计算机科学
COMPUTER SCIENCE

区块链共识算法综述

周凯, 陈福, 鲁添元, 曹怀虎

引用本文

周凯, 陈福, 鲁添元, 曹怀虎. 区块链共识算法综述[J]. 计算机科学, 2025, 52(11): 255-269.

ZHOU Kai, CHEN Fu, LU Tianyuan, CAO Huaihu. Review of Blockchain Consensus Algorithm[J]. Computer Science, 2025, 52(11): 255-269.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.241100140

区块链共识算法综述

周 凯 陈 福 鲁添元 曹怀虎
中央财经大学信息学院 北京 102206
(mulberrychow@163.com)

摘 要 共识算法是区块链的核心支撑技术,本质上是分布式系统各节点就特定数据达成一致性的问题。目前,共识算法存在的最大瓶颈是通信复杂性带来的延迟和吞吐量对区块链性能产生的影响。据此,在系统综述共识技术发展脉络的基础上,分析了基于轮次(Basic-Round,BR)的DAG(Directed Acyclic Graph)分类标准,深入研究了BR-DAG共识算法的核心原理、共识过程,重点阐述了BR-DAG类共识算法降低网络通信延迟、提升共识收敛速度以及提高交易吞吐量的问题。进一步总结了BBCA-Chain等前沿共识算法的研究现状、存在的问题及发展趋势。此外,根据既定的分类标准,提出综合评价体系对各类共识算法在吞吐量、延迟等性能维度上进行对比分析。最后,讨论了目前共识算法面临的挑战,提出未来研究可以围绕BR-DAG和Rho-calculate构建基于消息交互传递的并发计算模型。通过形式化验证的方式,实现高吞吐量、低延迟并且稳健的共识算法。

关键词:共识算法;区块链;拜占庭容错;BBCA-chain;Rho-calculate
中图分类号 TP309

Review of Blockchain Consensus Algorithm

ZHOU Kai,CHEN Fu,LU Tianyuan and CAO Huaihu
School of Information,Central University of Finance and Economics,Beijing 102206,China

Abstract The consensus algorithm is a critical technological cornerstone of blockchain,facilitating consistency among nodes within a distributed system concerning specific data. The primary bottleneck in current consensus algorithms lies in the impact of communication complexity on blockchain performance,specifically in terms of latency and throughput. To address these challenges,this paper presents a comprehensive analysis the development of consensus technology,with a particular focus on the Basic-Round(BR)-based Directed Acyclic Graph(DAG) classification criterion. The present study aims to analyse the core principles of the BR-DAG consensus algorithm and the consensus process. The objective of this analysis is to mitigate the inherent limitations of BR-DAG consensus algorithms by reducing network communication latency,enhancing convergence speed,and increasing transaction throughput. This study offers an extensive review of the current state of research,existing challenges,and emerging trends in advanced consensus algorithms,with a specific emphasis on BBCA-Chain. Furthermore,we propose a robust evaluation framework designed to facilitate the comparative analysis of various consensus algorithms based on throughput,latency,and other performance dimensions,aligned with established classification criteria. Finally,it discusses the prevailing challenges faced by consensus algorithms and proposes future research directions that should focus on BR-DAG and Rho-calculate. This includes the development of concurrent computation models based on message interaction delivery and the formal verification of consensus algorithm correctness. To achieve a high-throughput,low-latency,and robust consensus algorithm,formal verification methods can be employed.

Keywords Consensus algorithm,Blockchain,Byzantine fault tolerance,BBCA-Chain,Rho-calculate

1 引言

2008年中本聪首次提出了比特币[1],拉开了以区块链为底层技术的数字货币序幕。区块链技术的提出,标志着分布式系统迎来了新的发展机遇。共识算法作为区块链的关键基础设施,旨在让分布式系统中的节点就某一决策达成一致性[2]。

最早的分布式一致性算法可以追溯到1978年Gray[3]提出的两阶段提交算法(Two-phase Commit,2PC),其首次尝试解决分布式数据库中的一致性问题。传统分布式一致性算法通常解决集群部署下节点出现宕机等故障导致的数据不一

到稿日期:2024-11-25 返修日期:2025-03-09
基金项目:国家自然科学基金(61672104);中国高校产学研创新基金(2021FNA01002)
This work was supported by the National Natural Science Foundation of China(61672104) and China University Industry University Research Innovation(2021FNA01002).
通信作者:陈福(chenfu@cufe.edu.cn)

致问题[4]。但是,在脆弱非可信网络中会存在恶意节点。1982 年 Lamport 等[5]提出了拜占庭将军问题,将拜占庭错误引入到分布式系统中,保障了共识信息不会被恶意节点操纵[6],并给出了在同步网络下的两种解决方案。此后,分布式共识算法可以根据容错类型分为拜占庭容错和非拜占庭容错 2 类[7]。1985 年,Fischer 等[8]提出了按照其姓氏首字母命名的 FLP(Fischer,Lynch 和 Paterson)不可能性定理,即在异步网络中,只要存在一个故障节点,就不存在一种可以正确终止的解决一致性问题的共识算法。

为规避 FLP 不可能性定理的约束,共识算法的设计通常会在某些方面做出妥协。譬如,1988 年 Dwork 等[9]提出一种介于同步网络与异步网络之间的部分同步模型。在该模型中,FLP 不可能性定理的限制被部分打破,即当系统处于异步状态时共识暂时停止,系统恢复至同步网络时共识继续推进。2002 年,Gilbert 等[10]证明了 Brewer 在 2000 年提出的 CAP 猜想,并正式确立为 CAP(Consistency,Availability,Partition tolerance)定理。CAP 定理,即在一个分布式系统中,一致性(Consistency)、可用性(Availability)、分区容错性(Partition Tolerance)三者无法同时被满足,最多只能满足其中两个特性。该定理为共识算法的设计提供了新的切入点。

Paxos 是首个在异步模型下能够保证正确性且具备容错机制的共识算法[11]。之后在该算法的基础上衍生出许多不同的改进版本,并逐渐形成一个协议族。其中,Raft 是 Multi Paxos 的一种变体[12]。2009 年,中本聪发布了比特币系统,并首次将工作量证明(Proof of Work,PoW)算法引入到区块链中。比特币中的 PoW 通过牺牲强一致性来满足最终一致性、可用性与分区容错性的平衡。

共识算法通过解决分布式系统中节点间的通信交互问题,实现了去中心化的信任机制,为区块链的安全稳定发展奠定了基础[5-6]。随着共识算法的持续发展,区块链技术已在数字货币、物联网和去中心化金融(Defi)等多个领域得到了广泛应用,因此也衍生了不同类型的共识算法。共识算法在区块链系统中的重要性愈发显著[13-16]。其中,文献[17-19]主要关注经典的区块链共识算法分类及对比。随着传统区块链系统在高并发场景下的吞吐量和交易延迟等性能瓶颈问题日益凸显,一些研究开始探索有向无环图(Directed Acyclic Graph,DAG)数据结构的共识算法。然而,现有研究主要将 DAG 应用在区块链分片等特定场景中,其在提升区块链系统整体吞吐量和降低交易延迟等方面的效果有限[20-21]。因此,有学者将研究目标聚焦于 BR-DAG 共识算法,以寻求高吞吐量、低延迟和稳健的共识算法解决方案[22-27]。但是,目前尚未有学者对 BR-DAG 的算法进行梳理总结和分类。此外,近期一些研究[28-29]从加密经济学、博弈论等方面探讨了共识算法,为 BR-DAG 共识算法在吞吐量和延迟性的设计上提供了新的视角。

本文首先深入研究共识算法的演化历程,探讨了不同共识算法中各类共识机制的内在联系;其次,分析了共识算法的发展趋势、目前的研究热点和存在的核心问题;然后,深入研究基于 BR-DAG 的 BFT 算法及其核心逻辑;最后,提出基于轮次的共识算法分类标准及新的评价体系,并在目前共识算法分类标准的基础上补充了 BR-DAG 类的研究。本文通过建立新的共识算法分类标准以及评价指标体系,期望对未来共识算法在吞吐量和延迟性等性能方面的创新提供参考。目前主流分类标准如图 1 所示。

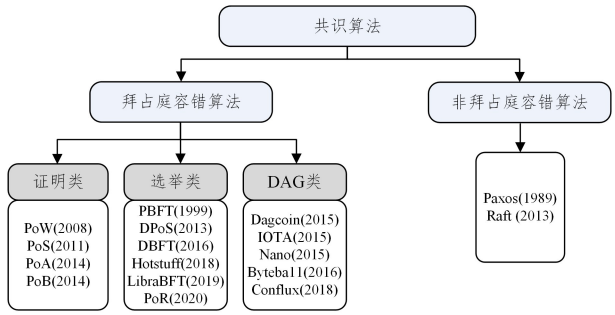

图 1 现有分类标准

Fig. 1 Existing classification criteria

2 共识算法分类

在区块链系统中,共识算法是确保网络节点达成一致的重要机制。首先,根据节点故障类型,现有共识算法主要分为拜占庭容错算法和非拜占庭容错算法 2 类。其中,拜占庭容错算法能够应对节点恶意行为以及非恶意故障,而非拜占庭容错算法则仅处理节点非恶意故障,无法应对节点恶意行为。其次,根据共识策略的不同(如节点出块方式),拜占庭容错算法又可分为证明类、选举类以及 DAG 类算法。其中,证明类通过资源竞争(如算力、权益)随机决定出块权,节点间无需直接通信交互;选举类通过节点间通信,依赖投票机制选举出块节点;DAG 类无严格的出块节点,而是通过直接确认交易的依赖关系构建 DAG,并行处理交易。共识算法的对比如表 1 所列,本文将进一步通过多维性能指标对各类共识算法的技术实现及应用场景进行系统性对比分析。

表 1 共识算法的对比

Table 1 Comparison of consensus algorithms

共识算法分类	组内分类	组内对比	组间对比
拜占庭共识算法	证明类	链式结构,通过资源竞争决定出块权	在存在恶意节点的情况下达成共识
	选举类	链式结构,选举领导节点并由其负责系统负载均衡	
	DAG 类	DAG 结构,并行出块;BR-DAG,数据传播和共识逻辑分离	
非拜占庭共识算法			仅能处理节点故障,无法处理恶意节点

2.1　非拜占庭容错算法

早期的共识算法一般为非拜占庭容错算法,也称崩溃容错协议(Crash Fault Tolerant,CFT)[30]。非拜占庭容错是指在分布式系统中,诚实无恶意节点的网络能够容忍因硬件故障、软件缺陷或网络问题导致的系统故障问题。非拜占庭容错算法的重点在于解决分布式系统中节点存在故障但无恶意行为的共识问题[31]。具体而言,若系统中存在 n 个节点,其中 f 个节点故障,那么需要满足 $n\geqslant 2f+1$ 才能保证系统达成一致性决策。

经典的非拜占庭容错算法有 Paxos 和 Raft 等。下面介绍经典的非拜占庭容错类算法的共识过程。

2.1.1　Paxos

1989 年,Lamport[32]首次提出在异步网络下能够保证系统正确性且具有容错能力的共识算法 Paxos。由于 FLP 不可能性定理的限制,Paxos 以牺牲部分活性为代价,以保证系统达成共识。具体而言,在系统处于异步状态时暂停推进共识,待系统中超半数节点恢复至同步状态后,共识过程继续推进。

随着 Paxos 的不断优化与完善,最终演进形成一个完整的 Paxos 协议簇[33]。理论和实践结果表明,Paxos 极大地提升了分布式共识效率,奠定了分布式一致性算法的基础。然而,由于其晦涩难懂,直至 2006 年 Burrows[34]将其应用到 Chubby 中,实现了分布式锁服务,Paxos 才逐渐为人所知。

2.1.2　Raft

2013 年,Ongaro 等[35]设计了一种比 Paxos 更易于理解和实现的共识算法 Raft,该算法如今已被广泛应用于联盟链和私有链中。Raft 共识算法通过领导选举、日志复制和安全性保障,确保了分布式系统中多个节点的一致性[36]。

Raft 通过将系统复杂性封装在领导者节点中,简化了系统架构。在运行过程中,当集群领导者节点因网络故障或分区失效时,系统能够快速选举出新领导者节点。当网络恢复后,根据任期号规则,历史领导者节点会变成跟随者,并且需要回滚失联期间产生的数据,接受新领导者节点的更新[37]。Raft 通常适用于分布式数据库管理场景,如 Etcd 系统[38]和 TiDB 数据库等。

CFT 类算法以大多数节点正确为前提假设,如果系统中存在恶意节点发送错误消息,会导致系统无法正常运行。例如,在高度实时和高对抗的环境中,Raft 中的恶意节点可能篡改客户端发送的日志项,导致其他节点接收到错误的日志,造成安全性问题。但随着区块链的快速发展,在利益驱使下,节点可能会产生恶意行为。因此,设计共识算法时必须考虑拜占庭节点存在的情况。CFT 类代表算法的对比结果如表 2 所列。

表 2　CFT 类代表算法的对比分析

Table 2　Comparative analysis of representative algorithms for the CFT class

	Paxos	Raft
基本原理	基于选举,分为提议、投票和决策阶段	分为领导选举、日志复制和安全性 3 个模块
网络模型	同步模型	同步模型
角色	提议者、接受者、学习者	领导者、跟随者、候选者
一致性	最终一致性	强一致性
缺点	实现难度大,不适应大规模系统	无法抵御恶意节点、选举过程公平性问题
应用场景	Chubby	Etcd 系统、TiDB 数据库

2.2　拜占庭容错共识算法

拜占庭容错(Byzantine Fault Tolerance,BFT)是共识算法的关键特性,它使区块链能够在部分节点失效或发生恶意行为时仍然保持系统的安全性和可用性[39]。对于拜占庭问题来说,假设系统节点总数为 n,故障节点数为 f,当满足 $n\geqslant 3f+1$ 时才能达成共识。目前解决拜占庭问题的算法有 2 种思路:1)通过提高作恶节点的成本来降低作恶节点出现的概率,如 PoW、权益证明(Proof of Stake,PoS)等;2)在允许一定的作恶节点存在的前提下,依然使得系统各节点之间达成一致性,如实用拜占庭容错算法(Practical Byzantine Fault Tolerance,PBFT)以及 BR-DAG 类共识算法等。根据共识策略,可以将拜占庭类共识算法分为证明类、选举类和 DAG 类 3 种。

2.2.1　基于证明类的共识算法

基于证明类的共识算法通常部署在无权限的公有链环境中,对所有节点开放注册,通过随机选择的方式为参与节点分配出块权。然而,为确保区块链网络的公平性,节点需要通过算力、权益、名誉等多种方式来证明其具备成为区块生成节点的资格[40]。下面介绍以 PoW 和 PoS 为代表的证明类算法的共识过程。

1)PoW

1993 年,Dwork 等[41]首次提出工作量证明的思想,要求用户求解一定难度的函数来获得邮件发送的权限,以此减少垃圾邮件的泛滥。1997 年,Back[42]在此思想的基础上,独立地提出 Hashcash 机制,要求用户完成计算工作并证明其拥有足够的资源调度这些服务,从而有效地防止垃圾邮件和拒绝服务攻击。直至 1999 年,Jakobsson 等[43]正式提出了工作量证明的概念,其为后来比特币的共识算法提供了思路。

2008 年,PoW 作为共识算法应用在比特币系统中。PoW 利用 SHA-256 函数正向高效计算与逆向求解困难的特性,让参与节点竞争解决一个难题,旨在消耗一定时间获得解,同时确保其他节点高效验证解的有效性[44]。PoW 共识流程要求区块链中的矿工节点在区块数据(BlockData)上,通过改变区块头的 Nonce 字段找到一个合适的哈希值,使其小于某个目标难度值(D),即 $SHA256(BlockData,Nonce)<D$。当某一节点成功找到满足条件的随机数后,系统将该区块广播到区块链中进行验证,验证通过后将该区块添加到主链末端,成功出块的节点将获得相应奖励。

PoW 共识算法采用求解数学难题的方式构建了完备的安全体系,当大多数节点参与到求解运算时,作恶节点攻击会付出极大的成本,进而保证了区块链的安全。然而,PoW 算法也存在一定的缺陷。首先,PoW 需要投入大量矿机进行算力支撑,导致大量能源消耗,造成环境负担[45]。其次,PoW 的

挖矿奖励，造成计算资源过度集中，使 PoW 共识的公平性和去中心化特性遭到破坏。最后，PoW 为降低区块链的分叉概率，设置了较大的区块生成间隔，进而导致交易确认时间长、吞吐量低的问题。

2）PoS

2011 年，Mechanic[46] 在 Bitcointalk 论坛首次提出了基于权益证明的共识算法，即节点拥有代币比例越高，获得记账的概率就越大。随即在 2012 年，King 等[47] 发布了点点币（Peercoin），首次实现了基于 PoS 共识算法的加密货币。点点币在运行机制中引入了币龄（Coinage）概念，用来计算节点持币时间和数量。币龄越大，交易优先级越高，越容易生产区块。与 PoW 在广域空间搜索 Nonce 值不同，PoS 将搜索空间限制在可控范围内去考虑币龄与难度值（D）[11]，使得节点只需消耗少量计算资源就可以完成区块生成，即 $SHA256(Blockheader) \leqslant D \times Coinage$。2013 年发布的 Nextcoin[48]，是首个完全基于 PoS 共识算法的数字货币项目。

虽然 PoS 显著降低了算力需求，大幅节省了算力消耗，但是当大量代币集中在少数节点上时，这部分权益较高的节点更容易生成区块，导致中心化现象。另外，纯粹的 PoS 区块链无法实现冷启动，在实际运行时，一般优先采用 PoW 启动区块链系统，再切换到 PoW&PoS 混合方式，最后切换到完全的 PoS[39,49]。证明类代表算法的对比结果如表 3 所列。

表 3 证明类代表算法的对比分析

Table 3 Comparative analysis of representative algorithms for proof classes

	PoW	PoS（以及变种）
基本原理	通过计算复杂数学题来验证交易和生成区块	通过持有代币数量和时间验证交易和生成区块
去中心化程度	高度去中心化，因矿池集中化而降低去中心化	依赖于持币者的参与度，导致富者愈富的现象
能源消耗	计算资源消耗较多	资源消耗较少
应用场景	比特币、莱特币	以太坊、Solana、Nextcoin

2.2.2 基于选举类的共识算法

投票选举是分布式系统中达成共识的经典方法。基于选举类的共识算法允许节点票选出领导节点，并由其生产区块和达成共识决策[17]。选举类共识算法通常部署在节点数量固定且受许可控制的环境中。证明类和选举类的共识算法各有优点，因此部分区块链算法融合了两者的思想[50]，采用了混合的方式，如 2-hop[51]，PoA[52] 和 Casper FFG[53] 等。此外，还有其他为特定领域而设计的共识算法，如非匿名证明算法、数据编校算法和 DAG 类算法等[54]。下面对具有代表性的选举类共识算法的工作过程进行描述。

1）PBFT

1999 年，Castro 等[55] 提出了 PBFT 共识算法，这是一种在拜占庭节点存在的情况下，能够实现 $O(n^2)$ 算法复杂度的实用解决方案。此外，PBFT 共识算法的吞吐量通常高于证明类共识算法，适用于高交易吞吐量、低延迟的应用场景[56]。

PBFT 是一种状态机复制的拜占庭一致性算法，在保证系统活性和安全性的前提下提供了 $(n-1)/3$ 的容错性[57]。PBFT 算法的目的是各节点响应客户端发出的请求，并按照同一个确定顺序执行。在 PBFT 中有且只有一个主节点，其他为副本节点。主节点负责对客户端的请求进行排序，副本节点遵循主节点提供的顺序来执行相应的请求。所有节点在相同的配置下工作，当主节点更换时，随之切换到新的视图周期[58]。PBFT 共识流程如图 2 所示。

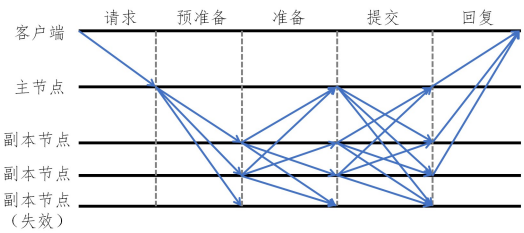

图 2 PBFT 共识流程图

Fig. 2 Flow chart of PBFT consensus

PBFT 开创了异步网络中拜占庭容错的实用解决方案，为后续共识算法（如 BFT-SMaRt[59]，SBFT，HotStuff 和 Prosecutor[60]）追求更高的吞吐量和更低的延迟奠定了基础。由于 PBFT 在中小规模的场景中能够满足高性能、低延迟和容错性需求，因此 FISCO BCOS 区块链平台选择 PBFT 作为共识算法。

2018 年，Gueta 等[61] 在 PBFT 算法的基础上进行了创新改进，提出了 SBFT 共识算法。SBFT 与 Zyzzyva[62] 相似，在无系统故障且网络保持同步的情况下，为系统提供了一条快速的共识路径。当网络不符合预期时，SBFT 自动切换到慢路径，即线下 PBFT。与 PBFT 相比，SBFT 利用阈值签名避免了二次拜占庭广播的复杂过程。

虽然 PBFT 较 PoW 等证明类算法在延迟和共识速度方面有着一定优势，但其也存在一些问题。比如，对恶意节点的检测和新领导节点的选举过程非常低效，在视图切换时具有 $O(n^3)$ 的高通信复杂度。

2）HotStuff

随着应用场景的不断拓展，区块链系统在吞吐量和通信开销等方面提出了更严苛的要求[63]。2019 年，Yin 等[64] 提出 HotStuff——一种创新的基于领导者的拜占庭容错算法。具体而言，在部分同步的网络中，HotStuff 选择性接收 $2f+1$ 条消息推进共识，其结合灵活的领导者轮换机制，不仅实现了快速响应，而且确保了链条质量[65]。与此同时，HotStuff 采用与 SBFT 相似的设计理念，利用阈值签名技术实现线性消息复杂度，相较于 PBFT 通过 P2P 通信达成共识的方式，显著降低了通信开销[66]。HotStuff 的复制过程始于客户端向所有副本广播请求，其客户端需要等待来自 $f+1$ 个副本的回复，以确认请求操作已被执行。

HotStuff 的共识过程需要经历 4 个阶段[67]。1）准备阶段。当接收到客户端请求后，主节点向所有副本节点广播一条准备消息。副本在确认消息连续且视图编号（ViewNum）增加后，对准备投票（Prepare-vote）消息进行签名并将其发送给主节点。主节点收到至少 $2f+1$ 个投票后，生成群体证明

(PrepareQC)。2)预提交阶段。在生成 PrepareQC 后,主节点进入预提交阶段。首先,主节点广播带有 PrepareQC 的消息。其次,各副本存储 PrepareQC 证书,随后向主节点返回带有签名的提交投票(Commit-vote)消息。最后,主节点收集到 $2f+1$ 个投票后形成预提交证明书(Pre-commitQC)。3)提交阶段。首先,主节点将带有 Pre-commitQC 的消息广播给所有副本。其次,副本回复主节点提交投票并更新本地 LockedQC。最后,主节点收集到足够的提交投票后,形成最终的共识决定。4)决定阶段。首先,主节点广播决定消息,各副本确认共识达成,并执行请求操作。随后,副本增加视图编号,并向新主节点发送新的视图消息(New-View)。同时,副本向提议请求的客户端发送确认消息,表示请求已在本地执行。最后,客户端等待 $f+1$ 条消息以确认其请求已被执行。HotStuff 共识流程如图 3 所示。

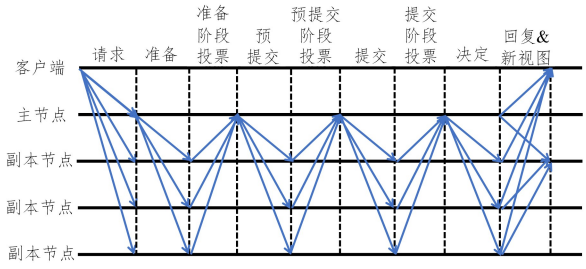

图 3　HotStuff 共识流程图

Fig. 3　Flow chart of HotStuff consensus

在共识过程中,HotStuff 每个副本存储 PrepareQC,LockedQC 和 ViewNum 这 3 个关键参数来达成客户端请求的共识。相较于其他基于领导节点的算法,视图变更是 HotStuff 的核心机制。HotStuff 支持高频视图变更,每个共识实例均由新的主节点发起,同时主节点负责广播消息和汇集阈值签名。具体而言,在每个共识阶段,领导者负责收集投票并构建可流水线化的阈值签名。这种设计不仅优化了决策过程,简化了链式 HotStuff 的构建,还显著降低了算法的时间复杂度。此外,它引入决策流水线化,允许多个共识实例并行进行,从而有效提升了系统吞吐量。因此,基于 HotStuff 算法优化改进的 LibraBFT 共识算法,在 Libra 区块链系统中实现了高效稳定的运行。该算法通过引入循环轮换机制,从所有副本中动态选举出主节点[68]。这一优化机制使得 Libra 实现了低成本、高效率的支付和转账,为区块链技术在金融支付领域的应用提供了重要的理论基础和实践参考。

Hotstuff 虽然具有良好的性能表现,但也存在一定的不足。首先,高频的领导权轮换机制在面对节点故障频发时,有一定的脆弱性。其次,HotStuff 只考虑了处理客户端请求的编号和去重问题,未深入探讨客户端故障时的处理方案,导致出现客户端请求丢失及状态不一致等问题。最后,HotStuff 系统中的负载能力随着故障节点达到系统最大承受能力时而降低,系统吞吐量也会受到影响,呈现下降趋势[69]。选举类代表算法对比结果如表 4 所列。

表 4　选举类代表算法的对比

Table 4　Comparison of representation algorithms for election classes

	PBFT	HotStuff
基本原理	基于投票的共识协议,通过多轮投票来达成共识	改进的投票机制,采用单轮提案和投票减少延迟
网络模型	同步网络	部分同步和异步网络
延迟	较高	较低
容错效率	节点有限时容错效率高,节点增加后性能下降	较好的容错效率,适用于中等规模的容错需求
应用场景	FISCO BCOS 区块链平台	Libra 区块链

2.3　基于 DAG 的共识算法

区块链共识算法的研究尽管在近年来取得了显著进展,但在实际应用中仍面临着诸多方面的性能瓶颈与挑战[70]。首先,以比特币为例,系统的平均交易处理能力仅为每秒 5～7 笔交易。与此同时,主流支付工具 Visa 每秒可以处理 2000 笔交易[71]。其次,许多研究认为降低通信复杂性是实现高性能的关键,过于追求线性通信协议。然而,低通信复杂度并没有使系统吞吐量得到实质性提高。例如,HotStuff 算法只能实现每秒 3500 笔交易的吞吐量,远低于当前区块链网络对吞吐量的要求。最后,在选举类算法中,领导节点承担着交易传播和共识决策的双重责任,使其面临巨大的工作负载,进而引发交易积压和系统瓶颈。

为了追求更高的性能,直观思路是采用并行数据结构来代替传统的链式结构。相比之下,基于 DAG 结构的 BFT 算法实现了交易的并行化处理,成为构建高性能、稳健性区块链系统的优选方案。DAG-BFT 不仅有效解决了传统 BFT 系统面临的性能瓶颈问题,更为未来区块链和分布式账本技术的发展提供了重要的技术基础和理论支撑。

DAG 的结构如图 4 所示。2015 年,Lerner[72] 提出了首个以交易为基本单元的 DAG 账本 DagCoin,其中每笔交易无需等待打包区块后确认,而是直接作为独立的节点融入 DAG 中,为后续 DAG 在实际项目的应用提供了诸多参考,如 IOTA[73],Obyte[74] 和 Hashgraph[75] 等。

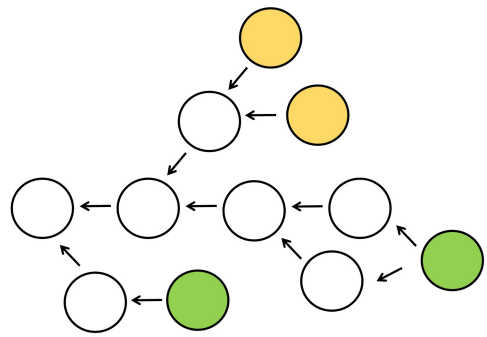

图 4　DAG 结构图

Fig. 4　Structure diagram of DAG

Tangle 是应用于 IOTA(Internet of Things Application)中的 DAG 网络结构。IOTA 是一种用于物联网场景的分布式账本,旨在突破传统区块链技术在小额支付和高效交易处理上的局限性。如图 5 所示,在 Tangle 中,每个节点代表一

笔交易，当节点创建新交易时，使用私钥对该交易进行签名，并进行轻量 PoW 计算。工作量越大，交易的自身权重越高。而累积权重则体现了交易在网络中的信任度，它由交易自身的权重以及在 DAG 中直接或间接引用该交易的所有交易的权重之和构成。

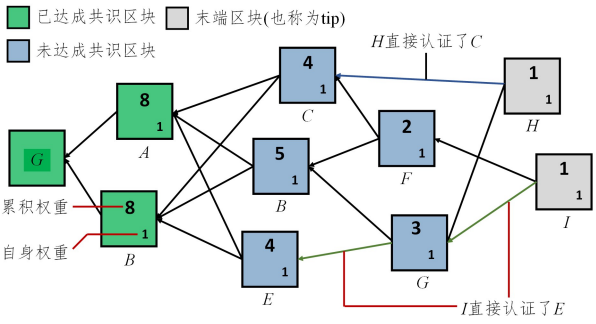

图 5 IOTA 中的 Tangle 网络

Fig. 5 Tangle network of IOTA

IOTA 使用马尔可夫链蒙特卡洛(Markov chain Monte Carlo，MCMC)算法对抗恶意攻击并惩戒节点懒惰行为。该算法根据累积权重在 Tangle 中随机引用两笔未确定的交易(tip)，累积权重越高，确认概率越大[76]。一旦新交易验证了两笔历史交易，并完成了轻量 PoW 计算，该交易将会被广播到整个网络，成为 Tangle 中的一部分。随着 Tangle 网络中参与节点的增加，节点累积权重加快，系统会缩短交易确认时间，降低网络延迟。

由于 Tangle 采用 DAG 结构以及轻量化的 PoW 算法，可以实现在低能耗条件下的高交易吞吐量，因此通常将 Tangle 部署在物联网等低功耗设备上。

Obyte 同样采用 DAG 数据结构处理交易数据。Obyte 与 IOTA 类似，每笔新交易都会确认一笔或多笔之前的交易。在 Obyte 中，交易的顺序与有效性由主链(MainChain)及一组 12 名见证人(Witnesses)共同决定，其中主链路径依据见证人所确认的交易构建[77]。尽管 DAG 结构能够并行处理交易，但主链上的依赖关系限制了其并发性能，从而在一定程度上影响了系统的吞吐量。与此同时，主链的依赖性要求交易必须等待与之关联交易的确认，不可避免地增加了交易的延迟。

在 Hashgraph 中，节点通过生成和传播事件(event)记录交易，并引用两个父事件：Self-Parent，即节点最近生成的事件；Other-Parent，即由相邻节点生成的事件。因此，在区块添加阶段接收事件时，节点必须验证父事件是否按照协议规定正确设置。与 IOTA 和 Obyte 相比，Hashgraph 中的节点通过 Gossip 协议与虚拟投票机制高效同步信息并快速达成共识，避免了传统共识算法的冗余计算和通信，具有较高的吞吐量和极低的延迟。

除此之外，关于 DAG 类共识算法的研究还在持续探索，综述[18,78-79]对 DAG 类的相关工作进行了系统性梳理。表 5 对现有基于 DAG 的分布式账本进行了详细对比。

表 5 DAG 类代表算法的对比

Table 5 Comparison of representative algorithms for the DAG class

	IOTA	Obyte	Hashgraph
基本原理	使用轻量级 PoW 进行交易确认，每笔交易需验证两笔历史交易	基于见证人机制，验证节点通过见证人获得共识	基于 Gossip 协议和虚拟投票算法，节点通过时间戳确定顺序
交易确认	Tips	Witnesses	2/3 节点
吞 吐 量	随着节点数量增加，吞吐量提高	依赖主链，吞吐量受限	Gossip 协议和虚拟投票吞吐量高
延迟	延迟较低，确认速度较快	延迟相对较高	Gossip 协议和虚拟投票延迟低
能源效率	轻量级 PoW 计算，消耗较低	无挖矿需求，资源消耗小	无需挖矿和 PoW 计算，能耗极低
应用场景	IOTA Rebased、EDAG CityBot	Byte 币、PolloPollo 平台	Hedera 账本、AdsDax 平台

本文系统性地对传统共识算法进行分类和总结，将其分为拜占庭和非拜占庭两大分支，并从容错度、复杂度、应用场景、代表应用以及优缺点等维度对各子类别进行了深入分析与对比，以期为后续在不同应用场景下选择合适的共识算法提供参考。

详细对比结果如表 6 所列。

表 6 主流共识算法的分类与比较

Table 6 Classification and comparison of mainstream consensus algorithms

	分类	共识算法	容错度	复杂度	应用场景	代表应用	优势	劣势
非拜占庭容错		Paxos	<1/2	$O(n)$	联盟链	Chubby	扩展性好，适合可信网络	无法应对恶意节点攻击，容错性较差
		Raft	<1/2	$O(n)$	联盟链	Etcd		
拜占庭容错	证明类	PoW	<1/2	$O(n)$	公有链	比特币	能够容忍网络节点存在恶意行为，适合开放性网络	实现复杂，性能较低，扩展性受限，吞吐量较低
		PoS	<1/2	$O(n)$	公有链	以太坊		
	选举类	PBFT	<1/3	$O(n^2)$	联盟链	超级账本		
		HotStuff	<1/3	$O(n)$	联盟链	Libra		
	DAG 类	IOTA	<1/2	—	公有链	IOTA	具备高并发、高吞吐量和可扩展性	存储成本高，容易遭受攻击，全局一致性难以保障
		Obyte	<1/2	—	私有链	Obyte		
		Hashgraph	<1/3	$O(1)$	联盟链	—		

3 BR 类共识算法

随着 DAG 类共识的快速发展，BR 类共识算法因在吞吐量和延迟性方面具有显著优势，已逐步成为区块链解决方案的重要演进方向。BR 类算法的可行性，源于 Hashgraph 将数据传播与共识逻辑解耦[75,80]。具体而言，BR-DAG 类算法将数据传播从共识逻辑中分离出来，所有节点(计算机实体)同时传播数据，而共识算法只对少量元数据进行排序。基于 BR-DAG

结构不但实现了高吞吐量，而且具有低延迟和可扩展性。

BR类共识算法的汇总如图6所示。

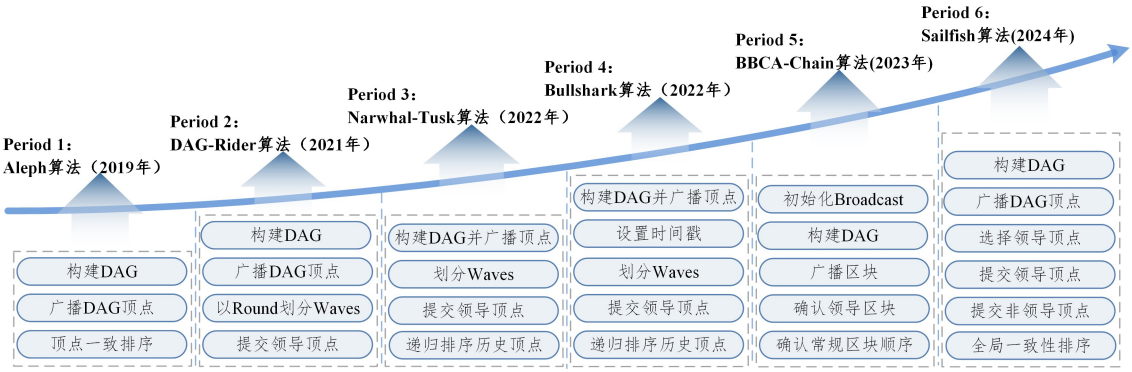

图6　BR类共识算法

Fig. 6　BR class consensus algorithms

目前，BR-DAG类的共识算法呈现出基于领导者和无领导者独特的结合方式。与传统选举类票选领导节点不同，其领导顶点具有随机性且数量不固定。BR-DAG类的共识算法在数据传播阶段不涉及领导者，而在共识阶段需要领导者存在；而且在BR-DAG类中，每个节点可以异步地计算共识状态与提交序列，并能自主解析其本地DAG。即使在网络异步性条件下，验证节点（Validator Nodes）在同一时刻拥有不同的DAG，系统仍能就提交序列最终达成一致[67]。

在BR-DAG共识算法中，当一条消息被确认嵌入到DAG时，系统能够保证具有可靠性（Reliability）、非模糊性（Non-Equivocation）和因果排序（Causal Ordering）3个关键特性[80-81]。1）可靠性：副本消息存储在足够多的验证节点上，并且所有诚实节点最终都可以下载。2）非模糊性：在两个验证节点的本地DAG中具有完全相同的顶点和引用关系。3）因果排序：顶点引用上一轮次的交付消息，可以保证只要验证节点提交了某个顶点，其余验证节点的局部视图中对该顶点引用历史轮次的消息也是完全相同的。

3.1　Aleph

Aleph[22]是首次基于BR方式构建DAG的共识算法，其核心是通过可靠的广播机制（Reliable Broadcast，RBC）确保在异步网络存在拜占庭节点的情况下，所有诚实节点最终输出相同的交易排序。在Aleph中，每个DAG顶点都与一个轮次（Round）相关，每个节点每轮可以广播一个顶点，每个顶点需要引用前一轮中至少 $n-f$ 个顶点，并且每个顶点通过RBC实现广播，以确保拜占庭节点不能在同一轮内将不同的顶点分发给不同的节点。Aleph采用将交易传播与共识过程分离的方式构建DAG，避免了视图更改或视图同步等复杂机制。

与传统BFT算法相比，Aleph采用了更简单的排序机制，在交易延迟方面相比于HoneyBadgerBFT[82]等异步共识算法实现了更低的延迟和更快的交易处理速度。在实际应用中，以Aleph为核心的Aleph Zero区块链可以实现接近毫秒验证时间和每秒近十万笔的交易，为追求高效、低延迟、可扩展和高吞吐量的区块链系统提供了可行的方案。

3.2　DAG-Rider

2021年，Keidar等[23]提出基于异步拜占庭原子的广播协议DAG-Rider。DAG-Rider由2层组成。1）通信层：节点通过Bracha广播发布尚未确认的交易，并通过构建BR-DAG调节通信开销，优化吞吐量。2）共识层：节点对所有DAG顶点进行本地计算并完成排序，该过程没有额外的通信开销。

在DAG-Rider中，每个节点独立维护自己本地的DAG，而DAG中的顶点代表各节点对所有节点之间的消息拓扑解释。各节点通过创建和连接顶点，将新消息与历史消息关联起来，串联它们的因果关系。DAG-Rider的共识进程以轮次为单位进行，并将4个连续的轮次划分成1个Wave。在每个Wave中，各节点利用完美硬币机制在第1轮中随机选出唯一的领导顶点。其中领导顶点的提交条件是，需要被下一轮 $2f+1$ 条强边引用，如果未满足提交条件，则直接跳过该Wave，直到新的Wave中有满足条件的领导顶点被提交。该条件保证了网络中的每个Wave中所有节点都向同一个领导顶点提交。此外，DAG-Rider定义了强边和弱边两组边。强边，是在一个回合中直接引用前一轮中的顶点。弱边，则是没有直接引用前一轮的顶点，其作用是保持DAG的连通性，提升协议的容错性，避免因通信延迟导致消息遗漏。

DAG-Rider的共识过程如图7所示。当客户端发起交易时，节点主动接收相应的顶点，将其存储在缓冲区中，并持续监控这些顶点的状态，以确定缓冲区内的顶点是否成功接收到所有必要的前导顶点。一旦满足条件，节点就根据其各自的轮次编号，将这些顶点整合到DAG中。每个顶点在每轮中至少引用 $2f+1$ 个前一轮的顶点，才能推进到下一轮次。DAG-Rider利用完美硬币机制，使节点能够独立检查各自的DAG，并推断出要提交的区块以及交易顺序，而无需在副本之间进行额外通信。在跨节点交付DAG中的顶点后，系统启动共识操作。图7中，顶点 V_2 和 V_3 分别是在第 w 个和 $w+1$ 个Wave中的领导顶点。DAG-Rider在Wave w 的第 r 轮尝试提交 V_2，但由于在第 $r+3$ 轮中少于 $2f+1$ 条边引用 V_2，因此在第 $r+3$ 轮时 V_2 不满足提交条件。于是继续往前推进轮次，直至在第 $r+7$ 轮中有 $2f+1$ 条路径通往 V_3，因此 V_3 符合提交规则，并在该Wave结束时被提交。同时，由于存在一条从 V_3 到 V_2 的路径（弱边），因此 V_2 作为 V_3 因果历史的一部分，将会一起被提交。

DAG-Rider底层利用可靠广播协议保证所有正确的进程最终都具有一致的DAG视图。通过引入基于轮次的设计，

DAG-Rider 显著提高了系统的稳定性，实现了 1.5 个 Wave 的预期提交时间。与传统基于领导节点的 BFT 算法相比，DAG-Rider 可以并行处理交易，在吞吐量上表现出优势。但由于交易和共识分离，在一定程度上会造成延迟的增加，这也侧面反映了区块链系统设计中吞吐量与延迟之间的博弈。DAG-Rider 的理论为后续 Mysten Labs 开发 Sui 链提供了启发和技术支持。

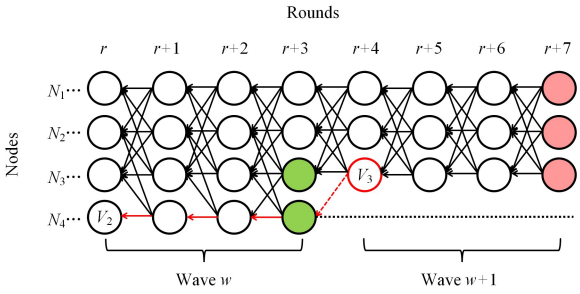

图 7 DAG-Rider 共识流程图

Fig. 7 Flow chart of DAG-Rider consensus

3.3 Narwhal-Tusk

2022 年，Danezis 等[24]在 DAG-Rider 算法的基础上进行了优化，提出了基于 DAG 的内存池协议和高效的 BFT 共识算法——Narwhal-Tusk。Narwhal 摒弃了传统模式下交易先至内存池再汇总广播的方式，使用基于 DAG 的内存池将交易传播职责与数据排序分离，将可靠的交易传播交由内存池协议处理，而共识算法只对少量元数据进行排序，从而显著提高了系统性能[81,83]。其中，Tusk 是部署在 Narwhal 上的 BFT 共识算法。

Narwhal 继承了 DAG-Rider 创建 DAG 的方式，同时利用 DAG 的特征，创新性地提出了可用性证书的可靠广播机制。在 Narwhal 中，DAG 由多个区块构成，每个区块包含一个哈希签名、一组交易列表以及前一轮区块的可用性证书集合。证书列表包含了当前区块的哈希值、来自其他验证者的 $2f+1$ 个签名以及轮次编号 r，证书的组成元素保障了区块的可靠交付。

Narwhal 构建 DAG 的过程如图 8 所示。系统中各节点持续接收客户端交易，并将其存储于交易列表中，同时将接收到的证书保存在证书列表中。当共识过程推进至第 r 轮时，如果节点从当前轮的其他节点处收到 $2f+1$ 个证书，共识将进入下一轮。随后，系统将创建一个包含当前交易的区块，并由发送节点将该区块广播至其他节点(接受节点)。接受节点在接收到区块后，首先验证区块的签名，以确保它包含上一轮的 $2f+1$ 个证书。此外，接受节点还需要确认该区块是发送节点广播的唯一区块。当满足条件后，接受节点将对该区块的哈希进行签名，以确认该区块。发送节点一旦接收到对该区块的 $2f+1$ 个确认签名，就会给区块创建一个可用性证书并广播该证书，然后发送节点停止广播原始块。循环往复，最终生成一副 DAG。

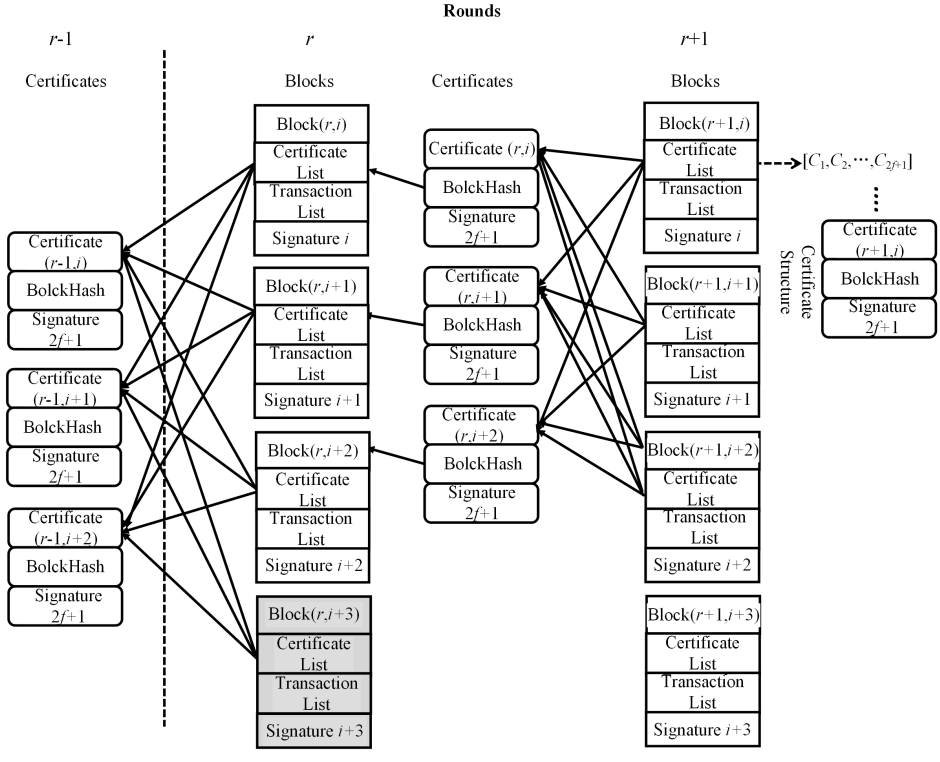

图 8 Narwhal-Tusk 共识流程图

Fig. 8 Flow chart of Narwhal-Tusk consensus

当 Narwhal 内存池完成传播交易后，则由 Tusk 接管共识过程。在共识中，Tusk 随机选择一个领导块，采用与 DAG-Rider 类似的共识逻辑[23]。不同之处在于，Narwhal-Tusk 缩短了用于达成共识的轮数。在 Narwhal-Tusk 中，以 3 轮划分为 1 个 Wave，并且下一个 Wave 的第 1 轮为上一个 Wave 的最后一轮，从而使共识的时间复杂度为 4.5 轮。

相较于传统的 BFT 算法(如 PBFT 和 HotStuff)，Narwhale-Tusk 具有更高的吞吐量。但是，Narwhal 中那些尚未

满足交付标准的交易需要经历更多轮次的等待，因此不可避免地导致延迟的上升。但整体上，Narwhal-Tusk 可以实现 160 000 的交易吞吐和大约 3 秒延迟的表现，较传统 BFT 系统的吞吐量和延迟有着巨大的性能提升。因此在实际应用中，最初 Sui 链选择将 Narwhal 和 Tusk 算法结合，并在高吞吐量和低延迟之间找到了巧妙的平衡。随着 Sui 链的成熟，逐渐发展形成 Sui 生态，如去中心化金融项目 MoveEX 等。

3.4　Bullshark

2022 年，Spiegelman 等[25,84]提出了 Bullshark 共识算法，并提供了同步与部分同步网络下的两种解决方案。Bullshark 改进了 Tusk 的公平性，确保在同步网络情况下，所有诚实节点的交易都能被打包。与此同时，Bullshark 引入了 Narwhal 内存池协议，使节点无需额外增加通信成本，仅凭本地 DAG 与 Bullshark 算法便能自主解析 DAG 中的交易关系，并达成全序一致。

Bullshark 将 DAG 中的轮次按奇偶类型进行划分。在奇数轮中，通过全局硬币机制选出领导顶点作为锚点。在偶数轮中，需要引用奇数轮的锚点对其投票。具体来说，当锚点获得超过 $f+1$ 选票时，该锚点就会被提交，节点便可借助排序规则为已确认的锚点及其因果历史达成全序一致。考虑到异步场景，在共识推进过程中会存在拜占庭节点或历史锚点遗漏的情况，导致各节点在其本地视图中拥有不同的 DAG。于是，Bullshark 设计了一套机制，确保即便本地 DAG 各异时，节点间也能就顶点的全序达成一致。其核心是，若 DAG 中存在一条连接历史锚点至未来锚点的路径，当未来锚点获得确认时，则历史锚点也可获得确认。若不满足此条件，则可忽略此旧锚点。节点可以遵循此逻辑递归追溯未提交的历史锚点，直至找到已确认的起点。

Bullshark 异步共识过程如图 9 所示。Bullshark 中，每个 Wave 中包含 4 个轮次，从每个 Wave 中选出负责推进轮次的 Steady-State Leader 锚点，S1A、S1B、S2A、S2B 和递归回溯的 Fallback Leader 锚点，以及 F1 和 F2。Bullshark 共识过程优先提交 Steady-State Leader 锚点，当其不满足提交条件时，投票类型变成 Fallback 类型。根据 Fallback 的特性，当其满足提交条件后，节点需要回顾其历史记录寻找未提交的 Steady-State 类型的锚点，并尝试提交。常规情况下，Bullshark 每两轮就能提交一个 Steady-State Leader，相较于 Tusk 算法做了进一步优化，同时 Fallback Leader 的设计也保证了即使在最坏的异步条件下，共识依然具有活跃性。

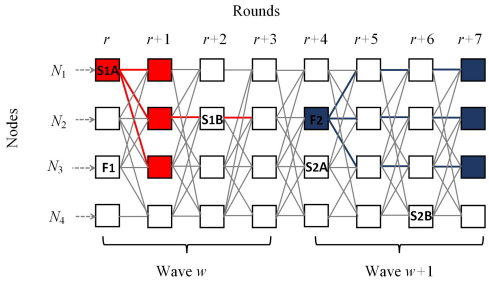

图 9　Bullshark 共识流程图

Fig. 9　Flow chart of Bullshark consensus

此外，Bullshark 引入垃圾收集机制，对 DAG 中的历史数据进行清理，以实现内存的有效释放。与此同时，对于系统中运行速度较慢的节点而言，历史数据的回收意味着增加了访问历史信息的难度，损害了共识算法的公平性。于是，Bullshark 引入全球稳定时刻（GST），当节点在广播顶点时添加时间戳，通过时间戳调整公平性的界定范围，仅在同步过程中确保公平性。在部分同步网络中，划分为异步与同步两个阶段，在达到 GST 之前保持异步，之后同步运行。与之前部分同步算法（如 SBFT，Tendermint[85] 和 Hotstuff）相比，Bullshark 具有最简单的算法实现。

由于 BullShark 相较于 Tusk 具有更优的数据处理速度和公平性，因此 2022 年 Sui 链的共识算法从 Tusk 过渡到 Bullshark，以缓解 Tusk 运行中的延迟、效率及公平性问题。在实际应用中，基于 Narwhal 和 Bullshark 的 Sui 链实现了每秒超十万次交易的处理能力，交易最终确认时间也仅为 400 多毫秒。这一性能指标极大地超越了传统区块链的性能，使得 Sui 能够支持高频交易和复杂的金融操作，且能够极大地提升用户体验，满足市场对快速交易的需求。

3.5　BBCA-Chain

2023 年，Malkhi 等[26]提出了 BBCA-Chain，旨在简化 Tusk 和 Bullshark 算法的共识逻辑，同时降低网络传输过程的延迟。BBCA-Chain 适用于异步网络，最多能容忍 $f<n/3$ 拜占庭节点，其中 BBCA-Chain 的广播原语 BBCA（Byzantine Broadcast With Commit-Adopt）是建立在因果有序的拜占庭一致广播基础上的。

BBCA-Chain 通过调用 BBCA 广播，将领导区块链接到 DAG 共识决策主干中。由于网络的异步性，各节点本地的 DAG 会以不同的方式演变，因此 BBCA 设计 Commit-Adopt 以确保已提交领导区块的安全性，而非领导者区块则采取尽力而为（Best-Effort Broadcast，BEB）的方式独立且并行地传输。BBCA-Chain 具体流程如图 10 所示。

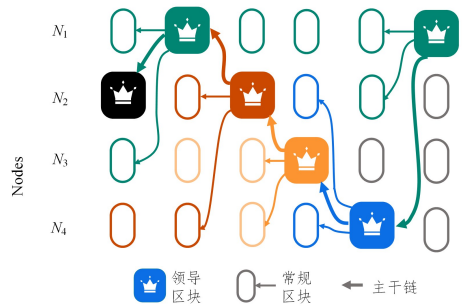

图 10　BBCA-Chain 共识流程图

Fig. 10　Flow chart of BBCA-Chain consensus

BBCA-Chain 是基于逐视图的共识算法，当节点进入到视图 v 时，领导者 B_v 启动计时器，在预定时间内完成 *BBCA-commit*(B_v) 来提交领导区块，而非领导区块则广播常规区块。当 B_v 被提交后，首先调用 *BBCA-broadcast*(B_v) 广播 B_v，当 $2f+1$ 个节点完成对 B_v 的签名认证后，随即进行 *BBCA-Complete*。然后，系统中引用其因果历史的常规区块也将一起被交付。最后，视图 v 成功前进到视图 $v+1$。这种提交方式被称为 Fast 路径。如果视图 v 的计时器超时，则会调用 *BBCA-probe*() 来检查 v 中领导块的状态，节点会得到 2 种响应结果，即〈*BBCA-Noadopt*〉或〈*BBCA-Adopt*，B_v〉，并将响应

通过 *BBCA-broadcast*()嵌入到广播中，随即视图推进到 $v+1$。如果满足 Adopt 条件，即至少拥有 $f+1$ 个可信任节点的证书，表明该领导块在网络中被接受，并且视图 $v+1$ 的领导块 B_{v+1} 应该引用通过 Adopt 路径提交的领导块。如果不满足 Adopt 条件，节点可以将 *BBCA-Noadopt* 值嵌入到下一个视图 $v+1$ 中。

总而言之，如果没有足够多的节点确认广播的信息，那么共识算法可能会引发视图更换，以尝试从一个新的状态达成共识。在进入视图 $v+1$ 后，系统随即启动一个新的定时器。视图 $v+1$ 的领导者会调用 *BBCA-broadcast*(B_{v+1})，广播视图 $v+1$ 的领导区块 B_{v+1}。B_{v+1} 会引用视图 v 里状态为 Completed 或者 Adopt 的领导块 B_v。如果没有这两种状态，则引用超过 $2f+1$ 个被签上 Noadopt 的区块。一般地，如果该视图里面的领导块被签上了 Noadopt 的状态，那么该视图里面的常规区块也会被标记上同样的标记。与此同时，BBCA-Chain 通过探测(Probe)机制确保了即使领导区块没有在规定时间内完成任务，也能根据已有的信息继续推进共识，并在进入新的视图后尝试解决未完成的状态，以保持网络的一致性和稳定性。该机制允许共识算法动态适应网络条件和节点行为的变化，确保了共识过程的鲁棒性。

BBCA-Chain 通过 3 项关键技术实现了简单的共识逻辑，在显著提升系统吞吐量的同时有效降低了网络延迟。1)在可靠广播之上添加了一个垫片来创建 BBCA 原语，支持节点在本地窥视协议内部，无需任何通信成本。2)该算法不需要专门的提议、投票过程，进一步降低了系统延迟。3)只有领导者块需要使用 BBCA，所有其他块则通过简单的尽力广播进行，这种单一广播的提交决策减少了通信次数，降低了共识延迟，也简化了共识逻辑。因此，BBCA-chain 在最佳情况下仅需 4 次网络通信交互，相较于 Bullshark 中的最佳情况减少了 8 次网络交互延迟。BBCA-chain 的提出，为构建高效、弹性的区块链系统提供了理论支撑。

3.6 Sailfish

目前在 BR-DAG-BFT 算法中，节点在每个轮次中均会创建一个包含提案信息的 DAG 顶点，并且依赖领导者推进共识过程。因此，选举领导者的频率和领导者提交的速度直接影响着共识效率。于是，2024 年 Shrestha 等[27]提出了首个支持每轮都有领导者的 DAG-BFT 共识算法 Sailfish。

在 Sailfish 中，节点每轮提出的提案表示为 DAG 顶点，顶点需要在规定时间内引用上一轮次中至少 $2f+1$ 个顶点，引用关系形成 DAG 边。Sailfish 利用 Bracha 可靠广播来传播顶点。当 Sailfish 共识过程推进到第 r 轮时，节点 N_i 需要等待至少 $2f+1$ 个第 $r-1$ 轮的顶点，以及该轮的预定义的领导顶点。在 Sailfish 中，对第 $r-1$ 轮的领导顶点引用被视为对其进行投票，并用以提交该领导顶点。若节点在超时前接收到必要的顶点和领导顶点，那么该节点立即进入第 r 轮并提出新的顶点。节点若在超时前未收到第 $r-1$ 轮的领导顶点，则会向第 r 轮的领导顶点发送不投票消息，表明未对 $r-1$ 轮的领导顶点进行投票。

在 Sailfish 中，第 r 轮的领导顶点需满足特定条件才能推进共识过程：1)具有通往第 $r-1$ 轮领导顶点的路径；2)持有第 $r-1$ 轮的不投票证书。不投票证书证明有足够多的参与节点未对第 $r-1$ 轮的领导顶点进行投票，从而使该顶点无法被提交，保证了系统的安全性。这 2 项条件确保了 Sailfish 在每轮中都能有效支持一个领导顶点。

在 Sailfish 中，只有领导顶点被提交，其他非领导顶点则作为领导顶点的因果历史的一部分并按确定性顺序进行排序。图 11 描述了节点在 r 轮和 $r-1$ 轮提交区块的 DAG 视图。在 r 轮中领导顶点获得超过 $2f+1$ 条 $r+1$ 轮的投票，符合提交规则，于是该领导顶点被提交，并确定性地排序引用该领导顶点的历史轮次中的顶点。另外，由于 r 轮的领导顶点没有引用 $r-1$ 轮的 N_4 顶点，因此该顶点目前还未进行排序。

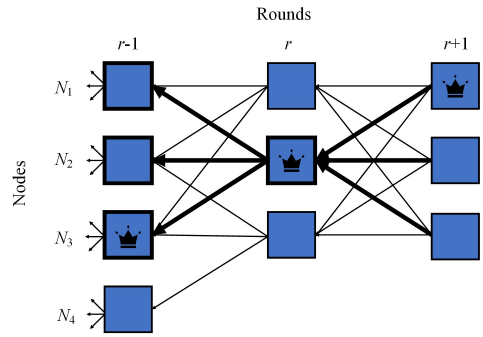

图 11 Sailfish 共识流程图

Fig. 11 Flow chart of Sailfish consensus

与现有的 BR-DAG-BFT 算法相比，Sailfish 在保持类似吞吐量的同时，还显著降低了系统延迟。譬如，Aleph 通过异步二元一致性协议决定顶点的顺序，这种方式会造成更大的提交延迟。虽然 BBCA-Chain 本质上也能实现每轮中容纳一个领导者，但其属于传统领导节点密集工作量型，当恶意节点有选择地仅向领导者节点发送提案时，其他节点下载缺失的提案会产生额外的延迟。在当前区块链系统中，随着交易处理能力的不断提高，系统延迟已成为影响交易最终确认时间的关键因素之一。于是，Sailfish 团队将其与 Bullshark 和 BBCA-Chain 等进行了对比测试，结果表明在延迟方面 Sailfish 具有显著的优势。BR-DAG 类共识算法的对比如表 7 所列。

表 7 BR-DAG 类共识算法的对比

Table 7 Comparison of BR-DAG class consensus algorithms

协议	Aleph	DAG-Rier	Tusk	Bullshark	BBCA-Chain	Sailfish
网络模型	异步	异步	异步	部分同步	异步	部分同步
协议	Aleph	DAG-Rier	Tusk	Bullshark	BBCA-Chain	Sailfish
容错度	1/3	1/3	1/3	1/3	1/3	1/3
选举领导频率	—	1/4	1/3	1/2	—	1
提交延迟	N	4	3	2	—	1
等待广播共识	抛硬币	抛硬币	抛硬币	计时器	计时器	计时器
应用场景	Aleph Zero	Defi	Sui 链	Sui 链	—	—

4　评价体系

根据本文从传统共识算法到最新的 BR-DAG 类共识算法的分析和总结,区块链共识算法可以从以下 6 个维度建立评价体系:去中心化、安全性、可扩展性、吞吐量、通信复杂度、确认延迟和资源消耗。与此同时,本文设计综合评分公式对评价维度进行量化:$D=\sum_{i=1}^{n} w_i \cdot Score_i$。其中,$n$ 为指标数目;$Score_i$ 为相关指标得分;D 为最终得分[86-87];w_i 是权重系数,反映各指标的重要性。根据各指标对结果的贡献对权重系数进行量化[88],使其满足 $w_1+w_2+\cdots+w_n=1$。权重系数的量化如表 8 所列。具体评价对比结果如表 9 所列。

表 8　量化权重系数

Table 8　Quantization weight coefficient

指标名称	权重范围	示例指标
关键指标	≥0.6	数据结构
一般指标	0.3～0.6	节点数量
轻度指标	≤0.3	容错度

表 9　共识算法的评价指标

Table 9　Evaluating indicators of consensus algorithms

算法	去中心化	安全性	可扩展性	吞吐量	确认延迟	资源成本
PoW	高	高	高	极低	极高	极高
PoS	高	高	高	低	高	高
HotStuff	低	中	中	中	低	中
Alehp	中	高	高	高	中	低
DAG-Rider	中	高	高	高	中	低
Tusk	中	高	高	高	中	低
Bullshark	中	高	高	高	中	低
BBCA-Chain	中	高	高	高	低	低
Sailfish	中	高	高	高	低	低

在系统评价体系中,指标的设定对后续系统的优化有至关重要的作用[17,88]。本文根据重要程度将指标划分为 3 类,并根据主观赋权法划分权重范围[89]。首先,关键指标是系统性能的主要驱动因素,其表现直接决定系统的性能。其次,一般指标对系统性能有显著影响,优化此类指标可提升整体表现。最后,轻度指标对核心性能影响较小,但优化后可在特定场景下带来一定收益,或支持系统的长远发展。本文的评价体系对与性能密切相关的指标赋予了更高的权重。

去中心化与参与共识的节点数量、节点记账方式以及共识机制 3 个指标相关,作为衡量去中心化的标准,根据各项指标对去中心化的影响,将其权重系数都量化为 1/3 权值,以确保各指标在综合评估中具有同等的重要性,从而更准确地捕捉到不同区块链系统去中心化特性的细微差别。证明类允许所有节点在任意时间参与共识,实现完全去中心化,其参与节点数量最多。选举类是所有参与节点票选出领导节点,并由其创建区块,与基于证明类的共识算法相比,选举类算法在参与节点数量相当的情况下的节点记账方式和共识机制的差异明显加强了中心化特性,因此在最终得分上要低于证明类。DAG 类算法的共识过程也依赖于特殊参与者,如 Obyte 中的见证人以及 BR-DAG 中的领导顶点,这会存在一定程度的中心化问题,但其共识机制和数据结构的不同避免了中心化的来源,在共识机制和记账方式的指标得分上要高于竞争类,而其节点数量要少于证明类,因此其最终得分高于竞争类而低于证明类算法。

安全性主要受系统的抗攻击能力、攻击成本以及容错度 3 个指标影响,具体而言,攻击力和攻击成本对系统安全性的影响更为直接,是确保系统稳定和防范潜在威胁的关键因素。因此,分别为抗攻击能力、攻击成本以及容错度赋予 2/5,2/5 和 1/5 的权重。其中,抗攻击能力主要指共识算法是否具有识别并排除恶意节点参与共识的能力;攻击成本是指恶意节点发起攻击所付出的代价;容错度则是系统容忍恶意节点或故障节点所占的比例。选举类算法对节点行为具有较强的控制力,在发现拜占庭节点之后,通过后续选举过程将其排除。BR-DAG 设置节点参与门槛,提高链路质量,以保障系统安全。对于竞争类算法,一旦领导节点受到攻击,将对系统产生较大影响。因此,竞争类算法在抗攻击能力指标上的得分最低。在攻击成本指标上,证明类算法中,由恶意节点发动攻击的代价远高于其他种类。在容错度方面,所有的算法在设计之初都已考虑恶意节点存在的情况,因此各分类算法在该指标上的得分差异不明显。综上,由评分公式可知,证明类算法的安全性最高,其次是 BR-DAG 类,最后是竞争类算法。

可扩展性主要与参与共识的节点数量和数据结构 2 个指标相关。虽然节点数量是可扩展性的核心衡量指标,但其对系统的影响更依赖于底层数据结构,过多的节点反而会导致系统性能下降。通过优化数据结构,可以降低这种负面影响。因此,其权重略低于数据结构指标,分别设为 2/5 和 3/5。在证明类算法中,节点增加会导致验证时间延长,限制系统的可扩展性。选举类和 BR-DAG 是通信密集型算法,适应于有限节点数量的系统,并且需要所有节点广播。因此,节点的增加会导致网络拥塞,使其难以扩展到上万节点而没有性能损失。而证明类共识算法,例如 PoW,一方面,其不需要广播投票;另一方面,其性能较差,节点数量对扩展性影响较小。因此,整体上在同样的链式结构下,证明类算法的扩展性要优于选举类和 BR-DAG 类算法。但是,BR-DAG 类因具有并行数据结构而存在天然的可扩展性优势,故在数据结构这一指标上的得分要远高于其他两类。因此,BR-DAG 具有最高的可扩展性评分,其次为证明类,最后为选举类。

吞吐量是指系统在单位时间内可处理的交易数量,主要受通信复杂度和网络结构 2 个指标影响。为了公平地评估各个指标对吞吐量的影响,每个指标的权重系数被设定为 1/2,确保了在综合评估过程中各指标具有同等重要性。选举类,通过优化共识流程(尤其是 HotStuff,最低可实现线性通信复杂度),通常情况下具有比证明类和 DAG 类更低的线性复杂度。其中,以 PoS 为基础的以太坊最高处理速度仅为每秒 15 笔交易,略高于比特币系统;而基于 PBFT 的 Fabric 和基于 HotStuff 的 Libra 可以实现每秒 1000+的交易。但从网络结构看,非 DAG 类共识算法的链式结构使交易序列化,极大地限制了吞吐量;而 DAG 类共识算法允许并行交易数据和广播区块,尤其是 BR-DAG 类进一步将数据传播和共识过程分离,吞吐量比非 DAG 类更高,如 Sui 链可以支持每秒超过 10 万笔交易。因此在网络结构指标评分上,BR-DAG 类算法遥遥领先。综上,虽然在通信复杂度指标上选举类具有最高评

分，但经过加权求和后，在吞吐量维度上，BR-DAG 具有最大的优势。

延迟性是指交易从生成到最终确认所需的时间，与节点属性以及时间复杂度挂钩。为确保在综合评估中平等考虑各指标的重要性，将延迟性的权重系数量化为 1/2。其中，节点属性的含义是领导节点更换频次以及工作量。证明类依赖于节点算力和权益，确认延迟通常较高，比特币的延迟可达 30～60 分钟。选举类则是通过领导节点在共识过程中的协调作用来降低消息复杂度，实现低延迟。而 BR-DAG 需要经过多轮领导节点才可以确认交易，这在一定程度上增加了系统延迟，但是 Sailfish 和 BBCA-Chain 在并行处理和通信效率上进行了优化，大幅降低了延迟性，因此其可以实现与竞争类相似的交易延迟。

资源成本包括 CPU、网络、内存、计算能力和存储等。证明类，尤其是 PoW，依赖于算力竞争，资源成本较高；PoS 的权益证明虽会消耗少量算力资源，但对资源的消耗已经远远少于 PoW。BR-DAG 将数据传输和共识逻辑分离，因此本地节点仅需极少的资源成本。选举类中只有领导节点的工作量是最大的，其他节点只需投票，但由于领导节点随机生成，每个节点都需要大量的算力储备，造成算力的浪费，故其最终的资源成本高于 DAG 类。因此，在资源成本维度上，以 PoW 为代表的证明类算法具有最高的资源成本，其次是竞争类算法，最后是 BR-DAG 类算法。

5 Rho-calculus 与共识机制

在区块链系统中，共识算法作为核心组件，通过协调分布式节点间的状态一致性，对系统的安全性与可靠性起到了关键性作用。然而，随着区块链应用需求的多样化发展，特别是面向高频交易和复杂金融操作等高性能需求场景，如何在兼顾系统吞吐量和延迟性的同时实现对分布式环境中多节点间共享状态的动态一致性协调，这一问题的本质就是区块链节点之间的通信、交互以及资源共识的并发计算问题。而在并发计算领域中，进程代数作为一种重要的建模工具，为理解、验证和建模并发系统提供了理论支持。进程代数是一种用于描述和分析并发系统行为的数学模型，被广泛应用于网络协议、分布式系统和软件工程等领域[90]。

Rho-calculus 是由 Meredith 等[91]提出的一种新的进程代数，它以一种直观且形式化的方式描述和分析分布式系统的行为。不同于传统的进程代数（如 π-calculus）[92]，Rho-calculus 更侧重于对系统状态及其随时间演化的过程的描述。Rho-calculus 的核心概念包括表示系统状态的 Rho-term 以及描述从一个 Rho-term 到另一个 Rho-term 转换演化的 Reduction。Rho-calculus 可以描述各种复杂的并发行为，包括异步通信、同步通信、移动性等。

区块链是一种分布式系统，而共识算法是其核心技术之一。Rho-calculus 作为一种强大的并发模型，可以为区块链共识机制的形式化建模与分析提供强有力的理论工具，从而有助于揭示其内在的逻辑复杂性及性能优化潜力。例如，将区块链的整个状态（包括区块、交易、节点等）看作一个 Rho-term。Rho-term 中的节点可以代表区块，边可以表示区块之间的链接关系。共识过程可以看作一系列的 Reduction，即从一个 Rho-term 到另一个 Rho-term 的转换。每个 Reduction 代表一次共识协议中的操作，如产生新区块、验证区块等。通过定义一系列的 Reduction 规则，可以形式化地描述各种共识协议，如 PoW 和 PoS 等。Rho-calculus 在共识机制的形式化验证、性能评估、智能合约的并发行为分析和跨链交互的安全性分析等领域具有很强的能力。其中，RChain 就是基于 Rho-calculus 设计的区块链平台。与以太坊等区块链平台采用顺序虚拟机不同，RChain 支持高并发的智能合约执行，提升了区块链系统的可扩展性和吞吐量。下面给出用 Rho-calculus 对 PoW 进行建模，分析矿工、交易池和网络之间的交互的过程，如算法 1 所示。

算法 1 Rho-calculus 形式化算法

```
输入：交易 tx
1. 初始化：内存池 T＝new tx in(receive(tx) | wait_for_miner)
2. 矿工 M＝new nonce in(collect_tx | build_block | compute_hash
   (nonce))
3. 区块链网络 N＝new b in(broadcast(b) | verify(b))
     /＊ 验证 PoW 共识算法的动态交互 ＊/
4. BlockchainSystem＝T | M | N/＊ 内存池、旷工和网络并行操作 ＊/
5. T＝new tx in(receive(tx) | wait_for_miner)
     M＝new nonce in(
          collect_transactions | /＊ 收集交易 ＊/
          build_block | /＊ 构建区块 ＊/
          (compute_hash(nonce) | /＊ 寻求随机数 ＊/
       if(hash ＜ target) then(broadcast(b) | N)
   else(nonce:＝nonce ＋ 1 | M)) /＊ 如果满足条件广播区块 b，否则
   遍历随机数，直至满足条件 ＊/
     )
6. N＝new b in(
        receive(b) |
        if(verify(b)) then(add_to_chain(b) | N)
   else(reject(b) | N)
     ) /＊ 区块 b 验证通过后，添加到网络中 ＊/
```

通过 Rho-calculus 对 PoW 共识算法建模，为分析共识算法的安全性、扩展性和公平性提供了新的方式。此形式化方法清晰地描述了矿工、交易池和网络之间的动态交互，为共识算法的优化和安全性设计奠定了理论基础。具体来说，矿工节点的作用为，在收集交易和构建区块的基础上，通过计算哈希值并根据目标值决定是否广播区块。网络节点则是负责接收区块并验证，进而决定是否将其添加到链上。该模型还能对矿工间的竞争公平性进行分析，探究是否存在某些矿工总是能优先发现符合条件的哈希值的现象。通过形式化验证的方式，可以为共识算法的特性和设计方向提供思路。

结束语 本文对共识算法进行了分类，并给出了系统的概述，特别关注到最新共识算法的发展。结合当前研究现状，确定了目前 BFT 算法的挑战仍然是在实现高吞吐量和低延迟之间寻找平衡，以及在不同区块链网络中实现跨链共识和互相操作问题。首先是在金融领域，共识算法的高吞吐量和低延迟特性对于处理海量数据具有至关重要的作用。其次，跨链协作可以促进不同平台间的数据要素可信流通，实现数

据市场的互联互通。

针对现有 BFT 共识算法解决方案中的主要问题，未来共识算法的发展方向将更加注重提高区块链的性能和跨链互操性，同时兼顾吞吐量和通信延迟。随着区块链技术的不断深入，共识算法还需要不断创新，以适应在物联网、去中心化金融等具有特殊需求场景中的应用。本文后续工作包括：首先，将继续从 BR-DAG 类共识算法如何降低通信复杂度、加快收敛速度、提高共识效率入手，在当前 BR-DAG 类算法的基础上进一步提高吞吐量，降低延迟，如设计新的共识原语，或者新的 DAG 内存池结构；其次，紧密结合 Rho-calculus 的并发执行和跨分片交易的能力，进一步实现低延迟的高吞吐量交易处理，以解决共识算法的可扩展性和跨链协作难题。通过解决以上关键问题，共识算法将持续为区块链技术的发展提供保障，为学术界和工业界开发出安全高效的区块链系统做出贡献。

参 考 文 献

[1] NAKAMOTO S. Bitcoin：A peer-to-peer electronic cash system [EB/OL]. https://bitcoin. org/bitcoin. pdf? kuid＝5e3c690d-b96e-4fb7-a9cd-c3119e61fe99-1743128278.

[2] ROLIM J，MUELLER F，ZOMAYA Y. Parallel and Distributed Processing[J]. Lecture Notes in Computer Science，1993，5(3)：217-221.

[3] GRAY J. The transaction concept：Virtues and limitations[C]// VLDB. 1981：144-154.

[4] SHAO Q F，ZHANG S，ZHU Y C，et al. Overview of Enterprise Blockchain Technology[J]. Journal of Software，2019，30(9)：2571-2592.

[5] LAMPORT L，SHOSTAK R，PEASE M. The Byzantine Generals Problem[J]. ACM Transactions on Programming Languages and Systems，1982，4(3)：382-401.

[6] HAN X，YUAN Y，WANG F Y. Security Problems on Blockchain：The State of the Art and Future Trends[J]. Acta Automatica Sinica，2019，45(1)：206-225.

[7] WANG Q，LI F J，NI X L，et al. Survey on Blockchain Consensus Algorithms and Application [J]. Journal of Frontiers of Computer Science & Technology，2022，16(6)：1214-1242.

[8] FISCHER M J，LYNCH N A，PATERSON M. Impossibility of distributed consensus with one faulty process[J]. Journal of the ACM，1985，32(2)：374-382.

[9] DWORK C，NAOR M. Pricing via processing or combatting junk mail[C] // Advances in Cryptology—CRYPTO’92. Berlin：Springer，1993：139-147.

[10] GILBERT S，LYNCH N. Brewer’s Conjecture and the Feasibility of Consistent，Available，Partition-tolerant Web Services[J]. ACM SIGACT News，2002，33(2)：51-59.

[11] LAMPORT L. The Part-Time Parliament [J]. ACM Transactions on Computer Systems，1998，16(2)：133-169.

[12] ONGARO D，OUSTERHOUT J. In search of an understandable consensus algorithm[C] // 2014 USENIX Annual Technical Conference(USENIX ATC 14). 2014：305-319.

[13] LIN L S，ZHENG H Q，SU S，et al. An On-Chain Security Mechanism Against DeFi Price Manipulation Attacks[J]. Journal of Computer Research and Development，2025，62(2)：443-457.

[14] DAMAŠEVIČIUS R，MISRA S，MASKELIŪNAS R，et al. Convergence of blockchain and Internet of Things：integration，security，and use cases[J]. Frontiers of Information Technology & Electronic Engineering，2024，25(10)：1295-1321.

[15] SHEN M，CHE Z，ZHU L H，Anonymity in Blockchain Digital Currency Transactions：Protection And Confrontation[J]. Chinese Journal of Computers，2023，46(1)：125-146.

[16] LI J ，YUAN Y，WANG F Y. Blockchain-based Digital Currency：The State of the Art and Future Trends[J]. Acta Automatica Sinica，2021，47(4)：715-729.

[17] 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.

[18] LIU Y Z，LIU J W，ZHANG Z Y，et al. Overview on Blockchain Consensus Mechanisms[J]. Journal of Cryptology，2019，6(4)：395-432.

[19] FU X，WANG H，SHI P. A survey of Blockchain consensus algorithms：mechanism，design and applications[J]. Science China Information Sciences，2021，64：1-15.

[20] GAO Z F，ZHENG J L，TANG S Y，et al. State-of-the-art Survey of Consensus Mechanisms on DAG-based Distributed Ledger[J]. Journal of Software，2019，31(4)：1124-1142.

[21] WANG Q，YU J S，CHEN S P，et al. SoK：DAG-based blockchain systems [J]. ACM Computing Surveys，2023，55(12)：1-38.

[22] GĄGOL A，LEŚNIAK D，STRASZAK D，et al. Aleph：Efficient atomic broadcast in asynchronous networks with byzantine nodes[C] // Proceedings of the 1st ACM Conference on Advances in Financial Technologies. 2019：214-218.

[23] KEIDAR I，KOKORIS-KOGIAS E，NAOR O，et al. All you need is dag[C] // Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing. 2021：165-175.

[24] DANEZIS G，KOKORIS-KOGIAS L，SONNINO A，et al. Narwhal and tusk：a dag-based mempool and efficient bft consensus [C] // Proceedings of the Seventeenth European Conference on Computer Systems. 2022：34-50.

[25] SPIEGELMAN，A，NEIL G，ALBERTO S et al. Bullshark：Dag bft protocols made practical[C] // Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. 2022：2705-2718.

[26] MALKHI D，STATHAKOPOULOU D，YIN M F. BBCA-CHAIN：One-Message，Low Latency BFT Consensus on a DAG [J]. arXiv：2310. 06335，2023.

[27] SHRESTHA N，SHROTHRIUM R，KATE A，et al. Sailfish：Towards Improving Latency of DAG-based BFT[C] // 2025 IEEE Symposium on Security and Privacy (SP). IEEE，2025：1928-1946.

[28] FENG B M，WANG S X，ZHANG J Q. Progress in Cryptoeconomics Research[J]. Economic Perspective，2023(11)：125-140.

[29] YUAN X Z. Consensus game and consensus equilibrium inblock-

chain ecology[J]. Operations Research Transactions, 2024, 28(3):1-26.
[30] XIA Q, DOU W S, GUO K W, et al. Survey on Blockchain Consensus Protocol[J]. Journal of Software, 2021, 32(2):277-299.
[31] YUAN Y, NI X C, ZENG S. Blockchain Consensus Algorithms: The State of the Art and Future Trends[J]. Acta Automatica Sinica, 2018, 44(11):2011-2022.
[32] LAMPORT L. The part-time parliament[J]. ACM Transactions on Computer Systems, 1998, 16(2):133-169.
[33] YI X C, WEI H F, HUANG Y, et al. TPaxos Consensus Protocol in PaxosStore: Derivation, Specification, and Refinement[J]. Journal of Software, 2020, 31(8):2336-2361.
[34] BURROWS M. The Chubby lock service for loosely-coupled distributed systems[C]// Proceedings of the 7th Symposium on Operating Systems Design and Implementation. USENIX Association, 2006:335-350.
[35] ONGARO D, OUSTERHOUT J. In search of an understandable consensus algorithm[C]// Proceedings of the 2014 USENIX Annual Technical Conference. 2014:305-319.
[36] GE N, HE Y K, ZHAI S M, et al. Formal Verification of Consensus Protocols: Survey and Perspective[J]. Journal of Software, 2022, 34(11):4989-5007.
[37] WANG D, DOU W S, GAO Y, et al. Raft Protocol Testing Based on TLA+ Formal Specification[J]. Journal of Software, 2024, 35(12):5363-5381.
[38] HAN S C, ZHU X R, ZHANG X X. Optimized scalable Byzantine fault tolerance algorithm[J]. Chinese Journal on Internet of Things, 2020, 4(2):18-25.
[39] ZHANG X, ZHONG W, YANG C, et al, BFT Consensus Algorithms[C]// 2023 IEEE 10th International Conference on Cyber Security and Cloud Computing. 2023:434-439.
[40] XU J, WANG C, JIA X. A survey of blockchain consensus protocols[J]. ACM Computing Surveys, 2023, 55(13):1-35.
[41] DWORK C, NAOR M. Pricing via processing or combatting junk mail[C]// Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology. 1992:139-147.
[42] BACK A. Hashcash—a denial of service counter-measure[EB/OL]. http://www.hashcash.org/papers/hashcash.pdf.
[43] JAKOBSSON M, JUELS A. Proofs of work and bread pudding protocols[C]// Secure Information Networks. 1999:258-272.
[44] YI L, LU X Y, TANG K, et al. Overview on consensus algorithms of blockchain [J]. Electronic Design Engineering, 2024, 32(6):161-170.
[45] WU Y, LI J X. Evolution process of blockchain consensus algorithm[J]. Application Research of Computers, 2020, 37(7):2097-2103.
[46] MECHANIC Q, Proof of stake[EB/OL]. https://en.bitcoin.it/wiki/Proof of Stake.
[47] KING S, NADAL S M. PPcoin: Peer-to-peer crypto-currency with proof-of-stake [EB/OL]. https://www.semanticscholar.org/paper/PPCoin%3A-Peer-to-Peer-Crypto-Currency-with-King-Nadal/0db38d32069f3341d34c35085dc009a85ba13c13.
[48] NXT Generation of Cryptocurrency. NXT whitepaper[EB/OL]. [2024-03-27]. https://nxtdocs.jelurida.com/Nxt_Whitepaper.
[49] QIN B, QIAO X. Development and Security of Blockchain Consensus Mechanism[J]. ZTE Technology Journal, 2018, 24(6):8-12.
[50] ZHU J M, ZHANG Q N, GAO S. Research Progress of Blockchain Key Technologies and Their Application[J]. Journal of Taiyuan University of Technology, 2020, 51(3):321-330.
[51] DUONG T, FAN L, ZHOU H S. 2-hop blockchain: combining proof-of-work and proof-of-stake securely [EB/OL]. (2023-09-30). https://eprint.iacr.org/2016/716.
[52] 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.
[53] BUTERIN V, REIJSBERGEN D, LEONARDOS S, et al. Incentives in Ethereum's hybrid casper protocol[C]// 2019 IEEE International Conference on Blockchain and Cryptocurrency. 2019:236-244.
[54] FU X, WANG H M, SHI P C, et al. Jointgraph: A DAG-based efficient consensus algorithm for consortium blockchains[J]. Software: Practice and Experience, 2021, 51(10):1987-1999.
[55] CASTRO M, LISKOV B. Practical Byzantine fault tolerance [C]// Proceedings of the 3rd Symposium on Operating Systems Design and Implementation. 1999:173-186.
[56] YANG, J, JIA Z H, SU R G, et al. Improved fault-tolerant consensus based on the PBFT algorithm[C]// IEEE Access, 2022, 10:30274-30283.
[57] FANG W W, WANG Z Y, SONG H L, et al. An optimized PBFT consensus algorithm for blockchain[J]. Journal of Beijing Jiaotong University, 2019, 43(5):58-64.
[58] ZHANG Z W, WANG G R, XU J L, et al. Survey on Data Management in Blockchain Systems[J]. Journal of Software, 2020, 31(9):2903-2925.
[59] BESSANI A, SOUSA J, ALCHIERI E E P. State machine replication for the masses with BFT-SMART[C]// 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. IEEE, 2014:355-362.
[60] ZHANG G, JACOBSEN H A. Prosecutor: An efficient BFT consensus algorithm with behavior-aware penalization against Byzantine attacks[C]// Proceedings of the 22nd International Middleware Conference. 2021:52-63.
[61] GUETA G G, ABRAHAM I, GROSSMAN S, et al. SBFT: A scalable and decentralized trust infrastructure[C]// 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks(DSN). IEEE, 2019:568-580.
[62] KOTLA R, ALVISI L, DAHLIN M, et al. Zyzzyva: speculative byzantine fault tolerance[C]// Proceedings of twenty-first ACM SIGOPS Symposium on Operating Systems Principles. 2007:45-58.
[63] CHENG A D, XIE S J, LIU A, et al. Efficient Quantum-secure Byzantine Fault Tolerance Consensus Mechanism Based on HotStuff[J]. Computer Science, 2024, 51(8):429-439.
[64] YIN M, MALKHI D, REITER M K, et al. HotStuff: BFT consensus with linearity and responsiveness [C]// Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing. 2019:347-356.

[65] GARAY J,KIAYIAS A,LEONARDOS N. The bitcoin backbone protocol: Analysis and applications[J]. Journal of the ACM,2024,71(4):1-49.

[66] DENG X H,WANG Z Q,LI K T,et al. ACT-BFT:Byzantine Fault Tolerant Consensus Mechanism Based on Adaptive Communication Topology[J]. Journal of Computer Engineering & Applications,2023,59(21):267.

[67] ZHANG G R,PAN F,MAO Y H,et al. Reaching consensus in the byzantine empire:A comprehensive review of bft consensus algorithms[J]. ACM Computing Surveys,2024,56(5):1-41.

[68] BAUDET M,CHING A,CHURSIN A,et al. State machine replication in the libra blockchain[C]// Proceedings of the 14th USENIX Symposium on Operating Systems Design and Implementation. 2020:633-649

[69] ZHANG G,JACOBSEN H A. Prosecutor:An efficient BFT consensus algorithm with behavior-aware penalization against Byzantine attacks[C]// Proceedings of the 22nd International Middleware Conference. 2021:52-63.

[70] SUN Z X,ZHANG X,XIANG F,et al. Survey of Storage Scalability on Blockchain[J]. Journal of Software,2020,32(1):1-20.

[71] SHI J,ZHANG A,BAI X Y,et al. Survey on Performance Optimization Technologies of Distributed Ledger System[J]. Journal of Software,2022,34(10):4607-4635.

[72] LERNER S D. DagCoin:A cryptocurrency without blocks [EB/OL]. (2024-03-11). https://bitslog. files. wordpress. com/2015/09/dagcoin-v41. pdf.

[73] POPOV S. The Tangle [EB/OL]. (2018-04-30) [2024-04-25]. https://assets. ctfassets. net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3. pdf.

[74] CHURYUMOV A. Byteball:A decentralized system for storage and transfer of value [EB/OL]. [2024-09-30]. https://byteball. org/Byteball. pdf.

[75] BAIRD L. The swirlds hashgraph consensus algorithm:Fair,fast,byzantine fault tolerance:Swirlds Tech Reports:SWIRLDS-TR-2016-01[R]. 2016:9-11.

[76] ZHAO G S,ZHANG H,WANG J. A Mobile Crowdsensing Data Security Delivery ModelBased on Tangle Network[J]. Journal of Electronics & Information Technology,2020,42(4):965-971.

[77] DONG Z,ZHENG E,CHOON Y,et al. Dagbench:A performance evaluation framework for dag distributed ledgers[C]// 2019 IEEE 12th International Conference on Cloud Computing (CLOUD). IEEE,2019:264-271.

[78] LU X F,JIANG C,WANG P. A Survey on Consensus Algorithms of Blockchain Based on DAG[C]// Proceedings of the 2024 6th Blockchain and Internet of Things Conference (BIOTC'24). 2024:50.

[79] ZHANG,Y Q,LEI G,KE W. A high-throughput DAG-based blockchain protocol[C]// Third International Conference on Signal Image Processing and Communication. 2023:533-540.

[80] MALKHI D,SZALACHOWSKI P. Maximal extractable value protection on a dag[J]. arXiv:2208.00940,2022.

[81] SPIEGELMAN A,ARUN B,GELASHVILI R,et al. Shoal:Improving dag-bft latency and robustness [EB/OL]. (2023-06-05) [2024-07-24]. https://arxiv.org/abs/2306.03058.

[82] MILLER A,XIA Y,CROMAN K,et al. The honey badger of BFT protocols[C]// Proceedings of the 2016ACM SIGSAC Conference on Computer and Communications Security. 2016:31-42.

[83] ARUN B,LI Z,SURI-PAYER F,et al. Shoal++:High Throughput DAG BFT Can Be Fast! [J]. arXiv:2405.20488,2024.

[84] SPIEGELMAN A,GIRIDHARAN N,SONNINO A,et al. Bullshark:the partially synchronous version[J]. arXiv:2209.05633,2022.

[85] BUCHMAN E. Tendermint:Byzantine fault tolerance in the age of blockchains[D]. Guelph:University of Guelph,2016.

[86] LIU F Y,ZHANG J F. Multiple-index Comprehensive Regression Scoring[J]. Journal of Applied Statistics and Management,2014,33(3):408-415.

[87] YU X F,FU D. Multiple-index comprehensive evaluation methods[J]. Statistics and Decision,2004(11):119-121.

[88] TAN P L,WANG R S,ZENG W H. Overview of Blockchain Consensus Algorithms[J]. Computer Science,2023,50(S1):691-702.

[89] LIU Q Y,WU X N,Review on the Weighting Methods of Indexes in the Multi-Factor Evaluation[J]. Knowledge Management Forum,2017,2(6):500-510.

[90] CHEN F,YANG J H,YANG Y,et al. A Study on Network Service Behavior Verification with Process Algebra and Its Application[J]. Chinese Journal of Computers,2011,34(9):1660-1668.

[91] MEREDITH L G,RADESTOCK M. A reflective higher-order calculus[J]. Electronic Notes in Theoretical Computer Science,2005,141(5):49-67.

[92] MILNER R. Communicating and mobile systems:the pi calculus [M]. Cambridge:Cambridge University Press,1999.

ZHOU Kai, born in 1994, Ph. D. His main research interests include blockchain technology and cyberspace mapping.

CHEN Fu, born in 1973, Ph. D, professor, Ph. D supervisor, is a member of CCF(No. 11386S). His main research interests include blockchain technology and information security.

(责任编辑:柯颖)