Bitcoin 挖矿与分叉的原理

吴嘉婧 副教授

中山大学 计算机学院

Outline of this Class

- Part 1: Math behind Mining
- Part 2: Forking —— 分叉
- Part 3: 比特币安全机制的保障

课前，（去年的学生问的）几个问题的答疑

- “老师，我有一个问题：比特币中的密码学原理主要就是哈希和数字签名，
  - 我的理解是哈希是用于矿工们挖矿的过程的，
  - 然后数字签名是为了去中心化账户管理，给用户自己非对称加密来验证身份的，
  - 这样的理解正确吗？”

- 解答：
  - 安全的数字签名机制可以防止恶意攻击者伪造比特币拥有者的签名
  - 防止窃取别人的比特币

课前，几个问题的答疑

- “老师我没有搞懂区块发布的意思”
- 解答：
  - 发布意思就是 一个矿工获得记账权的时候，他就有权利向整个网络广播它所打包生成的区块了
  - 别的节点收到这个新 block 之后，验证一下合法性之后，就要添加到区块链的尾部了

在比特币领域，我们选择奖励给矿工发布区块的权利。矿工可以通过哈希函数来运行他们准备发布的区块，并且输出给定范围内的数字。接着矿工会把格式数据用于一个有序的、格式化的方式中，并开辟一块地方用于存储少量无用的垃圾数据（称为“nonce”）。这如同我们举办一个比赛，我们告诉玩家添加一个或两个随机文章的文本并运行它，通过哈希函数来找到与输出相匹配的具体数字。

该截图展示误人子弟的错误网络知识

课前，几个问题的答疑

- “我不太理解 block 代表的意义， 发布一个新的block是指发布一个新的矿区吗？ 是指拥有比特币的矿工进行比特币交易的记录集合吗？”
- 解答:
  - 一个新区块就是一堆交易的集合
  - 所有的交易提交到比特币网络之后，不是以每个交易为单位放到链上，这样太繁琐了；
  - 而是以一组为单位打包批处理： 以区块为单位打包上链

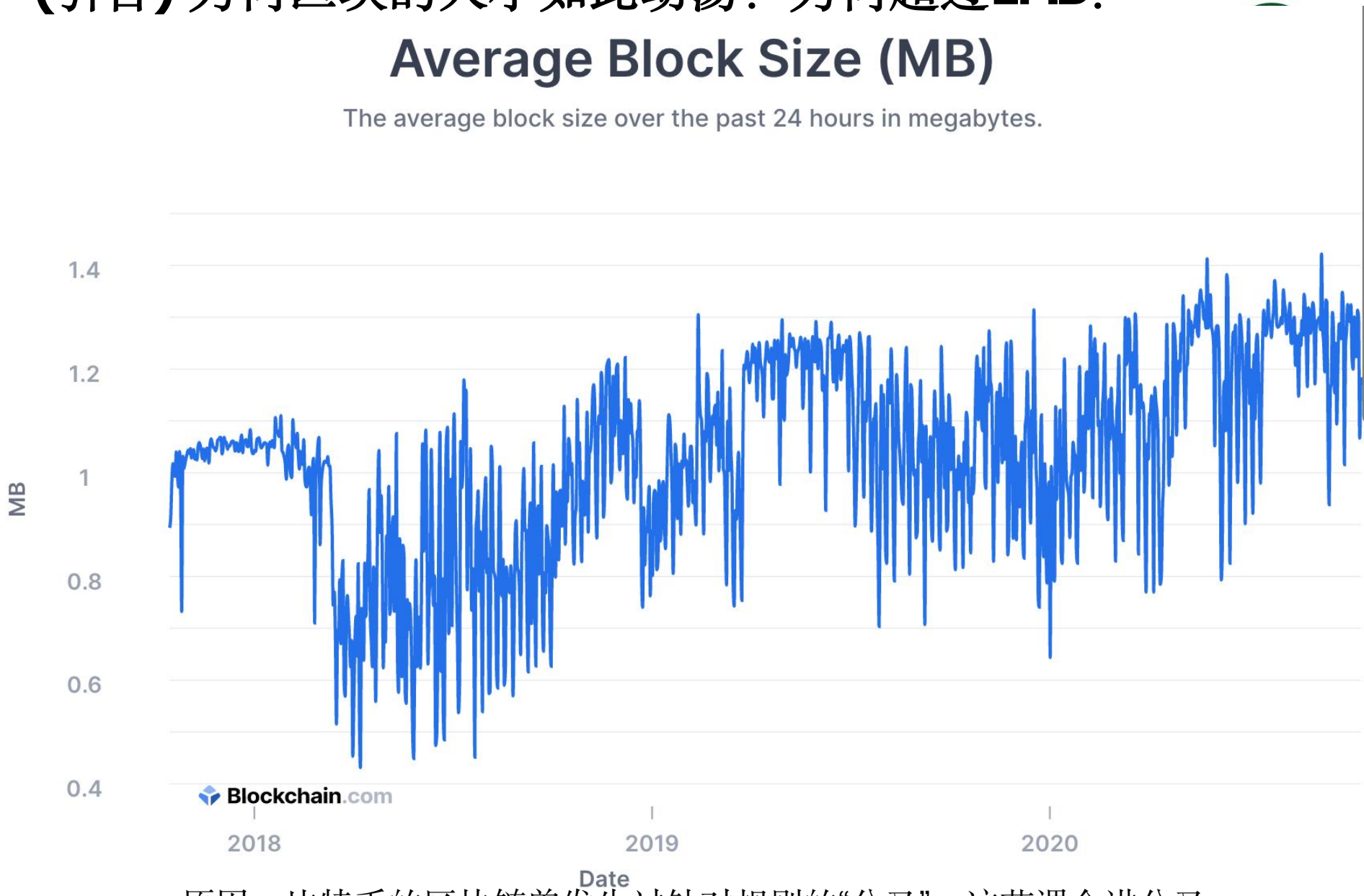
- “Block 是指拥有比特币的矿工进行比特币交易的记录集合吗？”
  - 解答：矿工只是为整个网络来帮助来记账的
  - 比特币网络中的交易是比特币持有者之间的转账记录

课前，几个问题的答疑

- “所以矿工不是指每一个拥有比特币的组织或者用户吗？矿工他是实际存在的人或者组织吗？”
  - 解答：矿工其实就一台可以具有哈希计算能力的电脑
  - 是用来“铸币”的
  - 矿机背后肯定有个所有者
  - 所有矿机都是按照去中心化的方式来组织的
  - 矿机可以通过挖矿来获得比特币，即出块奖励， 背后的所有人就持有了奖励的比特币了，但是它可以不花比特币：继续持有，或者卖给交易所

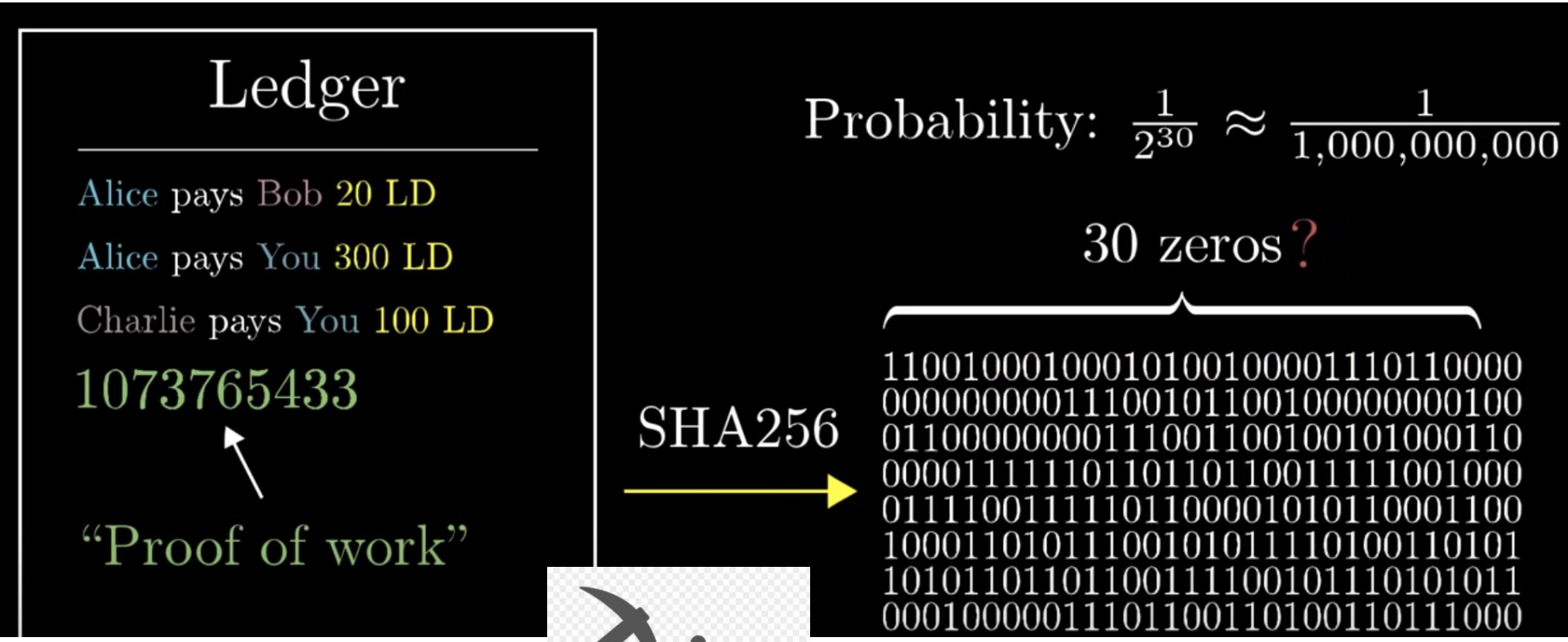
- “挖矿就是给定了输出，然后他们拿数据放进哈希函数试答案的过程吧？如何判断一个矿工挖到矿了？”
  - PoW 的原理其实已经在上节课讲过了

(引言) 为何区块的大小如此动荡？为何超过1MB?



原因：比特币的区块链曾发生过针对规则的“分叉”。这节课会讲分叉。

(引言) 第二课的剧透: Proof of Work, & Difficulty



今天就来讲更多的挖矿细节与原理。

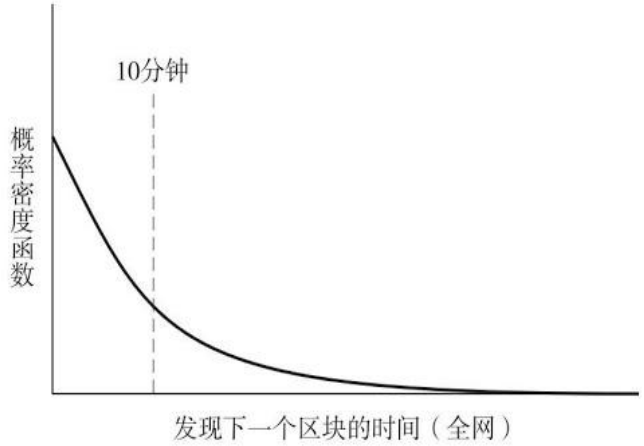
Outline

- Part 1: Math behind Mining
  - 挖矿的概率分析
  - 挖矿的难度设置
- Part 2: Forking
- Part 3: 比特币安全机制的保障

挖矿的概率分析

挖矿的概率分析

- 全网挖出新块的过程：伯努利试验， Poison process
  - 出块时间服从 exponential distribution, 具有“无记忆性 (memoryless)”
  - 当10分钟之后没有出块，全网的所有 miners 之后再经过多久可以出块?
    - 答案是：还是10分钟。
  - 背后是什么原理?



全网发现下一个区块所需时间的概率密度函数

PDF: Probability Distribution Function

定义

对于一维实随机变量X，设它的累积分布函数是 $F_X(x)$,如果存在可测函数 $f_X(x)$,满足： $F_X(x) = \int_{-\infty}^{x} f_X(t)dt$,那么X是一个连续型随机变量，并且 $f_X(x)$ 是它的概率密度函数。

连续型随机变量的概率密度函数有如下性质：

如果概率密度函数fX(x)在一点x上连续，那么累积分布函数可导，并且它的导数：

$dF_X(x)/dx = f_X(x).$

密度函数f(x) 具有下列性质：

① $f(x) \geq 0$ ；

② $\int_{-\infty}^{+\infty} f(x)\,dx = 1$ ；

③ $P(a < x \leq b) = \int_{a}^{b} f(x)\,dx$

离散型随机变量的概率分布

设离散型随机变量X的分布律是

$$P\{X = X_k\} = p_k \qquad k = 1,2,3\cdots$$

则 $F(x) = P(X \le x) = \sum_{x_k \le x} p_k$ 。

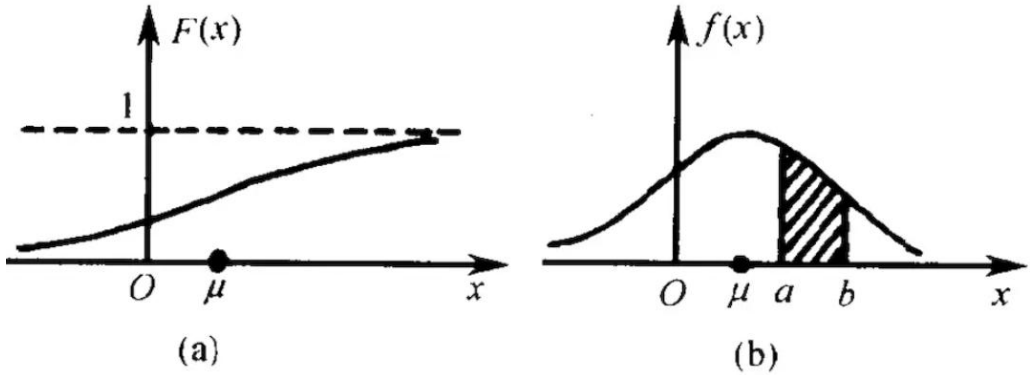
由于$F(x)$是X取$\le x$的诸值x_k的概率之和，故又称$F(x)$为累积概率函数.

连续型随机变量也有“概率函数”和“概率分布函数”吗？

概率函数 -- 概率密度函数

“密度函数”这名词的来由可解释如下. 取定一个点 x, 则按分布函数的定义, 事件 $\{x < X \leqslant x+h\}$ 的概率 ($h>0$ 为常数), 应为 $F(x+h)-F(x)$. 所以, 比值 $[F(x+h)-F(x)]/h$ 可以解释为在 x 点附近 h 这么长的区间 $(x, x+h)$ 内, 单位长所占有的概率. 令 $h \to 0$, 则这个比的极限, 即 $F'(x)=f(x)$, 也就是在 x 点处 (无穷小区段内) 单位长的概率, 或者说, 它反映了概率在 x 点处的“密集程度”. 你可以设想一条极细的无穷长的金属杆, 总质量为 1, 概率密度相当于杆上各点的质量密度.

陈希孺老师所著的《概率论与数理统计》



右图中的面积表示概率

$$P(a \leqslant X \leqslant b) = F(b) - F(a) = \int_a^b (x)\mathrm{d}x$$

泊松过程和泊松分布

- 泊松过程（Poisson process），是以法国数学家泊松（1781 - 1840）的名字命名的。泊松过程是随机过程的一种，是一种随机事件发生次数的独立增量过程。
- 日常生活中，常见的泊松过程
  - 某医院平均每小时出生3个婴儿
  - 某公司平均每10分钟接到1个电话
  - 某超市平均每天销售4包xx牌奶粉
  - 某网站平均每分钟有2次访问
- 泊松分布就是描述某段时间内，事件发生n次的概率。

$$P(N(t)=n)=\frac{(\lambda t)^n e^{-\lambda t}}{n!}$$

- 泊松过程具有平稳增量，即某个时间段内事件发生次数的分布只依赖于该时间段的长度

指数分布

- 指数分布（也称为负指数分布）是描述泊松过程中的事件之间的时间隔间的概率分布，即事件以恒定平均速率连续且独立地发生的过程。
- 下面这些都属于指数分布
  - 婴儿出生的时间间隔
  - 来电的时间间隔
  - 奶粉销售的时间间隔
  - 网站访问的时间间隔
- 若随机变量x服从参数为λ的指数分布， 指数分布的区间是 [0,∞)， 则记为 X ~ E(λ)

概率密度函数

$$f(x)=\begin{cases}\lambda e^{-\lambda x} & x>0\\ 0 & x\le 0\end{cases}$$

指数分布

- 指数分布（也称为负指数分布）是描述泊松过程中的事件之间的时间隔间的概率分布，即事件以恒定平均速率连续且独立地发生的过程。
- 若随机变量x服从参数为λ的指数分布， 指数分布的区间是 [0,∞)，则记为 X ~ E(λ)

概率密度函数

$$f(x)=\begin{cases}\lambda e^{-\lambda x} & x>0\\ 0 & x\le 0\end{cases}$$

- 无记忆性
  - 指数函数的一个重要特征是无记忆性（**Memoryless Property，又称遗失记忆性**）。这表示如果一个随机变量呈指数分布，

$$\text{当 } s,t\ge 0 \text{ 时有 } P(T>s+t|T>t)=P(T>s)$$

  - 如果 *T* 是某一元件的寿命，已知元件使用了 *t* 小时，它总共使用至少 *s+t* 小时的条件概率，与从开始使用时算起它使用至少 *s* 小时的概率相等。

指数分布

数学期望

期望值：$E(X)=\frac{1}{\lambda}$

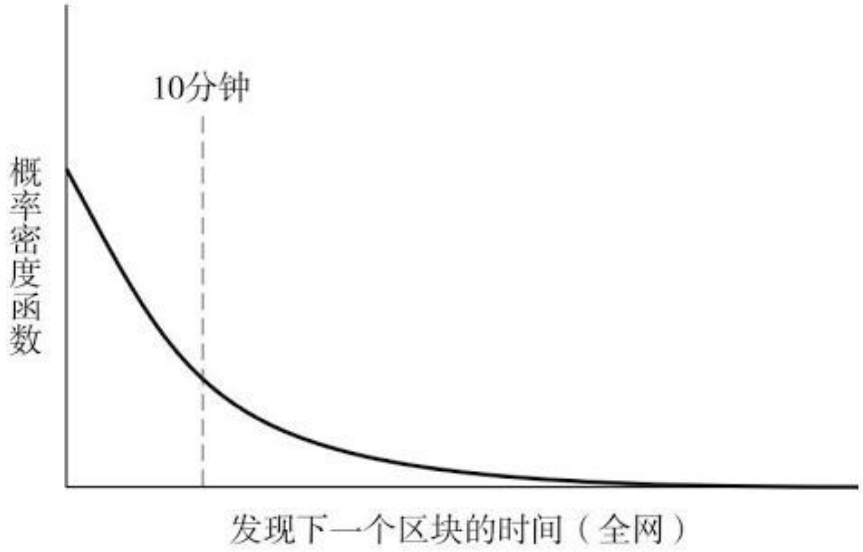
比方说：如果你平均每个小时接到2次电话，那么你预期等待每一次电话的时间是半个小时。

方差

方差：$D(X)=Var(X)=\frac{1}{\lambda^2}$

PDF of mining the next block

全网发现下一个区块所需时间的概率密度函数



But, to any miner:
The avg time spent on mining the next block = 10 min / ratio of its hash power

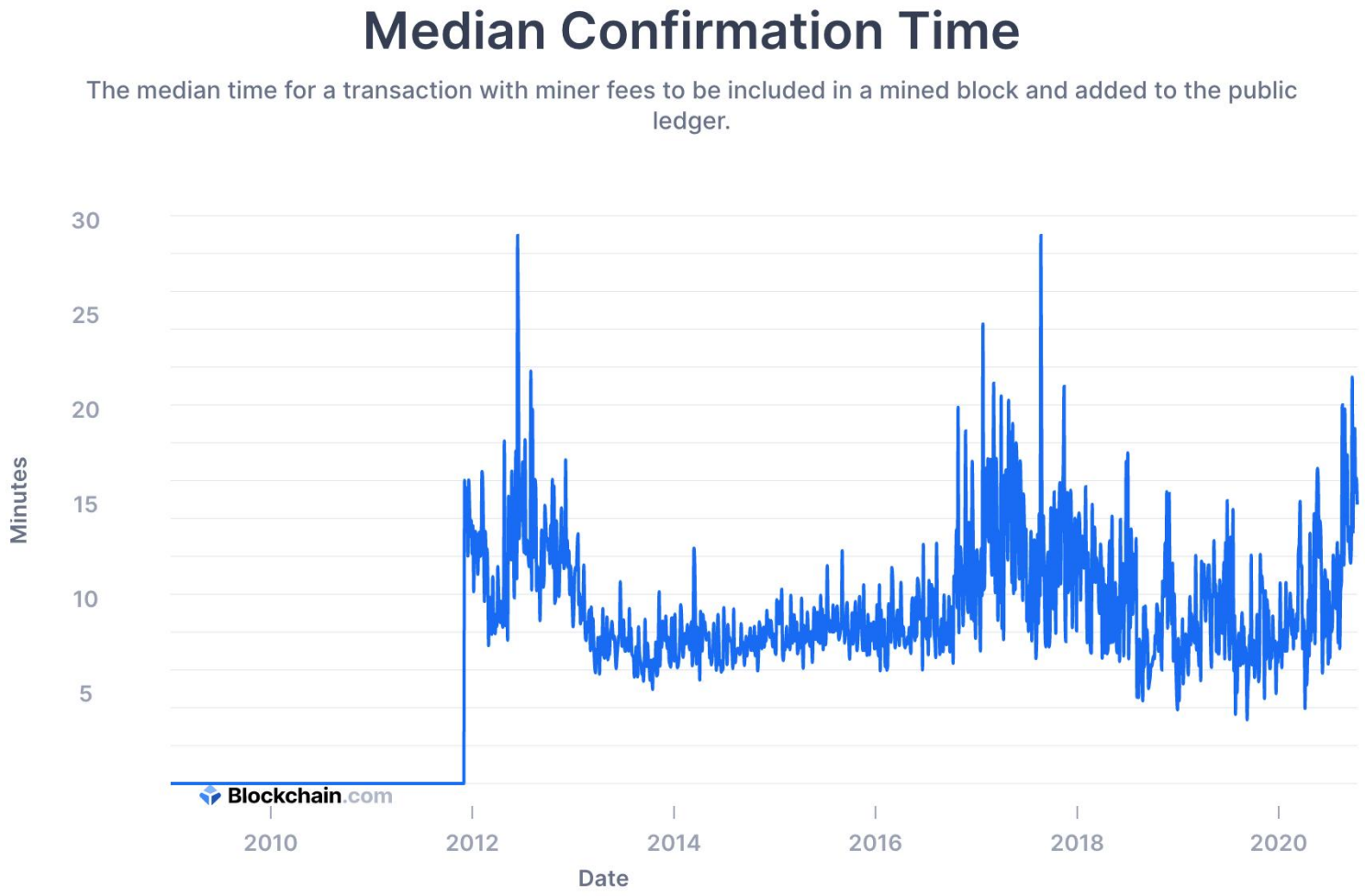
- 假如有全网 0.1% 的 hash power, 每 10,000 分钟能找到一个Block：one week.
- 时间间隔的波动会很大，靠运气

挖矿的概率分析 (cont.)

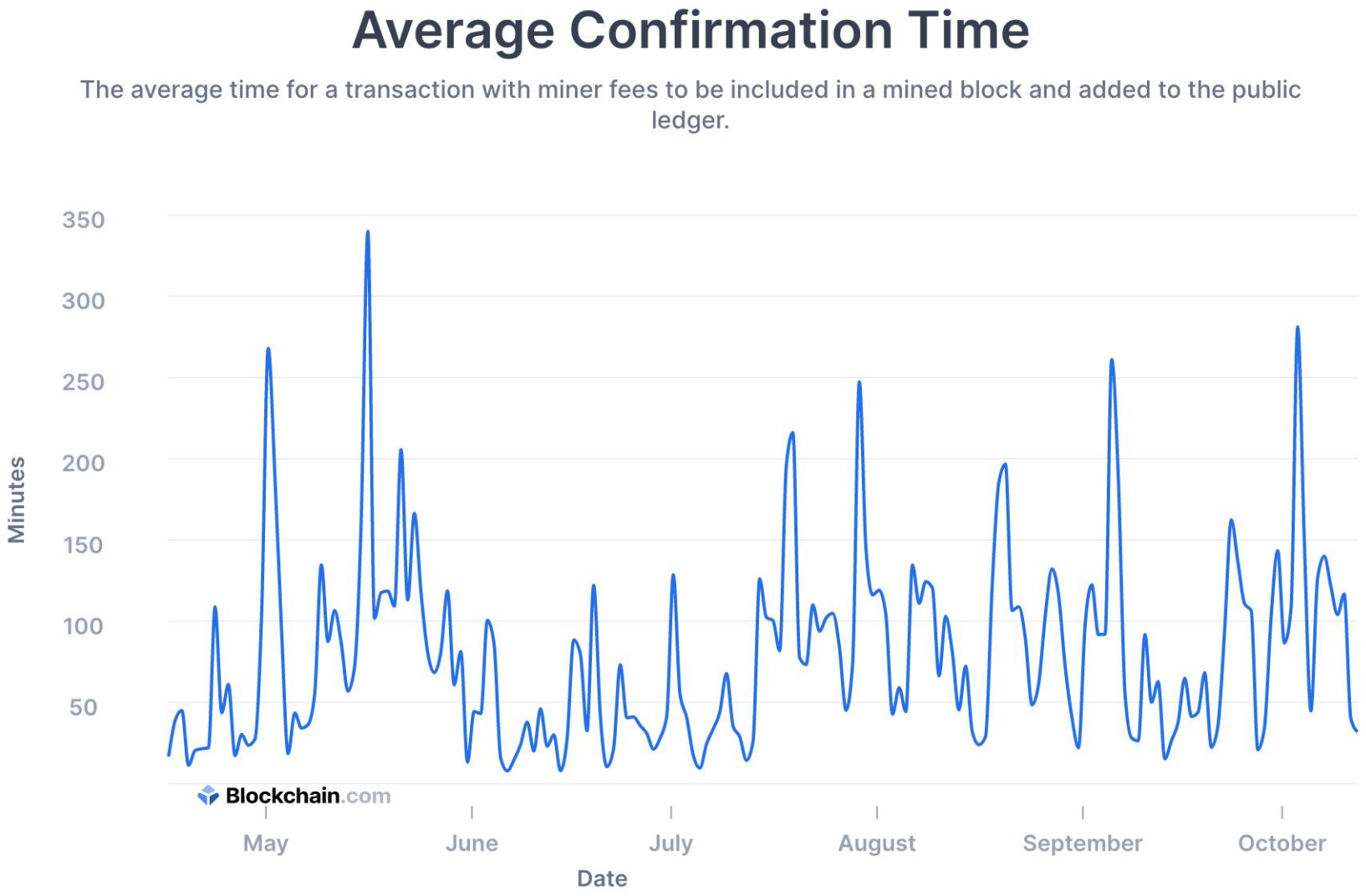
- 全网所有 miners 的伯努利试验：Poison process
  - 当10分钟之后没有出块，全网所有矿工节点之后再经过多久可以出块?
    - 答案是：还是10分钟。
  - 这个性质看似无情，其实无记忆性恰恰是保证公平挖矿的理论所在
    - 算力强的矿机与算力弱的矿机在每一次尝试“解题”的过程中，要具有相同的成功概率才行

一些挖矿的统计数据

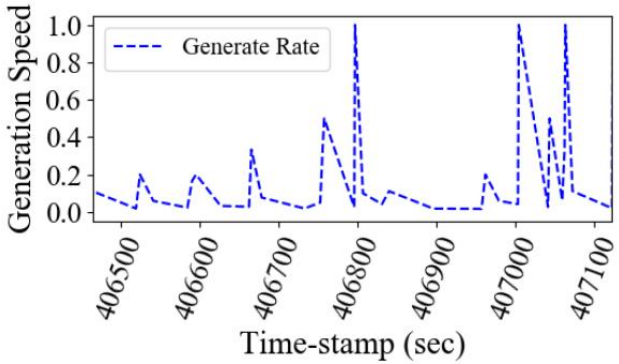
- Median Confirmation Time of a TX



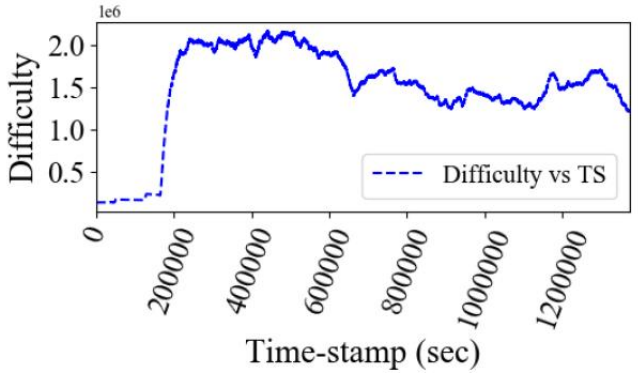
一些挖矿的统计数据（cont.）



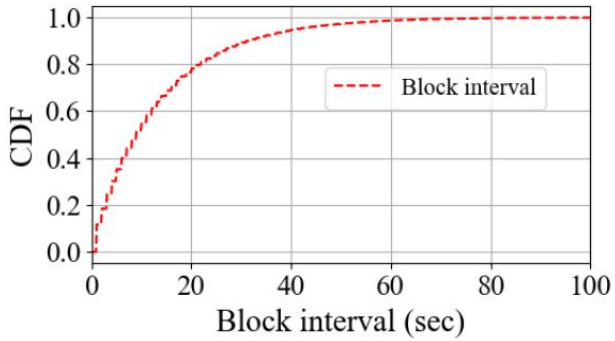
一些挖矿的统计数据 (cont.)



(a) Block-generation speed vs timestamp



(b) Difficult vs timestamp



(c) CDF of block generation intervals.

Fig. 12. Performance of the prototype deployed on the remote google cloud.

挖矿的难度

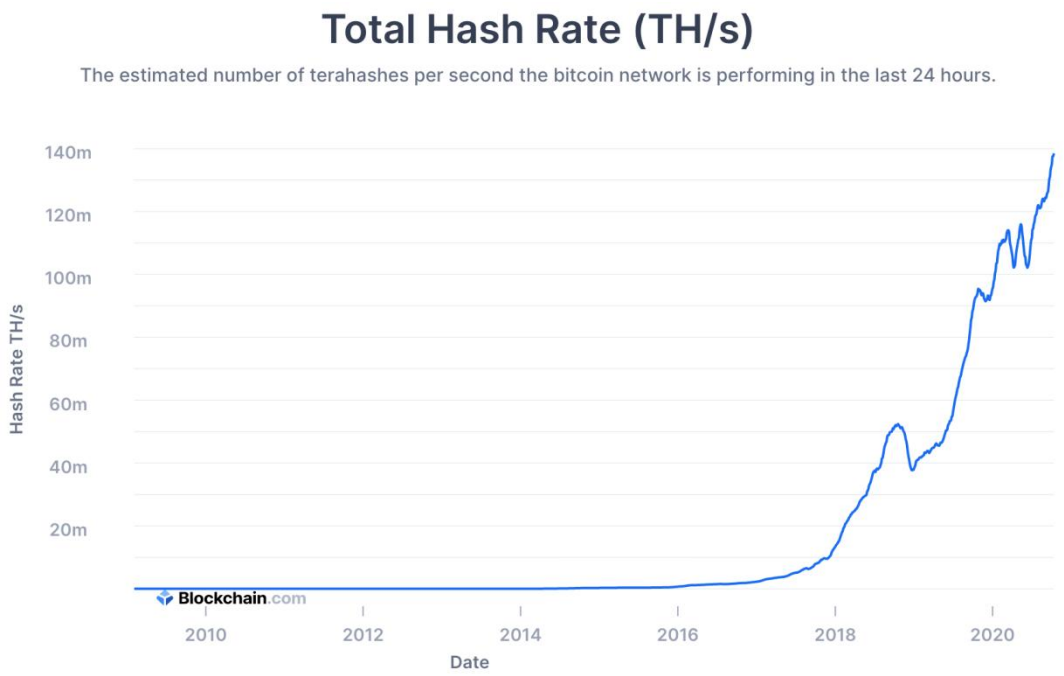
Mining Difficulty

- ***Difficulty*** is a measure of how difficult it is to mine a block,
  - (in more technical terms) to find a hash falling into a given target.

- A high difficulty indicates that it will take more computing power to mine the same number of blocks,
  - making the network more secure against attacks.

Mining Difficulty (cont.)

- The difficulty adjustment is directly related to the total estimated mining power
  - estimated in the Total Hash Rate (TH/s) chart.



设置挖矿难度的原因

- 难度调整
  - The difficulty is adjusted every 2016 blocks (every 2 weeks approximately) so that the average time between 2 consecutive blocks remains 10 minutes.
- 通过调整挖矿难度，使得出块时间相对稳定
- 让分叉攻击更难

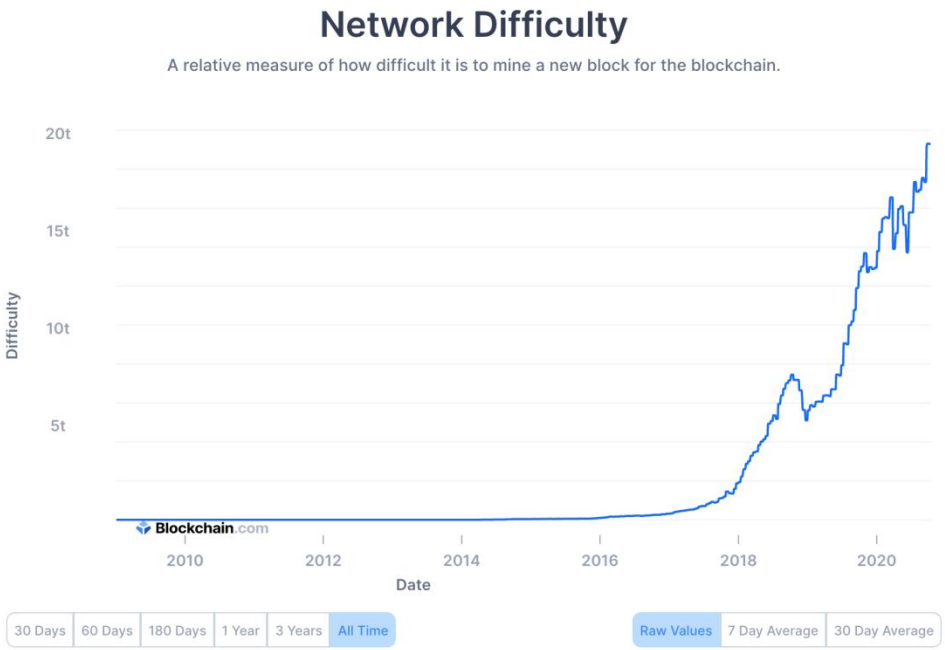
挖矿难度的动态调整

- 比特币的难度调整方法:
  - Every two weeks, to calcuate:
  - next_difficulty = previous_difficulty * (2 weeks) / (The time to mine the recent 2016 blocks)

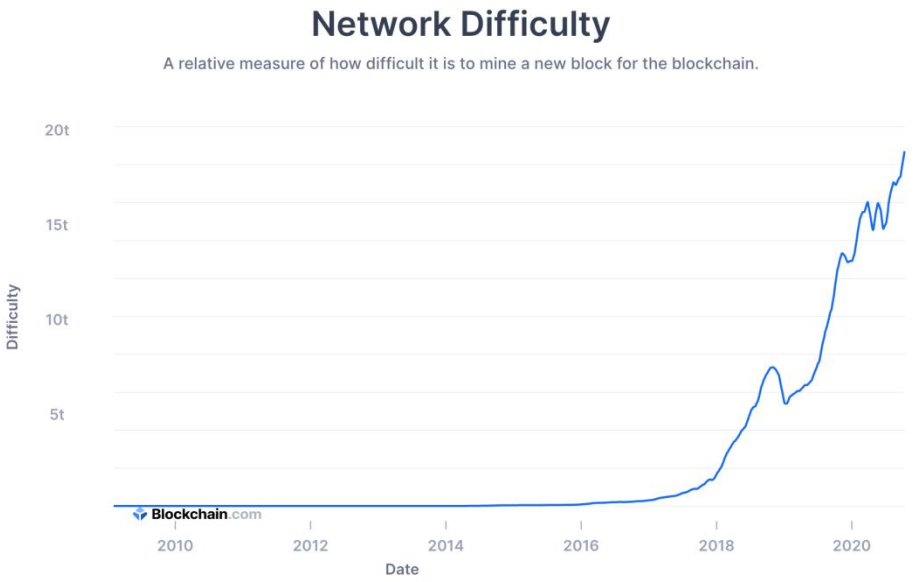
- Why 2016 blocks?
  - It is the expected number of blocks in 2 weeks at the specified rate 10 min/block.

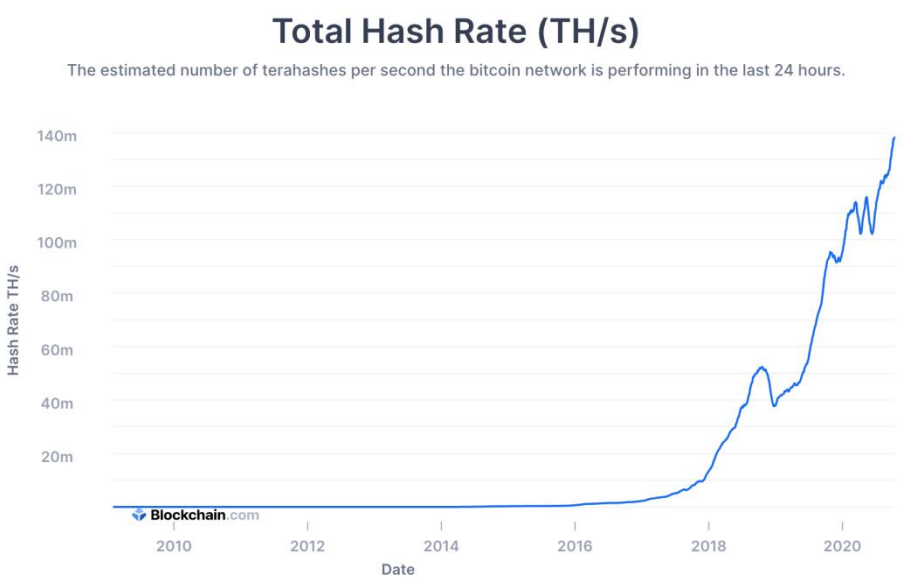
挖矿难度的动态调整 (cont.)

- 无奈之举：全网 Hashrate 的提高，逼着挖矿难度增加
  - Difficulty over time: Periodically increasing per two weeks.
  - 挖矿的人数越来越多，用的设备越来越先进.



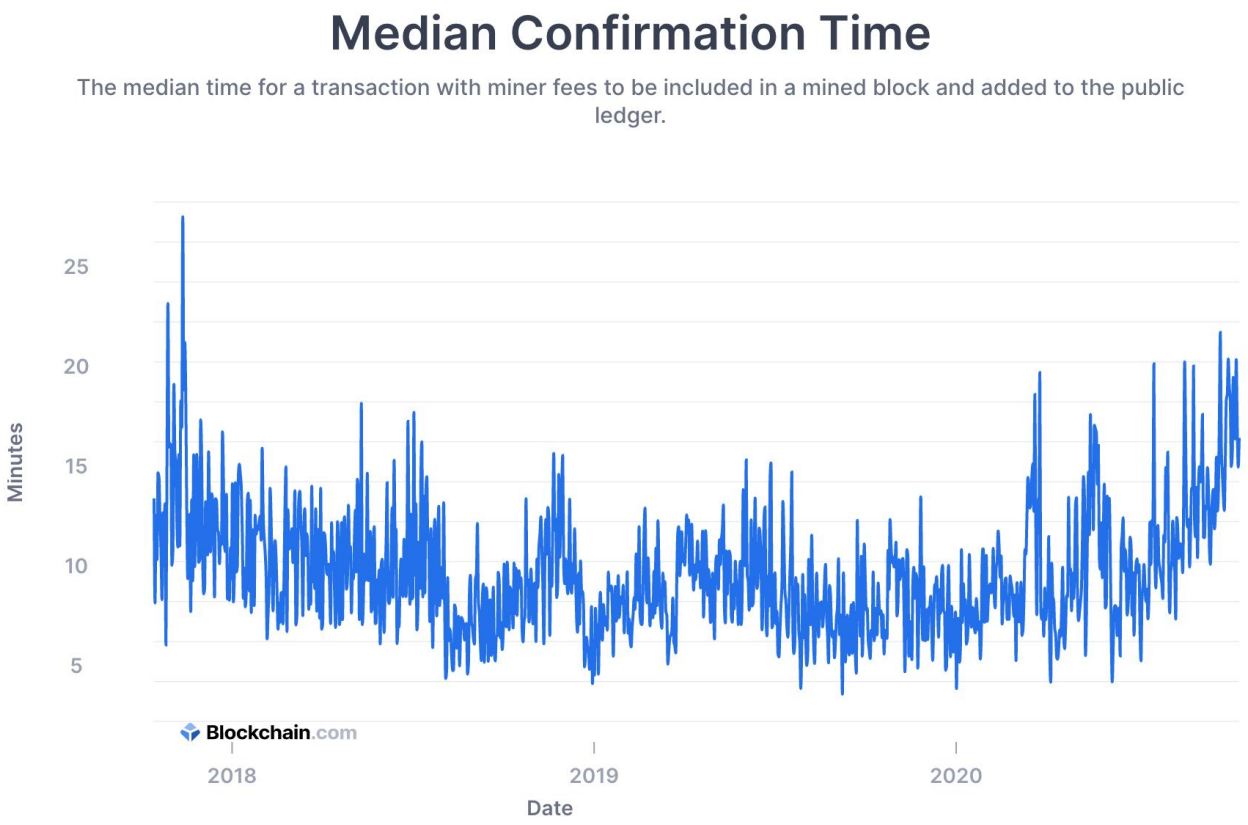
Difficulty vs. Hash Rate





挖矿难度的动态调整的效果

- 虽然 Hashrate 在提高，挖矿难度也在增加
  - 出块时间分布图：相对稳定在一个期望附近震荡



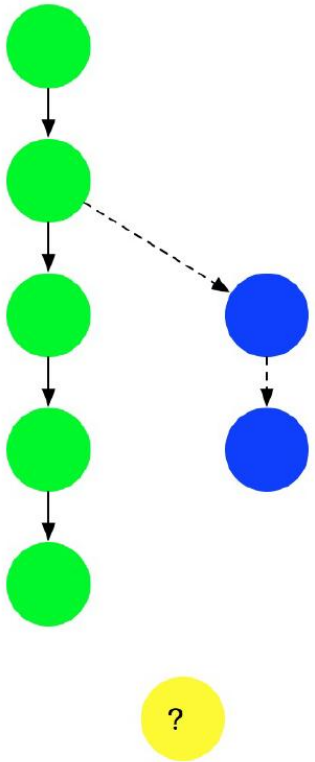
Outline

- Part 1: Math for Mining
- Part 2: Forking
  - 一般的分叉
  - 硬软分叉
  - 恶意分叉
- Part 3: 比特币安全机制的保障

一般的分叉

一般的分叉 与 PoW 最长链机制

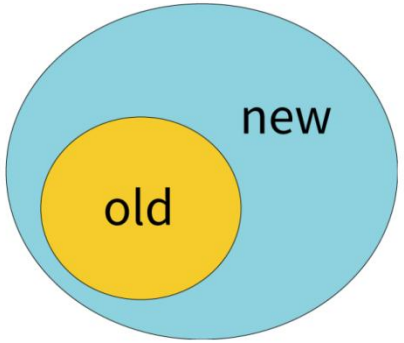
- P2P 区块链网络中存在什么典型问题？
  - 加入全网同时有两个合法提案会在网络中进行广播，收到的用户进行验证后，会基于用户认为的最长链基础上继续难题的计算。因此，系统中可能出现链的**分叉**（**Forking**）
  - **解决方案：** **比特币网络最长链机制**
  - 假定超市只有一个出口，付款时需要排成一队，可能有人不守规矩要插队。超市管理员会检查队伍，认为最长的一条队伍是合法的，并让不合法的分叉队伍重新排队。新到来的人只要足够理智，就会自觉选择最长的队伍进行排队



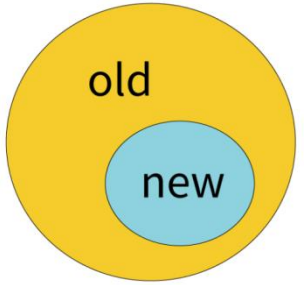
不得已的分叉
硬分叉 与 软分叉

不得已的分叉——对规则的分叉

- 硬分叉
  - 规则改变，产生一个不同的链
  - 新旧节点各不兼容

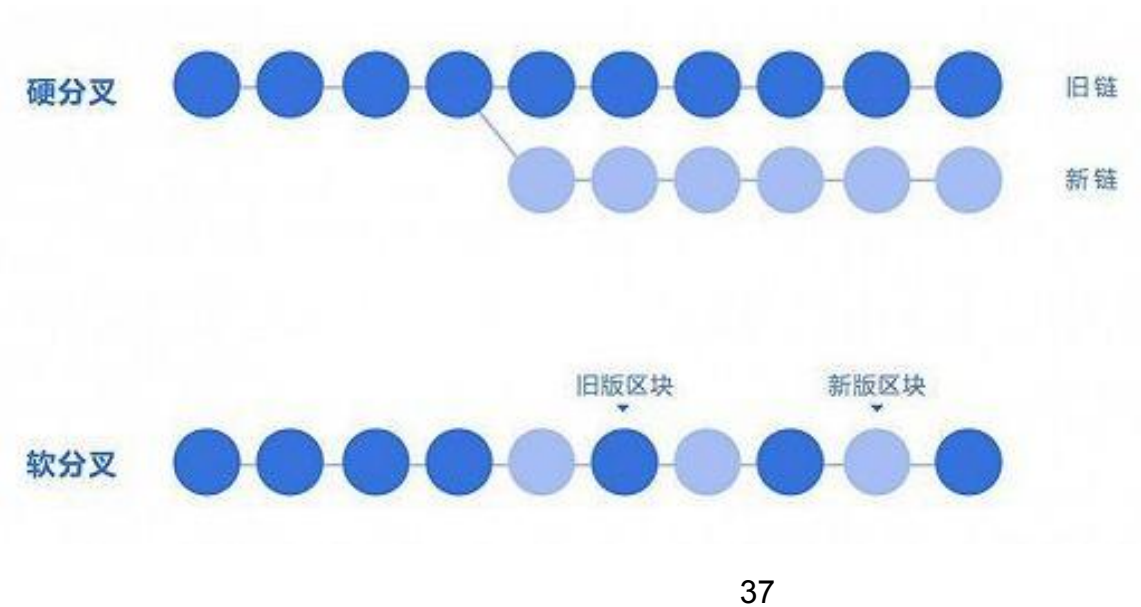


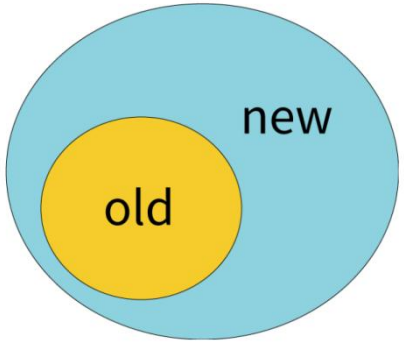
- 软分叉
  - 打补丁，加入新特性，让现有的规则更加严格
  - Such that the old nodes keep the same, but new nodes are upgraded

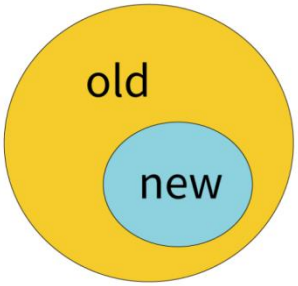


不得已的分叉——对规则的分叉(cont.)

- 硬分叉
  - 区块链发生永久性分歧，在新共识规则发布后，部分没有升级的节点无法验证已经升级的节点生产的区块，通常硬分叉就会发生。
  - 规则改变，产生一个不同的链





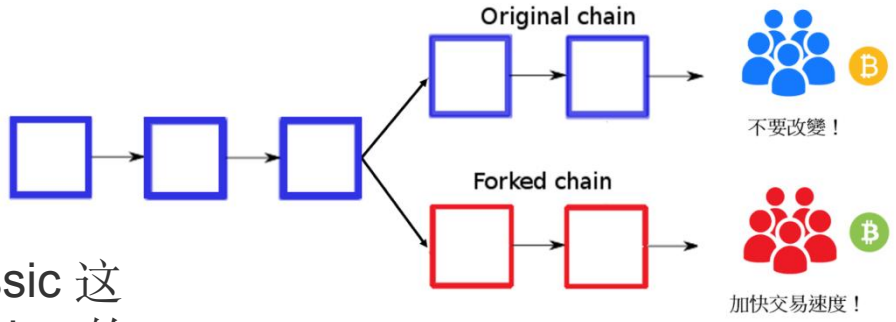


不得已的分叉——对规则的分叉(cont.)

- 软分叉
  - 当新共识规则发布后，没有升级的节点会因为不知道新共识规则下，而生产不合法的区块，就会产生临时性分叉。
  - 老版本节点可能做无用功：
    - 挖到无效块，因为这些块中包含了在新规则下无法被验证的交易
    - 转发给其他新节点，新块不会被接受
    - 这会强迫老节点更新协议/规则
    - 老节点转而扩展最长的链：分叉消失

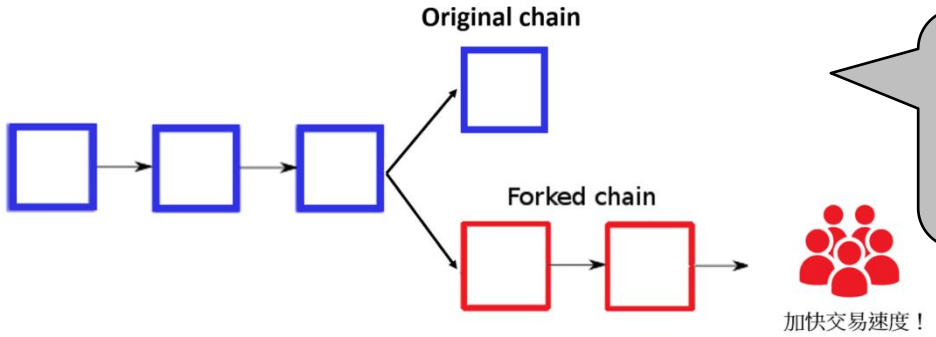
对规则的分叉(cont.)—— 硬/软分叉的例子

- 需求：加快交易速度，一些人提议：
  - （1）换一种共识机制
  - （2）或者，调整关键参数，比如增大block size



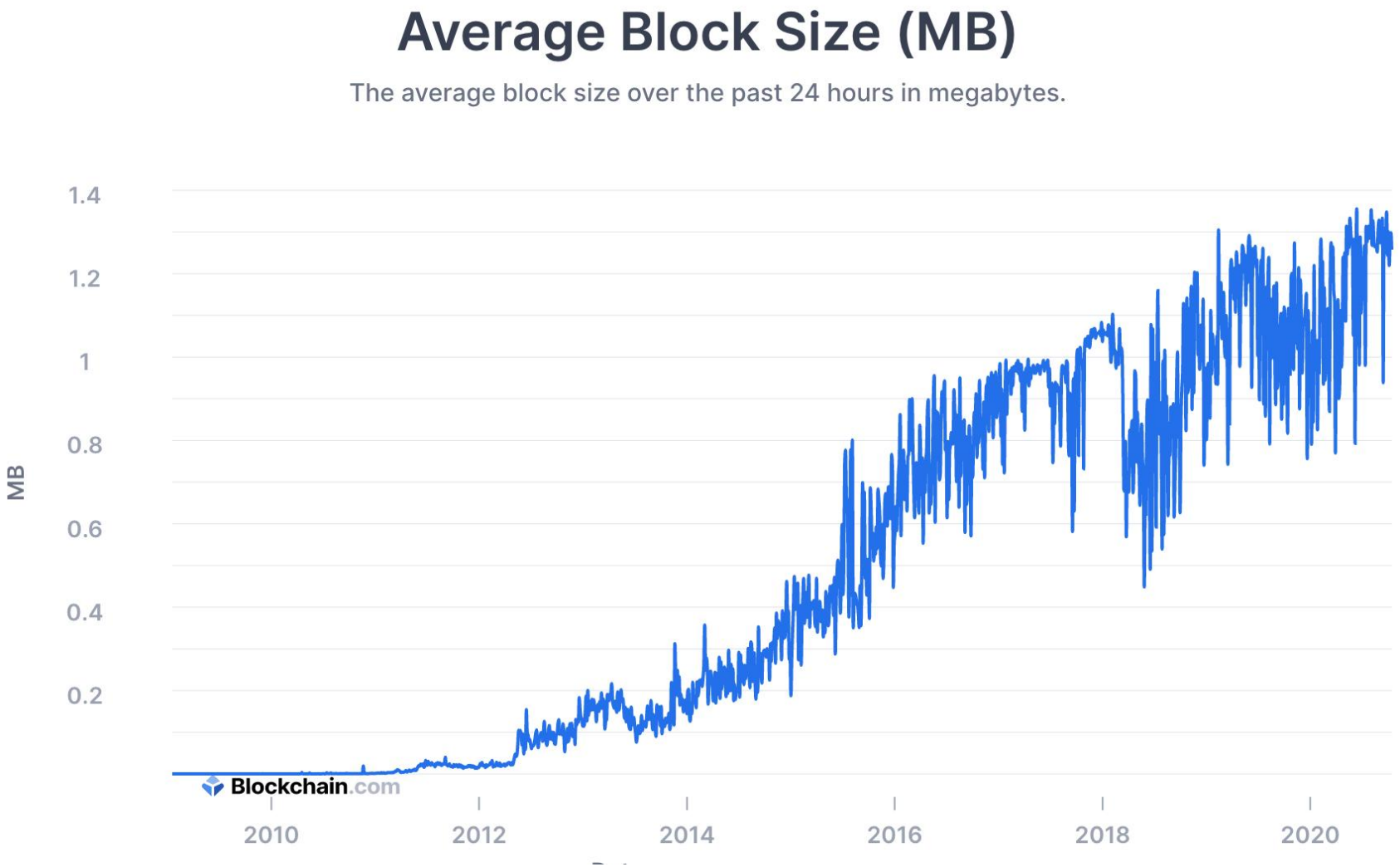
硬分叉

比如 BitcoinClassic 这个软件将 block size 的最大值调到 2M



软分叉：
如果旧节点妥协，
可以升级自己

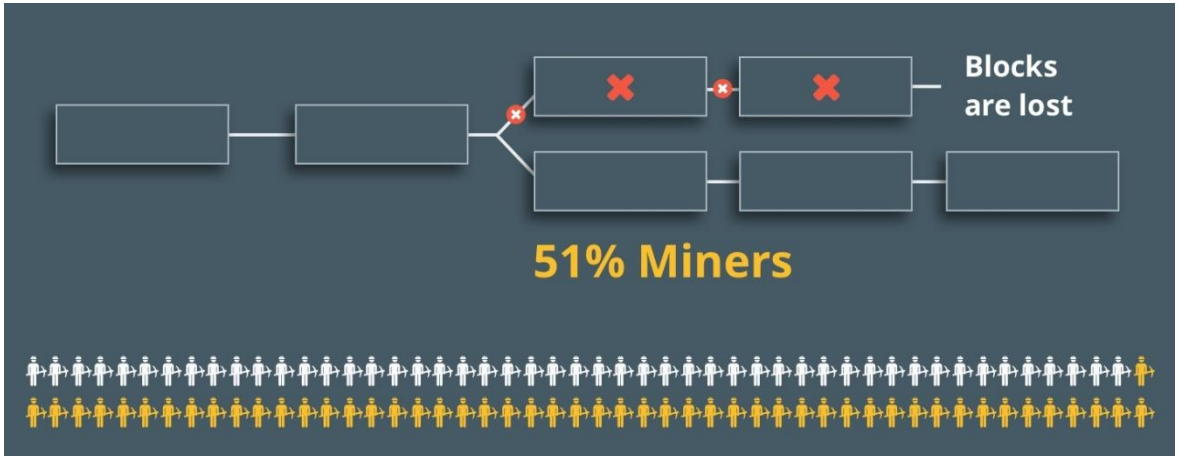
历史上 avg Block Size 确实是增大的



恶意的分叉

恶意的分叉

- 分叉攻击 —— 最长链机制带来的副作用
  - 如，为了double spending, 发动 51% attack



Outline

- Part 1: Math for Mining
- Part 2: Forking
- Part 3: 比特币安全机制的保障
  - 挖矿的安全性分析
  - 自私挖矿
  - 分叉攻击分析

挖矿的安全性分析

思考1：挖矿的安全性分析

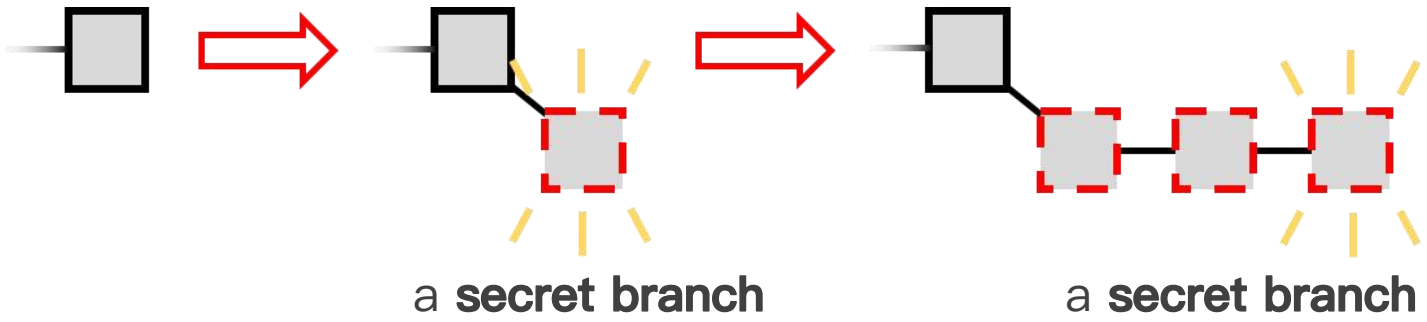
- 假设比特币的大部分算力是掌握在 honest 矿工手里，背后有什么安全保障？让我们思考3个小问题
  - #1. 恶意节点可以伪造一个交易把别人的钱转给自己吗？
    - 不能，因为有签名，别的 honest 节点不会承认，把它通过分叉废除了，该恶意节点白费力气又损失了钱
  - #2. 恶意节点可以 double-spending 吗？
    - 很难，除非有 51% 算力
  - 此外，如果大部分算力是诚实的，可以避免 selfish-mining，为什么？
    - 自私挖矿：悄悄挖不发布，为了获取更多的出块奖励
    - 回答这个问题，我们讲一下 selfish mining

自私挖矿

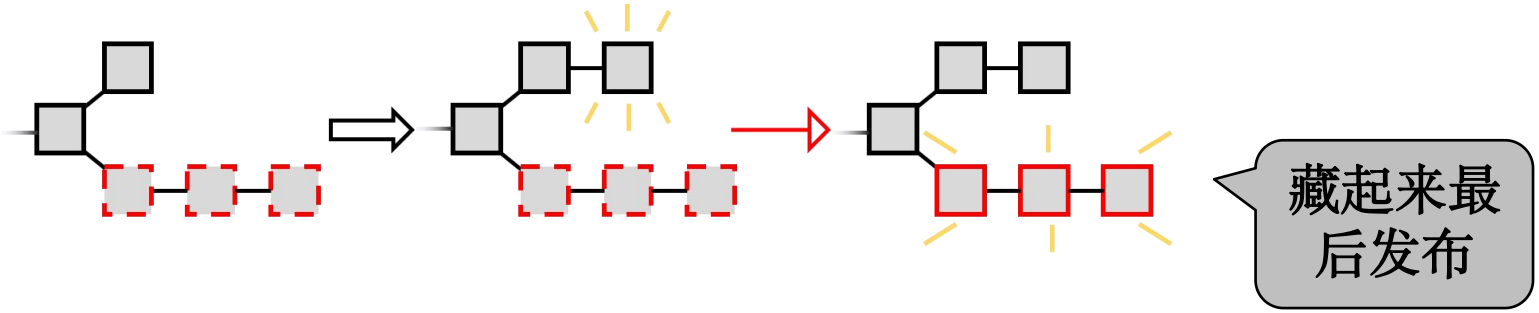
Selfish Mining —— 自私挖矿

- Definition
  - A selfish miner hides the new block it just mined, and keeps to mine the next following this hidden one.

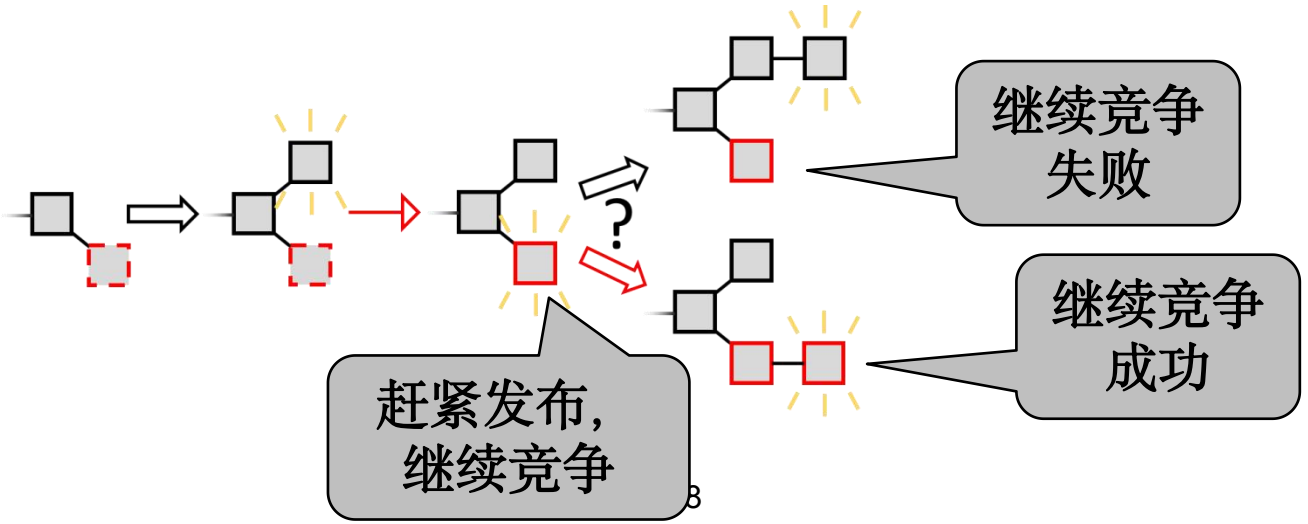
- Motivation: Why mine secretly?
  - Only himself knows a ***new*** block was just mined, such that others are mining following the ***old*** previous block
  - 一旦发布出去，大家都会在新区块后边平等地开始竞争



Selfish Mining —— 自私挖矿

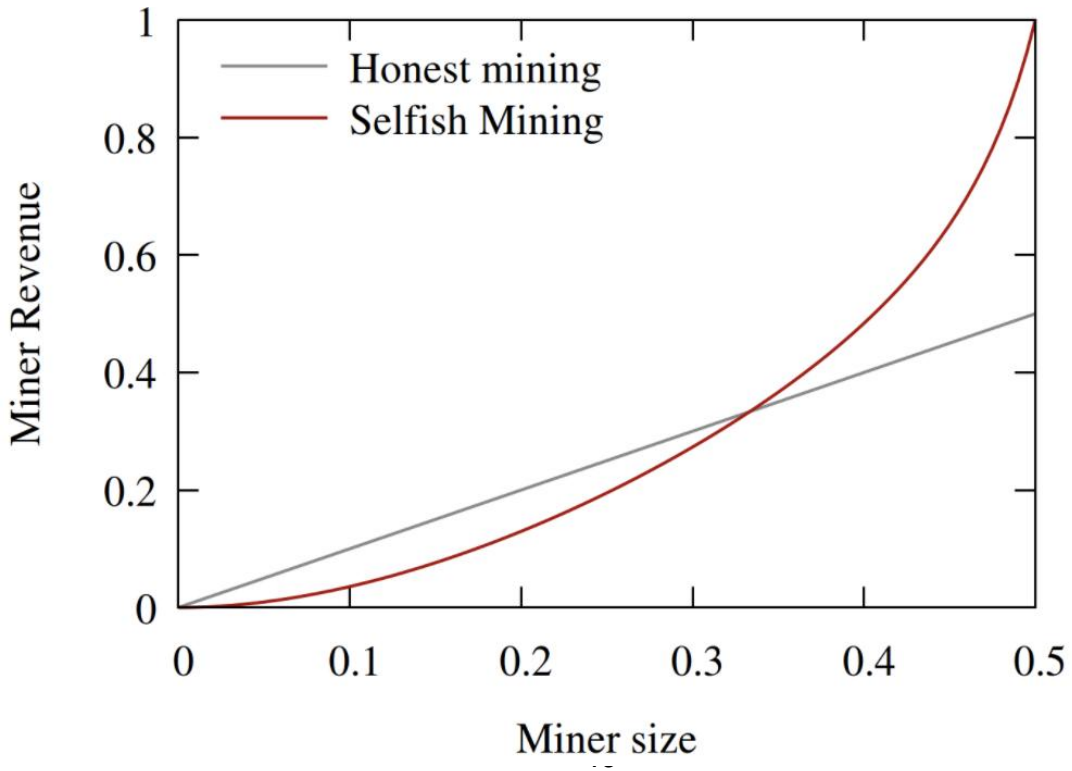


- Risks
  - 不发布的块有可能会浪费掉，所以还不如赶紧发布出去获取当前的出块奖励（落袋为安）



Selfish Mining —— 自私挖矿

- 收益曲线
  - （某项研究表明）如果一个矿工他的算力超过全网的三分一，他很可能为了更多的收益而选择自私挖矿



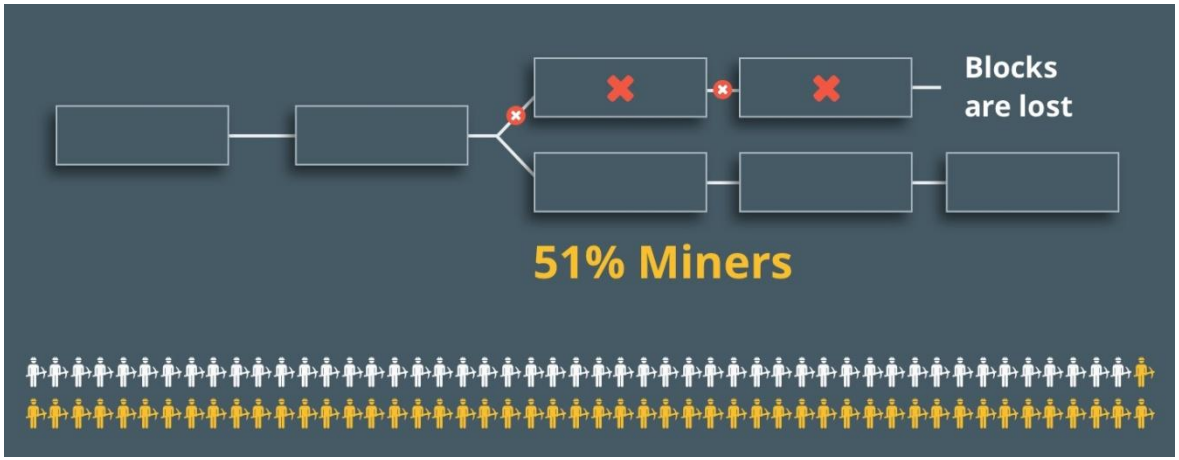
分叉攻击的一些事实与分析

思考2：挖矿的安全性分析

- 比特币的主要安全保障是什么？
  - 合谋发动“分叉攻击”，必须要占据系统中超过半数以上的计算力才可能成功
  - Fact: 当一个新区块来了，所有 miners 都需要停止当前的挖矿，把新 block 添加后，接着刚才的成果继续挖；这样可行吗？
    - 不可行，因为 PoW mining 是一个无记忆性的过程，
    - 从任何时候开始挖，成功率都是一样的。
    - 这样就可以防止“提前偷偷进行预先挖矿”

一种分叉攻击 —— 51% attack

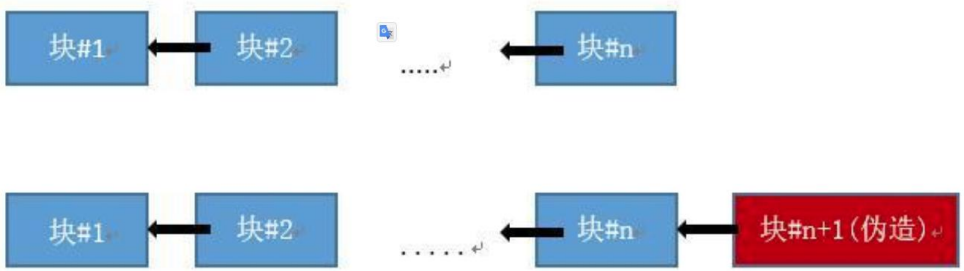
- A 51% attack is an attack on a blockchain by a group of miners who control more than 50% of the network's mining hash rate.



现实中的 51% 攻击

- 比特币区块的构建和算力的多少紧密相关，因此控制了算力就控制了区块链的生成

- 设想这样一个现实场景：
  - Alice 和 Bob之间使用比特币完成了一杯咖啡的交易，Bob在收到Alice的转账通知 (交易提交)，就给Alice提供了咖啡。
  - Alice不想支付这笔钱，在开始之前他把区块里的这笔交易改成Alice转给自己的一笔交易了(更改很容易，只要把接收地址和签名改掉即可)。
  - Alice开始尝试用这个伪区块参与挖矿 (挖矿成功后这个block会被加入主链中)，因为拥有 51% 的算力，Alice比别的节点更容易优先计算成功，导致一个伪造的区块加入主链。



- 一般来说，一个miner达到 1/3的计算力，比特币网络就存在被破坏的风险了，也就会出现双花问题

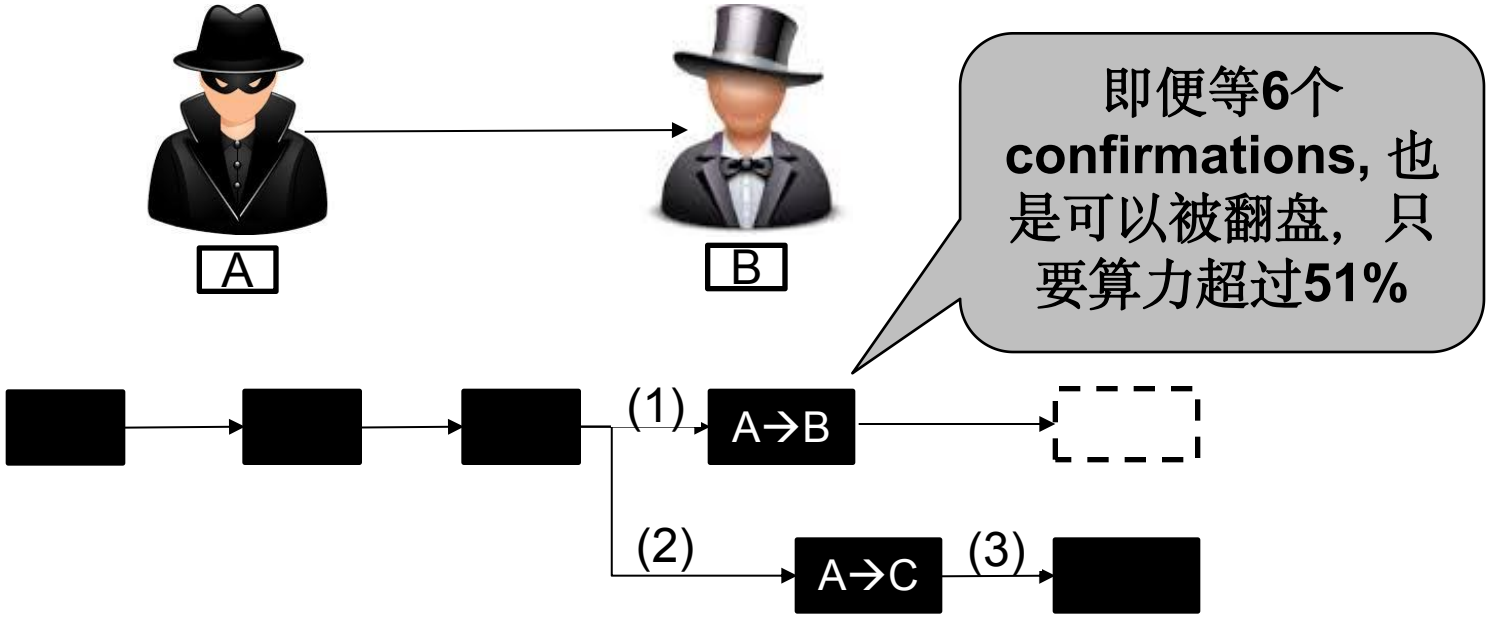
一种分叉攻击 —— 51% attack (cont.)

通过51%攻击，可以实现 double-spending attack

(1) A给B支付比特币，交易在一个块中确认

(2) A重新构造一笔交易A→C，并打包进区块公布（分叉）

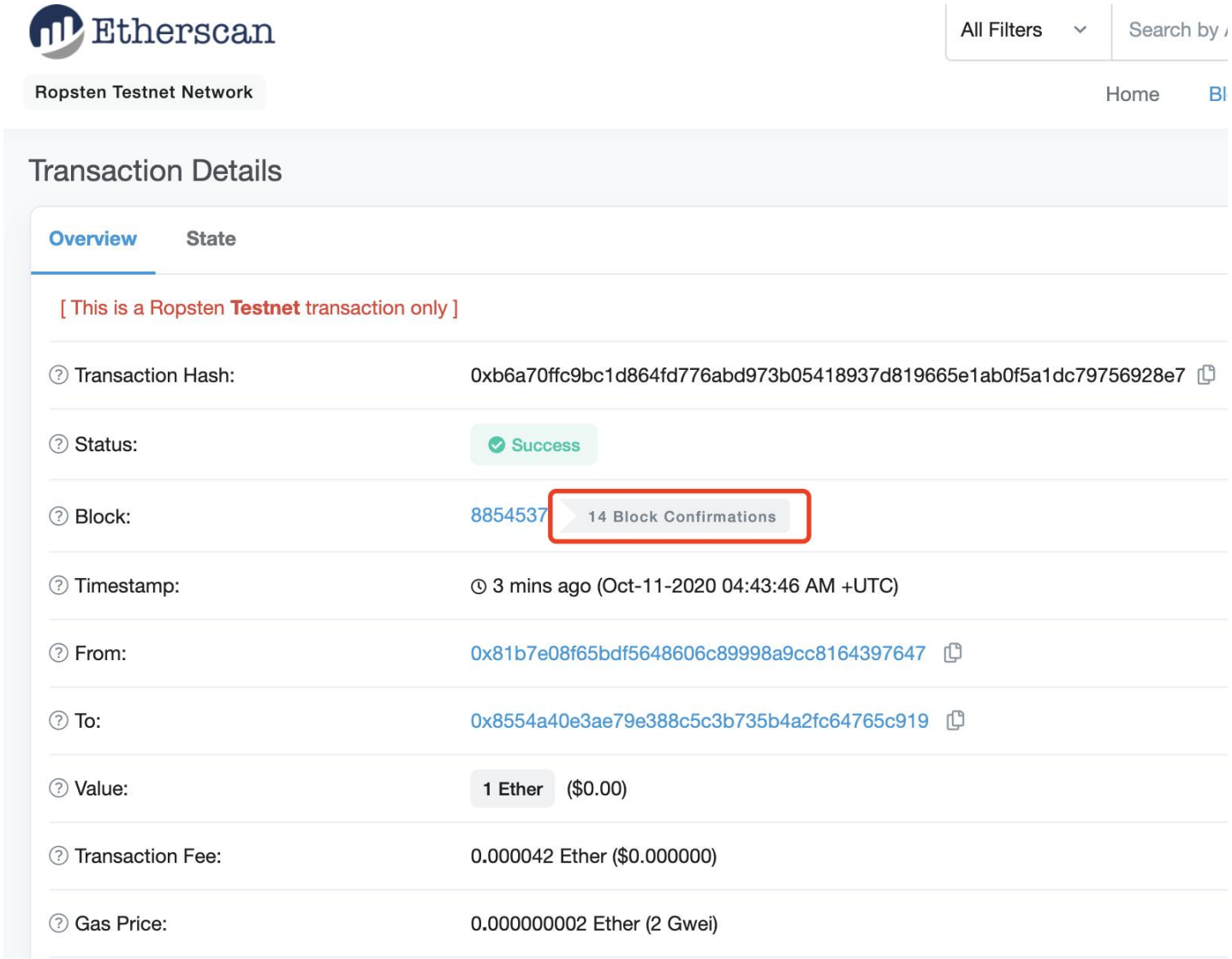
(3) 包含双重支付的块率先找到下一个块，全网认可A→C, 交易A→B无效



如何防范 51%攻击？

- 除了尽量避免计算力放到同一个组织手里，没太好的办法
  - 这是目前 PoW 机制自身造成的
- 绝大多数的矿工，都会通过诚实挖矿来维持整个比特币系统
  - 如果他们集体伪造交易，用户对比特币失去了信心，没人再去使用比特币。那么矿工伪造了交易盗取比特币就失去了意义
- 6个确认
  - 如果真有这样的51%攻击，建议是收款方等到全网的 6 个区块确认之后再交付商品
    - 按照10分钟一个区块的速度，只需一个小时就可以保证你的钱是否基本肯定收到
    - 6个区块后再对全网进行篡改的难度很高
- 即便如此，PoW仍是目前数学上可证的最安全的机制。

交易的确认次数 – 以太坊的交易例子



Summary of this class

- Part 1: Mathematics behind Mining
- Part 2: Forking —— 分叉的原理与类型
- Part 3: 比特币安全机制的一些探讨