

区块链共识机制新进展*

刘艺华, 陈康[†]

(清华大学 计算机系, 北京 100084)

摘要:随着区块链技术的发展,其共识机制也在不断进化,在公有链和许可链的不同场景下,区块链系统的共识机制也会有所不同。在公有链共识机制上,从PoW共识机制入手分析其优缺点,探讨了为解决PoW缺点所形成的不同解决思路及应运而生的一系列共识机制,并分析了不同解决方案的优缺点;在许可链共识机制上,根据是否需要解决拜占庭将军问题分类讨论了两类共识机制,并着重探索了拜占庭将军问题下的一些新的共识机制。最后展望了未来区块链共识机制的发展方向。

关键词:区块链; 共识机制; 公有链; 许可链

0 引言

2008年 Nakamoto^[1]提出了一种基于点对点网络的电子现金支付系统即比特币,这标志着区块链系统的诞生。狭义的区块链是一种链式结构,数据被打包为区块,并在头部包含上一区块的哈希值,从而形成逻辑上的链,并具备防篡改的特性。广义的区块链则是一种包含了密码学、分布式共识算法、智能合约、博弈论等复合技术而形成的去中心化架构^[2]。本文针对的是广义的区块链。根据系统是否具有准入机制,区块链系统可以进一步分为无许可的区块链和有许可的区块链。前者往往被称为公有链,后者则被称为许可链或联盟链。比特币系统是公有链的典型代表,任何人都可以运行比特币节点并参与^[3]。Hyperledger Fabric^[4]、Quorum^[5]等则是许可链的典型代表。只有经过许可的节点才能参与到共识流程中并参与区块的产生。准入机制的有无往往会影响到区块链系统所面临的环境假设,并导致系统采用不同的共识机制。

为了在开放性的点对点网络中达成一致,比特币采用了PoW (proof of work,工作量证明)作为系统的共识机制。在PoW共识机制下,矿工通过重复的哈希运算以获取下一个区块的记账权,具备单向性的哈希函数,使得记账权的获得只能通过消耗大量算力来获取。基于PoW共识机制的区块链系统往往具备良好的去中心化特性,但同时也面临着计算能力浪费、交易确认效率低、性能瓶颈等缺点^[6]。为了解决PoW所面临的问题,一系列共识算法如PoS (proof of stake,权益证明)、DPoS (delegated proof of stake,委托权益证明)等相继诞生。这些共识算法从不同角度着手以解决PoW共识机制所面临的问题,推动着区块链系统共识机制的发展。

区块链系统所要解决的共识问题并不完全是一个新事物。在分布式系统中,如何在各个节点之间达成一致一直都是分布式系统领域所研究的经典问题。早在1982年,Lamport等人^[7]就针对存在恶意用户的拜占庭将军场景进行了描述和论证;Castro等人^[8]则于1999年提出了PBFT (practical Byzantine fault tolerance,实用拜占庭容错)算法,让经典拜占庭算法开始进入到实用阶段。PoW等一系列新的共识算法为解决拜占庭问题提供了新的思路,一系列经典的分布式共识算法也逐渐应用到区块链系统领域。

区块链技术所具备的防篡改、去中心化等特性有助于更好地构建企业间合作,一大批企业如IBM、摩根大通、Facebook等也积极参与构建区块链系统。不同于比特币等所面临的开放性的点对点网络场景,企业间区块链系统面对的场景往往有着一定准入机制即许可链场景,同时需求也往往与公有链场景有所差异,如相对较弱的去中心化属性、追求更高的吞吐量和更低的延迟等。这些新的场景和需求也推动着区块链系统发展孕育出新的共识机制。

1 共识机制与拜占庭将军问题

共识机制占据着分布式系统的核心地位,对共识机制的研究也历时已久。在分布式系统中,多个节点组成了一个集群,节点之间需要通过通信来交流,以确定一系列事情并达成一致。集群中的各个节点除了需要面对通信的延迟、时钟不一致等假设外,还需要面临网络中断、机器宕机等意外发生,在部分假设中还需要考虑到恶意节点可能会故意不按照协议工作如恶意投票、发送随机数据或错误数据等,因此设计一个切实可行的算法非常不易^[9]。仅

需要考虑网络中断、机器宕机等意外的场景被称为CFT (crash fault tolerance,宕机容错),需要考虑恶意节点的场景被称为BFT (Byzantine fault tolerance,拜占庭容错)。

针对CFT场景,Lamport^[10,11]提出了Paxos算法,并逐渐成为应用最广的分布式一致性算法。时至今日,Paxos及其一系列变种如ZAB^[12]、Raft^[13]等都在工业界发挥了巨大的作用。

拜占庭将军问题于1982年由Lamport提出。在这一模型下,集群中的节点不仅可能面临网络故障、机器宕机等意外,集群中还存在一些恶意节点向其他节点发送随机数据、错误数据,或者不响应其他节点的请求,这些无法预测的行为使得一致性这一问题变得更加复杂。针对拜占庭将军问题,Lamport证明了如果节点总数为 N ,故障节点数为 F ,则当 $N > 3F$ 时,问题才能有解^[7]。但是长期以来,拜占庭容错算法都面临着运行过慢或者复杂度过高的问题。Lamport提出的原始拜占庭容错算法复杂度为指数级。直到1999年,Castro等人^[8]提出了PBFT算法,将拜占庭容错算法的复杂度降低到多项式级,拜占庭算法才逐渐得到了较为广泛的应用。

拜占庭将军问题也是大多数区块链系统尤其是公有链系统需要解决的一个问题。在公有链系统中,由于没有准入机制,所以恶意节点的存在是一定会面临的问题。在比特币系统中,除了面临存在恶意节点的问题外,还面临着网络节点众多且不可预知、节点可能随时退出或参与系统等问题。经典的拜占庭容错算法往往依赖于一些公共知识如系统节点数目,同时节点的进入和退出往往有一些规则或流程,因此经典的拜占庭容错算法已不再适用。为此,比特币系统采用了PoW共识算法,通过挖矿机制限制了在一定时间内的提案数,同时系统放弃了绝对确定性,从而巧妙地解决了其所面临的拜占庭问题,并且能容纳一半恶意节点。这一解决方案为解决拜占庭问题提供了新的思路,也极大地影响了后续的区块链共识算法设计。但是PoW共识算法也面临着计算能力浪费、交易确认效率低、性能瓶颈等缺点,因此继PoW共识算法之后,一系列适用于区块链系统的共识算法也陆续被提出。

2 公有链共识机制

典型的公有链系统有比特币、以太坊等。在公有链系统中,往往没有准入机制,也不存在具备公信力的权威机构,参与节点随时退出或加入,并通过点对点网络建立连接。相对开放的网络使得拜占庭将军问题是公有链系统必须面对和解决的问题,而不确定的节点数目使得经典的拜占庭算法难以直接应用于公有链系统,因此在公有链系统中出现了一批不同于经典拜占庭算法的共识机制。

PoW共识机制是区块链系统中的第一种共识机制。但是PoW也存在一系列缺点如计算能力浪费、交易确认效率低、性能瓶颈等。针对这些缺点,出现了不同类型的共识机制分别从不同角度进行改进发展,这也推动着公有链系统的共识机制发展进步。

2.1 PoW 共识机制

PoW并非在区块链中被首次使用,PoW于1992年由Dwork等人^[14]提出,并将其应用于处理垃圾邮件,Back^[15]则于1997年独立提出了PoW,并将其应用于hashcash系统。hashcash与前者不同在于其使用了更高效的哈希算法而不是弱签名算法,hashcash对比

收稿日期:2020-02-04; 修回日期:2020-04-28 基金项目:国家重点研发计划资助项目(2016YFB1000504);清华大学(计算机系)—北京阿尔山金融科技有限公司区块链技术联合研究中心资助项目

作者简介:刘艺华(1995-),男,河南项城人,硕士研究生,主要研究方向为区块链、分布式系统;陈康(1976-),男(通信作者),浙江温岭人,副研究员,硕士,博士,主要研究方向为分布式系统(chenkang@tsinghua.edu.cn)。

特币的产生有着重大影响。

在比特币系统中,各个节点共同求解一个验证简单但求解复杂的数学难题以争夺记账权,这一过程也被通俗地称为“挖矿”。该数学难题可简要描述为在某个难度值下,节点寻找一个随机数使得区块头的两次 SHA256 结果不大于目标值,难度值动态调整以使得区块产生的速率约为 10 min。通过付出大量的算力,达到了“一 CPU 一票”的目的^[16]。

由于一定时间内可能存在多个节点都找到了合法的随机数并广播到了网络,网络上的节点可能会分为多个集合即系统存在多条链,这一现象也被称为分叉。比特币系统中定义了最长链原则,即矿工始终选择最长的链进行挖矿,如果两条链长度一样,则选择最先收到的那一条链,如图 1 所示。因此,比特币中的确认是概率性的确认,一旦存在一条更长的链,系统就会切换到该链上去。如果系统的算力不够庞大或者过于集中,则更有可能遭遇 51% 攻击。所谓 51% 攻击,指的是恶意节点通过控制超过一半算力并试图制造更长的链以进行双花等行为的一种恶意攻击。2018 年 5 月,BTG(比特黄金)便发生了一次成功的 51% 攻击,攻击者逆转了 22 个区块,在这次攻击中 BTG 损失价值约 1 800 万美元^[17];Ethereum Classic(以太经典)、Verge(XVG)等也曾遭受过 51% 攻击,损失均超过百万美元,其中后者更是被连续攻击两次。因此,这一系列成功的攻击事件也提醒人们基于 PoW 共识机制的区块链系统遭受 51% 攻击的风险不容忽视。此外,ASIC 等专用芯片所打造出的矿机也使得算力更容易集中于少部分用户手中,这无疑对 PoW 所依赖的去中心化假设构成了一定的挑战,增加了 PoW 区块链系统遭受 51% 攻击的风险。

PoW 共识机制对区块链系统的共识机制构建产生了深远的影响,其创新性地提出了一种在开放性的点对点网络中达成一致的共识方法,为解决拜占庭将军问题提供了新思路。但是 PoW 也存在着效率低下、性能差、能源浪费严重等缺点。据 2018 年的一份报告显示,比特币系统每秒仅支持 2~3 个交易,但是其电力消耗超过 2.55 亿瓦^[18]。与传统的银行支付系统如 Visa 相比,比特币每处理一个交易所产生的碳排放要比 Visa 系统处理 10 000 笔交易还要多 62%^[19]。比特币系统也面临着交易成本相对较高的困扰,2017 年 12 月份比特币系统中每笔交易平均手续费达到 37 美元的高位^[20]。同样地,其他使用 PoW 共识机制的区块链系统如以太坊等也面临着类似的问题。

为了解决 PoW 的上述缺点,一些新的共识机制应运而生,这些共识机制往往选取不同方面来解决 PoW 所面临的困境。其中,一部分共识机制如 PoS 选择从挖矿的机制入手,通过重新定义挖矿的机制来降低无效的能源浪费以及提高区块的生成效率;一部分共识机制选择缩小参与共识的节点范围,将一定时期的共识限制在一部分节点内,而在共识节点之间选择相对传统的共识机制以避免资源浪费并提高效率,这类共识机制有 DPoS^[21,22]、基于 VRF 的共识机制^[23,24]等;一部分共识机制选择改变底层的数据结构如选择 DAG 作为底层的数据结构以提供更好的并行性、更强的安全性及更高的性能,这类共识机制有 Conflux^[25]等;一部分共识算法则将节点分片分区以提高系统的并行能力,这一类共识机制有分片技术^[26]、连弩挖矿^[27]等。

2.2 改进挖矿机制类共识机制

PoS 共识机制最早于 2012 年在 peercoin(点点币)上采用^[28]。在 PoS 系统中,尽管挖矿行为仍然存在,并且同样是共同求解一个验证简单但求解复杂的数据难题以争夺记账权,但是针对不同节点计算的难度值不再相同。在 PoS 共识机制中,定义币龄为货币数量和时间的乘积,节点所拥有的币龄越大,挖矿所面临的难度值也就越低,生成区块的概率也就越大。通过这种方式,PoS 将 PoW 中的算力消耗变成了币龄消耗,整个挖矿过程中消耗的能源大大减少。以太坊创始人 Buterin 认为,PoS 相较于 PoW 可以节省超过 99% 的能源消耗。相应地,系统中主链则被定义为币龄最高的链。为了避免“富者愈富”的情况发生,当节点成功生成区块时,币龄会被消耗;同时当货币发生转移时,币龄也会被清空。通过这样的机制,PoS 希望能够避免财富的过度集中以增强系统的去中心化能力。PoS 共识机制的支持者认为,在 PoS 区块链系统中,算力不再是系统中的硬通货,因此 ASIC 等专用芯片打造的矿机所发挥的作用也受到了抑制,从这一角度上 PoS 有助于提高区块链系统的去中心化程度。而发动 51% 攻击需要攻击者拥有全网大半的币或者币龄,同时攻击成功往往也意味着承担最大的代价,这不仅难以做

到,并且也不符合合理性,从这一角度 PoS 共识机制也能够更好地防范 51% 攻击。

PoS 被提出之后受到了越来越多的关注。除了 peercoin、nextcoin(未来币)采用了 PoS 外,以太坊也计划逐步将其共识机制迁移到 PoS 上来^[26]。Facebook 的 Libra 项目也有计划转型为非许可链,并选用 PoS 作为其共识机制。

但是 PoS 下系统的安全性还存在较多争议。在 PoS 共识机制下,即使节点离线其币龄仍然是累积的,所以节点可能会选择累计币龄到一定程度后再上线参与区块生成,这就导致了系统可能缺乏足够多的线上节点维护系统安全,同时也使得恶意节点可以通过这种方式以较小的成本等待时机发动攻击,这一可能性使得 PoS 更能抵御 51% 攻击的观点显得不那么可靠^[17]。PoS 共识机制还存在免权益证明(免费午餐)问题。所谓免费午餐问题,是由于在 PoS 系统中矿工并不需要类似于 PoW 中消耗大量的 CPU 资源进行挖矿以选择一条最长链进行跟随,所以其有更多的可能性在多条链上同时进行工作以获取更多收益。因为矿工付出的权益只是与现实世界没有直接关联的区块链系统中的资源,而不是不可撤销的 CPU 算力^[17,29]。为了解决 PoS 下的免费午餐问题,系统设计者往往选取惩罚措施以限制矿工同时下注。但是这一方面增加了系统的复杂度,另一方面矿工本身就可能放弃自己最初认定的区块并选择被多数人认同的区块。区分恶意下注并进行惩罚并不容易实施,并且更难证明惩罚带来的损失大于同时下注的收益且确实能改善系统中的免费午餐问题。此外,PoS 共识机制还存在一些可能的攻击方式如远程攻击^[30]等。所谓远程攻击,指的是攻击者可以购买一个过去拥有大量代币的私钥,并以此自己颁发越来越多的奖励,这些奖励的存在使得其得以生成比现有区块链更高的权益链并逆转区块,而为了缓解这一问题系统需要设置一些检查点以避免历史区块被重组。因此,PoS 共识机制所存在的这一系列问题导致其安全性尚且不能被普遍认可,并且 PoS 共识机制往往更加复杂而难以实现。

2.3 缩小共识范围类共识机制

不论是 PoW 还是 PoS,共识过程都发生于全网所有节点之间(只参与发起交易的轻节点除外)。受限于较多的节点数目、较高的网络延迟等客观现实,出块速度、区块大小等都难以进一步提高,这是导致系统性能受限如系统吞吐率低、交易延迟大等的重要因素。为了在较多参与节点中选取出块者,不可避免地依赖于挖矿机制或类似机制,这些机制不仅效率低下,同时也只能提供概率性的确认。为了进一步解决 PoW 和 PoS 中存在的上述问题,一些共识机制选择缩小参与共识的节点范围,将一定时期的共识限制在一部分节点内,而在共识节点之间选择相对传统的共识机制以避免资源浪费、提高系统效率等。这类共识机制典型的有 DPoS、基于 VRF 的共识机制等。

2.3.1 DPoS 共识机制

DPoS 是 PoS 共识机制的一个发展,其将共识范围从全网节点缩小到了选举产生的部分节点。DPoS 共识机制最早于 2014 年在 Bitshares(比特股)^[21]中被首次应用,EOS(enterprise operation system)^[22]等继承发展了这一思想。一般地,DPoS 可以分为以下两个环节:

a) 选举出委托人。这一环节通过某种机制在全网节点之间选举出一定数量的委托人在一段时间内参与区块共识环节。

b) 在委托人之间达成共识产生区块。这一环节在选举出的委托人间运行共识算法以在委托人之间产生区块并达成共识。

由于 DPoS 中共识环节仅有少数委托人所控制的节点参与,在这些少数节点中可以采取更为高效的共识算法如经典的 BFT 算法,从而实现高吞吐率、低延迟、确定性的区块生成等。以 EOS 为例,EOS 的代币持有者进行投票以在全球选出 21 个超级节点作为委员会,而委员会将在一年内负责参与 EOS 主网的区块产生。由于委员会内部节点确定且数目较少,所以 EOS 在委员会内部共识中采用了 aBFT 共识算法,并可以达到较短的区块产生时间、较高的 TPS(transaction per second,每秒交易数)。EOS 宣称其能够达到百万计 TPS,并且 99.9% 的交易确认延迟小于 250 ms,同时相对于需要等待多个区块确认以进行概率性交易确认的 PoW 和 PoS 共识机制,EOS 宣称所有的交易都能在 1 s 内进行不可逆的确认^[31]。遗憾的是,EOS 所宣称的这些性能指标并未能完全实现,截止 2018 年 4 月,其最终上线主网的平均 TPS 约为 3 000,即使在最好情况下 TPS 也仅有 6 000^[32],这与其所宣称的百万 TPS 相差甚远。一部分

研究者认为,其实际可用 TPS 不高于 500,并且在高压下会出现分叉情况^[33]。但总的来说,虽然 EOS 未能达到其宣称的性能,但是相较于 PoW 和 PoS 等共识机制,DPoS 展现出了更好的性能,如图 2 所示。

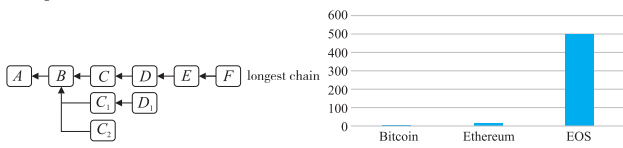


图 1 最长链法则

图 2 TPS 对比

但是 DPoS 存在的中心化的风险不容忽视。DPoS 在共识过程中只有少数节点参与,事实上系统在一定时间内由这些少数节点构成的委员会所控制。如果选举过程不能做到公平,便难以保证委员会的可靠性和公正性。此外,委员会也存在被恶意用户直接攻击的风险,这一系列缺点是 DPoS 相对中心化的机制所导致的。事实上,在 EOS 选举超级节点的过程中就暴露了贿选、用户参与度低等缺陷,主网上线时间也因此数次被推迟,同时也引发了对 EOS 是否是公有链的质疑。因此,虽然 DPoS 在吞吐量、延迟等方面获得了较 PoW 或 PoS 等共识机制更好的效果,但是其付出的代价也不容忽视。

2.3.2 基于 VRF 的共识协议

DPoS 的出现给区块链系统在提高吞吐率和降低延迟等方面开辟了新的可能性,但是其过于中心化的缺点也需要一些新的机制去弥补。因此,一批基于 VRF (verifiable random function, 可验证随机函数) 的共识协议应运而生。这一类共识协议往往具有如下特点: a) 通过 VRF 机制从一个开放的网络系统中选出一些节点来参与一段时期的共识; b) 在选出的少部分节点中使用 PoS 共识算法、经典的拜占庭算法如 PBFT 及其改进等共识算法进行共识。

VRF 最初于 1999 年被 Micali 等人^[34]提出,一般包括以下三个部分:

- a) 密钥生成 keygen。这一步生成一对密钥对,分别是私钥 sk 和验证公钥 vk 。
- b) 评估算法 evaluate。针对一个消息 m ,使用私钥 sk 生成一个伪随机字符串 r 和证据 ρ 。
- c) 验证函数 verify。验证函数将验证公钥 vk 、消息 m 、伪随机字符串 r 和证据 ρ 作为输入。如果 r 确实是评估算法基于私钥 sk 和消息 m 产生的输出,则验证函数返回真。

VRF 为在开放网络中去中心化的选举验证节点提供了一种可能性。借助 VRF 机制,区块链系统可以去中心化地在一个开放的网络中选出一批节点作为委员会来参与一段时期的共识。基于 VRF 的共识协议的区块链系统有 Algorand^[23]、DFINITY^[24]等。

以 Algorand 为例,系统中的用户都拥有一对安全生成的密钥对 sk 和 vk ,其中 vk 是公开的。用户想确定自己是否在生成区块 r 的委员会中,他将根据一个公开的 seed 运行评估函数,然后判断生成的随机字符串 r 是否在某个范围中。VRF 的特性保证了在 seed 确定的前提下没有用户可以生成一个虚构的 r 以加入委员会,seed 的值会随着上一轮区块的结果而变化,因此恶意用户也难以预测下一轮的 seed 值。通过 VRF 这种方式,委员会的选举保证了去中心化和可验证性。不过由于女巫攻击(恶意节点虚构出多个身份参与网络以增加自己比重)的存在,VRF 机制本身并不足以保证足够多的诚实节点进入委员会,所以往往需要 PoW 或者 PoS 等机制的配合以防范女巫攻击,如 Algorand 就计划将用户的代币金额纳入选举流程以保证系统的安全性。

基于 VRF 的共识机制弥补了 DPoS 下相对中心化的选举所带来的弊端,同时保留了 DPoS 具备较高的效率和性能的优点,很好地兼顾了去中心化和性能。但是基于 VRF 的共识机制也存在一些弊端,这一方面是由于其对现实世界的假设并不客观导致的,另一方面是其发展尚不成熟导致的。为了通过随机方式构建一个相对公平且不容易遭受攻击的委员会,需要相当大的种子集合参与委员会如超过 2 000 个节点,但是这在现实中并不容易构建。以 Algorand 为例,其对网络的假设是一个强同步网络,这与现实世界也并不符合;激励机制及惩罚机制的缺乏也使得网络更容易受到恶意攻击等。因此,基于 VRF 的共识机制要想成为一种成熟可靠的共识机制还有一段很长的路要走。

2.4 修改数据结构的共识机制

一般地,区块链是一种逻辑上的链式结构,但是有一些研究指出区块链这一链式结构导致了区块链系统的安全性并不如所宣称的那么安全,因为攻击者的算力总会聚集在同一条链上,而系统中诚

实用户则会由于分叉使得算力分散到不同的链上。因此诚实用户所形成的链所凝聚的算力会大大小于诚实用户的总算力,这会使得攻击者实际上并不需要 51% 的算力就可以发动 51% 攻击。同时,区块链的链式结构也使得系统缺乏足够的并行能力,这也制约了区块链系统性能的提高。一部分协议从修改区块链数据结构入手,通过对区块链的简单链式结构进行拓展以期解决区块链系统所面临的一系列问题如增强安全性、增大吞吐量、降低延迟等。GHOST (greedy heaviest-observed sub-tree, 贪婪最重可观察子树协议) 协议^[35]就是其中的一种,其允许区块头部引用叔区块并选择最重子树来选择子链,使得系统在降低出块时间的同时也提高了安全性。这一协议被以太坊采用并实现^[36]。以此为起点,产生了 DAG (directed acyclic graph, 有向无环图),一批基于 DAG 的协议或系统也应运而生,如 Spectre^[37]、PHANTOM^[38]、Conflux^[25]、Hashgraph^[39]等。

随着 DAG 的发展,基于 DAG 的系统可以分为基于区块的 DAG 和基于交易的 DAG 两类。前者通过修改区块的数据结构,允许在区块头部引用多个历史区块将区块链的链式结构拓展为 DAG 结构。虽然数据结构有所差异,但是仍然属于区块链技术的发展,因此仍然属于广义的区块链的一部分,这一类系统的典型有 Conflux 等;后者则放弃了区块这一属性,交易直接形成 DAG 结构,虽然借用了区块链系统的许多技术及机制,但是已经与传统的区块链系统大不相同,这一类系统的典型有 Hashgraph 等。本节所要进行描述的主要是第一类基于区块的 DAG 系统。

以 Conflux 为例,区块之间由多条连接组成,分别是一个父链接和多个引用链接,从而在多个区块之间形成 DAG 结构。其中,父链接代表了投票关系,引用链接代表了先后顺序。不同于比特币中的最长链原则,Conflux 基于 GHOST 定义了枢纽链并使得区块之间得以形成全局序,如图 3 所示。虽然 Conflux 等底层的数据结构不再是传统的区块链的简单链式结构,但是交易仍然被打包形成区块,并且区块之间仍然保持了全局顺序,因此相当程度上保留了区块链特征。通过 DAG 结构,系统的并发性能和安全性得以提高,Conflux 宣称其测试系统网络下 TPS 可以达到 6 400,并且相较于比特币系统有着更低的延迟和更好的安全性。需要注意的是,在狭义的共识机制上 Conflux 仍然采用了前文中所描述的 PoW 共识机制,在这点上并没有显著的创新,而是通过降低挖矿难度及数据结构的调整从而缩短挖矿间隔,提高吞吐量及降低延迟等,所以仍然不可避免地需要耗费大量的能源进行挖矿,并且延迟仍然相对较高,如 Conflux 在测试系统网络下交易确认延迟为 4.5 ~ 7.5 min^[25]。同时,相对复杂的 DAG 数据结构也使得激励机制等相对于比特币等更加难以设计,系统的最终效果还有待于进一步考察。此外,并非所有基于 DAG 的区块链系统都像 Conflux 一样保留了区块之间的全局序,因此,如何在缺少全局序的部分 DAG 系统上实现智能合约也会带来一些新的挑战。

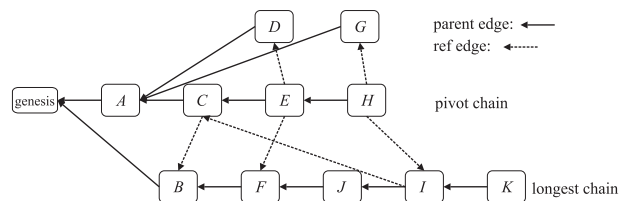


图 3 枢纽链法则

2.5 分片分区处理的共识机制

在传统数据库中,为了提高数据库容量和吞吐量,会将数据库分割成多个碎片并放置在不同的服务器上,这一概念叫做分片 (sharding)。为了解决区块链系统的吞吐量,一些系统也采用了类似的技术,并将其称为分片或者分区等。通过这一机制,将数据分散到不同的链或者区域进行分散处理,理论上有助于增加系统的并行度、提高系统的吞吐率和容量等,因此也吸引了一批区块链项目如 ZILLIQA^[40]、以太坊等加入以期解决性能瓶颈。其中,ZILLIQA 宣称自己是第一个实现分片技术的区块链系统。

一般地,分片技术将区块链系统通过网络、交易或者状态分成若干区域,区域内部形成独立的处理并达成共识,区域之间彼此得以并行处理交易,从而获得数量级的性能提升。但是实际情况却要复杂得多,分片分区机制在获得这些优点的同时也伴随着一些显著的缺点。首先,这一机制使得系统的交易变得更加复杂,由于区域之间的交易彼此独立,当发生跨区域交易时往往需要将一笔交易分成多笔交易依次完成,这一过程的原子性相对难以保持,同时跨区域之间的数据验证带来了额外的复杂度和风险,其他区域

如若发生区块逆转,则会使得这一问题更加严重;其次,由于分片系统将区块链系统分成了若干区域,算力也被分散到不同区域,这使得有可能通过控制一个区域进而干扰整个系统,对系统发动 51% 攻击的难度大大降低,以至于可能形成 1% 攻击。为了解决这些缺点,往往需要设计复杂的机制以保证系统的安全性。为此,以太坊放弃了 PoW 共识机制并计划采用更复杂的 Casper 共识机制,Monoxide 则引入了“连弩挖矿”机制以保证使用 PoW 共识机制时的系统安全性^[27]。

在以太坊 2.0 的分片方案中,全网按地址被划分为多个分片,各分片维护一条子链。为了将分片链加入到主链中,系统上需要部署名为 VMC (validator manager contract, 验证人管理员合约) 的特殊合约,这个合约负责管理验证者的保证金,并且产生随机数以周期性地选择验证人验证分片链上的数据。但是分片技术的应用分散了全网算力,因此发动 51% 攻击的难度大大降低,以太坊选择了基于 PoS 机制的 Casper 共识算法而非 PoW 共识算法^[41]。

在 Monoxide 中,系统被分为多个共识组,用户则根据地址前缀被分配到确定的共识组中。为了抵制可能的 1% 攻击,连弩挖矿允许矿工同时参与多个编号连续的共识组,每次出块时哈希函数将覆盖多个将要出块的块头进行计算,同时这些块头将共用一个 Nonce。通过这种方式,矿工可以同时多个共识组挖矿以放大自身算力并获得更多的收益,最终网络会收敛到主流的矿工都会采用连弩挖矿,并且参与到所有的共识组中。

分片分区机制将一条链拓展到多条链,提高了系统处理交易的并行能力,拓展了系统的容量,但是伴随的缺点却也不容忽视。首先,当系统分片分区超过一定比例后,跨片区的通信便不容忽视,当跨片区通信比例超过一定比例后,系统的性能反而会随着片区的增多而下降,因此分片技术带来的性能提升仍然存在着天花板;其次,跨片区交易往往需要拆分成多个片区内部交易以完成,因此交易原子性将更加难以保证;第三,片区内部可能出现的分叉行为使得系统行为更加复杂,为了避免跨片区交易受区内分叉影响,此类交易往往有一定的锁定时间,因此交易的延迟问题更加突出。此外,分片技术也对构建智能合约系统带来了不小的挑战,因为不同于传统区块链系统构建了全局的有序账本,分片技术将系统内数据、状态等从全局变为局部,无疑会增加智能合约系统的复杂性。因此,分片技术会导致系统的复杂度成倍上升,如何构建一个正确的基于分片分区处理的共识机制的区块链系统仍伴随着挑战。

从这一系列改进上来看,提高系统吞吐量、降低系统能耗等是各个改进所努力追求的目标,而在实现这一目标的过程中也难免存在一些取舍并带入一些缺点如去中心化程度降低、系统复杂度升高等。这些改进的共识机制在去中心化、性能和能耗等方面的对比如表 1 和图 4 所示。

表 1 公有链共识机制比较

| 共识机制 | 较 PoW 改进 | 去中心化 | 性能 | 能耗 | 缺点 |
|----------|----------|------|----|----|-----------|
| PoW | - | 高 | 差 | 高 | 能耗高、性能差 |
| PoS | 修改挖矿机制 | 较高 | 较高 | 较低 | 安全性受质疑 |
| DPoS | 缩小共识范围 | 低 | 高 | 低 | 去中心化属性受质疑 |
| VRP | 引入随机性选举 | 较高 | 较高 | 较低 | 发展尚不成熟 |
| DAG | 数据结构修改 | 高 | 较高 | 较高 | 智能合约支持较差 |
| Sharding | 数据分散处理 | 较低 | 高 | 较高 | 系统复杂度高 |

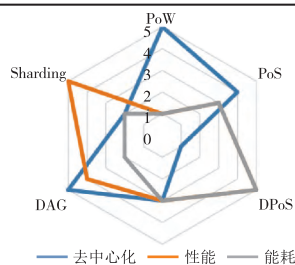


图 4 公有链共识机制雷达图

3 许可链共识机制

典型的许可链有 Hyperledger Fabric、Quorum、Libra 等。相较于开放程度很高的公有链,许可链存在一定的许可机制,系统中往往被预先设置了一些节点参与共识,并且约定参与和退出机制,因此系统中节点信任度较高,并且一定时期的节点数目往往是固定的。部分许可链系统中由于参与共识节点信任度更高,所以会假定系统不存在拜占庭将军问题,在该类系统中可以运行传统的非拜占庭共识机制以提高性能。

在许可链的场景下,经典的分布式共识算法如 Raft、Paxos、PBFT 等有着更多的应用。基于区块链系统所具有的一些特点,这些经典的共识算法往往经过改进以更适用于区块链场景,这也促进了一批新的共识算法的涌现。根据共识算法是否能够处理拜占庭场景,许可链共识机制可以分为 CFT 类共识算法和 BFT 类共识算法。前者能够处理网络故障、断电、宕机等错误,后者则能够处理拜占庭错误。一般地,前者往往有着更强的性能,后者则有着更多的适应场景。

3.1 CFT 类共识算法

相较于 BFT 类共识算法,CFT 类共识算法不需要考虑恶意节点可能导致的拜占庭错误,仅需要考虑网络中断、磁盘故障、机器宕机等意外的场景。这一类算法的典型是 Paxos 类算法。Paxos 算法于 1990 年由 Lamport 提出以解决非拜占庭场景下的分布式一致性场景,在其基础上陆续产生了 ZAB^[12]、Raft^[13] 等共识算法。相较于拜占庭将军容错场景下的共识算法,Paxos 类共识算法复杂度较低,具备更好的拓展性并拥有更好的性能。

在部分许可链场景下,由于参与共识节点信任度更高,所以会假定系统中不存在拜占庭将军问题。在此类场景中,Paxos 类共识算法往往可以取得更好的性能和容错能力。在一系列 Paxos 类共识算法中,使用最为广泛的是 Raft 算法。

Raft 算法于 2014 年由 Ongaro 等人提出。针对 Paxos 算法存在的复杂难懂、实现困难等问题,Raft 算法将易于理解放在了核心的地位。由于 Raft 算法相对清晰易懂、易于实现,并且能够提供优异的性能,自问世以来 Raft 算法便很快被工业界广泛采用。作为一类强 leader 共识算法,其核心流程可以分为 leader 选举和日志同步。前者解决的是如何进行 leader 的选举问题,后者则解决了日志如何在 leader 和 follower 节点同步的问题。Raft 算法的共识流程如图 5 所示。其可以简要描述如下:a) client 向 leader 发起一次共识请求;b) leader 为请求分配一个序号,并向 follower 节点广播;c) follower 将数据安全存储在本地并告知 leader;d) leader 收到超过一半回复后认定该请求被 commit 并返回 client,commit 消息在下次广播时告知 follower 节点。

在非拜占庭场景下的许可链使用最多的共识算法就是 Raft 算法,这得益于 Raft 算法为数众多的实现。据统计,Raft 算法的各类实现数目近百种^[42],这为构建基于 Raft 这一共识算法的许可链系统提供了良好的基础。目前,在许可链的非拜占庭场景中 Raft 占据了统治性的地位,如摩根的 Quorum 也将 Raft 作为其所支持的两种共识机制之一^[5]。IBM 的 Hyperledger Fabric 最初曾使用 Kafka 以实现分布式共识,在 1.4 版本实现了基于 Raft 的分布式共识并在 2.0 中废弃了 Kafka。Kafka 本身是一个开源于 2010 年的较为成熟的分布式消息系统,支撑其实现分布式共识的是 ZAB 协议,在工业界有着较为普遍的应用。但是 Kafka 的设计场景是运行得比较紧密的主机集群,往往依赖于统一的组织管理运维,这一点与 Hyperledger Fabric 的使用场景有所不同,一定程度上制约了 Hyperledger Fabric 的发展。相较于使用 Kafka 实现分布式共识,Raft 更加适用于较大规模的网络,维护及部署难度更低,并且展现了相对于 Kafka 更好的性能。针对 Hyperledger Fabric 在不同共识机制下的性能对比如图 6 所示。选用 Hyperledger Caliper 中的 marbles 样例进行测试,其中 Fabric 版本为 1.4.1,通过 Docker 部署到阿里云 ecs.g 5.3xlarge 平台。总的来说,相较于拜占庭场景下的许可链,非拜占庭场景下的许可链参与节点之间信任程度更高,这类系统也往往拥有着更好的性能。

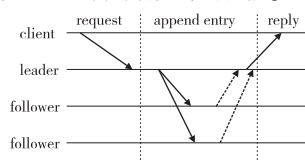


图 5 Raft 正常工作流程

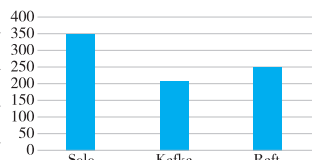


图 6 Hyperledger Fabric 不同共识机制性能对比

3.2 BFT 类共识算法

拜占庭将军问题于 1982 年由 Lamport 提出;1999 年,Castro 等人提出了 PBFT 算法,将拜占庭算法的复杂度降低到多项式级别,从而使得拜占庭算法进入实用阶段。PBFT 是 BFT 类共识算法的典型代表,其在 Hyperledger Fabric 0.6 被采用,而其变种 Istanbul-BFT 算法则在摩根大通的 Quorum 中被选为所支持的两种共识机制之一。PBFT 算法的共识流程如图 7 所示。其可以简要描述如下:a) 客户端向主节点发起一次共识请求;b) 主节点向其他节点广

播;c)其他节点之间广播执行 prepare 和 commit 过程;d)所有节点都将执行结果返回给客户端。由于系统中存在 F 个可能的恶意节点,所以客户端接收到 $F + 1$ 个相同的执行结果后认为网络达成共识。

尽管 PBFT 是目前最为成熟的拜占庭容错算法之一,但是其算法复杂度还是相对较高,正常流程下的算法复杂度为 $O(n^2)$, leader 切换时的算法复杂度更是高达 $O(n^3)$, 较高的算法复杂度使得 PBFT 不能在较大规模的网络中使用。为了更好地适应区块链系统的需求,基于 PBFT 诞生了一批新的拜占庭算法。在这些拜占庭算法中,一些共识算法虽然对 PBFT 进行了一定的改进,但是继承了算法的核心流程,并保持了与 PBFT 的相同复杂度。这一类算法典型的有 Istanbul-BFT^[5]等,这类共识算法可以认为是 PBFT 的变种而不需要再进行深入讨论。接下来所讨论的主要是另一类改进算法,虽然借鉴了 PBFT 中的某些思想或流程,但是对其核心流程进行了大胆的修改乃至变更,并且充分考虑了区块链的链式结构特点和系统主要使用场景,显著降低了其复杂度的共识算法如 Tendermint^[43]、HotStuff^[44]等。这两种算法将正常流程下的算法复杂度从 PBFT 中的 $O(n^2)$ 降低到了 $O(n)$, 从而得以更好地适应网络节点的数目增长,充分考虑了区块链系统中 leader 的切换场景,并将复杂度从 PBFT 中的 $O(n^3)$ 降低至 $O(n^2)$ 乃至 $O(n)$ 。这一系列改进使得这类算法能够达到更高的性能并支持更大的网络规模,通过将这类算法在许可链系统上的应用也进一步拓展了许可链的应用场景。

在 Tendermint 共识算法中,区块成为了共识的基本单位。借助区块链的链式特点,共识流程得以被充分简化,算法复杂度也得以进一步降低。其中,正常流程下算法复杂度虽然与 PBFT 一样保持为 $O(n^2)$,但是 leader 切换时的算法复杂度降低为 $O(n^2)$ 。如果在共识实现上加入门限签名的手段,上述流程的复杂度则均降低为 $O(n)$ ^[44]。其中,算法正常流程^[8]如图 8 所示。其可以简要描述如下:a)验证节点发起创建区块提议,这一过程称为 propose;b)其他验证节点验证区块是否正确,如果正确则对该区块进行投票,这一过程称为 pre-vote;c)如果在上一轮搜集到超过 2/3 的节点的投票,则对该区块发起 precommit 过程,否则放弃该区块;d)如果在上一轮搜集到超过 2/3 的节点的投票,则对该区块发起 commit 过程,并最终产生新的区块,否则进入下一轮区块生成。

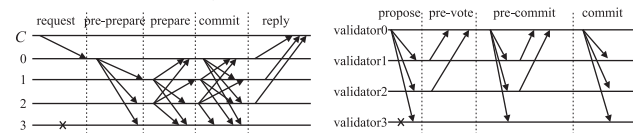


图 7 PBFT 正常情况下工作流程

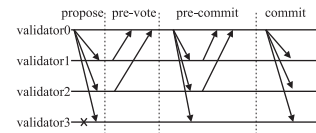


图 8 Tendermint 正常情况下工作流程

在 HotStuff 共识算法的设计上也充分考虑了与区块链系统的结合。基于区块链的链式结构 HotStuff 得以将共识流程进一步简化,相对于 Tendermint,HotStuff 的一个突出贡献是将配置变更与正常流程结合在一起,通过增加消息确认的步骤以及引入门限签名^[45]将正常流程和 leader 切换的复杂度均降低到了 $O(n)$,不仅提高了系统的性能,也简化了 leader 的切换场景;同时,HotStuff 将共识的流程流水化,从而使得多个 leader 同时工作以产生多个区块。这些特性使得 HotStuff 可以支持更大的网络规模,也更加适用于区块链这一相对去中心化的场景。得益于这些优点,HotStuff 被 Facebook 的 Libra^[46]所采用,以支持数百个节点构成的区块链系统。HotStuff 共识流程如图 9 所示。其可以简要描述如下:a)主节点搜集到足够多新视图变更后开启新视图阶段,发送 prepare 消息给其他节点;b)其他节点对 prepare 消息进行投票,主节点收集到足够多投票后发送 pre-commit 消息;c)其他节点对 pre-commit 消息进行投票,主节点收集到足够多投票后发送 commit 消息;d)其他节点对 commit 消息进行投票,主节点搜集到足够多投票后发送 decide 消息,其他节点收到 decide 消息后进行状态迁移并执行新视图变更。

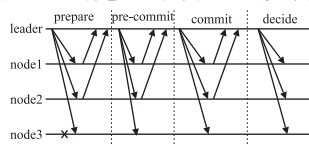


图 9 HotStuff 正常情况下工作流程

值得注意的是,尽管 Tendermint 及 HotStuff 等共识算法在许可链领域取得了一些较为显著的成绩,如两者分别在 HyperLedger Burrow 及 Libra 中被作为底层的共识协议所采用,但是这些算法的

应用并不仅限于许可链领域。以 Tendermint 为例,部分公有链如 Cosmos^[47]就采用其作为底层的共识机制。HotStuff 虽然出现时间较晚,但是已经有一些公链团队如 Cypherium^[48]开始尝试将 HotStuff 作为底层的共识机制。因此,在共识层面上许可链和公有链的差异远小于应用场景上的差异,两者的界限在逐渐淡化,以至于出现了一些同时在两个场景下都有着显著应用的共识算法。

总的来看,CFT 类共识算法适用于节点信任度较高以至于假定不存在拜占庭将军问题的许可链场景,在这类算法中 Raft 以其简单易懂的流程、种类众多的实现占据了绝对的优势地位。而在需要考虑拜占庭将军的许可链场景中,PBFT 以其成熟可靠的优点成为了一些项目的首选,但是是一些新的共识算法如 Tendermint、HotStuff 等的出现也推动了 BFT 类共识算法的发展,并且模糊了许可链和公有链系统在共识机制层面的差异。上述的 BFT 类共识算法在复杂度和应用上的对比如表 2 所示。

表 2 BFT 类共识算法在复杂度及应用上的对比

| 共识算法 | 正常流程复杂度 | leader 切换复杂度 | 应用 |
|--------------|----------|--------------|--------------------|
| PBFT | $O(n^2)$ | $O(n^3)$ | Fabric 0.6, Quorum |
| Tendermint | $O(n^2)$ | $O(n^2)$ | Burrow, Cosmos |
| Tendermint * | $O(n)$ | $O(n)$ | 同上 |
| HotStuff | $O(n)$ | $O(n)$ | Libra |

注:通过门限签名的实现来降低算法复杂度。

4 结束语

伴随着区块链技术的发展,区块链系统的共识机制也在不断发展变化。根据系统准入机制的有无,区块链系统又可以分为公有链和许可链,在不同的场景假设下有着不同的共识机制。本文将区块链系统的共识机制进行了系统的分类,针对公有链的共识机制,首先分析了 PoW 共识机制并指出了其缺点,然后分类讨论了几种解决思路及其优缺点;针对许可链共识机制,分类讨论了 CFT 类共识机制和 BFT 类共识机制,并且针对性地探索了新的共识机制算法如 Tendermint、HotStuff 等。本文中所述的一些典型的共识机制的诞生时间、典型应用及场景如表 3 所示。

表 3 共识机制诞生时间、典型应用及场景

| 时间 | 共识机制 | 典型应用 | 典型拜占庭容错率 | 应用场景 |
|--------|------------|-------------------|----------|---------|
| 2008 年 | PoW | Bitcoin, Ethereum | 1/2 | 公有链 |
| 2012 年 | PoS | Peercoin | 1/2 | 公有链 |
| 2013 年 | DAG | Ghost | 1/2 | 公有链 |
| 2014 年 | DPoS | Bitshares, EOS | 1/2 | 公有链 |
| 2017 年 | VRF | Algorand | 1/3 | 公有链 |
| 2017 年 | Sharding | Zilliqa | 1/2 | 公有链 |
| 1999 年 | PBFT | Quorum | 1/3 | 许可链 |
| 2014 年 | Raft | Fabric | - | 许可链 |
| 2016 年 | Tendermint | Burrow, Cosmos | 1/3 | 许可链/公有链 |
| 2018 年 | HotStuff | Libra | 1/3 | 许可链/公有链 |

从区块链系统的共识机制的发展过程(表 3)中可以看到,在增加吞吐率、降低能源消耗、降低延迟等现实需求的驱动下,诞生了一系列新的共识机制。这些机制从选取不同侧重点入手,一定程度从技术上解决了部分需求,但是也引入了新的问题如去中心化程度减弱、系统复杂度过高等。需要注意的是,在公有链场景下 PoW 仍然占据主导地位,这一方面固然是用户惯性使然,另一方面也是因为这些改善的共识机制难以与市场上主流应用如比特币、以太坊无缝对接;同时也存在一些用户对这些改善的共识机制有一些不信任心理,因此这些经过改善的共识机制要想真正地在市场上收获成功还有很长的路要走。同时也注意到,伴随着共识机制的发展,公有链与许可链之间的界限也在逐渐淡化,部分公有链在相对加强中心化或者引入传统数据库的优化方法等;许可链也在从公有链的进展上吸取新的思想,具备了更多去中心化理念以及更强的多节点支撑能力等。而在两者均需要面对的拜占庭场景,更是出现了一些同时适用于双方场景的共识算法如 Tendermint、HotStuff 等,公有链和许可链底层共用一套共识机制及实现的场景逐渐变为现实。

回望过去,展望未来,区块链系统的共识机制将会有以下可能的发展趋势。对于公有链系统,依旧会朝着进一步提高吞吐率、降低能源消耗及延迟的方向前进。为了达到这一目标,部分公有链会继续从经典分布式系统解决方案中寻找思路,如过去所做的适度引入中心化理念、加入分片分区等那样,但是仍然会追求通过引入随机属性等避免过度的中心化。智能合约作为区块链系统中的关键组成部分,也会在未来的公有链乃至区块链系统中发挥更大的作用,因此有理由相信难以支撑智能合约的共识机制的发展前

景将会极大受限。对于许可链系统,为了更好地支撑更多的应用场景,会将共识机制作为系统中的一个配置选项并根据场景灵活选择,同时进一步探索更合适的共识机制以支持更大规模的网络,增强系统的去中心化能力以及提高性能等。一些新的适用于双方场景的共识机制已经出现,在未来也会发挥越来越大的作用。因此在共识机制上,公有链系统和许可链系统的界限将会进一步缩小,两者会互相借鉴对方的先进理念,共同促进区块链系统共识机制的发展和进步。

参考文献:

- [1] Nakamoto S. Bitcoin: a peer-to-peer electronic cash system [EB/OL]. (2008) [2020-04-26]. <https://bitcoin.org/bitcoin.pdf>.
- [2] 沈鑫,裴庆祺,刘雪峰. 区块链技术综述[J]. 网络与信息安全学报,2016,2(11):11-20.
- [3] Garay J A, Kiayias A. SoK: a consensus taxonomy in the blockchain era [C]// Proc of Cryptographers' Track at the RSA Conference. Cham: Springer,2020:284-318.
- [4] Androulaki E, Barger A, Bortnikov V, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains [C]//Proc of the 13th EuroSys Conference. New York: ACM Press,2018:1-15.
- [5] Chase M J P. A permissioned implementation of ethereumsupporting data privacy [EB/OL]. (2018-02-20) [2020-04-26]. <https://github.com/jpmorganchase/quorum>.
- [6] 武岳,李军祥. 区块链共识算法演进过程[J]. 计算机应用研究,2020,37(7):2097-2103.
- [7] Lamport L, Shostak R, Pease M. The Byzantine generals problem [J]. ACM Trans on Programming Languages and Systems,1982,4(3):382-401.
- [8] Castro M, Liskov B. Practical Byzantine fault tolerance [C]//Proc of the 3rd Symposium on Operating Systems Design and Implementation. Berkeley, CA: USENIX Association,1999:173-186.
- [9] 王江,章明星,武永卫,等. 类 Paxos 共识算法研究进展[J]. 计算机研究与发展,2019,56(4):692-707.
- [10] Lamport L. The part-time parliament [J]. ACM Trans on Computer Systems,1998,16(2):133-169.
- [11] Lamport L. Paxos made simple [J]. ACM SIGACT News,2001,32(4):51-58.
- [12] Hunt P, Konar M, Junqueira F P, et al. ZooKeeper: wait-free coordination for internet-scale systems [C]//Proc of USENIX Conference on USENIX Annual Technical Conference. Berkeley, CA: USENIX Association,2010.
- [13] Ongaro D, Ousterhout J. In search of an understandable consensus algorithm [C]//Proc of USENIX Conference on USENIX Annual Technical Conference. Berkeley, CA: USENIX Association,2014:305-319.
- [14] Dwork C, Naor M. Pricing via processing or combatting junk mail [C]// Proc of the 12th Annual International Cryptology Conference. Berlin: Springer,1992:139-147.
- [15] Back A. Hashcash: a denial of service counter-measure [EB/OL]. (2002-08-01) [2020-04-26]. <http://www.hashcash.org/papers/hashcash.pdf>.
- [16] Sybil attack [EB/OL]. [2019-10-27]. <https://www.geeksforgeeks.org/sybil-attack/>.
- [17] Sayeed S, Marco-Gisbert H. Assessing blockchain consensus and security mechanisms against the 51% attack [J]. Applied Sciences,2019,9(9):1788.
- [18] De Vries A. Bitcoin's growing energy problem [J]. Joule,2018,2(5):801-805.
- [19] Digiconomist. Bitcoin electronic waste monitor [EB/OL]. [2019-10-02]. <https://digiconomist.net/bitcoin-electronic-waste-monitor/>.
- [20] Bitcoinfees. Bitcoin transaction fees [EB/OL]. [2019-09-20]. <https://bitcoinfees.cash/>.
- [21] Schuh F, Larimer D. Bitshares 2.0: general overview [EB/OL]. (2017-04-04) [2020-04-26]. <https://cryptorating.eu/whitepapers/BitShares/bitshares-general.pdf>.
- [22] Xu B, Luthra D, Cole Z, et al. EOS: an architectural, performance, and economic analysis [EB/OL]. (2018-11-19) [2020-04-26]. <https://blog.bitmex.com/wp-content/uploads/2018/11/eos-test-report.pdf>.
- [23] Gilad Y, Hemo R, Micali S, et al. Algorand: scaling byzantine agreements for cryptocurrencies [C]//Proc of the 26th Symposium on Operating Systems Principles. New York: ACM Press,2017:51-68.
- [24] Hanke T, Movahedi M, Williams D. Definity technology overview series, consensus system [EB/OL]. (2018-05-11). <https://arxiv.org/pdf/1805.04548.pdf>.
- [25] Li Chenxing, Li Peilun, Zhou Dong, et al. Scaling Nakamoto consensus to thousands of transactions per second [EB/OL]. (2018-08-31). <https://arxiv.org/pdf/1805.03870.pdf>.
- [26] Buterin V, Griffith V. Casper the friendly finality gadget [EB/OL]. (2019-01-22). <https://arxiv.org/pdf/1710.09437.pdf>.
- [27] Wang Jiaping, Wang Hao. Monoxide: scale out blockchains with asynchronous consensus zones [C]// Proc of USENIX Symposium on Networked Systems Design and Implementation. Berkeley, CA: USENIX Association,2019:95-112.
- [28] King S, Nadal S. PPOin: peer-to-peer crypto-currency with proof-of-stake [EB/OL]. (2012-08-19). <https://decred.org/research/king2012.pdf>.
- [29] Rose D, Machery E, Stich S, et al. Nothing at stake in knowledge [J]. Nous,2019,53(1):224-247.
- [30] Deirmentzoglou E, Papakyriakopoulos G, Patsakis C. A survey on long-range attacks for proof of stake protocols [J]. IEEE Access,2019,7:28712-28725.
- [31] EOS. IO technical white paper v2 [EB/OL]. (2018-03-16) [2019-09-20]. <https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md>.
- [32] Larimer D. EOSIO dawn 3.0 now available [EB/OL]. (2018-04-06) [2019-09-20]. <https://medium.com/eosio/eosio-dawn-3-0-now-available-49a3b99242d7>.
- [33] Ma Sungmin. EOSIO benchmark [EB/OL]. (2018-04-23) [2019-09-20]. https://github.com/eoseoul/docs/blob/92daf7b3b2325b00a7c96329a4bda80b1a70b23/reports/eoseoul_tps_benchmark_20180423.pdf.
- [34] Micali S, Vadhan S, Rabin M. Verifiable random functions [C]//Proc of the 40th Annual Symposium on Foundations of Computer Science. Washington DC: IEEE Computer Society,1999:120-130.
- [35] Sompolinsky Y, Zohar A. Secure high-rate transaction processing in bitcoin [C]//Proc of the 19th International Conference on Financial Cryptography and Data Security. Berlin: Springer,2015:507-527.
- [36] Wood G. Ethereum: a secure decentralised generalised transaction ledger [EB/OL]. (2014). <http://gavwood.com/Paper.pdf>.
- [37] Sompolinsky Y, Lewenberg Y, Zohar A. SPECTRE: serialization of proof-of-work events: confirming transactions via recursive elections [EB/OL]. (2016-11). <https://eprint.iacr.org/2016/1159.pdf>.
- [38] Sompolinsky Y, Zohar A. PHANTOM, GHOSTDAG: two scalable block-DAG protocols [EB/OL]. (2018). https://pdfs.semanticscholar.org/72ef/7506c2cc017558acea90297a593d96847540.pdf?_ga=2.140994864.342072990.1593223759-1444469156.1514776825.
- [39] Baird L. The Swirls Hashgraph consensus algorithm: fair, fast, Byzantine fault tolerance, SWIRLDS-TR-2016-01 [R/OL]. (2016-05-31). <http://leemon.com/papers/2016b.pdf>.
- [40] ZILLIQA Team. The ZILLIQA technical whitepaper [EB/OL]. (2017-08-10) [2020-04-26]. <https://docs.zilliqa.com/whitepaper.pdf>.
- [41] 邵奇峰,金激清,张召,等. 区块链技术: 架构及进展 [J]. 计算机学报,2018,41(5):969-988.
- [42] The Raft consensus algorithm [EB/OL]. [2020-01-15]. <http://raft.github.io>.
- [43] Buchman E. Tendermint: Byzantine fault tolerance in the age of blockchains [EB/OL]. (2016-06) [2020-04-26]. <https://allquantor.at/blockchainbib/pdf/buchman2016tendermint.pdf>.
- [44] Yin Maofan, Malkhi D, Reiter M K, et al. HotStuff: BFT consensus in the lens of blockchain [EB/OL]. (2019-07-23). <https://arxiv.org/pdf/1803.05069.pdf>.
- [45] Shoup V. Practical threshold signatures [C]//Proc of the 19th International Conference on Theory and Application of Cryptographic Techniques. Berlin: Springer-Verlag,2000:207-220.
- [46] Baudet M, Ching A, Chursin A, et al. State machine replication in the Libra blockchain [EB/OL]. (2019) [2020-04-26]. <https://www.libraplus.org/docs/assets/papers/libra-consensus-state-machine-replication-in-the-libra-blockchain.pdf>.
- [47] Kwon J, Buchman E. Cosmos: a network of distributed ledgers [EB/OL]. [2020-01-15]. <https://cosmos.network/resources/whitepaper>.
- [48] Cypherium. CypherBFT: enabling decentralization for HotStuff [EB/OL]. [2020-01-15]. <https://medium.com/@cypherium/cypherbft-enabling-decentralization-for-hotstuff-1713da26628b>.