VICTORLAMP

区块链技术基础篇
之四：白话拜占庭将军问题

e休，爱编程的葫芦娃：exiu@victorlamp.com

多媒体课程：《深入浅出区块链技术基础篇》

课程目录

《区块链技术基础篇之一：白话非对称加解密》
《区块链技术基础篇之二：白话哈希算法》
《区块链技术基础篇之三：白话P2P网络》
《区块链技术基础篇之四：白话拜占庭将军问题》
《区块链技术核心篇之一：比特币区块链起源及原理》
《区块链技术核心篇之二：比特币区块链密钥与地址》
《区块链技术核心篇之三：比特币区块链交易共识》
《区块链技术核心篇之四：比特币区块链核心架构》

讲师介绍

网名：一休

2015 年开始，从事区块链技术开发，先后成功的研发出：区块链技术的数字版权管理（DRM）系统、基于区块链IPFS的CDN产品开发。

是多项区块链技术专利发明人。

对于比特币区块链、IPFS项目源码非常熟悉。

本人负责的基于区块链的创新技术方案，获得了当年华为公司年度十大发明奖。

本课程带领读懂区块链技术之核心—拜占庭将军问题

聊到区块链技术，都绕不开拜占庭将军问题，拜占庭将军问题到底是什么问题呢？

拜占庭将军问题是P2P网络通讯中如何达成共识的经典问题。

在中本聪发明比特币以前，世界上并没有一个非常完美的方法来解决“拜占庭将军问题”。

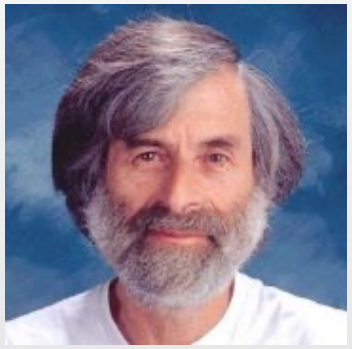
什么是共识机制

所谓"共识机制"，是通过特殊节点的投票，在很短的时间内完成对交易的验证和确认；对一笔交易，如果利益不相干的若干个节点能够达成共识，我们就可以认为全网对此也能够达成共识。再通俗一点来讲，如果中国一名微博大V、美国一名虚拟币玩家、一名非洲留学生和一名欧洲旅行者互不相识，但他们都一致认为你是个好人，那么基本上就可以断定你这人还不坏。

区块链为什么需要共识机制（算法）

要想整个区块链网络节点维持一份相同的数据，同时保证每个参与者的公平性，整个体系的所有参与者必须要有统一的协议，也就是我们这里要将的共识算法。比特币所有的节点都遵循统一的协议规范。协议规范（共识算法）由相关的共识规则组成，这些规则可以分为两个大的核心：工作量证明与最长链机制。

所有规则（共识）的最终体现就是比特币的最长链。共识算法的目的就是保证比特币不停地在最长链条上运转，从而保证整个记账系统的一致性和可靠性。



Leslie Lamport
Distinguished Scientist

Microsoft Research Lab – Redmond
Microsoft Building 99,
14820 NE 36th Street,
Redmond, Washington, 98052
USA

About

Leslie B. Lamport is an American computer scientist. Lamport is best known for his seminal work in distributed systems and as the initial developer of the document preparation system LaTeX. Leslie Lamport was the winner of the 2013 Turing Award（图灵奖）for imposing clear, well-defined coherence on the seemingly chaotic behavior of distributed computing systems, in which several autonomous computers communicate with each other by passing messages. He devised important algorithms and developed formal modeling and verification protocols that improve the quality of real distributed systems. These contributions have resulted in improved correctness, performance, and reliability of computer systems.

什么是拜占庭将军问题?

The Byzantine Generals Problem

LESLIE LAMPORT, ROBERT SHOSTAK, and MARSHALL PEASE
SRI International

Reliable computer systems must handle malfunctioning components that give conflicting information to different parts of the system. This situation can be expressed abstractly in terms of a group of generals of the Byzantine army camped with their troops around an enemy city. Communicating only by messenger, the generals must agree upon a common battle plan. However, one or more of them may be traitors who will try to confuse the others. The problem is to find an algorithm to ensure that the loyal generals will reach agreement. It is shown that, using only oral messages, this problem is solvable if and only if more than two-thirds of the generals are loyal; so a single traitor can confound two loyal generals. With unforgeable written messages, the problem is solvable for any number of generals and possible traitors. Applications of the solutions to reliable computer systems are then discussed.

Categories and Subject Descriptors: C.2.4. [**Computer-Communication Networks**]: Distributed Systems—*network operating systems*; D.4.4 [**Operating Systems**]: Communications Management—*network communication*; D.4.5 [**Operating Systems**]: Reliability—*fault tolerance*

General Terms: Algorithms, Reliability

Additional Key Words and Phrases: Interactive consistency

1. INTRODUCTION

A reliable computer system must be able to cope with the failure of one or more of its components. A failed component may exhibit a type of behavior that is often overlooked—namely, sending conflicting information to different parts of the system. The problem of coping with this type of failure is expressed abstractly as the Byzantine Generals Problem. We devote the major part of the paper to a discussion of this abstract problem and conclude by indicating how our solutions can be used in implementing a reliable computer system.

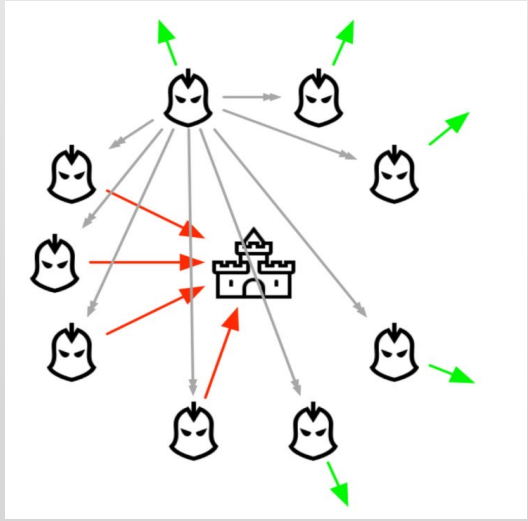
We imagine that several divisions of the Byzantine army are camped outside an enemy city, each division commanded by its own general. The generals can communicate with one another only by messenger. After observing the enemy, they must decide upon a common plan of action. However, some of the generals

This research was supported in part by the National Aeronautics and Space Administration under contract NAS1-15428 Mod. 3, the Ballistic Missile Defense Systems Command under contract DASG60-78-C-0046, and the Army Research Office under contract DAAG29-79-C-0102.
Authors' address: Computer Science Laboratory, SRI International, 333 Ravenswood Avenue, Menlo Park, CA 94025.

ACM Transactions on Programming Languages and Systems, Vol. 4, No. 3, July 1982, Pages 382-401.

1982年，Leslie Lamport等科学家提出了著名的拜占庭将军问题（Byzantine failures），其讨论的是允许存在少数节点作恶场景下的一致性达成问题。

假设拜占庭帝国的几支军队在敌人的城池外扎营，每支军队听命于自己的将军，这些将军之间只能通过信使传递消息。在对敌军进行侦察后，将军们必须制订一份共同行动计划。但是，有些将军可能是叛徒，这些叛徒会阻碍那些忠诚的将军达成共识。
这个问题的简洁描述：在已知有间谍的分布式军队中，将军们如何达成共识，执行共同的作战计划，来取得战争的胜利。



拜占庭将军问题难点

问题的难点

这个问题困扰了程序员们很多年，直到比特币区块链的出现。通常会遇到的问题有：

1. 这些将军离得很远，不能每遇到一个问题，就聚到一起开会商量对策
2. 这些将军中可能有少量叛徒，叛徒会乱发消息
3. 信使在传递消息时可能会把信弄丢
4. 信息可能会被敌国截获
5. 无法确认消息是否真的来自某位将军
6. 将军们在商量过程中可能会浪费很多天时间，贻误战机

拜占庭将军问题中并不去考虑通信兵是否会被截获或无法传达信息等问题，即消息传递的信道绝无问题。Lamport已经证明了在消息可能丢失的不可靠信道上试图通过消息传递的方式达到一致性是不可能的。所以，在研究拜占庭将军问题的时候，已经假定了信道是没有问题的.

拜占庭将军问题数学陈述和在互联网技术的意义

对于拜占庭问题来说，假如节点总数为N，叛变将军数为F，则当N≥3F+1时，问题才有解，由拜占庭容错算法BFT进行保证。

简单说一下论证过程：假设节点总数为N，作恶节点总数为F，有效的善良节点数为L1，无效（发生故障）的善良节点数为L2； 那么系统要安全的达成一致则必须满足2点：有效的善良节点超过作恶节点，同时也必须超过故障的善良节点。

转化为数学公式则：L1≥F+1，L1≥L2+1（L2=F），又L1=N-L2-F，即N-F-F≥F+1，得出N≥3F+1；因此当叛变者不超过1/3时，存在有效的拜占庭容错算法。但BFT一直存在复杂度过高的问题，并没有真正落地到实际场景中。

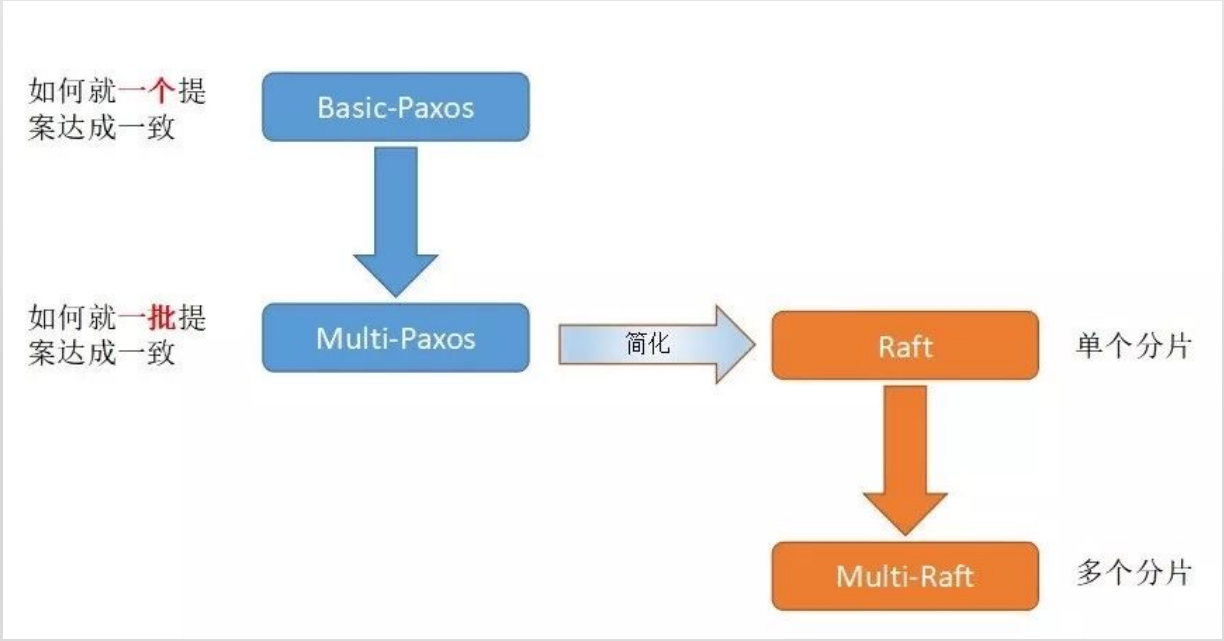
“拜占庭将军问题”延伸到互联网生活中来，其内涵可概括为：在互联网大背景下，当需要与不熟悉的对方进行价值交换活动时，人们如何才能防止不会被其中的恶意破坏者欺骗、迷惑从而作出错误的决策。进一步将“拜占庭将军问题”延伸到技术领域中来，其内涵可概括为：在缺少可信任的中央节点和可信任的通道的情况下，分布在网络中的各个节点应如何达成共识 。

一致性共识算法演进

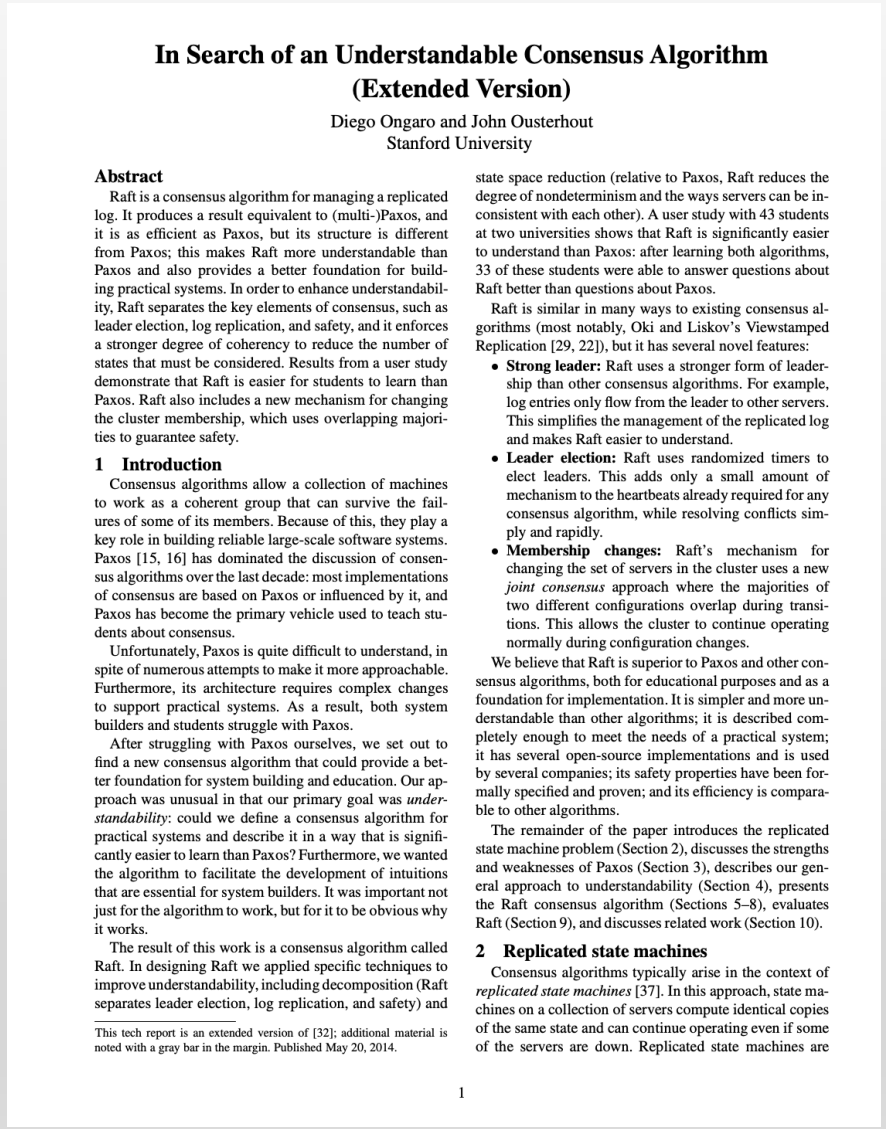
Paxos

1990年，Paxos算法是Leslie Lamport提出的一种基于消息传递的一致性算法。由于算法难以理解，起初并没有引起大家的重视。
1998年，Lamport将论文重新发表到TOCS上，即便如此Paxos算法还是没有得到重视。
2001年，Lamport用可读性比较强的叙述性语言给出算法描述。

06年Google发布了三篇论文，其中在Chubby锁服务使用Paxos作为Chubby Cell中的一致性算法，Paxos的人气从此一路狂飙。



Raft算法



In Search of an Understandable Consensus Algorithm (Extended Version)

Diego Ongaro and John Ousterhout
Stanford University

Abstract

Raft is a consensus algorithm for managing a replicated log. It produces a result equivalent to (multi-)Paxos, and it is as efficient as Paxos, but its structure is different from Paxos; this makes Raft more understandable than Paxos and also provides a better foundation for building practical systems. In order to enhance understandability, Raft separates the key elements of consensus, such as leader election, log replication, and safety, and it enforces a stronger degree of coherency to reduce the number of states that must be considered. Results from a user study demonstrate that Raft is easier for students to learn than Paxos. Raft also includes a new mechanism for changing the cluster membership, which uses overlapping majorities to guarantee safety.

1 Introduction

Consensus algorithms allow a collection of machines to work as a coherent group that can survive the failures of some of its members. Because of this, they play a key role in building reliable large-scale software systems. Paxos [15, 16] has dominated the discussion of consensus algorithms over the last decade: most implementations of consensus are based on Paxos or influenced by it, and Paxos has become the primary vehicle used to teach students about consensus.

Unfortunately, Paxos is quite difficult to understand, in spite of numerous attempts to make it more approachable. Furthermore, its architecture requires complex changes to support practical systems. As a result, both system builders and students struggle with Paxos.

After struggling with Paxos ourselves, we set out to find a new consensus algorithm that could provide a better foundation for system building and education. Our approach was unusual in that our primary goal was *understandability*: could we define a consensus algorithm for practical systems and describe it in a way that is significantly easier to learn than Paxos? Furthermore, we wanted the algorithm to facilitate the development of intuitions that are essential for system builders. It was important not just for the algorithm to work, but for it to be obvious why it works.

The result of this work is a consensus algorithm called Raft. In designing Raft we applied specific techniques to improve understandability, including decomposition (Raft separates leader election, log replication, and safety) and state space reduction (relative to Paxos, Raft reduces the degree of nondeterminism and the ways servers can be inconsistent with each other). A user study with 43 students at two universities shows that Raft is significantly easier to understand than Paxos: after learning both algorithms, 33 of these students were able to answer questions about Raft better than questions about Paxos.

Raft is similar in many ways to existing consensus algorithms (most notably, Oki and Liskov's Viewstamped Replication [29, 22]), but it has several novel features:

- **Strong leader:** Raft uses a stronger form of leadership than other consensus algorithms. For example, log entries only flow from the leader to other servers. This simplifies the management of the replicated log and makes Raft easier to understand.
- **Leader election:** Raft uses randomized timers to elect leaders. This adds only a small amount of mechanism to the heartbeats already required for any consensus algorithm, while resolving conflicts simply and rapidly.
- **Membership changes:** Raft's mechanism for changing the set of servers in the cluster uses a new *joint consensus* approach where the majorities of two different configurations overlap during transitions. This allows the cluster to continue operating normally during configuration changes.

We believe that Raft is superior to Paxos and other consensus algorithms, both for educational purposes and as a foundation for implementation. It is simpler and more understandable than other algorithms; it is described completely enough to meet the needs of a practical system; it has several open-source implementations and is used by several companies; its safety properties have been formally specified and proven; and its efficiency is comparable to other algorithms.

The remainder of the paper introduces the replicated state machine problem (Section 2), discusses the strengths and weaknesses of Paxos (Section 3), describes our general approach to understandability (Section 4), presents the Raft consensus algorithm (Sections 5–8), evaluates Raft (Section 9), and discusses related work (Section 10).

2 Replicated state machines

Consensus algorithms typically arise in the context of *replicated state machines* [37]. In this approach, state machines on a collection of servers compute identical copies of the same state and can continue operating even if some of the servers are down. Replicated state machines are

This tech report is an extended version of [32]; additional material is noted with a gray bar in the margin. Published May 20, 2014.

1

Raft算法的首要目标就是容易理解（UnderStandable），这从论文的标题就可以看出来。当然，Raft增强了可理解性，在性能、可靠性、可用性方面是不输于Paxos的。

Raft算法图形演示：

http://thesecretlivesofdata.com/raft/

VICTORLAMP 煤油灯科技

区块链中的共识机制

回顾拜占庭问题，简单分析一下核心问题：

1. 所有将军在任何时候都可以发起任何时间进攻的提议
2. 将军中存在叛徒，如果只有6人同意进攻，6人中有2人是叛徒，进攻是失败
3. 由于其中存在叛徒，消息传递过程中是否会被篡改进攻时间

区块链的解决方案

1. 把军队想像成计算机节点，把信使想像成计算机间的网络通讯，攻占敌军就是写入一个大家公认的区块记录。
2. 区块链技术在发送信息中加入了成本，降低了信息传递的速率，并采用了工作量证明（PoW），即一个节点必须经过大量尝试性计算才能得出一个结果，而其它节点只需极少的时间就能证明其真伪，这样能够减少垃圾消息、假消息在节点间传播的状况。
3. 挖矿节点把一段时间内的交易信息打包成一个区块，盖上时间戳，与上一个区块衔接在一起，每个区块都包含了上一个区块的索引（哈希值），然后再写入新的信息，从而形成新的区块，首尾相连，最终形成了区块链。
4. 用工作量证明、公钥加密等技术，使比特币网络从一个去中心化的不可信网络变为可信网络，使所有参与者可以在某些事情上达成一致，使价值传递成为了可能。

区块链中的共识机制

共识机制解决了区块链如何在分布式场景下达成一致性的问题，通过算法创造了信任，从而极大地减少当前金融系统及其他社会系统中的信用成本。小到菜市场的价格形成，大到一个国家的建立，这背后都意味着一种共识，所以共识本身就是价值，但价值大小要看其运用的场景。

那么区块链是如何解决这个问题的?

1. 首先区块链上主链是唯一的，争取出块的矿工可以有多个，但是最终记账的都是唯一一个矿工，不会存多个矿工同时记账的问题（也就是不会存在多个将军同时提起进攻的提议）。

2. 共识机制，每次出块都在链上进行广播，对该区块的合法性达成一致。并且每次提议的结果不需要所有节点马上跟上，节点选择最长的链进行链接即可。

3. 区块链采用了加密技术，对信息进行加密存储，并且每个区块都存储这条信息，除非有51%攻击，否则无法篡改。

PoW（Proof Of Work）：工作量证明机制

比特币的区块链网络在设计时，针对PBFT中同时存在多个提案和最终一致性确认的问题进行了创造性的改进，提出并采用了PoW算法。 PoW解决的是拜占庭下的共识，保证分布式账本的最终一致性，解决双花攻击；同时也建立和维护了一个分布式的时钟。

通过消耗大量能源来计算一个满足条件的Hash值来获得记账权（发起提案），某个节点成功找到满足条件的Hash值之后，会马上对全网进行广播打包区块，网络中的节点收到区块后，会立刻对其进行验证。如果验证通过，则表明已经有节点成功记账，自己就不再竞争当前的区块，而是选择接受这个区块，记录到自己的账本中，然后进行下一个区块的竞争记账。同时记账节点会得到代币奖励，进而吸引更多节点参与竞争。假如节点有任何的作弊行为，都会导致验证不通过，并直接丢弃其打包的区块，作弊的节点不但得不到奖励，还损失了巨大挖矿成本。这样全网节点始终维护着拥有最大工作量的链，从而保证整个账本的相对一致性。

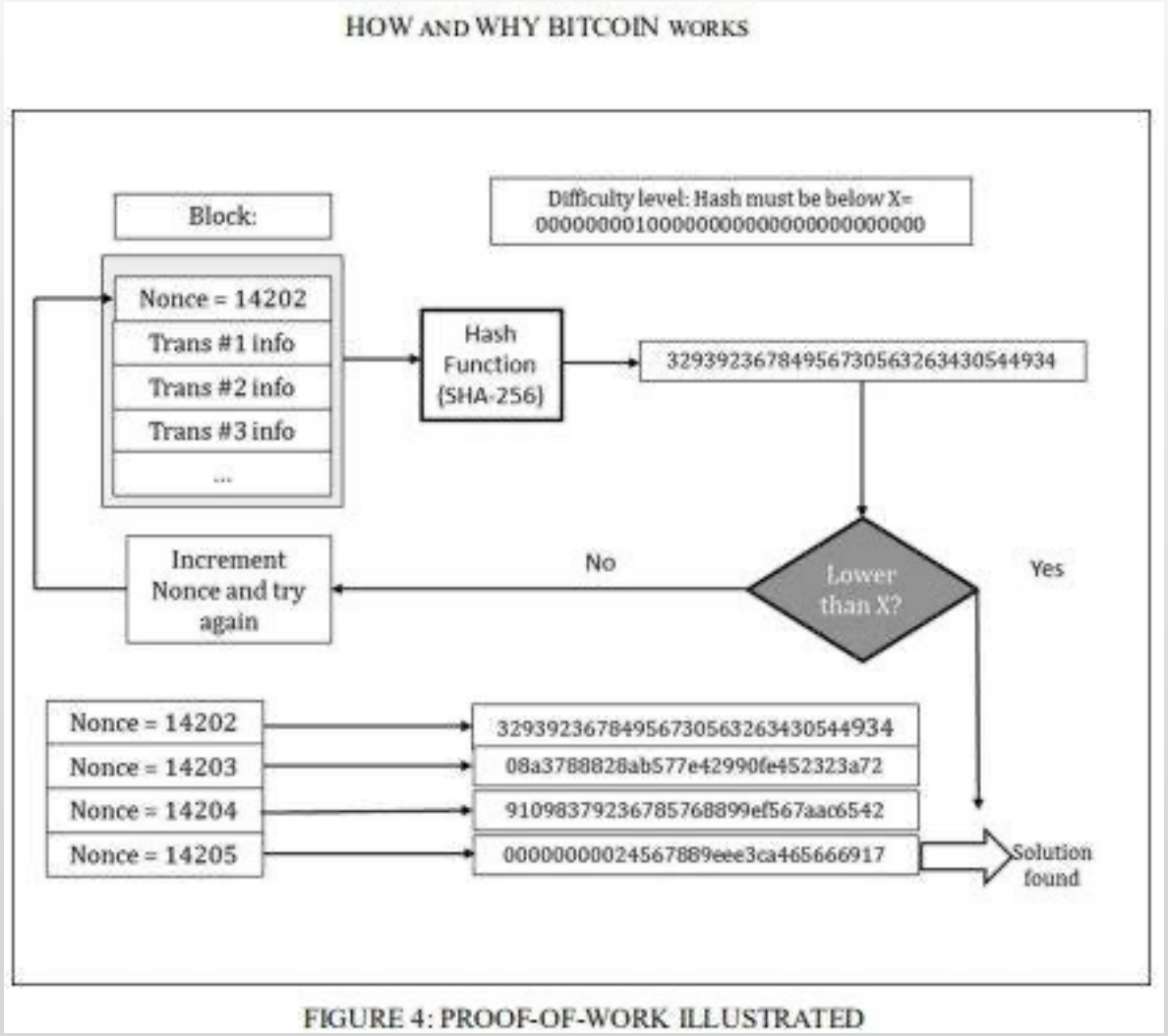


FIGURE 4: PROOF-OF-WORK ILLUSTRATED

PoW（Proof Of Work）：工作量证明机制优缺点

优点：

①完全去中心化，节点自由进出，避免了建立和维护中心化信用机构的成本。

②只要网络破坏者的算力不超过全网总算力的50%，网络的交易状态就能达成一致，并不可篡改历史记录。

③投入越多算力，获得记账权概率越大，越有可能产生新的区块奖励。

缺点：

①目前比特币挖矿造成大量的算力和能源浪费。

②挖矿的激励机制也造成挖矿算力的高度集中

③共识效率低下，结算周期长，每秒最多结算7笔交易。

PoS（Proof of Staked）：权益证明机制

权益证明机制，是针对工作量证明机制存在的不足而设计出来的一种改进型共识机制，其英文全称Proof of Stake，简称POS。与工作量证明机制要求节点不断进行哈希计算来验证交易有效性的机制不同，权益证明机制的原理是：要求用户证明自己拥有一定数量的数字货币的所有权，即“权益”。

PoS也称股权证明机制，其诞生的初衷是为了解决PoW带来的能耗问题。这种模式下持有币的数量越多、时间越长，记账成功率就越高（持有越多，获得越多），类似于利息制度。举例来说，PoS算法中有一个名词叫币龄，每个币每天产生1币龄，如果你持有100个币，总共持有了30天，那么此时你的币龄就为3000。这个时候如果你发现了一个PoS区块，你的币龄就会被清空为0。你每被清空365币龄，你将会从区块中获得0.05个币的奖励(相当于年利率5%)，在这个案例中，奖励 = 3000 * 5% / 365 = 0.41个币，即持币有利息。

PoS作为PoW的一种升级共识机制，根据每个节点所持有代币的数量和时间，等比例的降低挖矿难度，在一定程度上缩短了共识达成的时间，但最重要的是不再需要消耗大量能源进行挖矿。典型项目如以太坊，从PoW切换至PoS机制。

PoS（Proof of Staked）：权益证明机制优缺点

通过持有Token（代币）的数量和时长来决定你获得记账的机率，类似于股票的分红制度，持有股权越多的人就能够获得更多的分红。Token相当于区块链系统的权益。目前有很多数字资产用pow发行新币。

优点：

①降低了PoW机制的资源浪费。

②加快了运算速度，也可以理解为工作量证明的升级版

缺点：

①拥有币龄越长的节点获得记账权的几率越大，容易导致马太效应，富者越富，权益会越来越集中,从而失去公正性。

DPoS（Delegated Proof of Stake）：委托权益证明机制

DPoS是权益证明的一种改进版本，共识过程不再需要所有参与节点进行验证，而是委托部分代表来进行，很大程度上提高了共识效率。

BitShares社区首先提出了DPoS机制，并引入了见证人的概念。见证人可以生成区块（记账并获得奖励），每一个持有比特股的人都可以投票选举见证人。得票数前100名的候选者可以当选为见证人，见证人的候选名单每个维护周期更新一次。见证人通过随机排列后，依次轮流生成区块（限时2s内出块），若见证人在2s内未能出块，则自动跳到下一个见证人。由于持股人可以随时通过投票更换见证人，因此见证人为了获得奖励和避免损失保证金，就必须提供稳定高效的出块能力。

可以看出，DPoS实际上是对共识进行了分级，先通过投票选举达成见证人共识（选出极少数可信的见证人），然后见证人之间再达成交易验证共识，这样大大提高了整个系统的共识效率。从某种角度来看，DPoS与议会制度或人民代表大会制度有相似之处。如果代表不能履行他们的职责，例如未能按时出块，就会被网络选出的新见证节点所取代。

DPoS算法从性能和能耗的角度来说完全可以满足商用，但也不可避免地带来了过于中心化的问题。比如现在很火的EOS超级节点竞选就变成了鲸鱼们的合纵游戏，甚至被质疑是伪区块链项目。

DPoS（Delegated Proof of Stake）：委托权益证明机制优缺点

是基于POS衍生出的更专业的解决方案，类似于董事会投票，指拥有Token的人投票给固定的节点，选举若干代理人，由代理人负责验证和记账。不同于POW和POS的全网都可以参与记账竞争，DPOS的记账节点在一定时间段内是确定的。为了激励更多人参与竞选，系统会生成少量代币作为奖励。

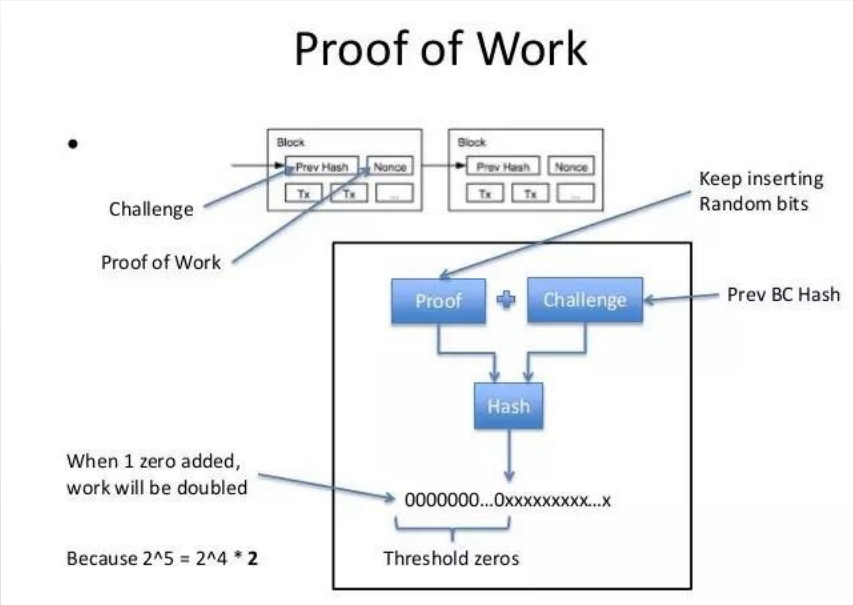
优点：

相较PoW， DPoS大幅提高区块链处理数据的能力，甚至可以实现秒到账，同时也大幅降低维护区块链网络安全的费用。

缺点：

去中心程度较弱，节点代理是人为选出的，公平性相比POS较低，依赖于代币的增发来维持代理节点的稳定性。

每隔10分钟左右，会新生成一个区块，**达成共识**，加入到这个链里面，永无止尽。区块链网络就是一个创建区块的“永动机”。



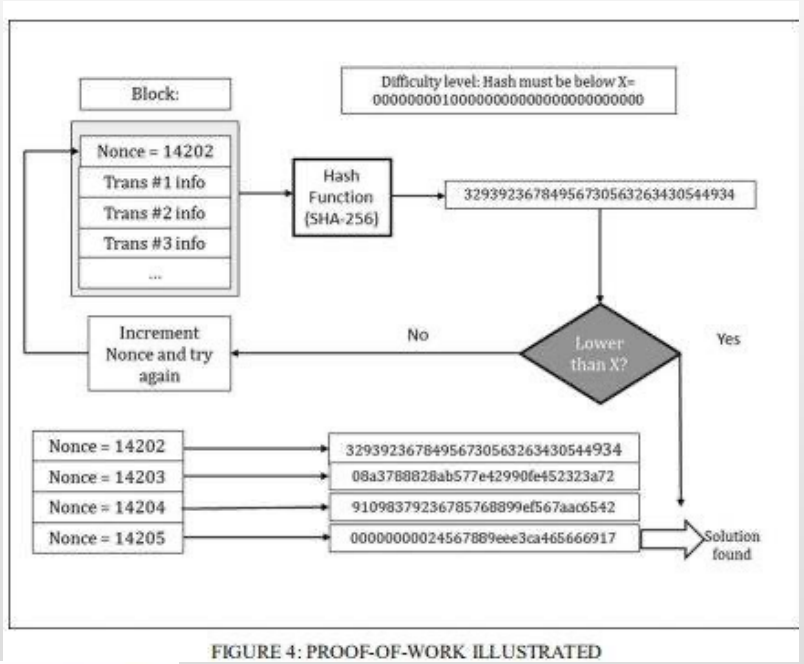
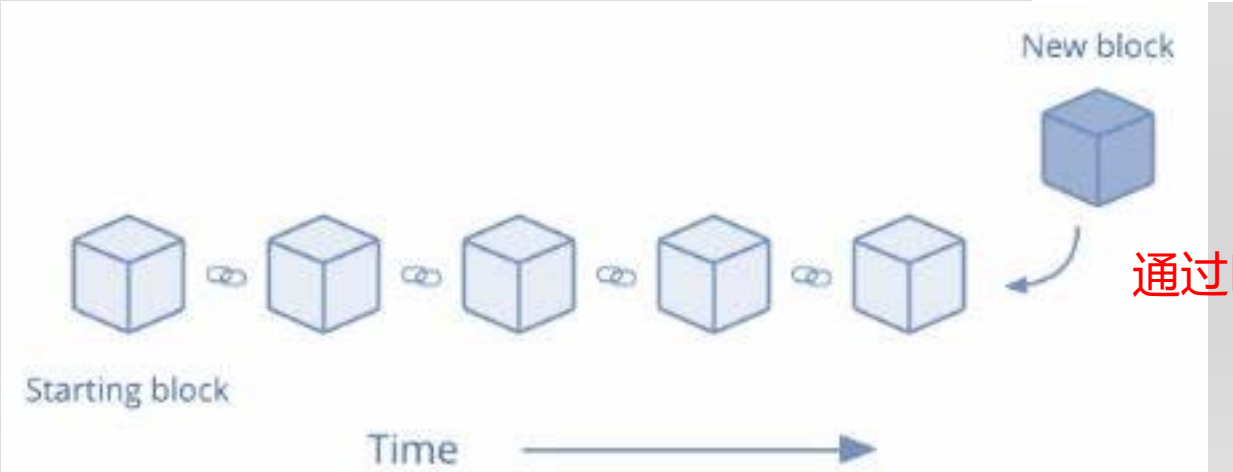


FIGURE 4: PROOF-OF-WORK ILLUSTRATED



通过PoW达成共识

谢 谢!

VICTORLAMP

Shenzhen VictorLamp Technologies CO. Ltd.
http://www.victorlamp.com

多媒体课程：《深入浅出区块链技术基础篇》

深入浅出区块链技术