十年架构 成长之路

SACC 第十届中国系统架构师大会

SYSTEM ARCHITECT CONFERENCE CHINA 2018

2018年10月17-10月21日　北京海淀永泰福朋喜来登酒店

IT168.com　ChinaUnix.net　ITPUB

Generality Hardness

根本上抵抗 ASIC 的挖矿原理

张翰 – 初链基金会

区块链行业所处的历史阶段

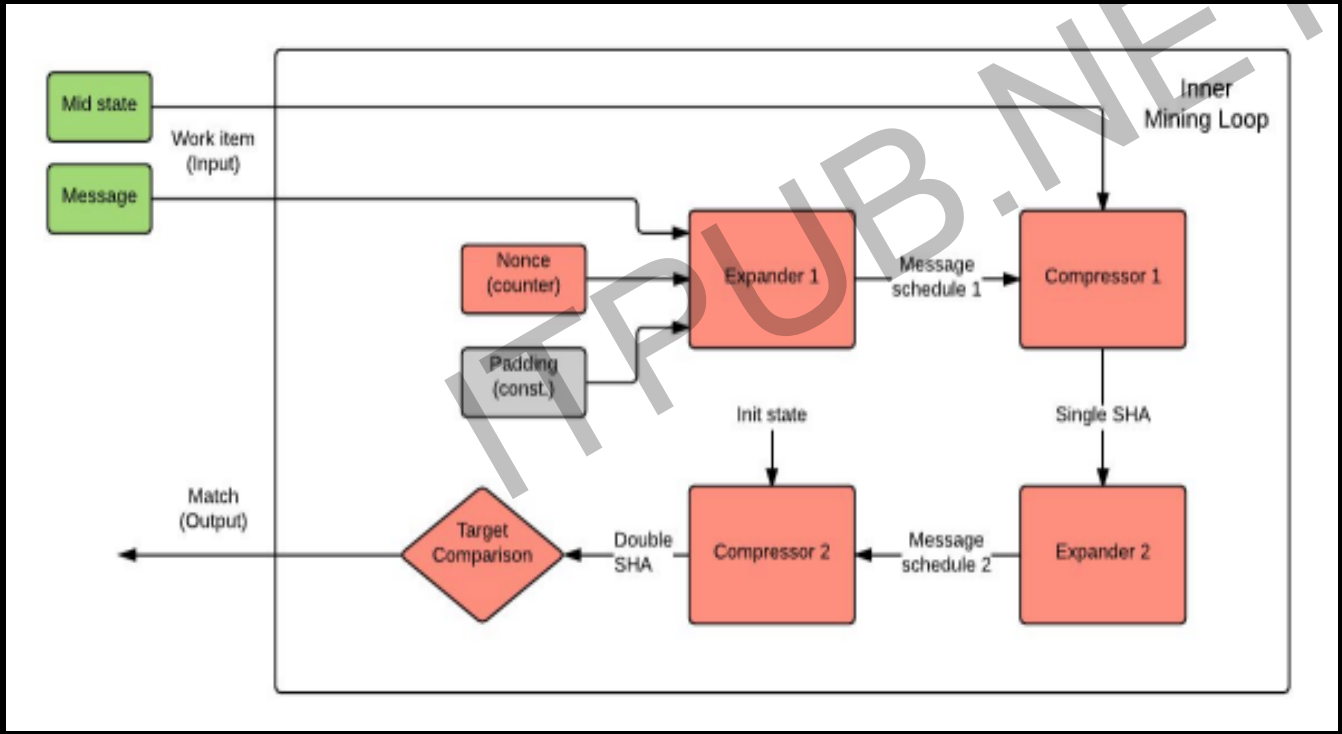
- 第一代公链
  - 比特币、莱特币、达世币等等
  - 支付问题，去中心化，价值互联网
  - 问题：没有实体经济
- 第二代公链
  - 以太坊 -> 去中心化应用
  - 性能低 ->无法落地
  - 不可能三角：安全、性能、去中心化不可兼顾
- 第三代公链
  - 混合共识，解决不可能三角问题
  - PBFT + PoW：Permissioned -> Permissionless
  - 需要对两者进行创新性修改

PoW 早期历史

- PoW 早期历史
  - PoW 算法最初是由 Cynthia Dwork 与 Moni Naor 于 1993 年提出来的。
  - 当时主要的应用场景是反 ddos 攻击和反 email spam。
  - 1999 年，Jacobsson 发明了 partial hash inversion 算法，中本聪在比特币中也选用的是这个方案。
- 中本聪用 PoW 解决的问题
  - 如何在节点之间互无信任的基础上实现去中心化记账
  - 记账权如何在参与节点中分配？例如，用均值随机数分配会遭到女巫攻击
  - 每次争夺记账权需要消耗算力资源 – Sha256d

PoW: Partial hash inversion

- 问题：Sha 256d 运算结构过于简单，BTC 算力被 ASIC 垄断



PoW 的困局

- 2008 年，中本聪创造了比特币网
  - 创造一个任何人都能自由参与的去中心化金融体系。
- 2012 年 ASIC 出现，算力空前集中在少数矿池手里
  - 从而降低了网络的去中心化。
  - 此后出现的 PoW 算法，ASIC resistant 一直是核心考量重点。
  - 例如：Scrypt, x11, Cryptonight, Ethash, Equihash
- 这些算法：
  - 增加了设计 ASIC 的难度，
  - 或者降低 ASIC 与 GPU 的性价比，
  - 但没有实现根本上的 ASIC resistant。

PoW 的困局

- 在足够经济利益的驱使下，ASIC 厂商将它们一一攻破。

发行年	项目	Ticker	创始人	挖矿算法	ASIC
2009	比特币	BTC	中本聪	SHA 256d	Antminer S
2011	莱特币	LTC	Charlie Lee	Scrypt	Antminer L
2013	狗狗币	DOGE	Palmer, Markus	Scrypt	Antminer L
2013	Siacoin	SC	Vorick, Champine	Blake2b	Antminer A
2014	达世币	DASH	Duffield, Hagan	X11	Antminer D
2014	门罗币	XMR	Monero Core Team	CryptoNight	Antminer X***
2014	Verge	XVG	Sunerok	Scrypt, groestl, x17, blake2s, lyra2rev2	Antminer A
2015	以太坊	ETH	Vitalik Buterin	Ethash	Antminer E
2016	Zcash	ZEC	Zooko Wilcox O'Hearn	Equihash	Antminer Z

PoW 的困局

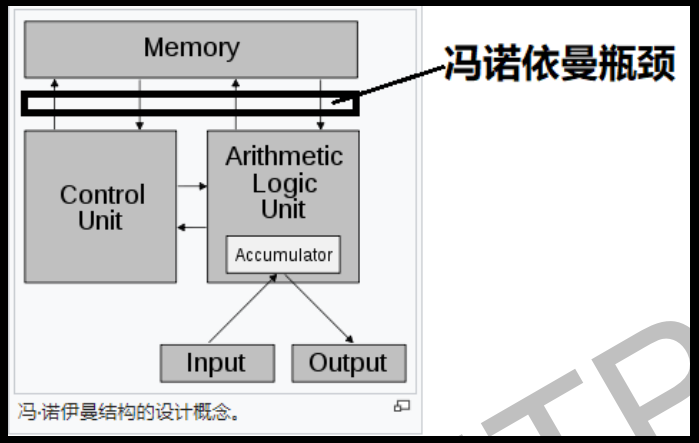
- 战术层面
  - ASIC 厂商完胜项目方
  - 主流项目所有 ASIC resistant 尝试都以失败而告终
- 战略层面
  - 以 Casper，EOS，ADA，NEO 为首的项目，抛弃 PoW 转而使用 PoS 或 DPoS。
- 后果
  - PoW 面临灭顶之灾
  - PoW 应该被历史淘汰么？

PoW 的困局

- PoW 不该被淘汰的几个理由
  - PoS / DPoS 的项目尚不成熟，PoW 项目则有 10 年运行历史
  - PoS / DPoS 项目方话语权过强，PoW 更包容、更去中心化
  - PoS / DPoS 中心化交易所拿客户的币挖矿、投票
- PoW 的自身问题
  - ASIC 矿池使 PoW 去中心化效果变弱
  - PoW 节能环保问题：哈希计算器无用论
  - 交易速度问题

ASIC 为什么高效？

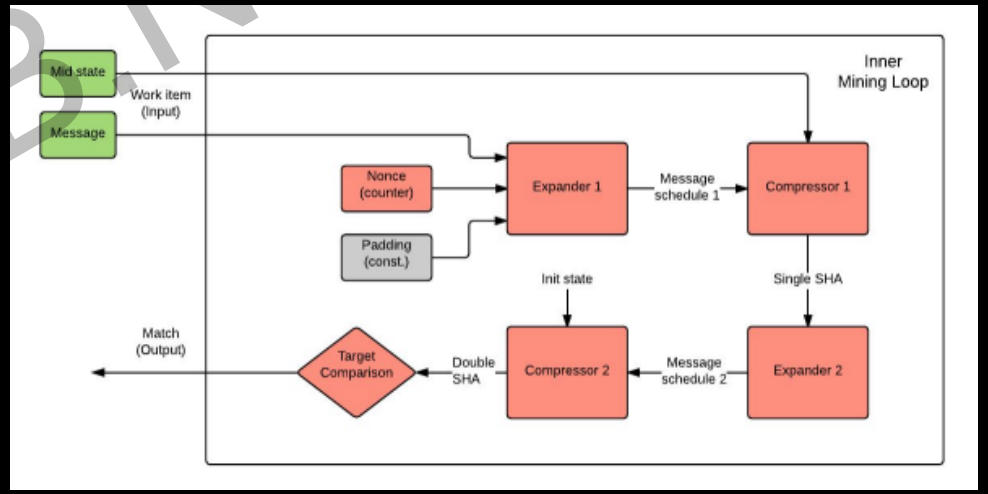
- CPU 设计需要满足多样性计算，采用的是冯诺依曼架构。



- 冯诺依曼架构包括：内存，计算单元，控制单元
  - 数据、程序储存在内存中，执行计算时向计算单元传输
  - 传输中产生的瓶颈，叫“冯诺依曼瓶颈”

ASIC 为什么高效？

- Sha 256d 算法过于单一
  - 通过循环一个 nonce 变量，不停计算哈希值
  - 等哈希小于预设难度系数，就算挖矿成功
  - 输入：block header
  - 算法：固定、单一
- 加速方案
  - 摒弃冯诺依曼架构
  - 将 Sha 256d 直接刻录在计算单元
  - 规避冯诺依曼瓶颈



抗 ASIC 的尝试

- 设计比 Sha 256d 更复杂的哈希
  - 多种算法：X11，XVG 算法
  - 读表：Ethash，Equihash
  - 只能增加 ASIC 难度，不能根本上抵抗 ASIC

Generality hard 原理

- CPU / GPU 为什么慢？
  - 通用性 -> 冯诺依曼架构 -> 冯诺依曼瓶颈
- 抵抗 ASIC 算法设计
  - 必须绕不过冯诺依曼瓶颈 -> 必须对通用性有要求
- 门罗案例
  - 每 6 个月社区投票更改算法
  - 成功抵抗 Antminer X
- 缺点
  - 投票过程项目方话语权太大
  - 如何证明没有提前私自准备 ASIC

Generality hard 原理

- Generality Hardness
  - 令 S 为一个算法集合，
  - 欲实现 S 中的所有算法，必须经过冯诺依曼瓶颈。
  - 每 T 个区块切换一次算法。
- 切换算法要求
  - 切换方式必须满足，可验证且不可预测。
  - 算法切换没有人为干预。

如何改算法：抽象原理

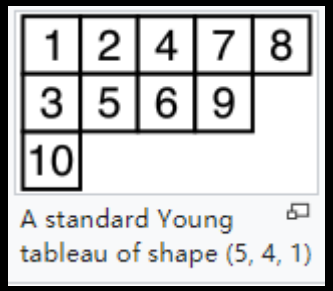
- 令 G 为一个群，对每个群元素 g，令 rho_V(G) 为 G在向量空间 V 上的表示。
- 在普通挖矿算法中，将 blockheade, nonce等信息经过 padding 等运算之后，会形成一个向量 v(nonce)
- 通过穷举不同的nonce 值，来寻找 hash(v(nonce)) 小于难度系数的结果。
- 我们将 hash(v(nonce)) 改为 hash(rho(g)*v(nonce))
- 只要 G 足够复杂（Truehash 用的是置换群 S_2048，该群有 2048! 个元素），这个算法集合就不可能全部写死在计算单元内。
- 由于算法会随机切换，冯诺依曼瓶颈将不可避免。

如何改算法：生成规则

- 每 12000 个 PoW区块换一次群元素，12000 个 PoW 区块大概需要 83 天的时间生成。
- 新的群元素信息由上个周期的第 1 – 8192 个区块所组成。
- 形状参数通过分析第 11001 - 11256 个区块的哈希值所产生。
- 由于区块的哈希值不可提前预知，在第 11256 个区块出现之前，任何人都不可能知道关于新算法的任何信息。
- 从上周期的第 11257 个区块，到该算法作废，总共只有 88 天的时间，这么短的时间内生产 ASIC 没有任何意义。
- 生成方式：区块哈希数据 -> Young tableaux -> RSK Correspondence

如何改算法：Young Tableaux

- Young tableaux 规则

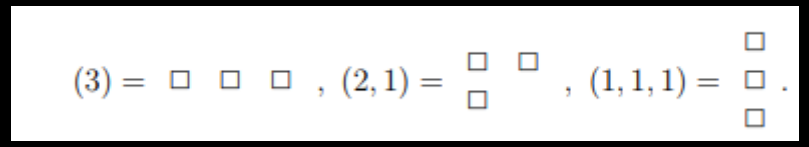


A standard Young tableau of shape (5, 4, 1)

- 每行向右，数字递减；每列向下，数字递减。
- 形状参数 lambda = (5,4,1), |lambda| = 10。
- Young tableaux 的复杂度：1, 1, 2, 4, 10, 26, 76, 232, 764, 2620, 9496

如何改算法：Young Tableaux

- 举例：S_3 群（三元素置换群）



- (3) – 幺矩阵（Trivial representation）
- (2,1) - 自然表示 (Natural representation / 置换矩阵)
- (1,1,1) - 奇偶置换表示（Sign representation）

如何改算法：RSK Correspondence

- 两个同样形状的 Young tableaux

$$\begin{array}{cccccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} \\ 2 & 4 & 3 & 6 & 5 & 1 \end{array} \xrightarrow{\text{R–S}} \left(\begin{array}{ccc} 1 & 3 & 5 \\ 2 & 6 & \\ 4 & & \end{array} \;,\; \begin{array}{ccc} \mathbf{1} & \mathbf{2} & \mathbf{4} \\ \mathbf{3} & \mathbf{5} & \\ \mathbf{6} & & \end{array} \right)$$

可得出唯一一个置换群元素。

rho(g) =

0	1	0	0	0	0
0	0	0	1	0	0
0	0	1	0	0	0
0	0	0	0	0	1
0	0	0	0	1	0
1	0	0	0	0	0

- 挖矿哈希：hash(rho(g)*v(nonce))

TRUE

初链开发者平台

长按识别二维码
关注初链开发者平台，竞赛答题赢奖励

初链开发者平台