

## 区块链共识算法及应用研究

王 群<sup>1,2</sup>, 李馥娟<sup>1+</sup>, 倪雪莉<sup>1,2</sup>, 夏玲玲<sup>1</sup>, 王振力<sup>1</sup>, 梁广俊<sup>1,2</sup>

1. 江苏警官学院 计算机信息与网络安全系, 南京 210031

2. 江苏省电子数据取证分析工程研究中心, 南京 210031

+ 通信作者 E-mail: lfj@jspi.edu.cn

**摘 要:**作为区块链核心技术的共识算法,为区块链的去中心化、开放自治、信息不可篡改、匿名溯源等功能的实现提供了机制支撑和保障,实现了分布式系统中强一致性和最终一致性的高效达成。以比特币出现为时间节点,将共识算法分为之前的经典分布式共识算法和之后的区块链共识算法,在此基础上根据算法的实现原理对共识算法又进一步分类,并选择其中的典型算法,重点从去中心化、可扩展性、安全性、一致性等方面进行了讨论。首先,提出了区块链共识算法的一般模型,给出了共识算法的基本定义。其次,在介绍经典分布式共识算法特点的同时,研究了两军问题、拜占庭将军问题、FLP不可能性定理、CAP定理和Paxos等分布式一致性算法及其改进,分析了算法的执行流程和功能特征。再次,对于区块链共识算法,根据实现原理和应用场景的不同,将其分为PoW共识算法、PoS共识算法、PoW+PoS混合共识算法和PoW/PoS+BFT/PBFT混合共识算法,在每一类中选择了具有代表性的算法后分别给出了算法流程,并结合具体应用场景进行了深入分析。最后,指出了区块链共识算法在性能与可扩展性、激励机制、安全与隐私、并行处理等方面的研究热点和发展方向。

**关键词:**区块链;共识算法;分布式系统;拜占庭容错

**文献标志码:**A **中图分类号:**TP309.7

## Survey on Blockchain Consensus Algorithms and Application

WANG Qun<sup>1,2</sup>, LI Fujuan<sup>1+</sup>, NI Xueli<sup>1,2</sup>, XIA Lingling<sup>1</sup>, WANG Zhenli<sup>1</sup>, LIANG Guangjun<sup>1,2</sup>

1. Department of Computer Information and Cybersecurity, Jiangsu Police Institute, Nanjing 210031, China

2. Jiangsu Electronic Data Forensics and Analysis Engineering Research Center, Nanjing 210031, China

**Abstract:** The consensus algorithm, as the core technology of blockchain, provides mechanism support and guarantee for the realization of functions such as decentralization, openness, autonomy, information tamperability and anonymous traceability, and realizes efficient achievement of strong and final consistency in distributed system. The consensus algorithms are divided into the previous classical distributed consensus algorithms and the subsequent blockchain consensus algorithms by taking the emergence of bitcoin as the time node. On this basis, the consensus algorithms are further classified according to the implementation principle, the typical algorithms are selected, and then the discussion is focused on decentralization, scalability, security, consistency and so on. Firstly, a general model of blockchain consensus algorithm is proposed, and the basic definition of consensus algorithm is given. Secondly,

**基金项目:**国家自然科学基金(61802155);江苏省高校自然科学研究重大项目(20KJA520004);江苏省高校优秀科技创新团队项目;公安技术、网络空间安全“十四五”江苏省重点学科项目;江苏省社会科学基金项目研究成果(21MLD012)。

This work was supported by the National Natural Science Foundation of China (61802155), the Natural Science Research Major Project of Jiangsu Province (20KJA520004), the Project of Excellent Scientific and Technological Innovation Team of Jiangsu Universities, the Key Disciplines Project of Jiangsu Province in the 14th Five-Year Plan: Public Security Technology and Cyberspace Security, and the Social Science Fund of Jiangsu Province (21MLD012).

**收稿日期:**2021-12-20 **修回日期:**2022-02-14

while introducing the characteristics of the classical distributed consensus algorithms, the distributed consistency algorithms and their improvements such as the two armed forces problem, the Byzantine generals problem, the FLP impossibility theorem, the CAP theorem and Paxos algorithm are studied, and then the execution process and functional characteristics of the algorithm are analyzed. Thirdly, the blockchain consensus algorithms are divided into POW consensus algorithm, POS consensus algorithm, POW+POS hybrid consensus algorithm and POW/POS+BFT/PBFT hybrid consensus algorithm according to different implementation principles and application scenarios. The algorithm flow is given respectively after selecting representative algorithms in each category, and then the specific application scenarios are deeply analyzed. Finally, the research hotspots and development directions of blockchain consensus algorithms in performance and scalability, incentive mechanism, security and privacy, parallel processing and so on are pointed out.

**Key words:** blockchain; consensus algorithms; distributed systems; Byzantine fault tolerance

共识是一个选择或意见集中与统一的问题,涉及社会生活、管理及计算机应用等众多领域。通俗地讲,共识过程类似于日常生活中的投票或举手表决,只是这里的表决环境变成了分布式系统,而“投票”或“举手”过程则是通过相应的算法隐性地进行的。

分布式系统出现后,首先需要解决的一个根本问题就是如何实现集群内部不同节点上数据的统一性和不同节点间针对特定操作状态(如“提交”或“回滚”)的一致性,即通过共识机制使分布式节点达成一致性或取得稳定的状态。由于早期的分布式系统应用主要集中于节点数量有限、运行环境相对可信的分布式数据库系统,系统中不存在恶意篡改或伪造数据的攻击节点,因此针对传统分布式系统中共识算法的研究一般不考虑拜占庭容错问题。与之不同,区块链系统在一个复杂和缺乏信任机制的开放互联网环境(尤其是公有链)中通过共识算法建立了节点之间的信任,最终实现了不同节点上账本的一致性和数据的安全性<sup>[1]</sup>,因此,区块链被称为“信任机器”<sup>[2]</sup>。由此可见,共识算法是区块链技术中需要重点研究和掌握的关键技术。

通常情况下,区块链系统中的节点首先是一个互联网接入节点,其接入配置与普通联网计算机没有区别。但作为一个构建在互联网架构和平台上的自治系统,区块链系统在继承了传统互联网开放性、匿名性以及节点之间松耦合关系等基本特性的同时,其内部采用点对点(peer-to-peer, P2P)方式组织节点之间的区块传输、验证与记账。P2P网络中,所有节点均处于相同的逻辑位置,每个节点在从其他节点获取服务的同时,也为其他节点提供路由选择与维护、区块传输与验证、新节点发现等服务。在P2P网络中,每个节点都维护着一个存放区块链账本数

据的共享文件夹<sup>[3]</sup>,共识算法的主要功能便是维护不同节点共享文件夹中账本数据的一致性和正确性。

在传统网络中,像微信、支付宝等在线支付平台的可靠性都由业务系统的提供者决定,个人身份信息、交易记录等信息的安全性也取决于第三方机构的信誉和安全技术服务能力。这种中心化的业务处理和管理模式在区块链系统中被打破。随着2008年11月1日中本聪在其论文“Bitcoin: a peer-to-peer electronic cash system”<sup>[4]</sup>中对比特币原理的简要阐述,以及2009年1月3日比特币系统的正式上线及创世区块(genesis block)的问世,比特币系统在随之不温不火地发展了将近10年后,人们才发现通过区块链这一底层技术,可以在一个完全去中心化的分布式环境中实现与传统中心化环境下相同的业务功能,而完成该功能的关键便是共识算法。与传统的业务处理系统相比,区块链具有去中心化、数据不可篡改、集体维护、高度透明、安全可信等特点,区块链技术以密码学和P2P网络为基础,将特定结构的交易数据(如比特币)或业务数据按照约定方式组织成区块,然后通过选定的共识算法将新区块添加到主链上,并利用密码学技术保证数据的安全性和不可伪造性。因此,从本质上讲,区块链就是一种不可篡改的分布式记账方法,在整个网络中共识算法每个节点都在维护着唯一的账本。为此,共识算法是区块链系统架构的核心。

根据是否严格维护去中心化这一机制的不同,可将区块链分为完全去中心化的“公有链”(public blockchain)和中心化的“私有链”(private blockchain)两类,另外在公有链和私有链之间还存在着部分去中心化的“联盟链”(consortium blockchain);如果按照准入机制的不同,可以将区块链系统分为无需对

节点身份进行认证的非许可链(permissionless blockchain)和需要对节点的准入身份进行认证的许可链(permissioned blockchain)。通常情况下,公有链属于非许可链,而私有链和联盟链属于许可链。联盟链未来的发展更趋向于私有链,因此在部署架构中一般可将联盟链作为私有链的一种类型。共识算法因区块链应用场景的不同而有所不同,公有链环境中的共识算法一般需要支持良好的可扩展性,需要在参与共识的节点随时变化(“加入”或“离开”)的情况下达成最终的共识,而且还要防止可能存在的拜占庭节点对系统的攻击。同时,受限于FLP(Michael Fisher、Nancy Lynch and Michael Paterson)定理、CAP(consistency、availability and partition-tolerant)定理和Paxos算法等分布式一致性算法对运行条件的严格限定,私有链中的共识算法一般很难保证强一致性;在私有链环境中,由于对接入节点的身份有严格的验证和控制机制(如目前广泛使用的基于公钥基础设施(public key infrastructure, PKI)机制的数字证书技术实现节点认证等),有效保证了节点的可信性,私有链中的共识算法一般是一种高效的强一致性算法,但算法的可扩展性和有效防范拜占庭节点攻击的能力有所减弱。

随着区块链应用领域的不断扩展和对区块链技术研究的不断推进,国内外研究者分别对共识算法从不同角度进行了梳理。在国内研究方面,文献[5]较为系统地整理了传统分布式系统和区块链系统中的共识算法,并选择不同的发展主线分析了共识算法的新进展;文献[6]在对区块链共识算法进行总结的基础上,根据不同的共识机制就算法实现、应用和发展进行了较为系统的介绍;文献[7]重点从出块节点选举和主链共识两方面系统分析了共识协议的实现,并提出了存在的问题及解决方案;文献[8]在调研并分析了代表性共识算法及其演进历程的基础上,提出了共识算法的分类模型,并对部分重要的共识算法进行了分析。在国外研究方面,文献[9]重点对区块链中基于证明的共识协议进行了讨论,并对工作量证明和权益证明进行了对比分析;文献[10]从应用开发的角度,重点对区块链分叉和一致性算法进行了讨论;文献[11]在介绍了分布式账本技术(distributed ledger technology, DLT)及其变种基本原理的基础上,对130种共识算法进行了对比分析,然后对分布式账本技术的发展前景进行了展望。

本文在广泛收集国内外各类文献资料以及对区

块链技术及其主要应用进行深入分析的基础上,归纳整理了共识算法中涉及到的主要概念。同时,以比特币的出现和应用作为一个大体的分界点,将共识算法分为分布式系统经典共识算法和区块链共识算法两类,其中对经典共识算法的介绍主要针对传统分布式系统工作原理和特点来展开,而对区块链共识算法的分析是在对现有共识算法和系统进行了梳理的基础上,认为拜占庭容错(Byzantine fault tolerance, BFT)、实用拜占庭容错(practical Byzantine fault tolerance, PBFT)、工作量证明(proof of work, PoW)和权益证明(proof of stake, PoS)是各类区块链共识算法的核心,而其他大量的共识算法是其功能扩展和组合。基于这一事实和基本思想,本文将主流区块链共识算法分为PoW、PoS、PoW+PoS、PoW/PoS+BFT/PBFT共四类进行讨论。

## 1 共识算法概述

在计算机学科领域,算法是为了实现某一目的而设计的一系列解决具体问题的清晰指令,协议是为了使通信双方或多方达到某一目的而规定的一组规则。协议用于解决具体问题,而算法是对协议的具体实现,体系结构和协议奠定了网络的主体架构。为此,本文在讨论共识算法时将其置于具体的网络环境 and 应用场景中。

### 1.1 区块链共识模型

早期的共识算法主要是在大多不考虑拜占庭容错问题的分布式系统中研究数据库操作的一致性。在这类分布式系统中,一般不考虑节点故意伪造、篡改数据等问题,认为系统节点只会出现宕机或网络故障等非人为攻击现象。与传统分布式环境相比,区块链共识算法是在网络结构复杂、体系架构各异以及节点之间缺乏信任的开放网络环境中实现操作和数据的一致性和正确性,在共识算法的实现过程中,多数情况下参与节点较多,而且存在恶意的拜占庭节点<sup>[12]</sup>。另外,传统的分布式系统一般工作在同步环境,节点之间达成一致性的难度较小,算法实现也相对简单,算法运行的效率较高。而区块链共识算法主要运行在基于分组交换的互联网环境中,不但要解决分组传输的延时、重复、丢失和出错等复杂问题,还要能够根据大多数应用场景的要求,在效率、可靠性和安全性等方面得到保障。因此,区块链共识算法设计时需要面对的环境更为复杂,需要面对的困难和解决的问题更多。

区块链技术的核心内容主要包括分布式数据库、密码学、共识算法和P2P网络,其中分布式数据库负责数据的写入、读取与存储,密码学中的非对称密钥加密机制和哈希(Hash)算法实现用户身份认证、地址标识、完整性验证等功能,共识算法实现交易信息在分布式账本中的一致性和安全性,P2P网络提供了区块链不同节点间的通信保障。在图1所示的区块链共识模型中,以比特币系统为例,结合区块链核心技术的应用,描述了一个交易从生成到最后被打包确认的5个过程:

(1)生成交易。交易信息由客户端(完整客户端、轻量级客户端或在线交易平台)生成,包括交易的输入金额、输入账户及输出账户的地址等信息。未经签名的“原始交易”(raw transaction)无法被矿机接收,只有在进行了用户签名后,该交易才会被矿机通过共识机制打包进新区块。

(2)交易签名。用户在客户端用自己的私钥对交易进行签名,形成合法的交易数据。

(3)交易广播。将签名后的交易数据通过调用本地应用程序接口(application programming interface, API)或以P2P节点方式广播到网络中。

(4)挖矿。挖矿即对交易数据的打包和确认。首先根据区块的数据结构,将交易打包进区块,再采用Merkle Tree算法生成区块体中所有交易的Merkle Root(哈希值),然后根据区块头部的元数据,采用相应的共识算法(如PoW、PoS等)获取新区块的记账权。

(5)全网节点写入。矿工将生成的新区块广播到区块链网络中,任意一个节点在接收到新区块后,对其进行验证,验证通过后将该区块链接到本地区块链上。同时,不同的区块链系统还将采用相应的算法防止分叉的产生,维护主链的唯一性和稳定性。

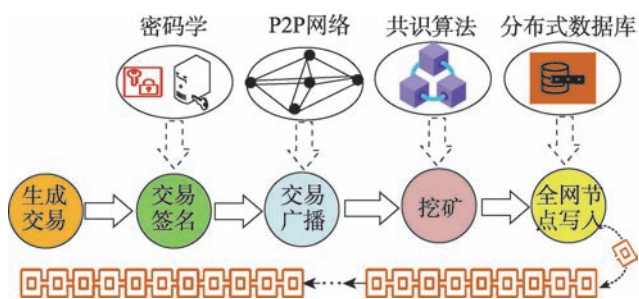


图1 区块链共识基础模型

Fig.1 Blockchain consensus base model

## 1.2 共识算法中的基本定义

针对区块链共识算法的研究涉及大量技术、经

济和社会等众多层面的问题,为便于对区块链共识算法的理解,下面给出一些基本定义<sup>[13-15]</sup>。

**定义1(一致性(agreement))** 所有的非缺陷进程都必须同意接受某一个值。

在区块链网络中,一致性存在强一致性(strict consistency)和弱一致性(weak consistency)两种类型。其中,强一致性是指任意时刻所有节点中的数据是相同的,而弱一致性(也称为最终一致性)是指不保证任意时刻任意节点上的同一份数据都是相同的,但最终会趋于相同。在弱一致性区块链系统中,并不是每一个合法挖矿者挖出的区块最终都会连接到主链,因此不可避免会存在分叉现象(如比特币(bitcoin)<sup>[16]</sup>、以太坊(ethereum)<sup>[17]</sup>等),而在强一致性区块链系统中,由于区块的生成由每一轮指定的节点负责(如Hyperledger<sup>[18]</sup>、Solida<sup>[19]</sup>等),不会出现分叉现象。

**定义2(正确性(validity))** 针对所有的非缺陷进程,其同意接受的值必须是非故障进程提议的值。

**定义3(可终止性(termination))** 对于每一个非缺陷进程来说,必须都有一个最终确定的值。

一致性、正确性和可终止性称为共识问题的三个属性,同时满足一致性和正确性时称为安全性(safety),满足可终止性时称为活性(liveness)。

**定义4(共识(consensus))** 就某种状态或数据达成一致性的过程。

一致性一般是指分布式系统中数据及其副本对外呈现的状态,描述的是多个节点对数据状态的维护能力,而共识描述的是分布式系统中多个节点间彼此对某个状态达成一致结果的过程。一致性描述的是状态的结果,而共识则是一种手段,达成某种共识并不意味着取得了完全一致。

**定义5(拜占庭缺陷(Byzantine fault))** 从不同角度表现出不同现象的缺陷。

**定义6(拜占庭故障(Byzantine failure))** 由拜占庭缺陷引起的故障。

**定义7(宕机缺陷(fail-stop fault))** 因系统进程停止运行导致的缺陷。

**定义8(宕机恢复故障(fail-recovery failure))** 因系统进程停止运行导致的故障,该故障在重启进程后恢复。

由定义5至定义8可知,并不是所有的缺陷或故障都是拜占庭的,像宕机缺陷或故障为非拜占庭缺陷或故障,而由人为破坏、恶意代码攻击等不可预测

的行为引起的缺陷或故障是拜占庭的。

**定义9**(通信模型(communication model)) 分组通信网络是一类串行通信网络,根据节点之间消息(分组)传输方式的不同,可以将区块链网络分为同步网络、异步网络和部分同步网络三种类型。其中,同步网络是指非拜占庭收发节点在时钟频率上保持一致,异步网络是指非拜占庭收发节点在时钟频率上不需要保持一致,而部分同步网络是指网络中的部分(而非全部)非拜占庭节点在时钟频率上保持相对一致。

通信模型在很大程度上决定着共识算法的设计与实现。在同步网络中,在一个消息到达节点之前,已经确认前一条消息成功接收;在异步网络中,由于消息的传输可能会产生延时、乱序等现象,无法保障消息会按照发送时的顺序有序到达;而在部分同步网络中,消息能够在系统限定的时间内到达所有非拜占庭节点,但前后顺序无法得到保障。

## 2 经典分布式共识算法

共识算法的发展是随着计算机应用技术及应用环境的变化而同步演进的。与目前复杂网络环境下的主流区块链共识算法相比,早期的共识算法一般不考虑作恶节点的存在,认为系统中只会存在服务器宕机或网络故障等非人为攻击现象,即大多数情况下不考虑拜占庭容错问题,因此,早期的共识算法也称为分布式一致性算法<sup>[20]</sup>。但需要说明的是,回顾经典的分布式一致性算法并不是说这些算法已经成为束之高阁的历史,而是通过系统的梳理和研究,为现代区块链共识算法的设计思想理清思路,为区块链共识算法的发展趋势把准方向。另外,像目前联盟链中使用的实用拜占庭容错(PBFT)和比特币的工作量证明(PoW)等典型共识算法均出现在区块链概念提出之前,最初也是为了解决分布式系统的一致性问题而提出,“新瓶装旧酒”也是一种应用创新。

### 2.1 两军问题

1975年,Akkoyunlu等人<sup>[21]</sup>首次提出了网络通信领域的“两军问题”(也有学者称之为“两军悖论”或“协同攻击问题”),指出在一条不可靠的通信链路上试图通过通信方式达成一致性是不可能的,成为计算机通信领域首个被证明无解的问题。研究两军问题,需要同时建立在以下两个条件的基础上:

**条件1** 通信信道是不可靠的(与拜占庭将军问题不同)。

**条件2** 不存在叛徒问题,即信使是可靠的。

两军问题是一个在信道存在干扰条件下的两节点通信问题,也是一个日常生活问题。如图2所示,分隔在“白军”两侧的总数大于白军人数的两支“蓝军”如果要获胜,必须同时对白军发起攻击。那么,是否存在一种方式(协议)能够使蓝军获胜呢?显然这是一个无解的问题。



图2 两军问题示意图

Fig.2 Two-army problem pattern

两军问题是研究分布式系统一致性的一个重要理论,对现代计算机网络技术的发展产生了重要影响,对区块链共识算法的研究具有重要意义。例如,在TCP/IP体系中,TCP协议为了解决两个节点之间通信的可靠性问题,就采用了基于“三次握手”建立连接和“四次握手”断开连接的机制,为解决两军问题不存在理论提供了一个简易可行的应用范式。

### 2.2 拜占庭将军问题

1982年,由Lamport等人提出了拜占庭将军问题(Byzantine generals problem)<sup>[22]</sup>,其核心思想是在明知军中存在叛徒的情况下,要保证行动(进攻或撤退)的一致性。随着区块链技术的出现和兴起,这一问题作为研究分布式系统中一致性机制的经典理论开始重入大众视野,并直接影响着区块链共识算法的设计与实现。

#### 2.2.1 拜占庭将军问题提出的条件

拜占庭将军问题的提出,建立在同时满足以下3个条件(条件3至条件5)的基础上。

**条件3** 通信信道的可靠性。拜占庭将军问题的研究建立在信使不会被敌方捕获这一前提下,即信使肯定会将消息传递给盟军中的其他将军(即信使是可靠的),但传递的消息是否真实并无限制。用在分布式系统中,拜占庭将军问题认为传递消息的网络信道是绝对可靠的,在此前提下再进行一致性和容错性的研究。

条件3可通过口头协议来实现:(1)每条发送的消息都能够被正确传递;(2)消息的接收方能够知道该条消息的发送方;(3)当消息没有发送时可以被检测到。

**条件4** 所有忠诚的将军能够最终达成行动的一

致性(进攻或撤退)。Lamport 等人对拜占庭将军问题的研究表明,如果在  $n$  个将军中存在  $m$  个叛徒,当  $m \geq \frac{1}{3}n$  时,拜占庭将军问题无解,否则将军们能够做出正确决定。图 3 所示的是最简单的有关拜占庭将军问题的论断,说明在 3 个将军中只要存在 1 个叛徒最后将无法达成一致(满足  $m \geq \frac{1}{3}n$  条件)。

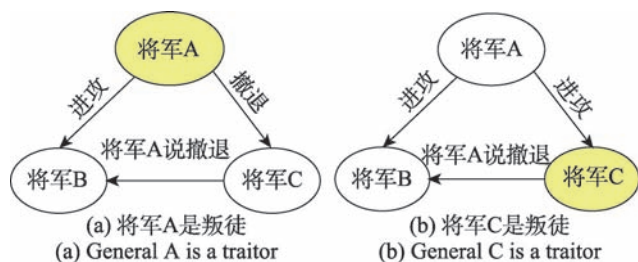


图 3 三个将军中一个是叛徒

Fig.3 One of three generals is a traitor

如果条件 4 成立,还需要满足另两个条件:每个忠诚的将军必须收到相同的命令值  $v(1), v(2), \dots, v(n)$ , 其中  $v(i)$  ( $1 \leq i \leq n$ ) 表示第  $i$  个将军传递给其他将军的消息,即所有忠诚的将军在收到消息后会遵守相同的命令;如果发送消息的第  $i$  个将军是忠诚的,那么所有忠诚的将军接收到的  $v(i)$  值是相同的。

**条件 5** 少数的叛徒无法使忠诚的将军做出错误的决定。假设  $v(i)$  是第  $i$  个将军发送给其他将军的消息,每个将军会根据收到的所有消息  $v(1), v(2), \dots, v(n)$  做出决定。这一条件在分布式系统中可以描述为每个节点在接收到消息后都会传递给其他节点,通过递归方式,直到最后每个节点中都存在其他所有节点发送的消息。最后,所有节点按照少数服从多数的原则达成行动上的一致。

需要说明的是,通过条件 5 虽然可以最终实现运行上的一致性,却无法知道谁是叛徒,即无法进行溯源。为了解决这一问题,在分布式系统中可以采用群签名、环签名、盲签名等数字签名技术,通过非安全通道实现对消息或文档真实性的认证。

## 2.2.2 拜占庭容错

共识算法是区块链技术的核心,拜占庭容错是研究区块链共识算法的关键。传统的独立磁盘冗余阵列(redundant arrays of independent disks, RAID)、帧检验和超时重传等分布式系统中使用的容错机制,一般只能解决局部或特定条件下的单一问题,无法适应分布式系统中可能存在的因人为攻击、安全漏洞、软件缺陷等综合问题引起的复杂错误。因此,

工程实践中需要一种能够容忍多种形式的软件错误和安全漏洞的机制,这一机制或方案称为拜占庭容错(BFT)技术<sup>[23]</sup>。

拜占庭将军问题类似于分布式系统。其中,服务器类比为将军,服务器之间的消息传递类比为在将军之间传递消息的信使。服务器可能会受到攻击而向其他服务器发送错误信息,发送错误信息的服务器扮演着叛徒的角色。因此,可以将拜占庭容错系统理解为一个由多台服务器组成的分布式系统中,对于每一个请求必须满足以下两个条件(条件 6 和条件 7)的系统:

**条件 6** 所有非拜占庭服务器(没有出现故障的服务器)使用完全相同的输入信息,产生完全一致的输出结果。

**条件 7** 当输入的信息正确时,所有非拜占庭服务器都要接受该信息,并计算相应的结果。

与此同时,对于一个已投入运行的拜占庭容错系统来说,所有拜占庭服务器(出现缺陷或故障的服务器)需要满足安全性和活性两个指标:

(1)安全性:对于任何已经完成的请求都不允许更改,而且对之后的请求是可见的。

(2)活性:对于非拜占庭客户端发出的请求,能够无条件地接受并执行。

根据节点(服务器或客户端)的容错类型,可以将分布式系统的共识机制分为拜占庭容错和非拜占庭容错两类,其中非拜占庭容错(如 Paxos<sup>[24]</sup>、views-tamped replication<sup>[25]</sup>、Raft<sup>[26]</sup>等算法)主要用于联盟链和私有链,而公有链主要使用拜占庭容错共识算法。

## 2.2.3 实用拜占庭容错

拜占庭容错系统能够容忍几乎所有形式的错误类型,已经成为解决分布式系统容错问题的一种通用方案。但早期的拜占庭容错协议在实现时过于强调算法的理论性而忽视了实用性,并且算法的复杂度随着系统中节点数的增加而呈指数级上升,另外还需要配置时钟同步机制,限制了其应用。实用拜占庭容错(PBFT)共识算法<sup>[27]</sup>能够实现对一定数量的拜占庭节点进行容错,在异步分布式系统中提供安全性和活性,并通过对 BFT 算法的改进,将系统的复杂度从指数级降到了多项式级别,使得算法在实际应用中变得可行。在区块链应用场景中, PBFT 共识算法适合于对一致性要求较高的私有链和联盟链。例如,在 IBM 主导的超级账本(hyperledger)项目中, PBFT 便是一个可选的共识协议。

根据协议实现时关注点的不同,目前的拜占庭系统主要分为状态机拜占庭系统和 Quorum 拜占庭系统两类<sup>[28-30]</sup>。前者强调系统中同一时刻只允许有一个请求被执行,这就使得服务器之间执行请求的顺序要求完全一致;后者对请求的执行顺序没有严格要求,但对响应时间极为敏感。PBFT 属于状态机拜占庭系统,即整个系统维护同一个状态,所有服务器采取的运行完全一致,具体通过一致性(agreement)、检查点(check point)和视图更换(view change) 3 个协议来实现。其中,一致性协议的功能是实现客户端发出的请求在每个服务器上都能够按照确定的顺序执行;检查点协议的功能是定期清理日志以节约资源,同时将系统中的服务器同步到某一个相同的状态;视图更换协议的作用是在节点出错时进行替换,并确保被非拜占庭服务器执行的请求不会受到影响。系统在正常运行时一般只执行一致性和检查点两个协议,只有在系统出现错误(如节点宕机或运行缓慢)时视图更换协议才开始启用。

因为本小节研究的主要内容是分布式系统的共识问题,所以在这里更关注 PBFT 的一致性协议。一致性协议约定来自客户端(client)的每一个请求按照确定的顺序在每个服务器上执行,每个服务器在相同的配置信息下工作(该配置信息称为“视图”),服务器分为主节点(primary)和副本节点(replica)两类,其中主节点仅有一个,它负责对客户端的请求进行排序,副本节点按照主节点提供的顺序执行请求。

一致性协议至少包括请求(propose)、预准备(pre-prepare)、准备(prepare)、确认(commit)和回复(reply)5 个阶段。图 4 是一个简化的 PBFT 协议运行示意图,其中副本节点 replica 3 为故障节点(标×),箭头表示消息的发送过程。整个协议的执行过程如下:

(1)请求(propose)。客户端向主节点发送请求,该请求消息  $m = [op, t, c-id, c-sig]$ ,其中  $op$  代表执行的操作,  $t$  为时间戳,  $c-id$  为客户端标识,  $c-sig$  为客户端数字签名。签名的目的是对客户端进行认证和授权,以适应联盟链或私有链对接入节点的管理要求。

(2)预准备(pre-prepare)。当主节点接收到请求消息后,给该请求分配一个序列号  $sn$ ,并构造一个预准备消息  $pre-prepare = [p-p, vn, sn, D(m), p-sig, m]$ ,之后将该消息发送给系统中的所有副本节点,其中  $p-p$  代表 pre-prepare 消息,  $vn$  为视图号,  $D(m)$  是客户端消息的摘要值(Hash 值),  $p-sig$  为主节点的签名。序号规定了服务器执行命令的顺序,视图号使副本节点知

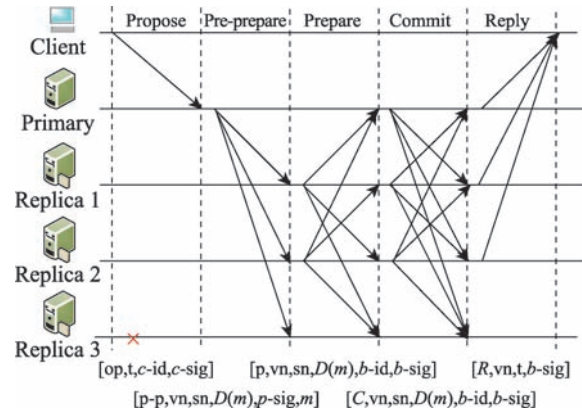


图4 PBFT协议运行示意图

Fig.4 PBFT protocol communication pattern

道当前视图,主节点签名是使副本节点知道谁是当前系统中唯一的主节点,消息摘要值防止消息在传输过程中被篡改。

(3)准备(prepare)。所有副本节点在接收到 pre-prepare 消息后,构成一个准备消息  $prepare = [p, vn, sn, D(m), b-id, b-sig]$ ,然后将其发送给其他服务器节点,其中  $p$  表示 prepare 消息,  $b-id$  为副本节点的标识,  $b-sig$  为副本节点的数字签名。

(4)确认(commit)。服务器节点在收到  $2f+1$  个 prepare 消息(系统中总共有  $3f+1$  个服务器节点,其中故障节点有  $f$  个)后,对视图内的请求和执行顺序进行验证,验证通过后执行客户端请求,并构成一个确认消息  $commit = [C, vn, sn, D(m), b-id, b-sig]$ 后发送给其他服务器节点,其中  $C$  表示 commit 消息。

(5)回复(reply)。当服务器节点收到  $2f+1$  个 commit 消息,构成一个回复消息  $reply = [R, vn, t, b-sig]$ 后发送给客户端,其中  $R$  表示回复消息。客户端等待所有服务器节点的回复消息,如果接收到  $2f+1$  个相同的回复,则该回复即为运算结果。

在异步系统中, PBFT 算法能够在拜占庭节点数少于  $1/3$  的情况下达成最终的一致性。同时,在 PBFT 中,消息发送者的身份都以数字签名方式经过了认证,确保了节点的可信性和消息来源的可追溯能力。另外, PBFT 提供了超时机制,从而保证了安全性和活性。

## 2.3 FLP 不可能性定理

1985 年, Fischer、Lynch 和 Paterson 三位科学家在文献[31]中提出:在一个多进程异步系统中,只要存在一个不可靠的进程,就不存在在有限时间内使所有进行达成一致的协议。按照三位科学家姓氏的首

字母,该定理被称为FLP不可能性定理。

### 2.3.1 FLP不可能性定理成立的条件

任何一个分布式算法都需要对系统场景进行假设或设置条件,从而形成系统模型,FLP不可能性定理建立在以下4个条件(条件8至条件11)的基础上<sup>[31-32]</sup>:

**条件8** 异步通信。与相对可靠的同步通信相比,异步通信的最大区别是没有时钟,无法实现时间同步,未使用超时机制,无法回溯失败原因,信息的延迟可能非常大,并且消息可能存在乱序。

**条件9** 通信健壮。只要进程没有失败,消息虽然会被无限延迟,但最终会到达接收端,并且消息只会被送达一次(不能重复)。

**条件10** 失败-停止模型。进程失败后不再处理任何消息。

**条件11** 失败进程数限制。系统中最多只能有一个进程失败。

### 2.3.2 FLP不可能性定理的基础理论

根据FLP不可能性定理的定义,一个共识协议 $P$ 针对一个具有 $n(n \geq 2)$ 个进程 $p$ 的异步通信系统,不同进程 $p$ 之间的通信通过相互发送消息进行;共识系统的配置(configuration)由进程的內部状态与消息缓存组成,如果在系统操作之后,没有进程做出决定(“提交”或“回滚”),则称为不确定的配置;相反,如果某个配置能够精确地做出决定,则称为确定性的配置。如果某个配置是确定的,则认为一致性可以达成。在此过程中,把从一个配置状态转换到另一个配置状态的过程称为一个步骤(step),每个步骤由单一进程的一次执行完成。对于FLP不确定性定理的理解,关键在于“确定性”(deterministic)和“不确定性”(un-deterministic)两个词应用场景的理解,即在共识协议 $P$ 中,尽管每个进程 $p$ 的状态转换函数是确定性的,但触发状态转换函数的消息接收和处理过程是不确定性的,因此,在一个存在故障进程的异步通信系统中,不可能在有限时间内达成共识状态。

FLP不可能性定理是众多共识算法中一个非常经典的基础性定理。对于FLP不可能性定理的理解和应用需要建立在相应假设及限制条件的系统模型上,在FLP不可能性定理假设条件中,因为没有对进程接收消息、处理消息和回复消息的时间进行限制,也就是说进程处理速度不确定,消息的延迟也可能很大,因此在判断一个进程问题时不知道是因宕机引起的进程故障还是因消息的处理时间太长而造成的延时所致。同时,FLP不可能性定理中所指的故障

并不是拜占庭故障,而仅仅是导致进程停止运行的宕机故障。

在异步系统中,当一个进程出现故障或响应时间延迟时,不存在一个分布式算法能够让所有的非故障进程达成一致共识。为规避FLP不可能性定理,只能通过对某些条件进行必要的取舍或调整来优化问题模型。例如,因为存在FLP不可能性定理的限制,许多区块链项目的共识算法一般都认为大部分节点是诚实的,并且满足一定的同步性,其中PoW算法认为至少51%的节点是诚实的,并且有一定的同步性,PoS和PBFT算法也认为2/3及以上的节点是诚实的。

## 2.4 CAP定理

在2000年的分布式计算原则大会(Symposium on Principles of Distributed Computing 2000)上,计算机科学家Brewer在主旨演讲中针对分布式环境下数据库的一致性(consistency)、可用性(availability)和分区容错性(partition-tolerant)提出了猜想。2002年,该猜想得到了来自麻省理工学院的两位教授Lynch和Gilbert的证明,被称为CAP定理<sup>[33-34]</sup>。

### 2.4.1 CAP定理概述

CAP定理是分布式系统设计中基础和最关键的理论之一。CAP定理提出:在一个分布式数据存储系统中,最多只能同时满足一致性、可用性和分区容错性3个属性中的2个属性(每个属性即一个条件)。

(1)一致性:数据对象的原子性,即在所有操作(“读”或“写”)中,必须存在一个完整、确定的顺序,使得所有操作就像在单个实体上完成一样。

(2)可用性:每个非故障节点在收到请求消息后必须给予回复。

(3)分区容错性:能够容忍网络丢失从一个节点发送给另一节点的任意数量的消息。

CAP定理在应用于分布系统时,需要根据不同的场景对3个属性进行相应的取舍,如图5所示。在网络环境中,由于分区容错是一个常态,在设计一个具体的分布式系统时需要在一致性与可用性之间进行权衡。例如,Amazon DynamoDB为了确保可用性而只能舍弃一致性,而谷歌文件系统(Google file system, GFS)在实现一致性的前提下只有舍弃可用性。

虽然CAP定理和FLP不可能性定理都是研究分布式系统中的一些无解问题,但两个定理成立的条件不同:FLP不可能性定理假设的异步通信链路是可靠的,只是消息存在延迟,但不会丢失,而CAP定理

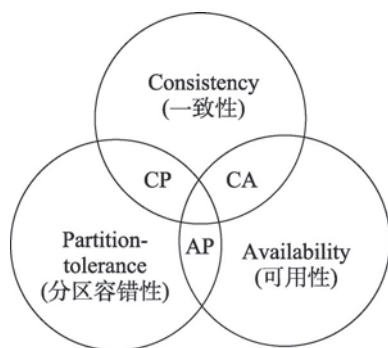


图5 CAP定理的属性取舍

Fig.5 Attribute trade-off of CAP theorem

中的网络分区容错性会导致消息丢失;由于FLP不可能性定理中的故障节点会从系统中完全隔离,不会对任意请求进行响应,但CAP定理中的可用性要求系统能够响应请示。但不管是FLP不可能性定理还是CAP定理,因为分布式系统都会对一致性、可用性和可终止性有要求,所以在具体的共识算法设计时都需要考虑两个定理的条件限制,通过合理取舍CAP定理的相关属性来规避FLP不可能性。

#### 2.4.2 区块链SHD完整性

CAP定理指出,分布式系统不可能同时提供一致性、可用性和分区容错性。同样,在区块链系统中也存在一个名为SHD完整性的问题(也有研究者称为“不可能三角”问题),即必须在安全性(security)、高性能(high-performance)和去中心化(decentralization)之间进行权衡<sup>[35]</sup>。

在比特币区块链系统中,在确保了安全性和去中心化的同时,需要以牺牲效率为代价(每秒只能处理7笔交易,每10 min只能产生一个区块)。比特币系统的缺陷不止一个,例如高性能ASIC矿机和矿池的出现破坏了去中心化(当ASIC矿机获得超线性回报时,普通CPU挖矿的概率几乎为0)。还有,随着矿池计算能力的逐渐增强,有人垄断网络总计算资源的51%似乎只是时间问题,因此系统的安全性也岌岌可危。因此,比特币区块链已经失去了SHD完整性。

为了避免ASIC矿机带来的影响,以太坊采用Ethash算法<sup>[36]</sup>实现了安全性和去中心化。与此同时,作为以太坊第一个大规模智能合约应用的CryptoKitties算法<sup>[37]</sup>却暴露出了效率低下的问题。另外,从PoW向PoS和DPoS的转变极大地提高了以太坊的性能,却严重影响了去中心化。

为此,如果要实现去中心化又要保证安全,那么在性能上需要做出牺牲(如比特币系统);如果在安

全前提下实现高性能,就需要在去中心化上作出让步(如私有链和联盟链);如果要同时实现去中心化和高性能,那么其安全性将无法得到保障。因此,在当前区块链系统中如果要实现SHD完整性还需要在理论和实践上不断探索。

## 2.5 Paxos算法

Paxos算法<sup>[24]</sup>是Lamport早在1990年提出的一种在分布式系统中基于消息传递的一致性算法,为后续各类典型一致性算法研究提供了理论基础和参照。不过,由于Paxos算法的描述过于晦涩难懂,在工程应用中很难实现。

### 2.5.1 Paxos算法概述

概括地讲,Paxos算法具体解决的是在一个随时可能出现节点故障或网络异常(如分组延时、丢失、重传、乱序等)的分布式系统中,在某一集群内快速实现数据或网络状态的一致性。Paxos算法建立在以下3个假设条件(条件12至条件14)成立的基础上<sup>[38]</sup>:

**条件12** 部分同步网络环境。

**条件13** 能够容忍一半以下数量的宕机恢复(非拜占庭故障)节点,即少数服从多数的“过半”规则。

**条件14** 不存在拜占庭将军问题,即信道是安全可靠的,且节点发送出去的信息可能会出现丢失或重复,但不会被篡改。

由于Lamport在文献[24]中对Paxos算法的描述很难理解,为使更多的研究者能够掌握Paxos算法的精髓并推动该算法的应用,后来作者在文献[39]中对该算法描述进行了简化,在该精简版本的Paxos算法中,分布式系统中的节点被分成proposer(提议者)、acceptor(接受者)和learner(学习者)共3种角色,而且一个节点可以同时扮演多种角色。其中,提议者向接受者提交提案(proposal),提案中包含有决议(value);接受者在接收到提案后对该提案进行审批;学习者在获取了通过审批的提案后,执行提案中包含的决议。在此过程中,如果同时满足只有提议者的决议才能被接受者审批、每次只能批准一个决议以及决议只有被批准后才能被学习者获取并执行这3个条件,就可以保证数据的一致性<sup>[40]</sup>。

### 2.5.2 Paxos算法的执行过程

如图6所示,Paxos算法的执行分为两个阶段,每个阶段又分为两个子过程<sup>[39]</sup>。

第一阶段(准确阶段):(1)提议者选择一个提案编号 $N$ ,然后向一半以上的接受者发送该编号的准备请求。(2)接受者在收到该准备请求后,如果其中的

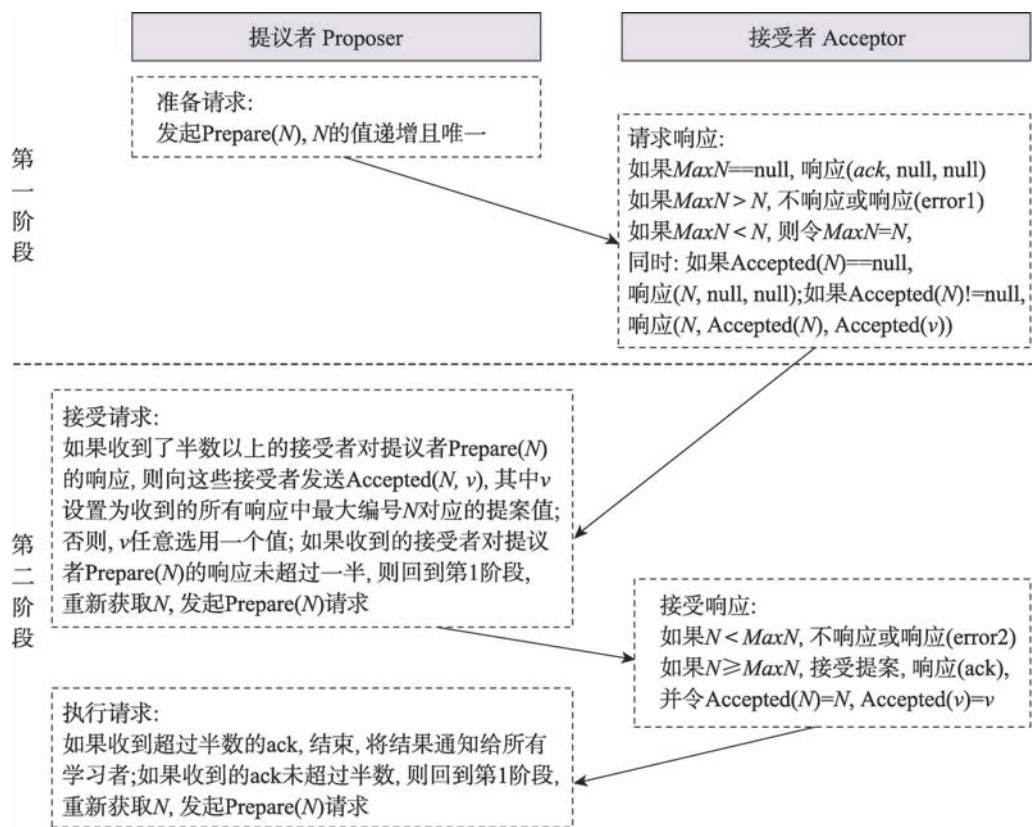


图6 Paxos算法运行过程描述

Fig.6 Description of running process of Paxos algorithm

编号  $N$  大于其已经响应过的所有准备请求的编号, 则对该请求做出响应, 并将其已经接受过的最大编号的提案 (如果有的话) 回复给提议者, 同时接受者承诺不再授受任何编号小于  $N$  的提案; 如果还没有接受过提案, 则返回当前的编号  $N$ 。

第二阶段 (提议阶段): (1) 如果某个提议者收到半数以上授受者对其编号为  $N$  的准备请求的应答, 那么该提议者就会向这些接受者返回一个针对  $[N, v]$  提案的授受请求, 其中  $v$  是应答中编号最大的提案的值; 如果应答中没有任何提案, 那么  $v$  就由提议者任意设置。(2) 如果接受者收到针对编号  $N$  的接受请求, 只要该接受者尚未对编号大于  $N$  的准备请求做出过响应, 便接受该提案, 并返回接受应答。

在以上两个阶段中, 每个阶段对应的请求和应答 (响应) 内容各不相同。

作为分布式共识设计时参考的一个重要算法, Paxos 算法已经应用到 Google Chubby<sup>[41-42]</sup>、Microsoft Autopilot<sup>[43]</sup>、微信 PaxoStore<sup>[44]</sup> 等重要的分布式服务中。另外, 在 Paxos 算法的基础上, 相继设计出了应用于不同场景且易于理解和实现的共识算法, 如 Raft<sup>[45]</sup>、Zookeeper<sup>[46]</sup>、etcd<sup>[47]</sup> 等。

表 1 对经典分布式共识算法进行了汇总, 并对各算法涉及到的主要特性进行了说明。

表1 经典共识算法

Table 1 Classical consensus algorithm

| 名称                      | 年份   | 拜占庭容错            | 节点是否可靠   | 链路是否可靠 | 通信模型 |
|-------------------------|------|------------------|----------|--------|------|
| 两军问题 <sup>[21]</sup>    | 1975 | 否                | 是        | 否      | 不限   |
| 拜占庭将军问题 <sup>[22]</sup> | 1982 | 是 ( $<1/3$ )     | 否        | 是      | 部分同步 |
| FLP <sup>[31]</sup>     | 1985 | 否                | 允许 1 个失败 | 是      | 异步   |
| CAP <sup>[33-34]</sup>  | 2019 | 是 ( $\geq 1/3$ ) | 是        | 否      | 异步   |
| Paxos <sup>[24]</sup>   | 1998 | 否                | 是        | 是      | 部分同步 |

### 3 区块链共识算法

当中本聪在其那篇经典的论文“Bitcoin: a peer-to-peer electronic cash system”中描述了一个在不需要第三方机构就可以实现点对点在线支付的比特币系统后, 支撑这一“神奇”应用场景且对传统中心化系统产生颠覆作用的区块链技术得到全球范围的关注, 也引发了区块链共识算法的研究热潮, 同时产生了大量创新性的应用。虽然各类共识算法伴随着区

区块链产品及应用的迭代不断推陈出新,但就其算法的核心仍然是BFT、PBFT、PoW和PoS共识算法,其他大量算法几乎都是以上4类基本算法的性能优化、功能扩展和应用组合。

### 3.1 PoW 共识算法

虽然比特币系统首次使用了工作量证明(proof of work, PoW)算法实现挖矿过程,但工作量证明的概念早在1999年就已经提出<sup>[48]</sup>,并在防范垃圾邮件和拒绝服务(denial of service, DoS)攻击等功能中得到了应用。

比特币区块链的PoW共识机制借鉴了哈希现金(HashCash)<sup>[49]</sup>的思想。哈希现金最早用来处理垃圾邮件,发件人发送的邮件正文中必须包含一段由收件人地址、发送时间和计数器(counter)组成的邮件签名,其中计数器功能需要使邮件签名满足条件:利用SHA-1算法对邮件签名生成一个160 bit长度的哈希值,该哈希值的前20位全为0。于是,邮件发送方在发送邮件前需要不断调整计数器的值(counter++),生成满足条件的邮件签名。对于邮件服务器来说,只需要进行一次SHA-1计算并判断签名是否满足条件即可。SHA-1的碰撞概率和散列算法的不可预测性,决定了HashCash系统的安全性。

#### 3.1.1 PoW 共识算法定义

**定义10**(比特币PoW共识算法) 在比特币网络中,给定一个难度值 $D$ ,存在一个区块中容纳的所有交易数据 $tradeData$ ,计算满足条件的 $Nonce$ (number only used once)值,使得两次SHA-256计算得到的值小于 $D$ :

$$SHA256(SHA256(tradeData||nonce)) \leq D$$

其中,在 $tradeData$ 确定(新区块中“区块体”中的交易信息已确定)的情况下,不断重复选取随机数 $Nonce$ ,直至找到满足条件的 $Nonce$ 为止。

由定义10可见,在PoW共识算法中,矿工只有不断改变输入的 $Nonce$ 值,才能在竞争中获胜,而竞争力的大小由矿工节点的算力大小决定,因此算法的竞争最后转变到比特币网络中哈希算力的竞争上<sup>[50-51]</sup>。另外,难度值 $D$ 的大小用于控制比特币的出块时间在系统约定的10 min左右,新难度值 $D_{new}$ 与旧难度值 $D_{old}$ 之间的关系为:

$$D_{new} = D_{old} \times \frac{Usedtime}{20\ 160}$$

其中, $Usedtime$ 代表产生前面2 016个区块所用的时间。虽然在PoW共识机制中引入了大小为32 Byte的难度值,但在比特币区块头部中却只用4 Byte的

“Bits”(难度位)字段来设置难度大小,因此在PoW共识算法的具体实现中使用了与难度值对应的“目标值”(用 $T$ 表示)概念,目标值与难度值之间的关系为:

$$T = \frac{T_{max}}{D}$$

其中, $T_{max}$ 表示最大目标值,即比特币创世区块(genesis block)的“Bits”字段值0x00000000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF,是一个64位的16进制固定值,该值用小端法(little-endian)表示为0x1d00FFFF。

在PoW共识系统中,一定会存在在计算上严重不对称的两个角色:工作者和验证者。其中,工作者需要进行一定难度的哈希计算得出一个结果,而验证者通过简单的计算就可以判断工作者是否做了相应难度的工作。

#### 3.1.2 PoW 共识算法的实现过程

如图7所示的是比特币系统中PoW共识算法的执行流程,其主要实现过程如下:

(1)根据区块数据结构,将铸币(coinbase)交易和节点缓存中已签名的交易打包进区块形成完整的区块体,通过默克尔树(Merkle tree)算法生成默克尔根(Merkle-root)。

(2)判断距上次难度值调整时是否已经产生了2 016个区块,如果是,马上进行难度值的调整,以获得新的难度位(Bits);否则使用原来的难度值继续进行

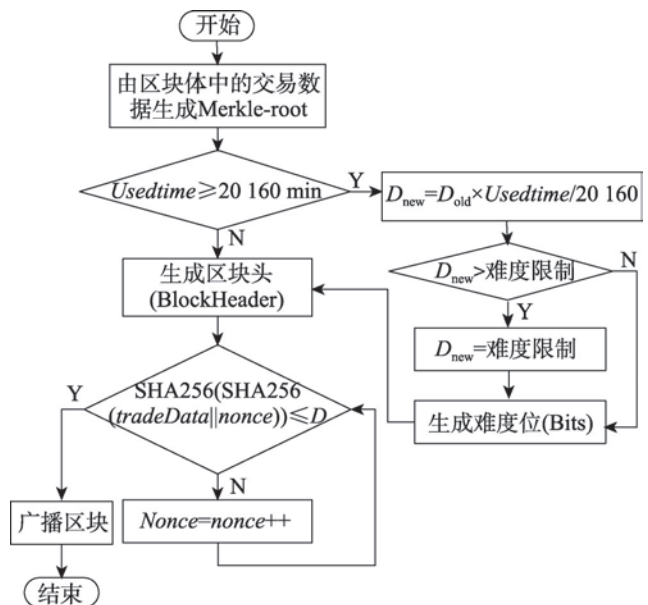


图7 比特币系统中PoW共识算法的实现流程

Fig.7 Realization process of PoW consensus algorithm in bitcoin system

行挖矿。

(3)将区块头部当前版本号(version)、前一区块哈希(prev-block Hash)、当前难度值(Bits)、随机数(nonce)、Merkle根(Merkle-root)以及时间戳(timestamp)6个字段共80 Byte数据作为工作量证明的输入值,不断调整随机数 *Nonce*,并对每次调整后的区块头进行双SHA-256运算,将运算值与当前的难度值 *D* 进行比较,如果小于难度值则挖矿成功。

(4)矿工将新挖出的区块向全网广播,希望获得新区块的记账权从而获得比特币发行奖励。节点对接收到的新区块进行验证,如果验证无误就将该区块添加到主链上,并以此区块为基础继续下一个区块的挖矿工作。

通过以上过程,交易信息将被写入各记账节点的区块中,形成一个分布式的一致性账本。

### 3.1.3 PoW 共识算法的典型应用与改进分析

由于比特币在加密货币领域取得巨大成功,PoW 共识算法也引起了区块链研究者的广泛关注,随即在原生 PoW 共识算法的基础上推出了一批具有影响力的算法。

#### (1)比特币 PoW 共识算法

PoW 共识算法最典型和成功的应用场景是比特币(Bitcon),后来针对工作量证明的共识算法基本上都是针对比特币系统存在的缺陷和不足衍生出来的,因此可以将比特币系统中的 PoW 共识算法称为“原生”PoW 共识算法。

比特币系统是一个基于互联网的分布式账本,节点缓存中的交易打包进矿工的区块,并通过算力竞争和P2P网络同步最终获得对区块的记账权。作为第一个成功的加密货币,挖矿过程扮演着货币发行的角色,矿工在成功获得区块的记账权后,系统会以一定数量的比特币(BTC)给矿工以奖励,该奖励以铸币交易(coinbase transaction)形式作为下一个区块的第一个交易打包进区块,从而实现了货币的发行过程。在比特币系统中,铸币交易只有一个输出,该输出地址即为矿工的比特币地址,其他交易的实质是账户间比特币的转移,根据具体的支付要求,一笔交易可能存在多个输入和多个输出,每一个输入或输出都代表着参与本笔交易各方的比特币地址。

比特币系统规定每个区块大小为1 MB,每笔交易限制在250 Byte左右,区块的出块时间限制在约10 min,因此比特币的交易量只有每秒约7笔( $1 \times 1\,024 \times 1\,024 / 250 \approx 7$ );另外,在比特币系统中,一个区块在

主链上的深度只有达到6个时才能被最终确认,因此一笔交易从发生到最后确认需要约1 h,如此小的交易量和超长的确认时间限制了PoW共识算法的应用推广;还有,中本聪为了解决拜占庭将军问题,在比特币系统中引入了竞争挖矿机制,所采用的PoW共识算法在最大可能实现公平性的同时,支持哈希运算挖矿所需的巨大算力唯一的价值是保障了比特币的安全性,除此之外对于现实社会没有任何意义。

#### (2)以太坊 Ethash 共识算法

虽然比特币和以太坊都使用了PoW共识算法,但两者分属于区块链技术发展的不同阶段。以比特币为代表的加密货币是区块链1.0时代的典型代表<sup>[52]</sup>,比特币系统存在着非图灵完备、脚本简单、缺少状态控制机制、扩展性差等不足,而以太坊<sup>[53]</sup>由于对智能合约的完美支持而被称为“全球计算机”,成为区块链2.0时代的典型应用场景。相对于比特币,以太坊提供了账户模式(没有使用比特币的UTXO模式<sup>[54]</sup>)、图灵完备、更好的隐私保护、更优的共识机制等功能。

由于比特币PoW共识算法中的哈希运算采用双SHA-256,只消耗CPU资源,而对内存要求不高,这样很容易设计和制造专用于算法需求的ASIC芯片进行挖矿。为了克服比特币存在的不足,以太坊中的PoW共识算法采用“内存困难”机制,主要思想是增加PoW算法的复杂性,使其在运算中需要大量内存的支持,以此来抵制ASIC专用芯片挖矿,其中以以太坊中使用的方案便是利用大内存基于有向无环图(directed acyclic graph,DAG)<sup>[55]</sup>机制的Ethash共识算法。

Ethash共识算法<sup>[56]</sup>是目前以太坊基于PoW的挖矿算法,它的前身是Dagger Hashimoto算法<sup>[57]</sup>。Dagger Hashimoto算法的设计目的是抵制ASIC专用芯片挖矿、轻客户端验证和全链数据存储。图8所示的是Ethash共识算法的实现流程。与比特币中的原生PoW共识算法相比较,Ethash共识算法不仅需要寻找*Nonce*值,同时还填入一项在挖矿过程中计算出来的混合摘要(*MixDigest*)值,*MixDigest*值不仅作为矿工消耗内存挖矿的工作量证明,而且验证者在验证区块的有效性时也需要使用到该值。Ethash共识算法的实现过程主要分为以下几个步骤:

①生成种子(seed)。一个窗口期(epoch length)等于30 000(以太坊约15 s产生一个新区块,因此窗口期约为15 h),即每30 000个区块形成一个窗口(epoch),每个窗口更新一次数据集(dataset)。Seed的初始值为0,每经过一个窗口期会取前一个seed的

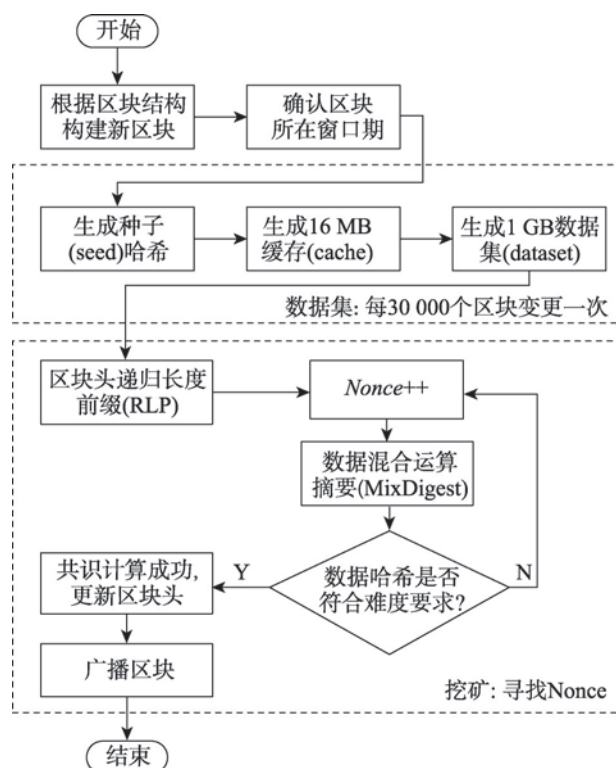


图8 以太坊 Ethash 共识算法的实现流程

Fig.8 Realization process of Ethash consensus algorithm in Ethereum system

哈希值作为下一个窗口的 seed。

②生成缓存(cache)。根据 seed 计算出一个 16 MB 的用于辅助校验区块和生成数据集(dataset)的伪随机 cache, 由轻客户端存储。

③生成数据集(dataset)。根据 cache 由特定算法生成一个 1 GB 的 DAG 数据集(dataset), 其中 dataset 中的每一项数据都是通过 cache 中的 256 项数据计算而来。cache 和 dataset 的大小每经过一个窗口期会重新计算一次, 由全节点(如矿工)存储, 而轻客户端只存储 cache。

④挖矿。矿工根据区块头递归长度(recursive length prefix, RLP)、从 dataset 中随机取出的数据和 Nonce 进行数据混合运算得到混合摘要值。当对混合摘要的哈希计算结果符合区块难度要求时挖矿成功; 否则, 继续 Nonce++ 运算, 直到符合条件。

⑤验证。节点收到新区块时, 根据存储在本地的 cache 重新生成 dataset 中所需要的数据, 再结合区块头中的信息来验证难度值。

以太坊中 Ethash 共识算法的主要功能可以分为生成和管理数据集以及挖矿两个阶段, 其中区块、种子、缓存和数据集之间的关系如图 9 所示。

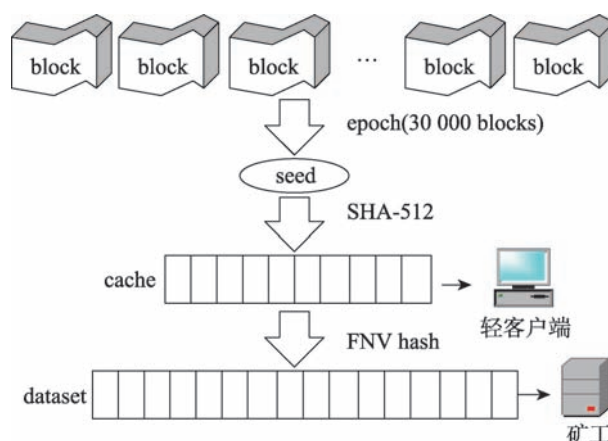


图9 Ethash 算法中区块、种子、缓存和数据集之间的关系

Fig.9 Relationship among block, seed, cache, and dataset in Ethash algorithm

除 Ethash 共识算法外, 莱特币 (Litecoin, LTC)<sup>[58]</sup> 通过刚性内存哈希函数 Scrypt 取代原生 PoW 中的 SHA-256, 达世币 (DASH)<sup>[59]</sup> 通过 11 种哈希函数的混合运算来提高算法的复杂性。目前, 以太坊正在从基于 PoW 的 Ethash 共识算法向着 PoW/PoS 混合过渡, 最后将完全运行在 PoS 共识算法下。

另外, 由于智能合约的引入, 使得以太坊应用场景不再限于加密货币, 而是延伸到了更广泛的应用, 因此在以太坊平台只有提供手续费 (gas) 的交易才能最终被打包到区块中, 以太坊手续费的基本标准为  $Gas_{Used} \times Gas_{Price}$ , 其中  $Gas_{Used}$  是系统根据交易难度设定的手续费, 而  $Gas_{Price}$  是交易者实际设置的手续费, 当实际提供的手续费低于标准手续费时, 矿工一般不会接受这笔交易, 某笔交易如果需要尽快被打包确认便会提高交易的手续费。

还有, 由于以太坊的出块速度约为 15 s, 相对较短的出块间隔在交易数量增多的情况下容易产生分叉。为了维护最长链规则, 以太坊使用了贪婪最重可见子树协议 (greedy heaviest observed subtree, GHOST)<sup>[60]</sup>, 也称为“幽灵”协议算法, 使没有被打包到主链上的区块 (称为“叔块”, uncle block) 的创建者以及对叔块的引用者都能够拿到一定的以太币 (ETH) 补偿。一方面, 使原来打包进叔块的交易被确认, 提高交易的效率; 另一方面, 将创建叔块的诚实节点的算力纳入到整个区块链的算力中, 提高了系统的安全性。

### (3) Bitcoin-NG

限制比特币系统效率的主要因素有区块大小 (每个区块约 1 MB) 和出块速度 (约 10 min)。如果单

纯地增加区块大小也可以增加单个区块容纳的交易数量,进而提高系统的效率,但是随着区块大小的增大,网络的延时也会随之增大,效率反而会降低;如果单纯缩短出块速度,可以缩短交易的确认时间,但将会导致频繁发生分叉,系统的安全性大大降低。为了解决比特币的效率问题,以提高区块链的吞吐量,Eyal等人提出了Bitcoin-NG<sup>[61]</sup>扩容算法。

在Bitcoin-NG算法中,每个窗口(epoch)仍然是10 min,但在每个窗口期内系统会进行两步操作:选择一位“领导者”(leader)和交易序列化(transaction serialization)。相应的存在两类区块:主块(key block)和微块(micro block)。其中,选择领导者的过程与比特币挖矿相同,由领导者挖出的区块为主块,但主块中不包含交易信息。领导者选出后,负责在窗口期内(下一个新领导者出现之前)收集交易信息,并将其打包进一个个微块,每个微块的大小不能超过预设值的上限,微块的形成时间约为10 s,生成微块的过程不需要工作量证明,即不需要消耗算力。由于微块的生成速度远远大于主块,如果简单地采用比特币的最长链原则将无法实现领导者的选择,因此Bitcoin-NG算法中引入了区块权重的原则,且规定微块是没有权重的,在Bitcoin-NG算法中矿工维护的是一条最长且权重最大的链。

如图10所示,与比特币的区块头部相比,Bitcoin-NG的主块多了一个用于存放当前领导者公钥( $PK_i$ )的字段,因为交易被打包进微块后,为了实现对微块所属领导者的验证,需要用领导者自己的私钥对生成的微块进行签名( $sig_i$ );为了激励矿工按照约定规则挖矿并打包交易,在Bitcoin-NG算法中,挖到主块的矿工获得类似于比特币中的挖矿奖励,而对于当前窗口期内生成微块的所有交易费中,40%归当前领导者所有,60%归下一个领导者所有,这样做的目的主要是防止贪心矿工进行自私挖矿(selfish mining)<sup>[62]</sup>。

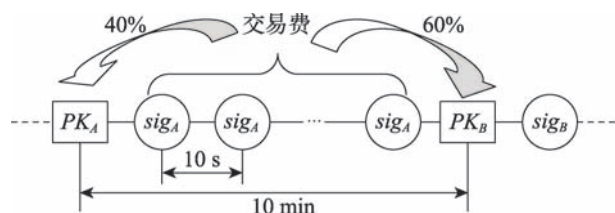


图10 Bitcoin-NG 共识算法示意图

Fig.10 Schematic diagram of Bitcoin-NG consensus algorithm

在Bitcoin-NG算法中,微块不需要进行挖矿竞争就可以轻松生成并发布,这样作恶领导者便可以构

造不同的微块并发送给不同节点实现对系统的双花攻击。为了防止双花攻击,Bitcoin-NG算法采用了一种称为“毒药交易”(poison transaction)的特殊交易。当领导者发现前面的某个领导者是作恶领导者时,他将作恶领导者的信息打包到自己的微块中并发布,这样作恶领导者原来获得的奖励将全部被收回,同时进行“举报”的领导者将获得作恶领导者总奖励5%的奖励。另外,为了防止区块链产生分叉,Bitcoin-NG算法还引入了“报酬审核期”的概念,规定交易费在100个主块后才能使用。

Eyal等人在约1 000个节点组成的比特币系统上对Bitcoin-NG算法进行了模拟实验<sup>[61]</sup>,对其良好的可扩展性、有效提升吞吐量和明显降低延迟等特性进行了有效验证,成为区块链扩容方案中具有代表性的共识算法。

#### (4)其他典型PoW共识改进算法

由于PoW共识算法在区块链技术中的奠基石作用,研究者通过算法优化与改进,在原生PoW算法基础上提出了大量有代表性的算法:2016年Kokoris-Kogias等人提出了ByzCoin<sup>[63]</sup>共识算法,该算法采用了与Bitcoin-NG同样的思路,将区块分为主块(key block)和微块(micro block)。只是在ByzCoin算法中,一旦发现有作恶领导者,矿工将通过投票方式,在票数超过67%阈值时便取消该作恶领导者的资格,并取消所有的奖励。ByzCoin算法可以有效防止Bitcoin-NG算法中可能存在的作恶领导者;2017年,Kokoris-Kogias等人在借鉴ELASTICO共识算法<sup>[64]</sup>并行跨分片处理功能的基础上,对ByzCoin共识算法进行了优化,提出了ByzCoinX共识算法<sup>[65]</sup>,使区块链系统的可扩展性和安全性都有了较大提高;2017年,Pass和Shi提出了FruitChains共识算法<sup>[66]</sup>。该算法提出了Fruit(水果)的概念,其中一个Fruit中可能包括多个交易,而一个区块中也可能包含多个Fruit,Fruit和区块利用同一个哈希寻找工作量证明,从而提高比特币系统中挖矿的公平性;2018年,来自清华大学的Li等人提出了Conflux共识算法<sup>[67]</sup>。该算法先用DAG图来算出区块顺序,再决定要保留的交易数据,然后生成没有分叉的单链结构,使交易的吞吐率提升到每秒6 000次以上。

除此之外,针对PoW共识算法主要在能耗和51%攻击方面存在的问题,研究人员提出了一些改进算法:为了有效缓解PoW算法的能耗问题,Lasla等人<sup>[68]</sup>提出了Green-PoW共识算法,通过对挖掘过程中

所消耗能量的再利用,可以将挖矿的能耗降低到50%左右;为了解决PoW共识算法高能耗和51%攻击问题,Kara等人<sup>[69]</sup>提出了CW-PoW算法,该算法通过引入多轮证明的方式在能耗和对51%攻击及Sybil攻击的鲁棒性方面取得了明显成效;针对PoW共识算法存在的不足,Qu等人<sup>[70]</sup>提出了联合学习证明算法(proof of federated learning, PoFL),该算法将联邦学习理论用于区块链工作证明,加强了对用户隐私信息的保护等。

表2列出了针对PoW共识算法及其主要改进算法的相关信息。

表2 PoW及其改进算法

Table 2 PoW and its improved algorithms

| 名称                          | 年份   | 能耗 | 出块速度               | 分叉 | 算力分布 |
|-----------------------------|------|----|--------------------|----|------|
| PoW <sup>[48]</sup>         | 1999 | 高  | 10 min             | 有  | 集中   |
| Ethash <sup>[56]</sup>      | 2014 | 较高 | 15 s               | 易  | 分散   |
| Bitcoin-NG <sup>[61]</sup>  | 2015 | 较低 | 主块 10 min、微块 10 s  | 易  | 较集中  |
| ByzCoin <sup>[63]</sup>     | 2016 | 较低 | 区块 1 MB 时约 1/100 s | 易  | 较集中  |
| ByzCoinX <sup>[65]</sup>    | 2018 | 较低 | —                  | 无  | 较集中  |
| FruitChains <sup>[66]</sup> | 2017 | 低  | —                  | —  | 分散   |
| Conflux <sup>[67]</sup>     | 2018 | 低  | < 1/6 000 s        | 无  | 分散   |

### 3.2 PoS 共识算法

对于比特币来说,成也PoW,败也PoW。PoW共识算法通过所有参与节点之间的算力竞争,提交一个计算困难但验证容易的计算结果,任何节点都可以通过简单的运算来验证结果的正确性进而承认被验证者的工作量,这一机制有效维护了比特币系统的安全性和稳定性。然而,PoW算法存在的算力不公平、资源浪费、效率低下等问题,严重限制了比特币区块链技术向加密货币领域之外的延伸和拓展。鉴于此,研究者提出了针对工作量证明机制的替代算法,权益证明(PoS)便是其中的一种。

#### 3.2.1 PoS 共识算法的概念

PoW共识算法是通过消耗现实生产生活中的价值(算力)来维护区块链的价值体系,而PoS共识算法则是在一套规则的支持下,利用已有的价值(权益)来达成共识从而不断衍生出新的价值,进而在一个“自治系统”(不需要像PoW那样从外界输入算力)中维护区块链的价值体系。

2011年,Quantum Mechanic首次提出了PoS共识算法<sup>[71]</sup>,算法中规定了用节点拥有的比特币数量来替代PoW共识算法中基于算力求解哈希值的过程,即矿工拥有的比特币数量越多挖到矿的可能性就越

大。2012年,King等人设计了点点币(PPCoin或PeerCoin)<sup>[72]</sup>并在系统中首次实现了PoS共识算法。

参照PoW共识机制的定义,将PPCoin中的PoS共识算法称为原生PoS共识算法。原生PoS共识算法中引入了“币龄”的概念,相关定义如下:

**定义 11(币龄)** 币龄(*coinAge*)是指持币数量(*coins*)与持币时间(*holdTime*)的乘积:

$$\text{coinAge} = \text{coins} \times \text{holdTime}$$

**定义 12(PoS 共识算法)** 给定一个全网统一的难度值  $D$ ,以及新打包进区块的元数据 *tradeData*,寻找满足条件的计数器 *timeCounter*,使得:

$$\text{HASH256}(\text{HASH256}(\text{tradeData} \parallel \text{timeCounter})) < D \times \text{coinAge}$$

PoS共识算法的记账规则与PoW共识算法基本相似,但PoS共识算法不需要矿工枚举所有的随机数 *Nonce*,而是在1 s内只允许一次哈希值,大大减轻了计算工作量,从而减缓了算力竞争带来的资源消耗。

由于PoS共识算法的提出主要是为了解决PoW共识算法的缺陷,通过与PoW共识算法的比较便可以体现PoS共识算法的优势:

(1)PoS是通过权益大小来寻找哈希值,不存在因算力竞争浪费能源的问题。诞生于2013年11月24日的Nextcoin(未来币)<sup>[73]</sup>是一个全新的使用PoS共识算法的第二代虚拟货币。

(2)由于PoS共识算法采用类似“币龄”的权益值,并对成功挖矿者采用了币龄“清零”机制,PoS共识算法更加公平。

(3)为了防止节点离线积累币龄,2014年Vasin提出了PoS2.0<sup>[74]</sup>并应用到黑币(BlackCoin)中,从而使黑币从当时的PoW+PoS共识模式发展到了纯PoS共识模式。PoS2.0共识算法中的权益大小与用户在线时间成正比,这种激励机制有效增强了区块链P2P网络的健壮性,防止了PoW共识算法中“公地悲剧”(tragedy of the commons)的发生,而且使得攻击者实现“51%攻击”的难度大大增加等。但PoS共识算法容易产生分叉,且区块链的去中心化特点被弱化。

需要说明的是:PoS共识算法最早运行在PPCoin系统,但PPCoin系统中使用的是PoS+PoW混合共识算法。

#### 3.2.2 PoS 共识算法的典型应用与改进分析

##### (1)DPoS 共识算法

针对PoS共识算法主要存在离线节点积累币龄和某些拥有权益的节点无意挖矿等缺陷,2014年4

月, Larimer 在 PoS 共识算法的基础上提出了委托权益证明 (delegated proof of stake, DPoS) 共识算法<sup>[75]</sup>。DPoS 共识算法引入了类似企业董事会的管理机制, 权益持有者通过投票方式选择能够代表自己利益的委托人 (票数与持有代币的多少成正比), 被选出的受信任的委托人需要交纳一定数量的保证金, 最后形成一个由 101 个委托人组成的集合。该集合中的委托人将以平等的权利按规则轮流作为出块者生成区块, 并从每笔交易的手续费中获得收益。在此期间, 如果某个委托人违反了出块规则, 其委托人身份将被新选出的委托人替代, 而且其保证金也被系统没收。

DPoS 共识算法通过选举委托人的方式来确定区块的生成者, 缩短了交易的确认时间, 提高了系统的效率, 但由于被选举出的负责挖矿的委任人数量仅为 101 个, DPoS 共识算法是一个去中心化 (针对权益持有者) 和中心化 (委托人) 相结合的共识算法。另外, DPoS 共识算法可能会存在委托人通过联合操纵区块链网络来损害普通节点利益, 以及通过选出不符合要求的委托人从而迫使网络的性能下降等现象。

目前, 使用 DPoS 共识算法的区块链应用系统主要有 Bitshares (比特股)<sup>[76]</sup>、Steem<sup>[77]</sup>、EOS<sup>[78]</sup>, 其中 Steem 和 EOS 中委托人的数量都为 21。

## (2) Ouroboros 共识算法

为了能够在运行 PoS 共识算法的区块链系统中实现类似 PoW 共识算法中的安全机制, Kiayias 等人提出了 Ouroboros 共识算法<sup>[79]</sup>。Ouroboros 共识算法因其在学术领域产生的影响力, 很快成为学术界认可和产业界采用的可证明是安全和健壮的 PoS 共识算法。目前, Cardano (ADA, 艾达币)<sup>[80]</sup>就使用了 Ouroboros 共识算法。

Ouroboros 共识算法的核心是根据权益的多少来随机选出一个出块者, 而且这个随机过程是无法预测的。Ouroboros 共识算法的运行过程为:

① 共识过程的时间被划分为间隔相同的时隙 (slot), 每个被选举出的出块者 (类似于 DPoS 中的委托者) 对应一个时隙。

② 每个时隙最多只能产生一个区块, 如果当时隙到来时对应的节点离线或生成的区块未得到大多数节点的确认, 则该时隙不会生成有效区块。

③ 由多个连续的时隙组成一个窗口 (epoch), 在一个窗口中, 权益的计算是以该窗口开始时的历史

值为基准, 期间权益的变化不会影响当前窗口中出块者的选择。

④ 每个窗口开始时, 在矿工节点的计算机内存中会生成一个 genesis block, 该特殊区块仅记录本窗口中可能参与出块的权益持有者的候选人信息以及随机数种子  $\rho$ , 并不会最终上链。

⑤ 在每个窗口中, 以 genesis block 中的每个候选人所持有权益的占比为概率, 选择每个时隙对应的出块者, 具体的选择函数 (Cardano 中使用了 follow-the-Satoshi 算法) 为  $U_k = F(S_n, \rho_n, r_k)$ , 其中  $U_k$  为窗口  $n$  中第  $k$  个时隙对应的出块者,  $S_n$  为在窗口  $n$  中由所有出块候选人组成的集合  $(U_1, U_2, \dots, U_n)$ ,  $\rho_n$  为窗口  $n$  中使用的随机数种子, 该随机数种子与当前窗口中每位候选人持有的权益  $(s_1, s_2, \dots, s_n)$  有关,  $r_k$  为第  $k$  个时隙对应的索引 (index) 信息。根据随机数种子  $\rho_n$ , 函数  $F()$  从候选人集合  $S_n$  中随机选出第  $k$  个时隙对应的出块者  $U_k$ 。

⑥ 由连续的窗口衔接生成的链便是 Ouroboros 共识算法形成的链, 形成链的每个区块之间的关系与比特币相同, 即当前区块通过上一个区块的哈希函数来连接。

在 Ouroboros 共识算法中, 每一个窗口的随机数  $\rho$  由安全多方计算 (secure multiparty computation, MPC)<sup>[81]</sup> 协议产生, 该协议使用可公开验证秘密共享 (publicly verifiable secret sharing, PVSS)<sup>[82]</sup> 方案。该方案基于非交互式零知识证明 (non-interactive zero-knowledge, NIZK)<sup>[83]</sup> 确保了任何出块候选人都可以在参与人数达到阈值的情况下重构密钥, 并验证其有效性。

Ouroboros 共识算法的执行流程如图 11 所示。从链的创世区块 (非每个窗口中的 genesis block) 开始, 硬编码了与用户身份相关联的公钥密钥对 (其中公钥  $pub_i$  公开, 私钥  $priv_i$  由用户保存)、权益  $s_i$  以及初始的随机数种子  $\rho_1$ 。之后, 后续的每个窗口会基于前一个窗口的这些基本数据运行, 其中每个节点以当前窗口的随机数种子  $\rho_n$  以及 genesis block 中的权益  $(s_1, s_2, \dots, s_n)$  和时隙对应的索引信息  $r_k$  为输入值, 独立运行函数  $U_k = F(S_n, \rho_n, r_k)$ , 根据概率获得当前时隙对应的出块者  $U_k$ ; 对于某个出块候选人来说, 如果正好自己被选择为本时隙对应的出块者, 则执行交易打包和分发等操作, 否则对接收到的区块进行验证操作 (此过程与比特币的挖矿过程相同); 如果

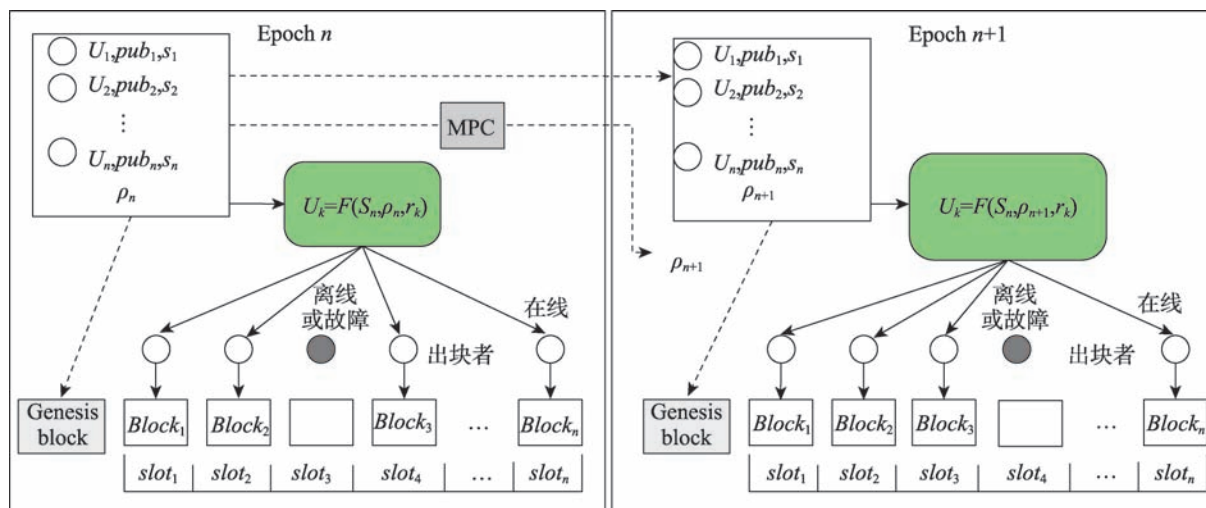


图 11 Ouroboros 共识算法的执行流程

Fig.11 Implementation process of Ouroboros consensus algorithm

被选择的出块者正好处于离线状态或出现运行故障,则在该时隙内不会产生新区块或生成的区块被废弃;在当前窗口中,根据时隙划分不断执行选择出块者、出块、验证等操作,直到最后一个时隙结束。在整个窗口操作结束后,会在每个节点的内存中生成下一个窗口的随机数种子  $\rho_{n+1}$  以及出块候选人集合,随后进入下一个窗口的操作流程。

Ouroboros 共识算法的最大特点是通过安全、多方执行的协议来实现每个时隙出块者选择的随机性,以确保出块者的选择不会被攻击者操纵。但该算法不适合完全去中心化的应用场景。

Ouroboros 共识算法中,在选择每个时隙对应的出块者时随机数种子  $\rho$  发挥着十分重要的作用,但由于该随机数种子生成函数是公开的,即在每个窗口(epoch)开始的时候,所有节点已经知道整个窗口中所有的候选人,作恶节点可以利用这一缺陷有针对性地进行贿赂攻击或 DDoS 攻击;另外,对于 MPC 来说,随着参与节点数量的增加,其性能则会降低;还有, Ouroboros 共识算法的攻击模型  $n = 2f + 1$  建立在同步网络模型中,对通信环境提出了较高要求。

### (3) Ouroboros Praos 共识算法

针对 Ouroboros 共识算法存在的不足, David 等人<sup>[84]</sup>在 Ouroboros 基础上提出了改进的 Ouroboros Praos 共识算法。该算法的主要改进是在选择出块候选人时,用可验证随机函数(verifiable random function, VRF)替代公开伪随机函数,从而使每个节点可以使用不同的随机函数判断自己是否是出块候选人。当某个节点被确定为某个时隙对应的出块候选

人时直接出块,该块中同时包含有区块验证时所需要的证明信息。其他节点只有在收到区块时才知道谁是出块者,这样系统的安全性和出块速度都得到了大幅度提升。另外,在 Ouroboros Praos 共识算法的基础上, Ganesh 等人<sup>[85]</sup>提出了匿名可验证随机函数(anonymous VRF)概念,通过构造一类通用的私有 PoS (private PoS) 协议,实现了对节点隐私的有效保护。

### (4) Ouroboros Genesis 共识算法

Ouroboros Genesis 是 Ouroboros 协议的第 3 个版本,由 Badertscher 等人<sup>[86]</sup>在 Ouroboros Praos 基础上提出。Ouroboros Genesis 主要解决的是在一个部分同步网络环境中,当一个全新节点或长时间离线节点加入网络时,如何利用其他节点中的账本信息来建立自己的区块链账本,即如何解决新节点的自举(bootstrap)问题。PoW 共识算法使用最长链规则来确定本地账本,而 PoS 使用替代的检查点(check point)协议或拜占庭容错机制来建立本地账本,但这些算法的实现都要求节点在始终在线的同步网络环境中。

为了解决自举问题, Ouroboros Genesis 算法中提出了一种称为丰富规划(plenitude rule)的主链选择机制,将从不同节点复制的账本进行比较,从中选出具有相同前缀且最长的链作为本地账本。Ouroboros Genesis 算法在没有采用检查点的前提下,有效防范了攻击者通过伪造一个符合系统规则的最长区块链,为新节点或离线节点加入网络时提供虚假账本信息服务的长程攻击(long range attack)<sup>[87]</sup>,并且在通用可组合(universal composable, UC)模型下形式化验证了算法的安全性。

### (5) Snow White 共识算法

Snow White 共识算法是由康奈尔大学 Elaine Shi 教授等人<sup>[88]</sup>在 2016 年提出的一个运行在去中心化的异构网络环境中、提供端到端 PoS 的共识算法。Snow White 采用了睡眠执行模型(sleepy execution model)实现了新加入节点或离线节点重新加入后实现共识时的具体要求。该模型中的节点状态与互联网中的节点状态高度契合。Snow White 使用委员会重组机制,每次重组过程都以节点最近拥有权益的多少为依据确定委员会成员,然后通过成员权益的多少随机选择出块者。Snow White 使用委员会重组机制的目的是根据系统中可能发生的节点权益变化,动态选择委员会成员和出块者,以防止后来破坏(posterior corruption)<sup>[88]</sup>攻击的发生,以提高系统的抗攻击能力。同时,该机制设置了较小的委员会重组时间间隔,适应了互联网环境中节点频繁加入和离开的连接特征。

Snow White 虽然设置了较小的委员会重组时间,但算法中规定的节点权益变化不能太频繁,以确保节点的投票权与其拥有权益成正比。另外, Snow White 采用了与 FruitChains 算法<sup>[66]</sup>相似的挖矿激励机制,交易信息放在水果中,水果放在区块中,将出块奖励和区块中包含的所有交易费分配给相应的出块者。

为便于加强对不同算法的理解,表 3 对几个典型的 PoS 共识算法的性能进行了分类比较。

表 3 典型 PoS 共识算法的性能比较

Table 3 Performance comparison of typical PoS consensus algorithms

| 名称                                | 年份   | 安全性 | 出块速度        | 分叉   | 去中心化  |
|-----------------------------------|------|-----|-------------|------|-------|
| DPoS <sup>[75]</sup>              | 2014 | 较弱  | 0.5 s (EOS) | 短期分叉 | 部分中心化 |
| Ouroboros <sup>[79]</sup>         | 2017 | 较强  | 20 s (ADA)  | 无    | 中心化   |
| Ouroboros Praos <sup>[84]</sup>   | 2018 | 强   | <20 s       | 无    | 中心化   |
| Ouroboros Genesis <sup>[86]</sup> | 2018 | 更强  | <20 s       | 无    | 中心化   |
| Snow White <sup>[88]</sup>        | 2016 | 强   | <20 s       | 分叉   | 去中心化  |

## 3.3 PoW+PoS 混合共识算法

PoW+PoS 共识算法结合了 PoW 和 PoS 的特点,节点在竞争记账权的过程中按算法要求生成 PoW 或 PoS 区块,有效解决了原来单一共识算法在安全风险、能源消耗、效率等方面存在的问题。

### 3.3.1 PoSV 共识算法

权益速度证明(proof of stake velocity, PoSV)<sup>[89]</sup>共识算法由 Ren 在 2014 年 4 月提出,并应用到 Red-

dcoin(蜗牛币)中。针对 PoS 中基于时间线性函数屯积币龄的问题, PoSV 将其函数修改为指数式衰减函数,从而使币龄的增长率随时间呈下降趋势,直到最后趋于 0。这样一来,新币的币龄增长要比老币快,直到达到上限阈值时趋于相同。

Reddcoin 是一种加密货币,其前期使用 PoW 算法实现代币的分发,而后期使用 PoSV 以提高 P2P 网络的安全性。

### 3.3.2 PoA 共识算法

出块奖励和收取交易费是区块链挖矿收入的主要来源。当出块奖励随着系统运行时间不断减少时(如比特币),挖矿者则将注意力集中到交易费的收取,这有可能导致“公地悲剧”的发生,从而对系统安全性造成威胁。另外,当交易费不足以吸引挖矿者时,要么部分挖矿者可能会因为无利可图而选择离开,要么只有通过收取更高的交易费来维持系统的运行。不管选择哪种方式,都会导致系统的安全性降低直至区块链网络无法正常运行。为了解决以上问题, Bentov 等人提出了活跃证明(proof of activity, PoA)共识算法<sup>[90]</sup>,其基本思想是尽可能激励持币者参与挖矿过程,在保持数字货币保值甚至增值的前提下,解决维护网络安全对交易费过度依赖的问题。PoA 共识算法的挖矿过程如下:

(1)每个矿工不断进行 Hash 运算,生成一个符合难度值的空区块头。该空区块头是一个特殊的区块,其中区块中不包含交易信息,只包含由前一区块 Hash 值、本节点的公钥地址、区块序号和 Nonce 值等字段组成的区块头。

(2)当某一矿工成功挖到一个空区块头后,将其立即向全网广播。

(3)所有活跃节点在收到空区块头后对其进行验证,验证通过后,便以组成区块头的字段信息作为数据源,随机选取  $N$  个股权持有者。具体选择过程为:首先将本区块头的 Hash 值与前一区块头的 Hash 值进行串联,再与一个固定值串联,然后计算串联后数据的 Hash 值;对计算得到的 Hash 值利用跟随聪(follow-the-Satoshi)算法进行  $N$  次随机运算,依次生成  $N$  个幸运股权持有者。

(4)所有活跃节点判断自己是否是幸运股权持有者。如果自己是前  $N-1$  个幸运股权持有者,则用自己的私钥对区块头签名后向全网广播;如果自己是第  $N$  个幸运股权持有者,则在空区块头的基础上通过添加区块体内容来构建一个新区块,其中区块

体中主要包括尽可能多的交易、前  $N-1$  个幸运股权持有者分别对 Hash 值的签名以及自己对完整区块 Hash 值的签名。然后,将生成的区块向全网广播。

(5)所有活跃节点在收到第  $N$  个幸运股权持有者的广播区块后,对其进行验证,如果验证通过则确认该区块并连接到主链,然后继续下一个区块的挖矿过程。否则,丢弃该区块。

在以上过程中,要求所有的  $N$  个幸运股权持有者都在线,其中任意一个不在线都会导致生成的区块验证失败。PoA 算法会周期性地通过计算丢弃区块的数量来调整  $N$  值,当某一周期内丢弃区块的数量增大时,则  $N$  值会减小;否则,  $N$  值会增大。PoA 算法中,打包进区块的交易费由挖出该空区块头的矿工和  $N$  个幸运股权持有者共享。

### 3.3.3 Casper FFG 共识算法

以太坊共识算法共有 Frontier(前沿)、Homestead(家园)、Metropolis(大都会)和 Serenity(宁静)4 个阶段,前 3 个阶段采用 PoW 共识算法,第 4 个阶段将引入 PoS 算法,即 Casper 投注共识算法。该共识算法增加了惩罚机制,并基于 PoS 思想在记账节点中选取验证者。

Casper 存在多个版本,其中 Casper FFG(friendly finality gadget)<sup>[91]</sup>是 PoW 和 PoS 的结合,用于 Ethereum2.0。Casper FFG 在初始阶段会发布一个 Casper 合约,每个用户在向该合约存入以太坊的同时,要求写入一个“验证代码”从而成为一个可以参与对 PoW 产生的区块进行 PoS 共识的验证者。验证者的权益大小与存入的以太坊数量多少有关。Casper FFG 并非对每一个区块都要进行共识,而是从创世区块开始,每 100 个区块设置一个检查点,仅对检查点区块进行 PoS 共识。如图 12 所示的是 Casper FFG 共识算法的流程示意图,从根(root)开始,由检查点  $B_1, B_2, \dots, B_n$  形成检查点树(checkpoint tree),从创世区块开始每经过 100 个 PoW 区块便形成一个 PoS 检查点,从创世区块开始主链上检查点的数量即为检查

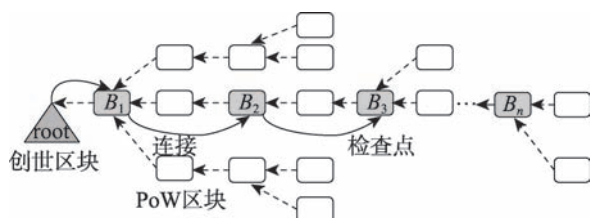


图 12 Casper FFG 共识算法流程示意图

Fig.12 Flow diagram of Casper FFG consensus algorithm

点的高度。每个验证者都要对检查点进行投票,投票的内容是由多个高度不同的检查点组成的连接,如果投票给某个连接的票数超过  $2/3$ ,则该连接被称为绝对多数连接,该连接上的所有区块将被最终确认并永久无法更改。

Casper FFG 的投票过程为:验证者生成投票消息,然后用自己的私钥进行签名后向全网广播。该消息主要由 4 条信息组成,如表 4 所示;Casper 合约在接收到投票消息后进行“苛刻条件”(slashing conditions)规则验证,以防范无利害关系(nothing at stake)<sup>[92]</sup>攻击和长程攻击。其中,针对无利害关系攻击,当验证者同时发送了  $\langle s_1, t_1, h(s_1), h(t_1) \rangle$  和  $\langle s_2, t_2, h(s_2), h(t_2) \rangle$  两条不同的投票消息时,要求  $h(t_1) \neq h(t_2)$ ,即一个矿工不能同时在两条链上挖矿,如果 Casper 合约检查到此类现象的发生,将没收该矿工(验证者)的以太坊押金;针对长程攻击威胁,Ethereum2.0 中引入了 LMD GHOST<sup>[93]</sup>分叉选择规则,如果区块链出现分叉,则选择当前检查点中验证者投票最多的链作为主链。

表 4 组成投票消息的 4 条信息

Table 4 Four pieces of information that makes up voting message

| 字段名    | 描述                      |
|--------|-------------------------|
| $s$    | 已经被确认的检查点(checkpoint)区块 |
| $t$    | 目标确认检查点区块               |
| $h(s)$ | 已确认检查点的高度               |
| $h(t)$ | 目标确认检查点的高度              |

### 3.3.4 2-hop 共识算法

如图 13 所示,2-hop(二跳)<sup>[94]</sup>共识算法以轮为单位,每轮分为 PoW 共识和 PoS 共识两个阶段,在 PoW 共识阶段生成 PoW 区块,而在 PoS 共识阶段对生成的 PoW 区块进行验证和确认。通过交替进行 PoW 共识和 PoS 共识,增加了攻击者的难度,有效防范了针对区块链 PoW 或 PoS 的 51% 攻击。

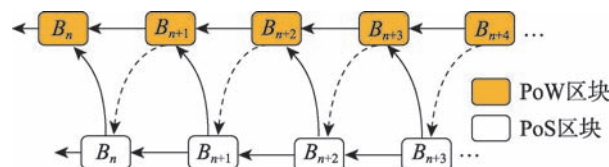


图 13 2-hop 区块链结构

Fig.13 2-hop blockchain structure

### 3.3.5 PoB 共识算法

在燃烧证明(proof of burn, PoB)<sup>[95]</sup>共识算法中,矿工通过燃烧自己的代币来竞争新区块的记账权,

燃烧的代币越多获得记账权的概率就越大。在此过程中,燃烧是指矿工将持有的代币转入到一个无法退回但可验证的特定地址的过程,矿工持有的代币可能是比特币、以太币或其他任何一个具有 PoB 协议的币种。

表 5 列出了典型的几类 PoW+PoS 共识算法主要解决的问题和典型应用场景。

**表 5 典型 PoW+PoS 共识算法的性能比较**

Table 5 Performance comparison of typical PoW+PoS consensus algorithms

| 名称                         | 年份   | 主要解决问题               | 典型应用        |
|----------------------------|------|----------------------|-------------|
| PoSV <sup>[89]</sup>       | 2014 | 屯积电币                 | Reddcoin    |
| PoA <sup>[90]</sup>        | 2014 | 持币者不积极参与挖矿           | Decred      |
| Casper FFG <sup>[91]</sup> | 2017 | 无利害关系攻击、长程攻击         | Ethereum2.0 |
| 2-hop <sup>[94]</sup>      | 2020 | 51%算力(PoW)或权益(PoS)攻击 | —           |
| PoB <sup>[95]</sup>        | 2020 | 屯积电币                 | Slimcoin    |

### 3.4 PoW/PoS+BFT/PBFT 混合共识算法

为解决非拜占庭容错的传统分布式共识算法难以适应大部分区块链应用场景的不足,提出 PoW/PoS+BFT/PBFT 系列混合共识算法。该类算法将典型区块链共识算法 PoW/PoS 与经典分布式共识算法 BFT/PBFT 结合,通常利用 PoW/PoS 选择产生一个或多个分片(shard),然后在分片内部使用 BFT/PBFT 生成区块。这里的分片也称为委员会,它是由负责对全网交易信息进行处理的节点组成的集合,如果所有交易在同一个集合中完成,则称为单一片,如果在不同集合内部分别处理不同的交易信息,则称为多分片。

#### 3.4.1 PoW+BFT 单一片共识算法

PoW 共识算法在比特币系统中取得了巨大成功,但在确认时间和激励机制方面遇到了挑战。为此,Abraham 等人<sup>[96]</sup>提出了 Solida 混合共识算法,该算法是一种基于可重构 BFT 共识的去中心化区块链协议。它通过借鉴 Paxos 共识算法中的领导者选举机制,有效解决了委员会重配置问题,并在恶意节点数小于节点总数 1/3 的情况下提供安全性和活性。Solida 共识算法主要包括以下两个过程:

(1)交易确认过程。①领导者对接收到的交易计算哈希值后打包进区块,并将区块和对应交易的哈希值向全网广播;②接收到领导者广播消息的成员验证交易的正确性和领导者身份的合法性,验证通过后将该消息继续向全网广播;③如果其中一个

成员接收到  $2f+1$  个其他成员针对同一区块消息的广播,就接受该区块消息,并将该  $2f+1$  个消息组成一个接受证明(accept certificate)消息后向全网广播;④其他成员在接收到该接受证明后,经验证无误后向全网广播;⑤如果一个成员接收到了  $2f+1$  个合法的接受证明消息,便对该区块信息进行最终确认,并将接收到的  $2f+1$  个成员的消息组成一个承诺证明(commit certificate)后向全网广播;⑥网络中的用户在接收到该承诺证明后验证其消息的合法性,通过验证后最终完成交易的确认。

(2)重配置过程。系统设定一个具有一定难度的哈希函数求解问题,求得 PoW 解的节点将其结果广播给委员会成员;委员会成员对 PoW 解进行验证,验证通过后将求得解的节点作为新的领导者;当新领导者产生后,所有节点将在“交易确认过程”阶段已确认的交易哈希值等相关状态消息发送给新领导者,当新领导者接收到  $2f+1$  个状态消息后,将其组合成一个状态证明(status certificate)消息,然后进行广播。接收到状态证明的节点确认已打包的交易信息,并进入由新委员会成员组成的新一轮一致性共识过程。

在 Solida 混合共识算法中,通过 PoW 共识算法实现新领导者的选举,完成新委员会成员的重新配置;利用 BFT 算法完成委员会内部成员之间的一致性共识,确保委员会内部有超过 2/3 诚实成员的前提下便能够取得一致性共识。

另外,在由 Pass 等人<sup>[97]</sup>提出的 Hybrid consensus 混合共识算法中,通过结合低效的 PoW 共识算法和快速的 BFT 授权共识算法,实现在非授权环境下达成快速共识。

#### 3.4.2 PoW+BFT 多分片共识算法

Chainspace 共识算法是由 Al-Bassam 等人<sup>[98]</sup>提出的一个支持多用户自定义智能合约的分布式账本平台。Chainspace 提出了一种称为分片拜占庭原子提交(sharded Byzantine atomic commit, S-BAC)的可跨多个拜占庭节点分片(委员会)处理通用智能合约交易的协议,S-BAC 协议结合了 BFT 和原子提交,实现了拜占庭和异步环境下交易处理的分片内共识。在 Chainspace 中,智能合约的执行和验证过程分开执行,其对象(交易或智能合约)的执行过程如下(如图 14 所示,其中用户提交了一个带有 2 个输入和 1 个输出的对象):

(1)准备(prepare)。用户执行交易启动程序,并

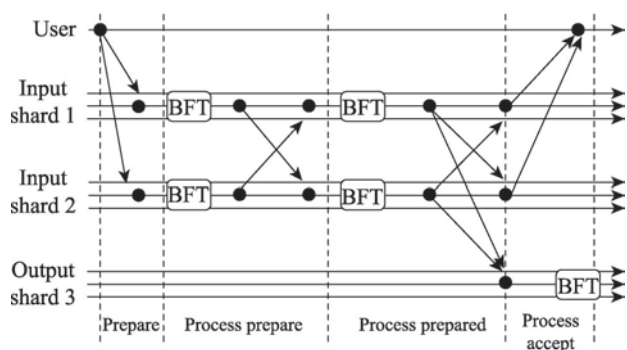


图 14 Chainspace 共识算法流程示意图

Fig.14 Flow diagram of Chainspace consensus algorithm

发送交易消息  $\text{prepare}(T)$  给至少一个诚实节点, 确保至少有一个诚实节点接收到该交易。本例中, 交易有分片 1 和分片 2 两个输入和分片 3 一个输出; 节点在接收到  $\text{prepare}(T)$  消息后, 每个分片中的节点将跨分片执行两阶段提交协议,  $\text{prepare}(T)$  消息在分片内部通过 BFT 达成共识。

(2) 过程准备 (process prepare)。分片 1 和分片 2 中的节点在接收到  $\text{prepare}(T)$  消息后, 在各委员会内部通过 BFT 共识算法来处理交易信息: 如果交易有效, 则提交该交易, 并向委员会内部所有节点广播  $\text{prepared}(T, \text{commit})$  承诺消息; 如果交易无效, 则终止该交易, 并向委员会内部所有节点广播  $\text{prepared}(T, \text{abort})$  取消消息。

(3) 过程准备完成 (process prepared)。通过原子提交协议在不同分片之间进行  $\text{prepare}(T)$  消息交互, 如果不同分片之间交互的全部都是  $\text{prepared}(T, \text{commit})$  消息, 则交易最终达成共识, 并广播  $\text{accept}(T, \text{commit})$  消息; 如果其中有一个分片提交的是  $\text{prepared}(T, \text{abort})$  消息, 则交易共识失败, 并广播  $\text{accept}(T, \text{abort})$  消息。

(4) 过程接受 (process accept)。当某一分片中存在  $\text{accept}(T, \text{commit})$  广播消息时, 将在输入分片的委员会内部运行 BFT 共识算法, 将交易交给分片进行管理; 当某一分片中存在  $\text{accept}(T, \text{abort})$  广播消息时, 所有分片对交易的输入状态进行解锁后, 将其丢弃。

本文在一个真实的基于云环境的测试平台上对 Chainspace 算法进行了评估, 其交易吞吐量随分片数量的增加而线性增长, 每增加一个分片, 每秒可提升 22 个交易的处理能力。Chainspace 为集中式的授权系统提供了一个高效的解决方案。

2018 年, Zamani 等人提出的 RapidChain<sup>[99]</sup> 混合

共识算法是一个基于分片的公有链协议, 它采用了委员会内共识算法, 在拜占庭节点不超过 1/3 总节点数的非许可网络环境下能够最终取得共识的一致性, 并实现处理交易的通信、计算和存储开销的完整分片。本文通过实验表明, 在一个由 4 000 个节点组成的网络中, RapidChain 算法的吞吐量超过 7 300 TPS, 而预期确认延迟约为 8.7 s。

### 3.4.3 PoW+PBFT 单一分片共识算法

虽然比特币取得了巨大成功, 但过长的交易确认时间严重影响了 PoW 共识协议的应用拓展, Decker 等人提出的 PeerCensus 算法<sup>[100]</sup> 被认为是最早成功地将 PBFT 与比特币区块链技术相结合形成的一种在确保安全的前提下达成快速一致性的混合共识算法。如图 15 所示, PeerCensus 算法主要由底层的区块链和位于中间层的链协议 (chain agreement, CA) 两部分组成。其中, 底层的比特币区块链运行 PoW 共识选出一定数量的节点, 对这些节点完成身份认证后提交给 CA; CA 通过 PBFT 算法来实现具体功能。CA 主要有两项任务: 一是跟踪系统中的成员, 即当前有哪些节点在线并参与共识, 以确保 PBFT 能够发挥其作用; 二是解决区块链分叉问题, 来实现最终的强一致性。

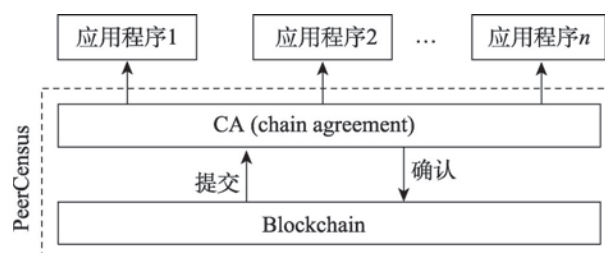


图 15 PeerCensus 共识算法组成示意图

Fig.15 Schematic diagram of PeerCensus consensus algorithm

为了演示 PeerCensus 的应用效果, 作者在论文中引入了一种名为 Discoin 的新型加密货币。在 Discoin 中, 交易提交给主服务器, 由主服务器给每一个交易分配序列号, 并将其保存在交易的历史记录中。由于交易是完全有序的, 这样有效解决了双重花费问题, 并且在提交时, 所有节点都会认可一个共同的交易历史记录, 提升了安全性。

另一个基于 PoW 和 PBFT 混合共识的典型算法是 ByzCoin<sup>[63]</sup>。该算法采用 PoW 选举分片节点 (委员会成员), 分片内部采用改进的 PBFT 算法完成共识。当区块大小为 32 MB、分片中的节点数为 144

时,算法每秒的交易量可以达到974。

### 3.4.4 PoW+PBFT 多分片共识算法

2016年,Luu等人提出了ELASTICO混合算法<sup>[64]</sup>,最先将多分片技术应用到区块链中。ELASTICO首先通过PoW共识竞争选举网络中的记账节点,然后按照预先确定的规则,将选举得到的记账节点分配到不同的分片委员会中。每个分片委员会内部执行PBFT算法打包生成交易集合。该交易集合经签名后被提交给共识委员会。共识委员会在验证签名后,最终将所有的交易集合打包成区块并记录在区块链上。

2018年,Kokoris-Kogias等人在ELASTICO算法的基础上提出了基于分片的区块链共识算法OmniLedger<sup>[101]</sup>。该算法使用RandHound<sup>[102]</sup>和VRF协议将节点随机分配到不同的分片上,每个分片内部的共识采用PBFT算法(具体为ByzCoinX共识算法)。OmniLedger的总体结构如图16所示,由一条用于记录不同窗口期(epoch length)分片及其包含的节点信息的身份链(identity blockchain)和多条负责产生和维护本分片内部交易信息的分片链(shard blockchain)组成。OmniLedger共识算法的执行过程如下:

(1)节点注册。在当前窗口期内,节点参与PoW共识,将计算得到的难度值和节点身份信息进行广播。领导者在当前分片内运行PBFT共识算法,将收集到的所有合法节点信息注册到身份链。

(2)节点分配。首先,每个已注册的节点通过 $ticket_{i,e,v} = VRF_{sk_i}("leader" || config_e || v)$ 计算自己的票据值,其中 $VRF_{sk_i}(\cdot)$ 代表VRF函数, $leader$ 表示本次VRF计算目的是选择领导者, $config_e$ 是注册在身份链上的所有的节点信息, $v$ 是当前计算的轮数(同一深度的

区块,可能需要多轮共识才能确定), $i$ 为节点序号, $e$ 为当前窗口期。在一定时间内,所有节点交换自己的 $ticket_{i,e,v}$ ,然后将 $ticket_{i,e,v}$ 值最小的节点选举为当前的领导者;在确定了领导者后,新的领导者启动无偏差随机数生成协议RandHound,将 $m$ 个合法节点分配到 $n$ 个分片中。

(3)跨分片操作。为了支持分片间的交易,OmniLedger使用了UTXO账户模型。假设用户分别从Shard 1和Shard 2向Shard 3转账,主要操作阶段为:①初始化。用户(client)上传交易,交易的输入分片为Shard 1和Shard 2,输出分片为Shard 3。②锁定。Shard 1和Shard 2中的领导者在接收到交易输入后,锁定请求并验证交易的合法性,然后记录锁定状态以及合法交易在本分片UTXO中的Merkle树路径。③解锁。分两种情况:一是提交跨分片确认请求。如果Shard 1和Shard 2中的交易输入都合法,那么用户将向Shard 3提交确认信息。Shard 3在接收到交易后,验证所有交易的合法性,通过验证后将交易打包进当前区块。另一种情况是交易不合法后的回滚。如果在“锁定”阶段发现某个分片中的交易不合法,那么客户端将创建一个“取消交易”消息,并广播给所有参与该交易的分片,所有输入分片中的领导者在接收到该消息后,将原交易的输入状态解锁到可用状态,实现跨分片交易的回滚。

(4)分片中节点的重配置。为了提高系统的稳定性,分片中节点的更换不能太频繁,具体可通过设置合理的PoW难度值,使得每次重新配置时,更换掉的节点数不超过总节点数的1/3。

OmniLedger采用了在出块速度 and 安全性之间进行平衡的“先信任后验证”(trust-but-verify)分层处理

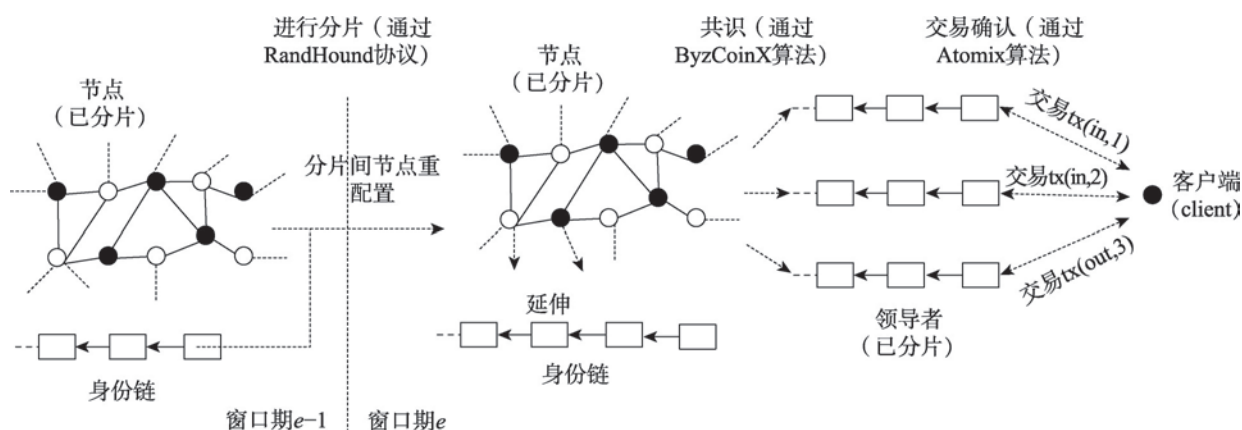


图16 OmniLedger共识算法总体架构

Fig.16 Overall architecture of OmniLedger consensus algorithm

机制,分片共识可采用较少的节点,以加快分片共识以及区块确认速度,分片形成的区块再被更多的节点进行再次验证。另外,OmniLedger还采用了区块镜像(snapshot)和区块并行处理技术,以减小对内存资源的占用,并提高共识效率。OmniLedger在25个分片的情况下,交易量可以达到13 000 TPS。

2019年,Wang等人提出了Monoxide算法<sup>[103]</sup>,该算法将分片技术和传统PoW共识有机结合,并借助提出的“Chu-ko-nu mining”算法,实现了每个矿工可以同时在不同的分片进行探矿,在解决了因分片造成的算法分散这一缺陷的同时,还提高了系统抗攻击能力和出块速度。

### 3.4.5 PoS+BFT单一分片共识算法

Algorand<sup>[104]</sup>是由Gilad等人提出的PoS+BFT单一分片共识算法中具有代表性的混合算法。Algorand算法的共识过程主要分为委员会成员选举和形成共识两个阶段:委员会成员选举阶段利用基于VRF协议的PoS共识选举出委员会领导者和成员(验证者),其中领导者负责创建区块,而验证者负责对创建的区块进行共识;共识阶段通过运行称为“BA★”的新的BFT协议对区块达成共识。BA★是Algorand算法的核心。

Algorand算法中主要使用的符号有:( $r,s$ )表示VRF函数的第 $r$ 轮中的第 $s$ 步; $\ell^r$ 表示第 $r$ 轮的领导者; $SV^{r,s}$ 表示( $r,s$ )的验证者集合; $PAY^r$ 表示区块 $B_i^r$ 中包含的交易集合; $B_i^r$ 表示由第 $r$ 轮中节点 $i$ 生成的区块; $Q^r$ 表示第 $r$ 轮的种子,该种子用于选择领导者和成员,其计算公式为 $Q^r = H(SIG_{\ell^r}(Q^{r-1}), r)$ ;  $SIG_{\ell^r}(Q^{r-1})$ 表示 $\ell^r$ 利用自己的私钥对 $Q^{r-1}$ 的签名; $\delta_i^{r,s}$ 表示节点 $i$ 的签名 $SIG_i(r,s,Q^{r-1})$ ,用于证明 $i \in SV^{r,s}$ ;  $m_i^{r,s}$ 表示节点 $i$ 在( $r,s$ )广播的消息。Algorand算法的主要运行过程包括:

(1)选择随机数。在VRF协议的每一轮,节点首先获取上个区块的信息 $B^{r-1} = (r-1, PAY^{r-1}, SIG_{\ell^{r-1}}(Q^{r-2}), H(B^{r-2}))$ ,然后通过计算 $Q^{r-1} = H(SIG_{\ell^{r-1}}(Q^{r-2}), r-1)$ 得到本轮的随机数。

(2)选择委员会领导者(出块节点)和成员(验证者节点)。每个节点运行VRF函数,根据函数值来判断自己的身份:如果满足 $H(SIG_i(r,1,Q^{r-1})) \leq p$ ,则该节点被选举为本轮的领导者;如果满足 $H(SIG_i(r,1,Q^{r-1})) \leq p'$ ,则该节点被选举为本轮委员会的成员。其中, $p$ 和 $p'$ 表示概率。

(3)创建区块。选举出的本轮领导者 $\ell^r$ 生成区块 $B_i^r = (r, PAY^r, SIG_{\ell^r}(Q^{r-1}), H(B^{r-1}))$ ,同时生成 $m_i^{r,1} = (B_i^r, esig_i(H(B_i^r), \delta_i^{r,1}))$ ,其中 $esig_i(H(B_i^r))$ 表示用 $\delta_i^{r,1}$ 进行的签名。将 $m_i^{r,1}$ 和 $B_i^r$ 广播到区块链网络中。

(4)验证区块。在当前轮 $r$ 的委员会内部运行BA★算法验证区块的合法性。BA★算法使用VRF协议以一种私密和非交互的方式来随机选择用户,该算法主要由分级共识(graded consensus, GC)和二元拜占庭一致(binary Byzantine agreement, BBA)算法组成。其中GC的主要功能是从收到的多个 $m_i^{r,1}$ 中找到 $\ell^r$ ,即 $H(\delta_i^{r,1})$ 值最小的节点,然后验证 $\ell^r$ 发来的区块 $B_{\ell^r}^r$ 的合法性,验证通过后将 $\ell^r$ 对应的 $m_{\ell^r}^{r,1}$ 进行广播;BBA的主要功能是对接收到的来自不同节点的 $m_i^{r,s'-1}$ 进行验证。如果验证通过的节点数超过2/3则表示该区块合法,并链接到主链上。

Algorand在由1 000台虚拟机组成的实验环境中模拟了多达500 000个用户,结果表明交易在不到1 min的时间内得到了确认,实现了125倍于比特币的吞吐量。

2017年,被称为“中国以太坊”的NEO(小蚁)区块链中采用了授权拜占庭容错(delegated Byzantine fault tolerance, dBFT)算法<sup>[105]</sup>。该算法是一种通过代理投票来实现大规模节点参与共识的BFT算法。在NEO中,管理代币的持有者通过PoS共识投票选出其所支持的记账人,然后被选出的记账人之间通过BFT算法达成共识并生成区块。在公有链中,NEO在dBFT的支持下每15~20 s可以生成一个区块,交易吞吐量达到1 000 TPS。

区块链混合共识算法充分吸收了原来单一共识算法的优势,并经相互结合后克服了原来单一共识在出块速度、吞吐量、安全性等方面存在的不足,使得区块链能够适应更多的应用场景。表6给出了几类典型混合共识算法在核心算法、分片类型、通信模型和应用场景等方面的主要性能比较。

## 4 总结与展望

区块链技术在与实体产业的融合过程中正在加速数字经济的发展。作为区块链核心技术的共识算法,能够在节点高度分散且不存在彼此间信任机制的网络环境中就某一具体事务达成结果的一致性,且对过程和结果都不存在分歧。严格地讲,除能源消耗外,共识算法不存在绝对的谁最适合谁的问题。

表6 典型 PoW/PoS+BFT/PBFT 共识算法的性能比较

Table 6 Performance comparison of typical PoW/PoS+BFT/PBFT consensus algorithms

| 算法名称                             | 年份   | 核心算法     | 分片类型 | 通信模型 | 应用场景        |
|----------------------------------|------|----------|------|------|-------------|
| Solida <sup>[96]</sup>           | 2016 | PoW+BFT  | 单一分片 | 异步   | 公有链         |
| Hybrid consensus <sup>[97]</sup> | 2017 | PoW+BFT  | 单一分片 | 异步   | 公有链         |
| Chainspace <sup>[98]</sup>       | 2017 | PoW+BFT  | 多分片  | 异步   | 联盟链         |
| RapidChain <sup>[99]</sup>       | 2018 | PoW+BFT  | 多分片  | 部分同步 | 公有链         |
| PeerCensus <sup>[100]</sup>      | 2014 | PoW+PBFT | 单一分片 | 部分同步 | 公有链         |
| ByzCoin <sup>[63]</sup>          | 2016 | PoW+PBFT | 单一分片 | 部分同步 | 公有链         |
| OmniLedger <sup>[101]</sup>      | 2018 | PoW+PBFT | 多分片  | 同步   | 联盟链/<br>公有链 |
| ELASTICO <sup>[64]</sup>         | 2016 | PoW+PBFT | 多分片  | 同步   | 联盟链/<br>公有链 |
| Monoxide <sup>[103]</sup>        | 2019 | PoW+PBFT | 多分片  | 同步   | 联盟链/<br>公有链 |
| Algorand <sup>[104]</sup>        | 2017 | PoS+BFT  | 单一分片 | 部分同步 | 公有链         |
| dBFT <sup>[105]</sup>            | 2020 | PoS+BFT  | 单一分片 | 异步   | 公有链         |

题。通常情况下,共识算法的选择是在针对具体应用场景的前提下,在效率、安全性、稳定性之间的折衷和平衡。

本文重点从区块链产生之前的经典分布式共识和区块链共识两方面着手,就单一共识算法和混合共识算法进行了系统性的分析和梳理,并选择了部分典型算法进行了较为系统的介绍。基于目前的研究现状,针对区块链共识算法的发展,重点需要研究和解决性能与可扩展性、激励机制、安全与隐私、并行处理等方面的问题。

首先,区块链的性能和可扩展性是影响当前区块链共识算法的关键因素,分片技术通过将节点分组后分割处理交易,多个分片并行交易,以最大限度地提高性能,同时大大减少每个节点的通信、计算和存储开销,从而使系统能够扩展到大型网络环境。然而,现有的单一共识和混合共识算法大多没有更好地发挥分片的更多优势,吞吐量和延迟仍然是主要的瓶颈。同时这些协议实现的安全保障很弱且故障率较高,许多性能的实现还依赖于大量的假设(如可信、同步网络环境等),限制了对主流系统的适用性。下一步还需要重点在算法优化和工程应用中进一步研究。其中,如何将经典的传统一致性算法更好地“移植”到区块链应用环境,在发挥传统算法已形成的特定性能优势的同时,解决某些区块链应用场景下对共识算法的具体要求,还需要在算法实现

的理论研究和工程实践上进行深入探讨。

其次,激励机制是区块链共识算法的重要组成部分,是决定具体算法能否满足特定应用场景要求的关键,比特币取得今天的成功有力地证明了这一点。受比特币的影响,早期的激励机制主要来源于经济刺激,矿工积极、忠诚、高效地参与挖矿主要是为了获得经济利益。然而随着区块链应用场景的多元化,在单一经济激励机制发挥的功能逐渐被弱化的情况下,如何实现共识算法的安全性、一致性和活性,还需要针对具体应用场景对算法进行优化,在共识与激励二元耦合过程中寻得平衡。试想一下,如果一个区块链系统的共识算法中不再存在激励机制,那么区块链的安全性、去中心化、开放自治等功能将如何实现。没有这些功能特征的系统已不再是一个区块链系统。但如果过于强调经济激励机制在共识算法中的作用,则势必在安全性、效率、过扩展性等方面带来新的挑战。因此,从某种程度上讲,区块链共识提供的是一种解决问题的思路,而不是一个固定的范式。

还有,在区块链共识算法中,隐私与安全相互依存,但各有侧重<sup>[106]</sup>。安全更关注于系统的抗攻击能力,而隐私则聚焦于对个人及相关范围信息的管控。共识过程是一个多节点通过频繁交换信息以达成最终一致性的过程,期间不但会受到传统网络攻击(如DDoS攻击、女巫攻击等)的威胁,攻击者更会利用具体算法存在的缺陷进行有针对性的攻击破坏(如针对特定共识算法的双花攻击、自私挖矿攻击、贿赂攻击等)。因此,针对区块链共识算法安全性的研究需要在综合各类复杂网络环境的前提下,再就具体的算法实现进行研究。同时,还要综合安全、去中心化和可扩展性之间的平衡;区块链共识过程中涉及到的隐私主要包括节点交易隐私、账户地址隐私、用户身份隐私和智能合约隐私等方面,攻击者基于算法设计和实现过程中存在的缺陷,分析账本内容和合约执行代码,从中获得与用户身份相关的信息,导致隐私泄露。针对共识环节的隐私保护,目前的研究主要集中在信息混淆、信息加密、通道隔离和访问与授权管理等领域。

最后,随着区块链应用领域的不断拓展,传统的单一链式数据结构因其效率低下而无法适应大规模高吞吐量应用场景的需求,以DAG为代表的“图型”数据结构和存储结构通过并行处理来替代链式处理有效弥补了区块链的原生缺陷。这些新型区块链数

据结构带来的共识算法研究成为当前和未来研究的趋势之一。例如,如何将 DAG 结构和分区技术进行结合,以此来改变区块链系统的数据结构和存储结构,进而可弥补分布式账本在可扩展性、吞吐率、交易确认时间等方面的原生缺陷;再如,随着区块链技术在物联网中应用的不断深入,区块链为解决物联网数据存储和共享的安全性提供了技术保障,但海量的物联网数据对区块链共识性能提出了挑战。为此,如何将区块链中的分片机制与物联网环境中的云计算/边缘计算等架构有机结合,从而提高数据处理的效率,将是一个迫切需要解决的热点问题等。

### 参考文献:

- [1] 王群,李馥娟,王振力,等. 区块链原理及关键技术[J]. 计算机科学与探索, 2020, 14(10): 1621-1643.  
WANG Q, LI F J, WANG Z L, et al. Principle and core technology of blockchain[J]. Journal of Frontiers of Computer Science and Technology, 2020, 14(10): 1621-1643.
- [2] MIRAZ M H. Blockchain: technology fundamentals of the trust machine[EB/OL]. (2017-12-23)[2021-07-12]. <https://www.legalanalytics.law.cuhk.edu.hk/post/2017/12/23/blockchain-technology-fundamentals-of-the-trust-machine>.
- [3] FANTI G, VISWANATH P. Anonymity properties of the bitcoin P2P network[J]. arXiv:1703.08761, 2017.
- [4] NAKAMOTO S. Bitcoin: a peer-to-peer electronic cash system[EB/OL]. (2018-06-10)[2021-07-13]. <https://bitcoin.org/bitcoin.pdf>.
- [5] 袁勇,倪晓春,曾帅,等. 区块链共识算法的发展现状与展望[J]. 自动化学报, 2018, 44(11): 2011-2022.  
YUAN Y, NI X C, ZENG S, et al. Blockchain consensus algorithms: the state of the art and future trends[J]. Acta Automatica Sinica, 2018, 44(11): 2011-2022.
- [6] 刘懿中,刘建伟,张宗洋,等. 区块链共识机制研究综述[J]. 密码学报, 2019, 6(4): 395-432.  
LIU Y Z, LIU J W, ZHANG Z Y, et al. Overview on blockchain consensus mechanisms[J]. Journal of Cryptologic Research, 2019, 6(4): 395-432.
- [7] 夏清,窦文生,郭凯文,等. 区块链共识协议综述[J]. 软件学报, 2021, 32(2): 277-299.  
XIA Q, DOU W S, GUO K W, et al. Survey on blockchain consensus protocol[J]. Journal of Software, 2021, 32(2): 277-299.
- [8] 靳世雄,张潇丹,葛敬国,等. 区块链共识算法研究综述[J]. 信息安全学报, 2021, 6(2): 85-100.  
JIN S X, ZHANG X D, GE J G, et al. Overview of blockchain consensus algorithm[J]. Journal of Cyber Security, 2021, 6(2): 85-100.
- [9] NGUYEN G T, KIM K. A survey about consensus algorithms used in blockchain[J]. Journal of Information Processing Systems, 2018, 14(1): 101-128.
- [10] YIU N. An overview of forks and coordination in blockchain development[J]. arXiv:2102.10006, 2021.
- [11] LASHKARI B, MUSILEK P. A comprehensive review of blockchain consensus mechanisms[J]. IEEE Access, 2021, 9(3): 43620-43652.
- [12] BOURAGA S. A taxonomy of blockchain consensus protocols: a survey and classification framework- ScienceDirect [J]. Expert Systems with Applications, 2021, 168: 114384.
- [13] LAMPORT L. How to make a multiprocessor computer that correctly executes multiprocess programs[J]. IEEE Transactions on Computers, 1979, 28(9): 690-691.
- [14] KRISHNA S, EMMI M, ENEA C, et al. Verifying visibility-based weak consistency[C]//LNCS 12075: Proceedings of the 29th European Symposium on Programming, Dublin, Apr 25-30, 2020. Cham: Springer, 2020: 280-307.
- [15] MA Z F. Handbook of blockchain technology development [M]. Beijing: Tsinghua University Press, 2021.  
马兆丰. 区块链技术开发指南[M]. 北京: 清华大学出版社, 2021.
- [16] HERRERA- JOANCOMARTÍ J. Research and challenges on bitcoin anonymity[C]//LNCS 8872: Proceedings of the 9th International Workshop on Data Privacy Management, Autonomous Spontaneous Security, and Security Assurance, Wroclaw, Sep 10-11, 2014. Cham: Springer, 2015: 3-16.
- [17] BADARI A, CHAUDHURY A. An overview of bitcoin and ethereum white-papers, forks, and prices[EB/OL]. (2021-05-11)[2021-08-10]. [https://papers.ssrn.com/sol3/papers.cfm?abstract\\_id=3841827](https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3841827).
- [18] CACHIN C. Architecture of the hyperledger blockchain fabric[EB/OL]. (2016-01-01)[2021-08-16]. [https://www.zurich.ibm.com/dcc/papers/cachin\\_dccl.pdf](https://www.zurich.ibm.com/dcc/papers/cachin_dccl.pdf).
- [19] ABRAHAM I, MALKHI D, NAYAK K, et al. Solida: a blockchain protocol based on reconfigurable Byzantine consensus[C]//Proceedings of the 21st International Conference on Principles of Distributed Systems, Lisbon, Dec 18-20, 2017: 25.
- [20] MANEAS A, CHONDROS N, DIAMANTOPOULOS P, et al. On achieving interactive consistency in real-world distributed systems[J]. Journal of Parallel and Distributed Computing, 2021, 147: 220-235.
- [21] AKKOYUNLU E A, EKANADHAM K, HUBER R V. Some constraints and tradeoffs in the design of network communications[J]. ACM SIGOPS Operating Systems Review, 1975, 9(5): 67-74.

- [22] LAMPORT L, SHOSTAK R, PEASE M. The Byzantine generals problem[J]. *ACM Transactions on Programming Languages and Systems*, 1982, 4(3): 382-401.
- [23] COWLING J, MYERS D S, LISKOV B, et al. HQ Replication: a hybrid quorum protocol for Byzantine fault tolerance [C]//*Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, Seattle, Nov 6-8, 2003. New Orleans: USENIX Association, 2003: 177-190.
- [24] LAMPORT L. The part-time parliament[J]. *ACM Transactions on Computer Systems*, 1998, 16(2): 133-169.
- [25] OKI B M, LISKOV B H. Viewstamped replication: a new primary copy method to support highly-available distributed systems[C]//*Proceedings of the 7th Annual Conference on Principles of Distributed Computing*, Toronto, Aug 15-17, 1988. New York: ACM, 1988: 8-17.
- [26] FAZLALI M, EFTEKHAR S M, DEHSHIBI M M, et al. Raft consensus algorithm: an effective substitute for Paxos in high throughput P2P-based systems[J]. *arXiv:1911.01231*, 2019.
- [27] CASTRO M, LISKOV B. Practical Byzantine fault tolerance[C]//*Proceedings of the 3rd USENIX Symposium on Operating Systems Design and Implementation*, New Orleans, Feb 22-25, 1999. New Orleans: USENIX Association, 1999: 173-186.
- [28] THAI Q T, YIM J C, YOO T W, et al. Hierarchical Byzantine fault-tolerance protocol for permissioned blockchain systems[J]. *The Journal of Supercomputing*, 2019, 75(11): 7337-7365.
- [29] CASTRO M, LISKOV B. Practical Byzantine fault tolerance and proactive recovery[J]. *ACM Transactions on Computer Systems*, 2002, 20(4): 398-461.
- [30] JEHL L. Quorum selection for Byzantine fault tolerance [C]//*Proceedings of the 39th International Conference on Distributed Computing Systems*, Dallas, Jul 7-10, 2019. Piscataway: IEEE, 2019: 2168-2177.
- [31] FISCHER M J, LYNCH N A, PATERSON M S. Impossibility of distributed consensus with one faulty process[J]. *Journal of the ACM*, 1985, 32(2): 374-382.
- [32] WANG Y G. Blockchain BFT protocol for complete asynchronous networks[J]. *arXiv:2005.04309*, 2020.
- [33] MUÑOZ-ESCOÍ F D, DE JUAN-MARÍN R, GARCÍA-ESCRIVÁ J R, et al. CAP theorem: revision of its related consistency models[J]. *The Computer Journal*, 2019, 62(6): 943-960.
- [34] KLEPPMANN M. A critique of the CAP theorem[J]. *arXiv:1509.05393*, 2015.
- [35] En-Tan-Mo. En-Tan-Mo science[EB/OL]. (2019-11-27) [2021-08-10]. [http://www.entanmo.com/docs/ETM%20Science\\_en-GB.pdf](http://www.entanmo.com/docs/ETM%20Science_en-GB.pdf).
- [36] HAN R, FOUTRIS N, KOTSELIDIS C. Demystifying cryptomining: analysis and optimizations of memory-hard POW algorithms[C]//*Proceedings of the 2019 IEEE International Symposium on Performance Analysis of Systems and Software*, Madison, Mar 24-26, 2019. Piscataway: IEEE, 2019: 22-33.
- [37] JIANG X J, LIU X F. CryptoKitties transaction network analysis: the rise and fall of the first blockchain game mania [J]. *Frontiers in Physics*, 2021, 9: 631665.
- [38] GARCÍA-PÉREZ Á, GOTSMAN A, MESHMAN Y, et al. Paxos consensus, deconstructed and abstracted[C]//*LNCS 10801: Proceedings of the 27th European Symposium on Programming*, Thessaloniki, Apr 14-20, 2018. Cham: Springer, 2018: 912-939.
- [39] LAMPORT L. Paxos made simple[J]. *ACM SIGACT News*, 2001, 32(4): 18-25.
- [40] LAMPORT L. Fastpaxos[J]. *Distributed Computing*, 2006, 19(2): 79-103.
- [41] BURROWS M. The Chubby lock service for loosely-coupled distributed systems[C]//*Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, Seattle, Nov 6-8, 2006. New Orleans: USENIX Association, 2006: 335-350.
- [42] KRZYZANOWSKI P. Google chubby[EB/OL]. (2021-03-01)[2021-07-12]. <https://people.cs.rutgers.edu/~pxk/417/notes/chubby.html>.
- [43] ISARD M. Autopilot: automatic data center management [J]. *ACM SIGOPS Operating Systems Review*, 2007, 41(2): 60-67.
- [44] ZHENG J J, LIN Q, XU J T, et al. Paxos store: high-availability storage made practical in wechat[J]. *Proceedings of the VLDB Endowment*, 2017, 10(12): 1730-1741.
- [45] HUANG D Y, MA X L, ZHANG S L. Performance analysis of the raft consensus algorithm for private blockchains[J]. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019, 50(1): 172-181.
- [46] HUNT P, KONAR M, JUNQUEIRA F, et al. Zookeeper: wait-free coordination for internet-scale systems[C]//*Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, Boston, Jun 23-25, 2010. New Orleans: USENIX Association, 2010: 11-25.
- [47] etcd. A distributed, reliable key-value store for the most critical data of a distributed system[EB/OL]. (2021-06-15) [2021-07-15]. <https://etcd.io/>.
- [48] JAKOBSSON M, JUELS A. Proofs of work and bread pudding protocols[C]//*Proceedings of the IFIP TC6/TC11 Joint Working Conference on Secure Information Networks*:

- Communications and Multimedia Security, Netherlands, Sep 20-21, 1999. Boston: Kluwer Academic Publishers, 1999: 258-272.
- [49] BACK A. Hashcash—a denial of service counter-measure [EB/OL]. (2002-08-01)[2021-08-30]. <http://www.hashcash.org/papers/hashcash.pdf>.
- [50] WAHAB A, MEHMOOD W. Survey of consensus protocols [J]. arXiv:1810.03357, 2018.
- [51] CONTI M, KUMAR E S, LAL C, et al. A survey on security and privacy issues of bitcoin[J]. IEEE Communications Surveys & Tutorials, 2018, 20(4): 3416-3452.
- [52] MUKHERJEE P, PRADHAN C. Blockchain 1.0 to Blockchain 4.0—the evolutionary transformation of blockchain technology[EB/OL]. (2021-05-01)[2021-09-01]. [https://www.researchgate.net/publication/351263701\\_Blockchain\\_10\\_to\\_Blockchain\\_40-The\\_Evolutionary\\_Transformation\\_of\\_Blockchain\\_Technology](https://www.researchgate.net/publication/351263701_Blockchain_10_to_Blockchain_40-The_Evolutionary_Transformation_of_Blockchain_Technology).
- [53] CHERNET H F, JILLEDI S K. A next-generation smart contract and decentralized blockchain platform: a case study on ethiopia[J]. Journal of Business Analytics and Data Visualization, 2020, 1(1): 28-34.
- [54] BRÜNJES L, GABBAY M J. UTxO- vs account-based smart contract blockchain programming paradigms[C]//LNCS 12478: Proceedings of the 9th International Symposium on Leveraging Applications of Formal Methods, Rhodes, Oct 20-30, 2020. Cham: Springer, 2020: 73-88.
- [55] BENCIC F M, ZARKO I P. Distributed ledger technology: blockchain compared to directed acyclic graph[C]//Proceedings of the 38th IEEE International Conference on Distributed Computing Systems, Vienna, Jul 2-6, 2018. Washington: IEEE Computer Society, 2018: 1569-1570.
- [56] Ethereum Wiki. ethash[EB/OL]. (2020-11-06)[2021-09-02]. <https://eth.wiki/en/concepts/ethash/ethash>.
- [57] TRADING C. X11 mining calculator[EB/OL]. (2020-07-21)[2021-09-02]. <https://www2.ict.ufvjm.edu.br/index.php/2020/07/21/x11-mining-calculator/>.
- [58] PADMAVATHI M, SURESH R M. Secure P2P intelligent network transaction using litecoin[J]. Mobile Networks & Applications, 2019, 24(2): 318-326.
- [59] DASH. What is dash?[EB/OL]. (2021-06-03)[2021-09-01]. <https://dashplatform.readme.io/docs/introduction-what-is-dash>.
- [60] SOMPOLINSKY Y, ZOHAR A. Secure high-rate transaction processing in bitcoin[C]//LNCS 8975: Proceedings of the 2015 International Conference on Financial Cryptography and Data Security, San Juan, Jan 26-30, 2015. Cham: Springer, 2015: 507-527.
- [61] EYAL I, GENCER A E, SIRER E G, et al. Bitcoin-NG: a scalable blockchain protocol[J]. arXiv:1510.02037, 2015.
- [62] EYAL I, SIRER E G. Majority is not enough: bitcoin mining is vulnerable[J]. Communication of the ACM, 2018, 61(7): 95-102.
- [63] KOKORIS-KOGIAS E, JOVANOVIĆ P, GAILLY N, et al. Enhancing bitcoin security and performance with strong consistency via collective signing[C]//Proceedings of the 25th USENIX Security Symposium, Austin, Aug 10-12, 2016. New Orleans: USENIX Association, 2016: 279-296.
- [64] LUU L, NARAYANAN V, ZHENG C D, et al. A secure sharding protocol for open blockchains[C]//Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Oct 24-28, 2016. New York: ACM, 2016: 17-30.
- [65] KOKORIS-KOGIAS E, JOVANOVIĆ P, GASSER L, et al. OmniLedger: a secure, scale-out, decentralized ledger via sharding[C]//Proceedings of the 2018 IEEE Symposium on Security and Privacy, San Francisco, May 20-23, 2018. Washington: IEEE Computer Society, 2018: 583-598.
- [66] PASS R, SHI E. FruitChains: a fair blockchain[C]//Proceedings of the 2017 ACM Symposium on Principles of Distributed Computing, Washington, Jul 25-27, 2017. New York: ACM, 2017: 315-324.
- [67] LI C X, LI P L, ZHOU D, et al. Scaling nakamoto consensus to thousands of transactions per second[J]. arXiv:1805.03870v4, 2018.
- [68] LASLA N, ALSAHAN L, ABDALLAH M, et al. Green-PoW: an energy-efficient blockchain proof-of-work consensus algorithm[J]. arXiv:2007.04086, 2020.
- [69] KARA M, LAOUID A, ALSHAIKH M, et al. A compute and wait in PoW (CW-PoW) consensus algorithm for preserving energy consumption[J]. Applied Sciences, 2021, 11(15): 6750.
- [70] QU X D, WANG S L, HU Q, et al. Proof of federated learning: a novel energy-recycling consensus algorithm[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(8): 2074-2085.
- [71] MECHANIC Q. Proof of stake[EB/OL]. (2011-07-11)[2021-09-13]. <https://bitcointalk.org/index.php?topic=27787>.
- [72] KING S, NADAL S. PPCoin: peer-to-peer crypto-currency with proof-of-stake[EB/OL]. (2012-08-19)[2021-09-05]. <https://www.peercoin.net/assets/paper/peercoin-paper.pdf>.
- [73] Nxt community. Nxt whitepaper[EB/OL]. (2014-07-12)[2021-09-30]. <https://www.jelurida.com/sites/default/files/NxtWhitepaper.pdf>.
- [74] VASIN P. BlackCoin's proof-of-stake protocol v2[EB/OL]. [2021-09-30]. <https://blackcoin.org/blackcoin-pos-protocol-v2-whitepaper.pdf>.
- [75] SCHUH F, LARIMER D, CRYPTONOMEX, et al. BitShare 2.0: financial smart contract platform[EB/OL]. (2015-11-01)[2021-10-02]. [http://docs.bitshares.eu/\\_downloads/](http://docs.bitshares.eu/_downloads/)

- bitshares-financial-platform.pdf.
- [76] Bitshares. What is bitShares?[EB/OL]. (2019-01-01) [2021-10-02]. [https://how.bitshares.works/en/master/technology/what\\_bitshares.html](https://how.bitshares.works/en/master/technology/what_bitshares.html).
- [77] Steem. Steem: an incentivized, blockchain-based, public content platform[EB/OL]. (2019-06-01)[2021-10-02]. <https://www.steem.com/steem-whitepaper.pdf>.
- [78] EOSIO. What is EOS?[EB/OL]. [2021-10-02]. <https://eos.io/eos-public-blockchain/>.
- [79] KIAYIAS A, RUSSELL A, DAVID B, et al. Ouroboros: a provably secure proof-of-stake blockchain protocol[C]//LNCS 10401: Proceedings of the 37th Annual International Cryptology Conference Advances in Cryptology, Santa Barbara, Aug 20-24, 2017. Cham: Springer, 2017: 357-388.
- [80] GAŽI P, KIAYIAS A, ZINDROS D. Proof-of-stake side-chains[C]//Proceedings of the 2019 IEEE Symposium on Security and Privacy, San Francisco, May 19-23, 2019. Piscataway: IEEE, 2019: 139-156.
- [81] HALEVI S, HAZAY C, POLYCHRONIADOU A, et al. Round-optimal secure multi-party computation[J]. Journal of Cryptology, 2021, 34(3): 1-63.
- [82] JZNBAS S, ASGHARI R, BAGHERPOUR B, et al. A fast non-interactive publicly verifiable secret sharing scheme [C]//Proceedings of the 17th International ISC Conference on Information Security and Cryptology, Tehran, Sep 9-10, 2020. Piscataway: IEEE, 2020: 7-13.
- [83] ROTHBLUM R D, SEALFON A, SOTIRAKI K. Toward non-interactive zero-knowledge proofs for NP from LWE [J]. Journal of Cryptology, 2021, 34(1): 472-503.
- [84] DAVID B, GAŽI P, KIAYIAS A, et al. Ouroboros Praos: an adaptively-secure, semi-synchronous proof-of-stake blockchain[C]//LNCS 10821: Proceedings of the 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Apr 29-May 3, 2018. Cham: Springer, 2018: 66-98.
- [85] GANESH C, ORLANDI C, TSCHUDI D. Proof-of-stake protocols for privacy-aware blockchains[C]//Proceedings of the 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, May 19-23, 2019. Cham: Springer, 2019: 690-719.
- [86] BADERTSCHER C, GAŽI P, KIAYIAS A, et al. Ouroboros Genesis: composable proof-of-stake blockchains with dynamic availability[C]//Proceedings of the 2018 ACM SIGSAC Conference, Toronto, Oct 15-19, 2018. New York: ACM, 2018: 913-930.
- [87] DEIRMENTZOGLOU E, PAPAKYRIAKOPOULOS G, PATSAKIS C. A survey on long-range attacks for proof of stake protocols[J]. IEEE Access, 2019(7): 28712-28725.
- [88] DAIAN P, PASS R, SHI E. Snow White: robustly reconfigurable consensus and applications to provably secure proof of stake[C]//LNCS 11598: Proceedings of the 23rd International Conference on Financial Cryptography and Data Security, Frigate Bay, Feb 18-22, 2019. Cham: Springer, 2019: 23-41.
- [89] REN L. Proof of stake velocity: building the social currency of the digital age[EB/OL](2018-04-10)[2021-10-15]. <https://assets.coss.io/documents/whitepapers/reddcoin.pdf>.
- [90] BENTOV I, LEE C, MIZRAHI A, et al. Proof of activity: extending bitcoin's proof of work via proof of stake[J]. ACM SIGMETRICS Performance Evaluation Review, 2014, 42(3): 34-37.
- [91] BUTERIN V, GRIFFITH V. Casper the friendly finality gadget[J]. arXiv:1710.09437v4, 2017.
- [92] ORLICKI J I. Fair proof-of-stake using VDF+VRF consensus[J]. arXiv:2008.10189v2, 2020.
- [93] BUTERIN V, HERNANDEZ D, KAMPERFNER T, et al. Combining GHOST and casper[J]. arXiv:2003.03052, 2020.
- [94] DUONG T, FAN L, KATZ J, et al. 2-hop Blockchain: combining proof-of-work and proof-of-stake securely[C]//LNCS 12309: Proceedings of the 25th European Symposium on Research in Computer Security, Guildford, Sep 14-18, 2020. Cham: Springer, 2020: 697-712.
- [95] KARANTIAS K, KIAYIAS A, ZINDROS D. Proof-of-burn[C]//LNCS 12059: Proceedings of the 24th International Conference on Financial Cryptography and Data Security 2020, Kota Kinabalu, Feb 10-14, 2020. Cham: Springer, 2020: 523-540.
- [96] ABRAHAM I, MALKHI D, NAYAK K, et al. Solida: a blockchain protocol based on reconfigurable Byzantine consensus[J]. arXiv:1612.02916v2, 2016.
- [97] PASS R, SHI E. Hybrid consensus: efficient consensus in the permissionless model[C]//Proceedings of the 31st International Symposium on Distributed Computing, Vienna, Oct 16-20, 2017. Leibniz: Zentrum für Informatik, 2017: 39.
- [98] AI-BASSAM M, SONNINO A, BANO S, et al. Chainspace: a sharded smart contracts platform[J]. arXiv:1708.03778, 2017.
- [99] ZAMANI M, MOVAHEDI M, BAYKOVA M, et al. Rapid-Chain: scaling blockchain via full sharding[C]//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, Oct 15-19, 2018. New York: ACM, 2018: 931-948.
- [100] DECKER C, SEIDEL J, WATTENHOFER R. Bitcoin meets strong consistency[J]. arXiv:1412.7935, 2014.
- [101] KOKORIS-KOGIAS E, JOVANOVIĆ P, GASSER L, et al. OmniLedger: a secure, scale-out, decentralized ledger via

sharding[C]//Proceedings of the 2018 IEEE Symposium on Security and Privacy, San Francisco, May 20-24, 2018. Washington: IEEE Computer Society, 2018: 583-598.

- [102] SYTA E, JOVANOVIĆ P, KOKORIS-KOGIAS E, et al. Scalable bias-resistant distributed randomness[C]//Proceedings of the 2017 IEEE Symposium on Security and Privacy, San Jose, May 22-26, 2017. Washington: IEEE Computer Society, 2017: 444-460.
- [103] WANG J P, WANG H. Monoxide: scale out blockchains with asynchronous consensus zones[C]//Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation, Boston, Feb 26-28, 2019. New Orleans: USENIX Association, 2019: 95-112.
- [104] GILAD Y, HEMO R, MICALI S, et al. Algorand: scaling Byzantine agreements for cryptocurrencies[C]//Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, Oct 28-31, 2017. New York: ACM, 2017: 51-68.
- [105] WANG Q, LI R J, CHEN S P, et al. Formal security analysis on dBFT protocol of NEO[C]//Proceedings of the 24th International Conference on Financial Cryptography and Data Security, Kota Kinabalu, Feb 10-14, 2020. Cham: Springer, 2020: 20-31.
- [106] 王晨旭, 程加成, 桑新欣, 等. 区块链数据隐私保护: 研究现状与展望[J]. 计算机研究与发展, 2021, 58(10): 2099-2119.
- WANG C X, CHENG J C, SANG X X, et al. Data privacy-preserving for blockchain: state of the art and trends[J]. Journal of Computer Research and Development, 2021, 58(10): 2099-2119.



王群(1971—),男,甘肃天水人,博士,教授,CCF会员,主要研究方向为信息安全、计算机网络体系结构与协议。

**WANG Qun**, born in 1971, Ph.D., professor, member of CCF. His research interests include information security, computer network architecture and protocols.



李馥娟(1974—),女,陕西西安人,硕士,教授,主要研究方向为计算机网络技术与应用、信息安全。

**LI Fujuan**, born in 1974, M.S., professor. Her research interests include computer network technology and application and information security.



倪雪莉(1990—),女,江苏南通人,硕士,讲师,主要研究方向为电子数据取证、网络空间安全。

**NI Xueli**, born in 1990, M.S., lecturer. Her research interests include digital forensics and cyberspace security.



夏玲玲(1988—),女,江苏射阳人,博士,副教授,江苏省网络空间安全学会会员,主要研究方向为网络安全、复杂网络传播动力学、网络爬虫、数据挖掘。

**XIA Lingling**, born in 1988, Ph.D., associate professor, member of Jiangsu Cyberspace Security Association. Her research interests include network security technology, spreading dynamics of complex networks, web crawler technology and data mining.



王振力(1977—),男,安徽寿县人,博士,副教授,主要研究方向为区块链技术与应用。

**WANG Zhenli**, born in 1977, Ph.D., associate professor. His research interests include blockchain technology and applications.



梁广俊(1982—),男,安徽芜湖人,博士,讲师,主要研究方向为电子数据取证、网络空间安全。

**LIANG Guangjun**, born in 1982, Ph.D., lecturer. His research interests include digital forensics and cyberspace security.