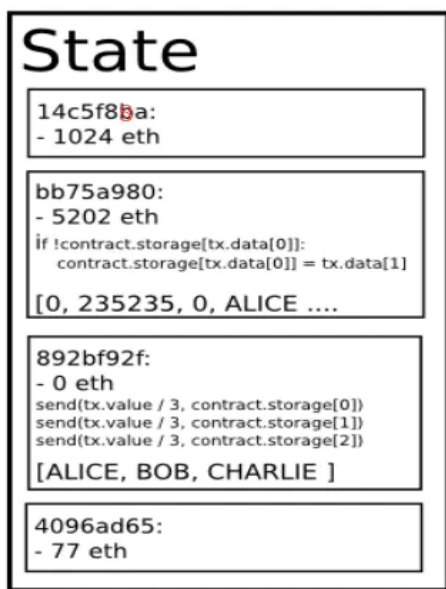
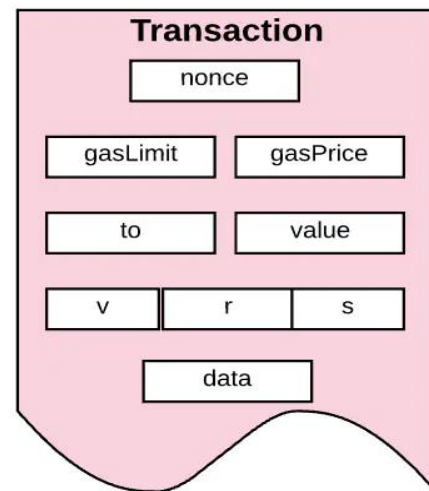


- Merkle Patricia树：树的底部数据是通过把我们想要存储的数据分割成块后而生成的。要求存在里面的值（value）都有一个对应的key。从树的根节点开始，key会告诉你顺着哪个子节点可以获得对应的值，这个值存在叶子节点





- 交易包含以下内容
- 消息的接收者
- 一个可以识别发送者的签名
- 发送方给接收方的以太币的数量
- 一个可选的数据字段
- gasLimit值, 表示执行这个交易允许消耗的最大计算步骤
- gasPrice 值, 表示发送方的每个计算步骤的费用





• 合约有能力向其他合约发生“消息”，消息包含以下内容

- 消息的发送者 (隐式)
- 消息的接收者
- 与消息一起传送的以太币的数量
- 一个可选的数据字段
- 一个 gasLimit 值

由一个合约产生的而不是一个外部用户。一个正在执行代码的合约，当执行到 CALL 代码时，会产生并执行一个消息。就像一个交易，一个消息会导致接收方的账户运行它的代码。因此，合约之间是可以互相发生作用的。

GAS费用

在以太坊网络上进行的每一笔交易都会产生费用，Gas不仅用于支付计算的费用，还用于支付存储的使用费用。

收费的目的就是使整个网络不会因用户的不当使用而变得负担过重，以太坊是一种图灵完备语言。这就允许了循环，如果没有费用，意图不良的人可以通过在交易中执行一个无限循环来扰乱网络。

每次交易，发送方都要设置一个Gas Limit和Gas Price。Gas Limit和Gas Price代表发送方愿意为执行交易支付的最大金额。

例如，发送方将Gas Limit设置为50,000，一个Gas Price设置为20 gwei。这意味着发送者愿意花费最多 $50,000 \times 20 \text{ gwei}$ ，也就是：
1,000,000,000,000,000 Wei (0.001以太币) 来执行这一交易

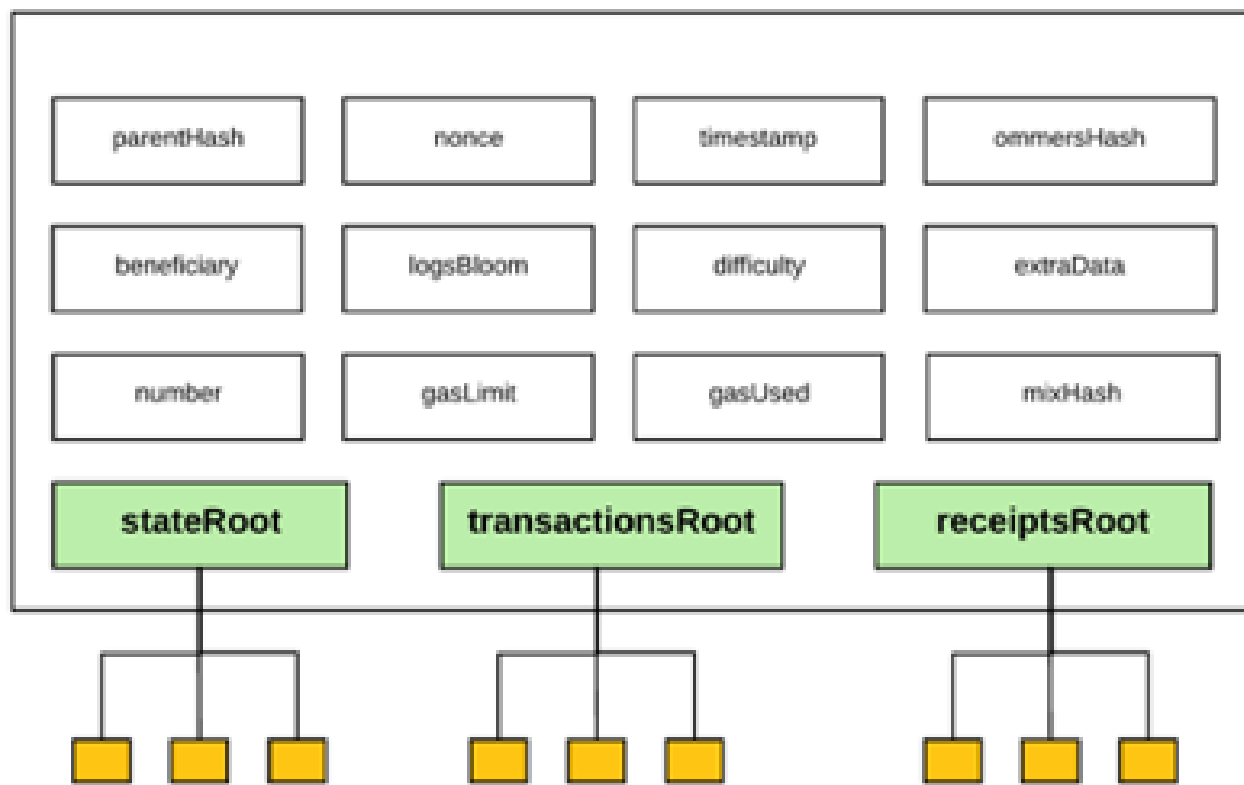


- Block结构体主要分为Header和Body，Header相对轻量，涵盖了Block的所有属性，包括特征标示，前向指针，和内部数据集的验证哈希值等；Body相对重量，持有内部数据集。都以[k, v]形式单独存储在底层数据库中。
- Blockchain管理Block组成的一个单向链表，HeaderChain管理Header组成的单向链表，并且Blockchain持有HeaderChain。
- Merkle-PatriciaTrie (MPT) 数据结构用来组织管理[k, v]型数据，它设计了灵活多变的节点体系和编码格式。
- StateDB作为本地存储模块，它面向业务模型，又连接底层数据库，利用两级缓存机制来存储和更新所有代表“账户”的stateObject对象。
- stateObject除了管理着账户余额等信息之外，也用了类似的两级缓存机制来存储和更新所有的State数据。



区块链头：存储三个不同Merkle树结构根节点的哈希，状态数，交易树和收据树。

Block header





- **parentHash**: 父区块头的Hash值（这也是使得区块变成区块链的原因）
- **ommerHash**: 当前区块ommers列表的Hash值
- **beneficiary**: 接收挖此区块费用的账户地址
- **stateRoot**: 状态树根节点的Hash值（回忆一下我们之前所说的保存在头中的状态树以及它使得轻客户端认证任何关于状态的事情都变得非常简单）
- **transactionsRoot**: 包含此区块所列的所有交易的树的根节点Hash值
- **receiptsRoot**: 包含此区块所列的所有交易收据的树的根节点Hash值
- **logsBloom**: 由日志信息组成的一个Bloom过滤器（数据结构）
- **difficulty**: 此区块的难度级别
- **number**: 当前区块的计数（创世纪块的区块序号为0，对于每个后续区块，区块序号都增加1）
- **gasLimit**: 每个区块的当前gas limit
- **gasUsed**: 此区块中交易所用的总gas量
- **timestamp**: 此区块成立时的unix的时间戳
- **extraData**: 与此区块相关的附加数据
- **mixHash**: 一个Hash值，当与nonce组合时，证明此区块已经执行了足够的计算
- **nonce**: 一个Hash值，当与mixHash组合时，证明此区块已经执行了足够的计算



以太坊目前的共识机制称为Ethash算法，提高内存级别难度，缓解用ASIC挖矿中心化的问题。算法：

$$(m, n) = \text{PoW}(H_n, H_n, d)$$

m代表的是mixHash，n代表的是nonce，

H_n 代表的是新区块的头（不包含需要计算的nonce和mixHash）

H_n 是区块头的nonce，

d是DAG，就是一个大数据集。

矿工可以随机抽取数据集中的部分并将它们放入一个数学函数中Hash出一个“mixHash”。矿工会重复生成mixHash直到输出的值小于想要的目标值nonce，挖矿成功。



- 每一个区块中，通过区块头来生成一个seed，这个seed称为种子，种子只和当前的区块有关。
- 种子产生一个16MB的伪随机缓存。
- 基于上述的缓存，那么再生成一个1GB的DAG数据集。
- 矿工所做的事情，就是在数据集DAG中随机去选择元素，且对其进行散列的一个过程。
- 缓存和数据集DAG每增加30000个区块更新一次
- 轻节点可以仅基于缓存来校验一个交易的有效性
- 使用这个缓存，节点可以生成DAG“数据集，全节点需要保存



一个矿工挖出一个区块的时候会获得奖励，包括：

- 为“获胜”区块提供的5 以太币静态区块奖励
- 区块中的交易在区块内所消耗的gas
- 纳入ommers作为区块的一部分的额外奖励

Ommers为了帮助奖励矿工纳入这些孤区块。必须在父区块的第6个子区块之内或更小范围内。



区块链3.0：可编程经济

区块链+DAO+人工智能+法律？

- 可扩展性：性能，规模化应用，高并发能力
- 互通性：跨网、跨链、协议互通
- 整体应用性：用户体验
- 技术进步：快速交易确认时间，低手续费，开发引擎，合约规模化产生，高安全性，大数据、大文件存储，分片、租赁、侧链，社区开发、群智和博弈、代码即法律等。



北京航空航天大学

Beijing University of Aeronautics and Astronautics

智能合约



Gartner 2015年10月9号的报告:

- Gartner 认为, **2016 年是数字化的时代元年。**
- 算法驱动的交易已经参与到我们的经济中, 在我们的企业、法律、经济和信托的规则下。

新的自动交易会自己定义规则(**智能合约**), 并以此作为**新经济范式**的基础。Gartner 把它称之为 **Programmable Economy**, 这会给现存经济带来巨大的变革。算法会以透明和开源代码的方式, 在区块链 (BlockChain) 中**自由设定**, 并且在银行、保险、市场、交易以及各种类型的金融工具中使用。

区块链: 去中心化, 人与机器的关系, 全球信用的基础协议



1994年左右，几乎与互联网（world wide web）同时出现，密码学家尼克萨博（Nick Szabo），提出了“智能合约”

“The Idea of Smart Contracts”

- 机器通过物理的密封自行控制财产，所以可以执行“合约”条款
- Szabo指出计算机代码可以代替机械设备，进行更复杂的数字财产交易
- 1998年，尼克萨博就设计出了一种叫“比特黄金”（bit gold）的去中心化的数字货币机制



- ✓ 电子数据交换(EDI)，
- ✓ 证券期权等合成型资产(synthetic assets)
- ✓ 智能资产：“智能财产可能以将智能合约内置到物理实体的方式，被创造出来”

定义1：

“智能合约就是执行合约条款的可计算交易协议”



Smart contracts are computer **protocols** that facilitate, verify, or enforce the negotiation or performance of a contract, or that obviate the need for a contractual clause. Smart contracts usually also have a user interface and often emulate the logic of contractual clauses. Proponents of smart contracts claim that many kinds of contractual clauses may thus be made partially or **fully self-executing, self-enforcing**, or both. Smart contracts aim to provide **security** superior to traditional **contract law** and to reduce other **transaction costs** associated with contracting.



- Alice从Bob处以每股50美元的价格，购买Bob在Acme Inc公司的100份股票

```
contract Option {  
    strikePrice = $50  
    holder = Alice  
    seller = Bob  
    asset = 100 shares of Acme Inc.  
    expiryDate = June 1st, 2016  
    function exercise ( ) {  
        If Message Sender = holder, and  
        If Current Date < expiryDate, then  
            holder send($5,000) to seller, and  
            seller send(asset) to holder  
    }  
}
```



- 发展缓慢，合约方无法直接观察与验证其他合约方的执行动作，只有让第三方审核各方合约执行的记录。

智能合约怎样控制实物资产保证有效地执行合约？

售货机通过将商品保存在内部控制财产所有权，可是计算机程序要怎么控制现实世界的现金、股份等资产呢？

计算机怎样执行这些条款以获得合约方的信任呢？

合约方应该不需要认可合约代码、以及解释和执行代码的计算机。不需要合约方亲自检查有问题的计算机。

区块链为完全数字化资产的记录和转移奠定了基础



结合带来的好处

智能合约通过协议与用户接口来促进合约的执行，

- 它不仅比传统“纸质”合约具有更大功能、生命力更强，
- 它还减少了交易在合约制定、控制协议和执行效能保障的人工花费与计算成本。
- 它还是形成“数字社会”主力军，是我们在互联网中形成安全、数字化关系的关键，使得我们可以在不信任的环境中保持协作。



- **场景1: 汽车交易**

交易和贷款行为将嵌入程序以取代复杂流程和过程。如果贷款者不还款，智能合约将自动收回发动汽车的数字钥匙

- **场景2: 出租房屋**

所有的门锁都是连接互联网的。当你为租房进行了一笔交易时，达成的智能合约将自动地为你打开房门。你只需要存储在智能手机中的钥匙就能进入房屋。



- **合法性**: 代码符合法律规制, 所控资产拥有所有权, 且合法有效;
- **公信性**: 合约代码产生机制必须具有公信、权威性, 结果可验证;
- **证据性**: 过程数据和场景必须被安全地存储, 可被用于法律证据;
- **一致性**: 智能合约应与文本一致, 经过专业人士制定审核;
- **智能性**: 能准确反映智力共识和智能执行;
- **可信性**: 静态产生和动态执行过程必须具有正确、安全、可靠和可监管;
- **可观察性**: 合约方能够通过用户界面去观察关于合约的所有状态
- **可验证性**: 合约方执行合约的过程是可验证;
- **自强制性**: 对于违反合约行为的制裁必须是强制性的
- **接入控制**: 合约相关的知识、控制、执行都应该作为资产保护起来, 只有发生争执的时候, 才把内容提供给第三方检验。

智能合约具有多面性: 角色性, 程序性, 智能性, 资产性

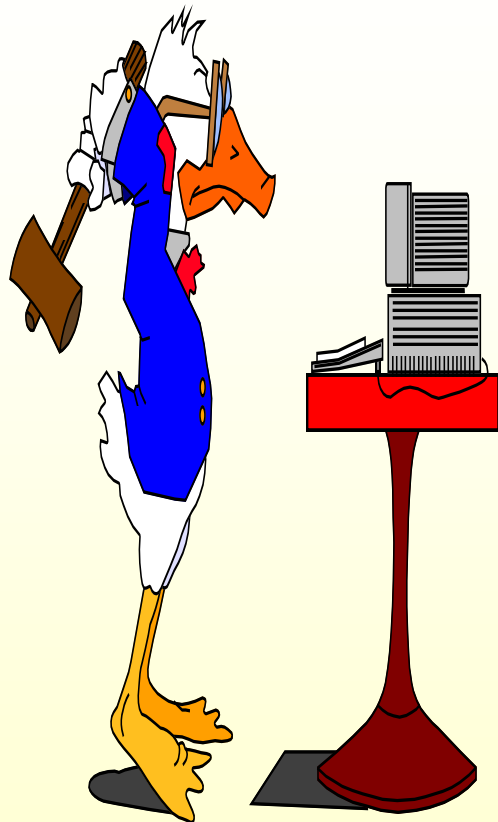


智能合约的编写

- 把这段代码的哈希存在区块链中，代码存在key-value数据库中，（key:code hax,value:code）



智能合约的挑战



- 2016年6月17日发生的DAO攻击事件，由于DAO智能合约自身的漏洞，导致6000万美元从账号中流失。
- 智能合约的核心是算法合同（Algorithmic Contract），
- 智能合约作为“代码即合约”
- 智能合约将是需要大规模生产
- 智能合约是应用服务性的。



案例1 The DAO事件

回顾： 2016年6月17日，运行于以太坊公有链中的The DAO项目中的智能合约遭受黑客攻击，此攻击被盗高达300多万以太币，并直接导致了以太坊的硬分叉。

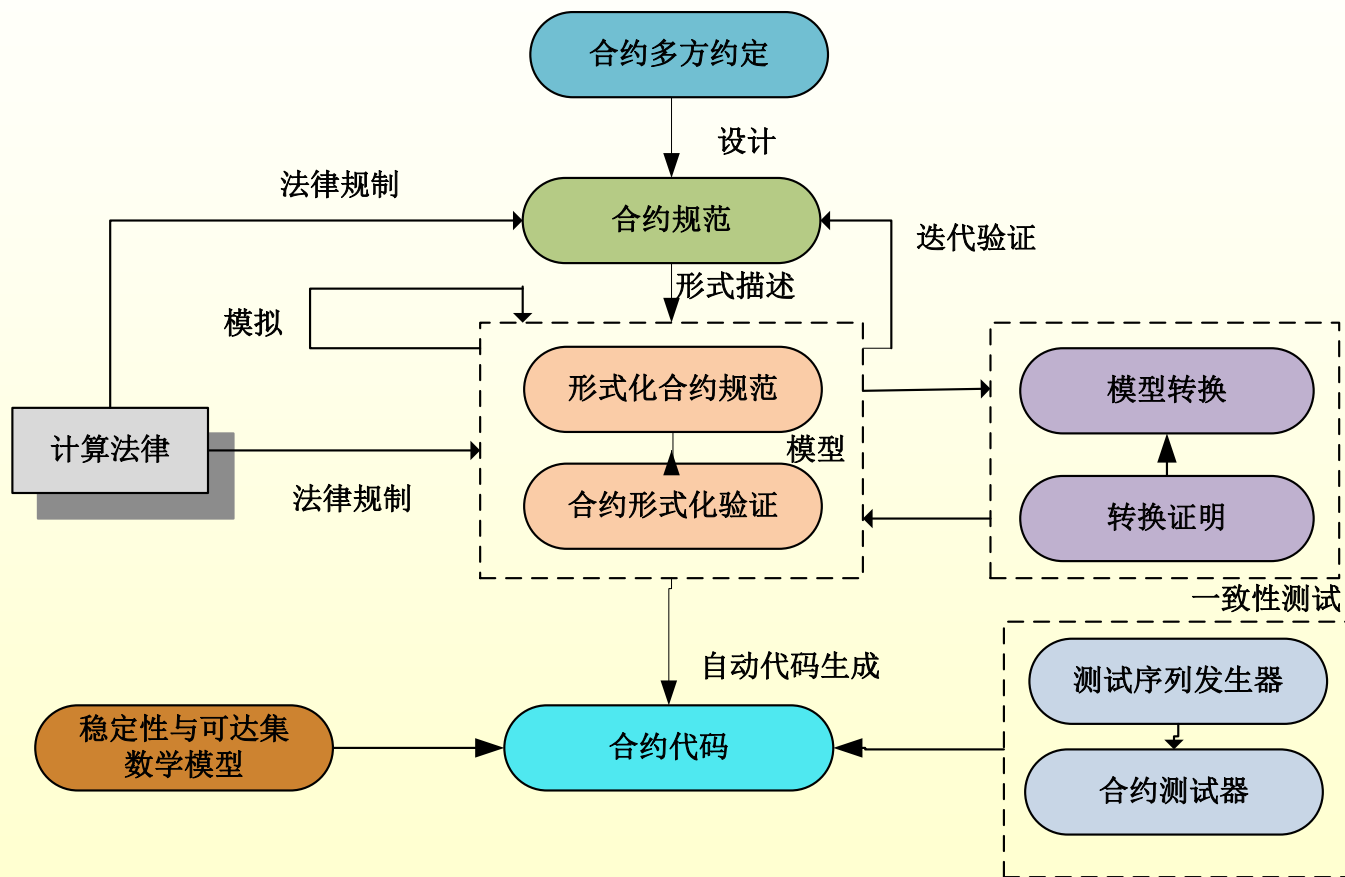
分析： 代码的主要问题出现在withdrawRewardFor方法中，该方法中的某个判断语句逻辑不严谨，导致攻击者可以通过一个递归函数将以太币转移到自己的账户中。

```
1 function withdrawRewardFor(address _account) noEther internal returns (bool success) {  
2     if ((balanceOf(_account) * rewardAccount.accumulatedInput()) / totalSupply < paidOut[_account])  
3         throw;  
4  
5     uint reward =  
6         (balanceOf(_account) * rewardAccount.accumulatedInput()) / totalSupply - paidOut[_account];  
7     if (!rewardAccount.payOut(_account, reward))  
8         throw;  
9     paidOut[_account] += reward;  
10    return true;  
11 }
```

The DAO事件问题代码



智能合约工程 (Smart Contract Engineering)





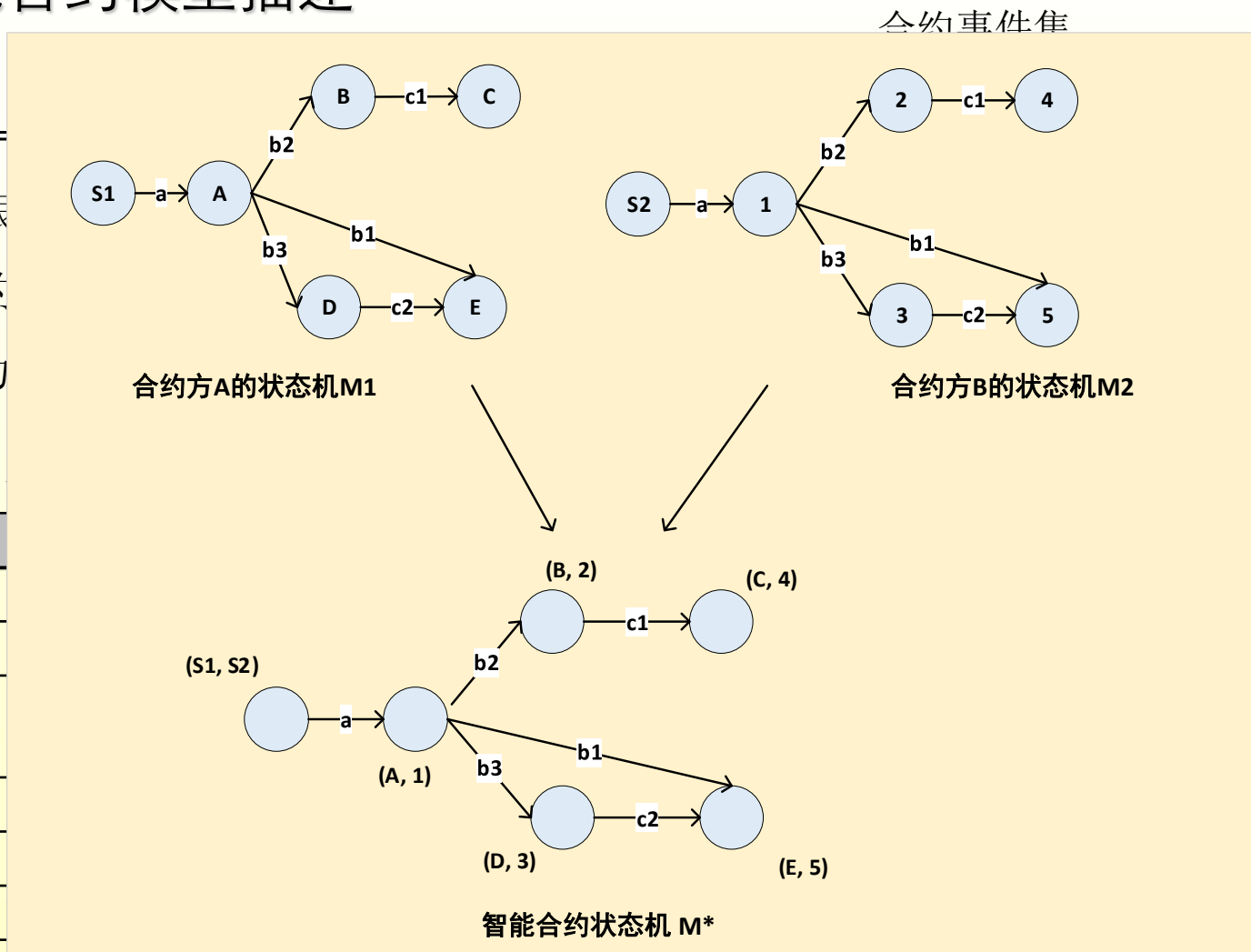
智能合约模型描述

■ $M^* =$

有限

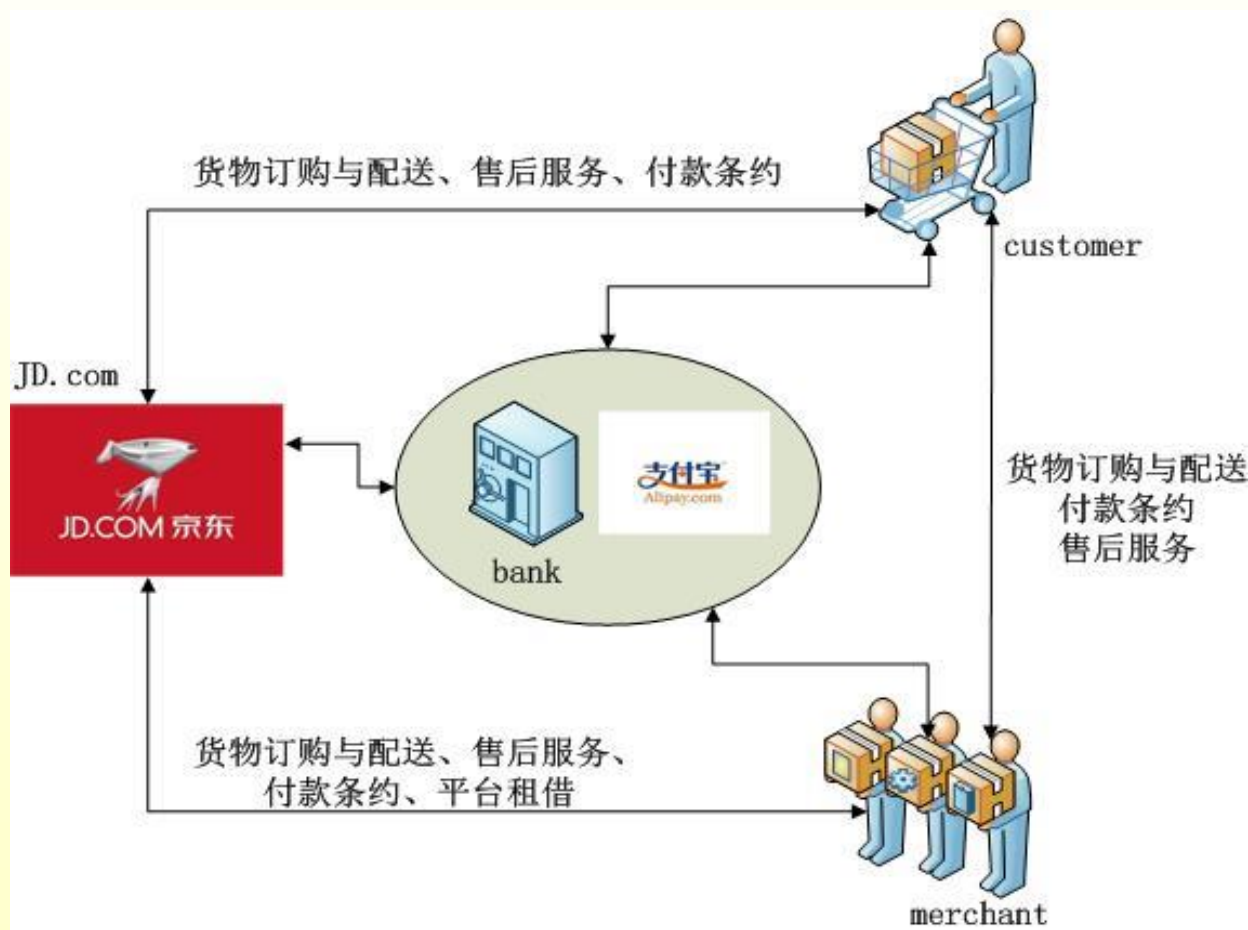
合约

s^* 为



状态
s_1
A
B
C
D
E

• 网络购物平台例子





- 货物订购与配送合约:
- Input: 货物订购交易信息 (货物信息goods、支付方式pay、配送方式distribution)
- 合约双方: A (the goods seller)、B (the goods buyer)
- 合约内容
- Begin
- IF goods are available AND pay is completed AND distribution is available
- Inform A to send the goods to B, set the terminator=timestamp + one week
- ELSE
- The transaction failed and return the money to B
- Wait one week for the acknowledgment message from customer
- IF acknowledgment message received
- Pay to merchant and quit
- IF TIMEOUT
- The transaction failed and return the money to customer
- END



- Input: 货物成交信息 (货物信息item、支付方式pay、售后截止日期terminator)
- 合约双方: A (the goods seller)、B (the goods buyer)
- 合约内容:
- Begin
- While(timestamp < terminator) {
- Wait for the feedback message from B
- IF A received message from B
- SWITCH(message) {
- Case goods return message:
- IF the timestamp of the message is before terminator
- A wait for the goods
- IF A received the goods
- A return the money back to B
- ELSE
- A reject and quit
- Case goods exchange message:
- IF the timestamp of the message is before terminator
- A wait for the goods
- IF A received the goods
- A send the new goods to B
- Other:
- NULL
- }
- }
- Quit
- End



平台租借合约

- 平台租借合约
- Input: 平台租借信息（租借信息message、支付方式pay）
- 合约双方: A（京东）、B（merchant）
- 合约内容:
- Begin
- IF message is confirmed and pay is completed
- Open the rights accessed to the platform to B and quit
- ELSE
- Reject and quit
- End



```
byte money=100
byte user_money=0
byte shop_money=0
byte day=0
bool isSend=false
lfl { <>isSend }
active proctype user() {
    do
        :: isSend -> atomic{
            shop_money=100;
            money=0;
            break;}

        :: (day>6)->{
            user_money=100;
            money=0;
            break;

        }
        :: else -> day=day+1;
    od
}
active proctype shop() {
    do
        ::(day<7)->isSend=true;
        ::break;
    od
}
init{
    atomic{run user();run shop();
}
```



```
jSpin Version 5.0
File Edit Spin Convert Options Settings Output SpinSpider Help
Open Check Random Interactive Guided Weak fairness Safety Verify Stop Translate Load LTL name SpinSpider Maximize

-ssc.pml-
1 byte money=100
2 byte user_money=0
3 byte shop_money=0
4 byte day=0
5 bool isSend=false
6
7 ltl { <>isSend }
8
9 active proctype user() {
10 do
11 :: isSend -> atomic{
12   shop_money=100;
13   money=0;
14   break;}
15 :: (day>6)->{
16   user_money=100;
17   money=0;
18   break;
19 }
20 :: else -> day=day+1;
21 od
22 }
23
24 active proctype shop() {
25 do
26 :: (day<7)->isSend=true;
27 :: break;
28 od
29 }
30
31 init{
32   atomic{
33     run user();
34     run shop();
35   }
36 }
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629

```



Thanks !

