一种基于双链的区块链共识机制

旋逸昭[1,2],赵红武[1,2],金瑜[1,2]

(1. 武汉科技大学计算机科学与技术学院,湖北 武汉 430065;
2. 湖北省智能信息处理与实时工业重点实验室,湖北 武汉 430065)

摘 要:共识机制是区块链系统的核心技术,目前针对"非币"区块链系统提出的基于贡献值证明与工作量证明(PoC+PoW)的共识机制存在共识效率较低、可靠性和安全性不高、算力消耗大的缺点。提出一种新的基于双链的共识机制 CON_DC_PBFT。在该共识机制中设计一种业务链-系统链双链结构,将贡献值等系统数据和主要业务数据分离到双链中各自完成共识处理,双链的共识表现为半独立的形式,业务链共识消息流受系统链监督协调,并且系统链根据贡献值随机指定业务链的记账节点,双链的分工与协同实现并行化和流水化,改善共识的效率。由于贡献值数据不能被轻易获取,通过拜占庭通信机制和节点随机选择算法,降低节点遭受攻击和系统停滞的风险。通过实验综合分析出块选择概率、单点故障率、节点数、区块传输速率、CPU 使用率对共识机制的性能影响,结果表明,与 PoC+PoW 机制相比,CON_DC_PBFT 共识机制节省了 50%以上内存、存储资源占用,在综合共识时延上有 30%以上的改善。

关键词:区块链;共识机制;双链结构;拜占庭容错协议;非币场景下的区块链;贡献值
中图分类号:TP311 **文献标志码**:A **DOI**:10.19678/j.issn.1000-3428.0067861

A Dual-Chain-Based Consensus Mechanism for Blockchain

XUAN Yizhao[1,2], ZHAO Hongwu[1,2], JIN Yu[1,2]

(1. College of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, Hubei, China;
2. Hubei Key Laboratory of Intelligent Information Processing and Real Time Industry, Wuhan 430065, Hubei, China)

【Abstract】 Consensus mechanism is the core technology of blockchain systems. Currently, a new consensus mechanism based on Proof of Contribution value and Proof of Work (PoC+PoW) has been proposed for "non-coin" blockchain systems. However, it presents some problems such as low efficiency, low reliability and security, and high computing-power consumption. Hence, this study proposes a new Consensus mechanism based on a Dual-Chain-PBFT(Con_DC_PBFT). In this mechanism, a dual-chain structure is designed to separate the contribution value and main business information for independent consensus processing. The consensus of the dual chain is of a semi-independent form, where the main-chain consensus message flow is supervised and coordinated by the subchain, and the subchain randomly designates the main-chain accounting nodes based on the contribution value. The division of labor and cooperation between the dual chains enables parallelization and pipeline processing, thus improving the efficiency of consensus. Moreover, the contribution value cannot be easily obtained, and the risks of node attacks and system stagnation are further reduced by adopting a Byzantine communication mechanism and a random-node-selection algorithm. A comprehensive experimental analysis pertaining to the effects of block-selection probability, single-point failure rate, number of nodes, block transmission rate, and CPU usage on the performance of the consensus mechanism shows that the Con_DC_PBFT consensus mechanism conserves more than 50% of memory and storage resources compared with PoC+PoW, as well as improves the overall consensus time delay by more than 30%.

【Key words】 blockchain; consensus mechanism; dual-chain structure; Byzantine fault tolerance protocol; blockchain on non-coin scenario; contribution value

0 引言

随着互联网技术的发展,区块链技术作为一种去中心化、安全性高、可追溯性强的分布式账本技术[1-4],可以为多个领域赋能,如金融、医疗、农业、信息安全[5]、物联网等,逐渐引起人们的关注和重视。近年来,以区块链技术为核心基础设施的元宇宙产业热度极高,或将形成庞大的新消费生态,成为数字经济的新热点和新方向[6-7]。其中,共识机制是区块链技术的基础和核心[8-9],保证所有参与者在没有一

收稿日期:2023-06-16 **修回日期**:2023-08-30
通信作者 E-mail:852053151@qq.com

个中心机构的帮助下,对一个特定的交易或者业务记录以及多个记录的顺序达成一致。共识机制的优劣直接影响区块链系统的性能[10-11]。

近年来,各种共识机制被提出并应用于不同的区块链系统中[12],其中被广泛研究和使用的有基于区块链的共识机制(包括工作量证明(PoW)[13]、权益证明(PoS)[14]、股份授权证明(DPoS))、基于经典分布式系统的实用拜占庭容错(PBFT)机制[15]、HotStuff[16]等,以上经典共识机制各有优缺点,这里不再赘述。近期出现了新的针对"非币"区块链系统,即基于贡献值证明和工作量证明(PoC+PoW)的共识机制[17],但 PoC+PoW 有以下缺点:此共识机制将贡献值作为 Leader 节点的选举依据,每个节点需要读取其他所有节点的贡献值数据到内存中做计算处理,还需要保存到区块链账本中做持久化,该系统设计层面的处理会占用节点资源,降低节点业务处理的效率;在安全性方面,由于贡献值数据能被轻易地获取,恶意者可以针对贡献值最高的几个节点进行攻击,系统的可靠性差;由于引入了 PoW 机制全部节点参与挖矿,会造成很大的资源浪费。

基于以上问题,本文提出一种 CON_DC_PBFT(Contribution-based-Dual-Chained-PBFT)共识机制。该共识机制基于贡献值的设定使用双链结构,其中业务链节点维护业务链的业务数据,系统链节点维护系统链的系统数据(贡献值等),使得对贡献值数据的存储、计算等处理任务不会拖累业务数据,实现并行化和负载均衡,并且在双链通信流程中实现流水化,改善共识的效率。另外,贡献值数据不能被轻易获取,通过拜占庭通信机制以及节点随机选择算法,降低节点遭受攻击和系统停滞的风险。

1 相关工作

在比特币出现后,基于区块链设计的共识算法被用于验证交易和生成新区块,保证网络安全和去中心化。最初,比特币采用的是 PoW 共识算法,它需要矿工通过高强度计算解决谜题,以获得记账权。然而,这种算法消耗大量能源,因此出现了其他的替代算法。

PoS 是一种可替代 PoW 的共识算法,它是基于持有代币的数量来选择下一个记账节点,而不是计算能力,不会浪费大量能源,但也存在公平性问题。随着时间的推移,许多其他的共识算法被提出,如权益证明/工作量证明(PoS/PoW)混合和股份授权证明(DPoS),以满足特定的需求和场景。

然而,每种共识算法都有自己的优缺点。PoW 在安全性方面表现出色,但会导致高能源消耗和网络拥塞。PoS 消耗较少的能源,但在长期持有代币的情况下可能导致中心化。PoS/PoW 组合的算法可以结合两者的优点,但也存在挑战。DPoS 通过代表节点的投票来确定下一个记账人,但需要信任这些节点。此外,还有其他一些共识算法,它们采用不同的机制,适用于不同的场景和需求。

区块链节点的信任问题与拜占庭将军问题要解决的是同一类问题[18],拜占庭将军问题源于分布式系统领域,涉及到多个将军发起进攻的情况下如何保证所有将军都能够达成一致的决策。拜占庭容错算法是区块链技术中的重要组成部分[19],它解决了节点之间可能存在的拜占庭错误问题。拜占庭错误指的是网络中的某些节点可能会出现不良行为,如恶意篡改数据或伪造交易等行为,这些错误可能会导致网络的故障或受到攻击。因此,拜占庭容错算法被广泛应用于区块链网络中,确保节点之间的数据一致性和可靠性。PBFT 算法是一种基于复制状态机的共识算法,具有高效、可扩展和安全的特点,被广泛用于企业级区块链平台,如 Hyperledger Fabric[20-21]等。近年来,随着区块链技术的发展和应用场景的增加,更多的拜占庭容错算法如 Algorand[22]、HotStuff 被提出,也有许多学者针对某种算法在具体应用假设下提出算法优化方案[23-25]。总地来说,各种共识机制和优化共识算法都根据其适用场景的不同有着不同的局限性。因此,按照具体场景来设计独特的共识机制才是最普适的规则。

针对"非币"区块链系统设计的 PoC 是将节点贡献值作为指标选举出块节点的共识机制,是面向非货币的共识系统所设计的共识机制,将参与系统的用户对系统做出的贡献进行量化,规定贡献值最大的节点获得新区块的出块权,体现了"按劳分配"的价值理念,激发节点参与维护区块链的积极性。在基于 PoC 的区块链应用中,所有节点共同维护一条相同的区块链,链上记录所有交易数据和各节点的行为数据,这些数据将作为贡献值的计算依据,在不同的应用场景中,也可以根据需要来丰富计算的依据。计算贡献值的算法全网统一,每个节点通过读取区块链数据独立计算自身和其他所有节点的贡献值。

PoC 共识机制的规则是贡献值最高的节点获得记账权成为出块节点,因此当出块节点因为许多原因无法正常出块时,系统将一直等待,造成性能严重下降,因此研究人员提出 PoW+PoC 机制来解决这一缺陷。在 PoC 机制中引入 PoW 形成双重选举模

式，将出块节点的确定性弱化，在贡献值最高的节点出现故障时给予其他节点也能成功出块的可能性，以此维持共识流程的持续推进。但 PoC＋PoW 的共识机制还存在以下问题：1）在基于贡献值的共识机制中，贡献值数据属于业务数据外的额外数据，如果每个节点都要计算其他所有节点的贡献值并存入区块中，效率较低；2）由于贡献值的计算方式是统一的，而且作为计算依据的数据可以直接获取，非常有利于恶意节点在每轮共识周期开始时就计算得知最可能出块的几个节点的身份并发动攻击，攻击目标十分明确，安全性有待提高，在遭受攻击后严重影响共识效率；3）由于引入了 PoW 挖矿机制，所有节点都要进行挖矿，但实际上前 3 个节点成功挖矿的概率超过 98%，极大地浪费了计算资源。

2 CON_DC_PBFT 共识机制

2.1 CON_DC_PBFT 框架

CON_DC_PBFT 共识机制的主要设计思想如下：将共识系统节点分成两组，贡献值前 50%的节点构成业务链节点群，后 50%的节点构成系统链节点群，每轮共识完成后业务链记账节点贡献值归 0，加入系统链节点群，同时系统链贡献值最高的节点上升成为业务链节点；在这种双链结构中，业务链节点完成对业务信息的共识，而系统链节点完成对贡献值信息的共识，双链的共识流程是半独立的，业务链共识消息流受系统链监督协调，并且系统链根据贡献值随机指定业务链的记账节点。业务链中拥有最高贡献值的系统节点大概率成为记账节点，遵循“贡献高的节点可靠性越高”的思想，贡献值稍低的节点按照某种规律也获得一定的记账概率。CON_DC_PBFT 框架如图 1 所示。

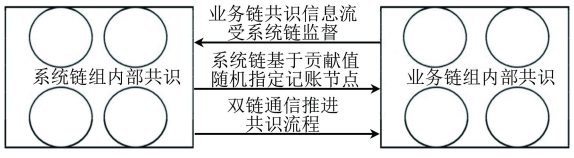

图 1 CON_DC_PBFT 共识机制框架

Fig. 1 Framework of CON_DC_PBFT consensus mechanism

2.2 双链设计

基于 PoC 的共识机制的特点就是需要计算节点的贡献值，因此需要把贡献值相关的计算依据以及贡献值本身的计算结果都附加到区块中的上链，节点通过区块链上经过共识统一的数据来计算贡献值，这样才能保证可靠性和安全性，但无疑会导致共识效率降低，因为贡献值的组成除了特定应用场景的业务数据外，通常都是在系统中业务外的参数，如节点在线时长、节点网络状态等，并且每轮计算都要计算全网其他节点的贡献值，然后进行排序，这将不利于区块链系统的扩展。基于这个需求，本文提出一个双链结构，业务链只需要维护原本的业务数据，而贡献值相关数据则交由系统链来维护，双链将互相协同推进共识进程。

双链区块结构体的定义如图 2 所示。

系统链区块
区块高度
父区块哈希
区块哈希
出块节点
时间戳
贡献值相关数据
//others……

(a)系统链区块

业务链区块
区块高度
父区块哈希
哈希
出块节点
时间戳
业务数据
//others……

(b)业务链区块

图 2 区块结构体的定义

Fig. 2 Definition of block structure

本文假设具有固定节点数量的系统，将节点分为两部分：一部分维护主业务相关的区块链；另一部分维护贡献值相关的区块链。两个部分的节点数各为 $n=3f+1$，总节点数为 $N=2n$，其中，f 是拜占庭节点数量的上界。系统中的所有节点通过互相传输信息进行通信，假设所有节点都可以通过直接或间接的方式连通，其中业务链节点的共识消息会发送给系统链节点，系统链的共识消息只在系统链节点之间传输。对于业务链和系统链的节点集合构成，业务链的节点都是系统中贡献值超过其他节点 50%的节点，而系统链的节点则相反。

双链之间的协同推进主要包括 3 个部分：

1）业务链与系统链节点之间的轮换。基于贡献值的挑选出块节点的方式需要保证贡献值最高的节点不能一直“独裁”，造成不公平的现象，故本文提出一种主系统链之间节点轮换的方式。当业务链节点完成一个区块的记账之后，贡献值将归 0，该节点将加入到系统链节点集合中。同时，系统链节点集合中贡献值最高的节点将加入到业务链节点集合中，这样双链的节点比例将保持动态平衡。

2）主系统链区块的同步协调。如图 3 所示，将业务链称作 A 链，A_x 表示为 A 链上的第 x 个区块，将系统链称作 B 链，B_x 表示为 B 链上的第 x 个区块。当 A_n 块被确认时，系统链节点根据 B_n 块中的贡献值计算结果选出业务链中下一轮的记账节点。其中一个设计的重点是两条链并不是齐头并进，而是交替前进。具体来说，在 A_n 块的共识周期中，B_n 块也在共识流程中，但是依据的是 A_{n-1} 区块的数据，因此 B_n 块的共识完成时间则比 A_n 要早。这种设计使得系统链在收到业务链区块的确认消息

后即可马上回复指定下一轮的记账节点，而不需要等待，有效地实现了双链的共识协同推进。

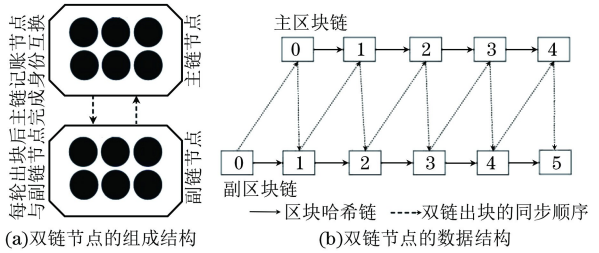

图 3 双链结构示意图

Fig. 3 Schematic drawing of dual-chain structures

3)系统链节点监督业务链共识。在系统中，系统链节点一直监听业务链节点的共识消息，主要有 2 个作用：

(1)根据业务链中的消息通信状况，系统链节点可以判断出业务链节点的大致情况，例如如果某个节点的消息总是比较慢，甚至没收到该节点的消息，则可判断该节点出现了网络故障等情况；当收到部分节点对于某条消息的错误反馈时，可判断该消息是恶意或者错误消息，可对该消息的发送方判定为有不良行为；当收到某个节点的所有符合共识流程的消息时，可判定该节点为诚信节点。具体的设计可根据实际应用场景而有所不同，根据不同的判断对节点贡献值进行增加或减少，该设计完全贴近 PoC 的设计理念。

(2)监督业务链的共识流程，即当贡献值最高节点无法正常出块时，可及时授权另一个节点完成出块。

2.3 出块节点选择

本文提出的共识机制在防止贡献值最高节点出现异常情况下线而无法出块的情况方面，除了前文提到的通过系统链监督与协调的方式，还设计了一种随机化算法，该算法的目的是结合贡献值，使得贡献值最高的节点不再必然成为记账节点，给予其他节点一定记账几率，可以降低单一记账节点被恶意针对攻击的风险，有利于提高系统的可靠性，从而提高整体的效率。

对于随机化的实现，本文提出了一种贡献初始值＋随机值的方法，具体公式为：$R_{\text{Result}}=I_{\text{Init}}+r$，其中，$R_{\text{Result}}$ 是最终贡献值，I_{Init} 是贡献初始值，r 是随机数。

对于贡献初始值，由于应用场景各异，贡献值的具体数值很难统一计算和管理，因此本文提出一种基于分组的方法，设计思想是无论贡献值具体是多少，在任何场景都能统一计算，因此先按贡献值大小排序，再根据排序分组，最后根据分组赋予贡献初始值，这样就能实现贡献值的统一计算。

基于二叉树的分组如图 4 所示。假设业务链节点数为 n，采用二叉树的思想进行分组，设二叉树的高度为 h。必然有 $2^{h-1}\leqslant n<2^h$，将 h 设为分组组数，二叉树的每一层为一组，得到最终的分组为：第 1 组由最高贡献值的节点组成；第 2 组由 2 个贡献值次低的节点组成，依此类推。

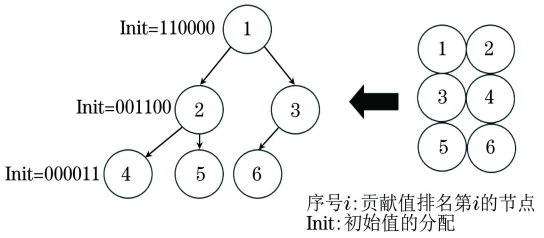

图 4 节点分组示意图

Fig. 4 Schematic drawing of node grouping

基于随机数的贡献值统一处理算法：根据二叉树的高度 h，也就是分组个数，设置初始值是位数为 $2h$ 的二进制正数，第 1 组赋予的初始值前 2 位为 1，后 $2h-2$ 位全为 0；第 2 组赋予的初始值第 3、4 位为 1，其他全为 0；设组别为 i，$0<i\leqslant h$。赋值规律为：第 $2i-1$、$2i$ 位为 1，其他位全为 0。随机数 r 是位数同样为 $2h$ 的二进制正数。根据 $R_{\text{Result}}=I_{\text{Init}}+r$，算出最终的贡献值及排名。

节点的分组和随机选择如算法 1 所示。

算法 1 随机出块算法

输入 业务链节点数 n，贡献值表 con，Com 消息

输出 分组，最终贡献值表

```
1. Group ← lb n+1
2. Sort(con)//按贡献值排序
3. For i=1; i < Group+1; i++
4. Tree[i] ← con[2^(i-1):2^i-1]//按贡献值进行二叉树
//分组
5. Bitlength←2 * Group //计算初始值位数
6. For i=1; i<n+1; i++
7. Init[i]←3 << Bitlength-2i //计算初始值
8. Result = Init + Com.V //根据 Com 消息中的随
//机数整合最终贡献值
```

2.4 共识中关键数据结构定义

为了方便描述本文提出的共识算法，下文给出关键数据结构的定义：

交易(Transaction)：客户端发送给节点群的待处理消息，主要包含客户端标识、时间戳、具体业务消息等。

区块(Block)：主节点将待处理的交易打包形成的数据结构，包含一定数量的交易(Tx)、出块节

点(Proposer)、节点签名(Signature)、时间戳(Timestamp)等,消息格式为 $<$ Block,Tx,P,S,T $>$。

准备消息(Prepare):主节点发送给副节点的消息,包含消息类型(MsgType)、视图编号(view)、区块(Block)、区块高度(height)、授权(Delegation)、区块哈希(Hash)、节点签名(Signature)、发送者(sender),消息格式为 $<$ Prepare,B,v,h,D,H,s,S $>$。

预确认消息(PreCom):节点发送给其他所有节点的消息,包含消息类型(MsgType)、视图编号(view)、区块高度(height)、区块哈希(Hash)、节点签名(Signature)、发送者(sender),消息格式为 $<$ PreCom,v,h,H,s,S $>$。

确认消息(Commit):节点发送给其他所有节点的消息,包含消息类型(MsgType)、视图编号(view)、区块高度(height)、区块哈希(Hash)、节点签名(Signature)、可验证随机数(Verifiable Random)、发送者(sender),其中随机数是基于该消息的其他参数生成,消息格式为 $<$ Commit,v,h,H,s,S,V $>$。

授权消息(Delegation):系统链节点向业务链节点发送的授权消息,包含消息类型(MsgType)、视图编号(view)、聚合签名(Signature)、指定记账者(Proposer)、时间戳(Timestamp),消息格式为 $<$ Delegation,v,S,P,T $>$。

举报消息(Feedback):节点在收到不合法的消息时向系统链节点发送的反馈消息,包含不合法消息本身(ErrorMsg)、签名(Signature),消息格式为 $<$ Feedback,E,S $>$。

2.5 CON_DC_PBFT 具体共识流程

CON_DC_PBFT 共识流程如图 5 所示,将分成 4 个阶段描述,在业务链中,节点在准备阶段会等待接收主节点发送的准备消息后发送预确认消息,在预确认阶段会收集足够的预确认消息后发送确认消息,在确认阶段会收集足够的确认消息后进入应答过程,应答过程中可以回复客户端本轮共识已完成的消息,同时等待系统链节点授权下一轮的主节点。

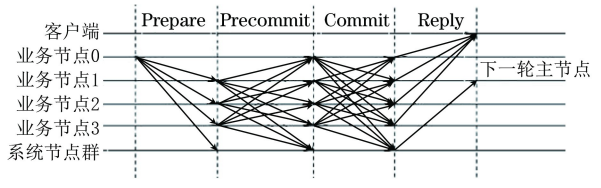

图 5 CON_DC_PBFT 的共识流程

Fig.5 Consensus procedure of CON_DC_PBFT

在系统链中除完成相似的共识过程外,还将监督业务链节点的共识过程。

2.5.1 准备阶段

业务链节点在上一个应答阶段会等待系统链节点发送授权消息,当收到授权消息 D 后,如果 D 中的 Proposer 字段就是自己,业务链节点就知道自己成为本视图的记账节点,于是进入准备阶段。首先从缓冲池中挑选交易 Tx,打包成区块 $<$ Block,Tx,P,S,T $>$,然后加工成准备消息 $<$ Prepare,B,v,h,D,H,s,S $>$,广播给其他节点,然后进入预确认阶段。

非记账节点收到准备消息后,验证发送节点 ID、授权 D、区块 B 以及其中交易是否合法,视图 v 和高度 h 是否与本地匹配,如果验证不通过,则生成举报消息 $<$ Feedback,E,S $>$,发送给系统链节点。如果验证通过,则进入预确认阶段,并广播预确认消息 $<$ PreCom,v,h,H,s,S $>$。准备阶段的节点行为算法如算法 2 所示。

算法 2 准备阶段节点行为算法

身份 主节点

输入 Tx, Delegation

输出 Block, Prepare

1. If verify(Delegation)==TRUE then //验证授权

2. Block←GenerateBlock(Tx,P,S,T)//生成区块结构体

3. Prepare←PrepareMsg(B, v, h, Delegation, H, s, S) //生成准备消息结构体

4. Broadcast(Prepare)

身份 从节点

输入 Prepare

输出 PreCommit

5. If verify(Prepare)==TRUE then

6. PreCom←PreComMsg(v, h, H, s, S)//生成预确认//消息

7. Else Broadcast(FeedbackMsg(Prepare))//广播举报//消息

2.5.2 预确认阶段

业务链节点在预确认阶段会等待接收预确认消息 $<$ PreCom,v,h,H,s,S $>$,当收到预确认消息后,验证视图 v、高度 h 是否与本地匹配,区块哈希 H 是否对应本轮区块,验证签名是否合法。若验证通过,则将 PreCom 消息计数器加 1。当收到 $2f+1$ 条不同的消息后,进入确认阶段,并且生成确认消息 $<$ Commit,v,h,H,s,S $>$,然后根据确认消息生成可验证随机数,加入到确认消息中生成 $<$ Commit,v,h,H,s,S,V $>$。若收到一条不合法的预确认消息,则生成举报消息发送给系统链节点。预准备阶段的节点行为算法如算法 3 所示。

算法 3 预确认阶段节点行为算法

输入 PreCom

输出 Commit

1. If verify(PreCom)==TRUE then //验证预确认消息有效
2. Count[PreCom]++
3. Else Broadcast(FeedbackMsg(PreCom))
4. If Count[PreCom]>2f then
5. Commit←CommitMsg(v,h,H,s,S) //生成确认消息
6. V←VerifiableRandom(Commit,Sig) //生成可验证随机数
7. Commit←V
8. Broadcast(Commit)

2.5.3 确认阶段

业务链节点在确认阶段会等待接收确认消息$<\text{Commit},v,h,H,s,S,V>$，当收到确认消息后，验证视图$v$、高度$h$是否与本地匹配，区块哈希$H$是否对应本轮区块，验证签名是否合法。如果收到$2f+1$条不同的合法确认消息，则说明该区块已经确认，可以进入应答阶段。如果收到一条不合法的确认消息，则生成举报消息发送给系统链节点。本轮的出块节点此时将进行身份转换，加入到系统链节点群中。确认阶段的节点行为算法如算法 4 所示。

算法 4 确认阶段节点行为算法

输入 Commit

输出 Reply

1. If verify(Commit)==TRUE then //验证确认消息有效
2. Count[Commit]++
3. Else Broadcast(FeedbackMsg(Commit)) //广播举报消息
4. If Count[Commit]>2f then
5. Reply… //应答客户端
6. Wait for delegation //等待下一轮授权

2.5.4 应答阶段

业务链节点在应答阶段中可以回复客户端表示本轮共识已经完成，相关业务数据已经保存在区块链中不再会被修改。该轮的主节点在此阶段进行身份的转换，加入到系统链节点群中。同时，业务链节点将等待系统链节点发送授权消息，收到授权消息$<\text{Delegation},v,S,P,T>$后将验证签名是否合法，如果合法，则将该轮主节点设置成授权消息中指定的主节点。

2.5.5 系统链的协同调控

系统链节点将持续收集每个业务链节点的各阶段消息以及举报消息进行相应的协调处理。伪代码如算法 5 所示。

算法 5 系统链监督授权算法

输入 业务链 Commit

输出 Delegation

1. If verify(Commit)==TRUE then //验证确认消息有效
2. Count[Commit]++
3. Else Punish…//执行某种处罚策略
4. If Count[Commit]>2f then
5. Calculate… //开始计算贡献值
6. Result = Init + Com.V //等待下一轮授权
7. Sort(Result
8. Delegation=DelegationMsg(v,S,lastResult[1],T)
9. Broadcast(Delegation)
10. Wait for Next Prepare…//等待下一轮的准备消息
11. If not then Broadcast(next Delegation)//超时则顺延授权新节点

当系统链节点收到$2f+1$条不同的确认消息后，说明业务链中该轮区块已经确定，记录下前$2f+1$条确认消息的发送节点 ID，并开始结算该轮区块共识中的各节点贡献，同时将上一轮计算好的根据贡献值算出的记账节点人选的结果广播给业务链节点，然后联合其他系统链节点生成时间戳更新的授权消息，Proposer 为顺延的下一个记账者，以防意外。与此同时，系统链节点群中贡献最高的节点将进行身份转换，加入到业务链节点群中。

在发送授权的一定时间内，系统链节点监听业务链节点的准备消息以及举报消息，如果在一定时间内没有收到任何准备消息，或者收到$f+1$条对于准备消息的举报消息，则记账节点的选择顺延到下一个节点，给所有业务链节点发送新的授权消息 Delegation。

系统链在监督业务链共识的过程中，如果发现系统出现了意外情况，将广播新的授权 Delegation，业务链节点在任何阶段中，如果收到了授权消息$<\text{Delegation},v,S,P,T>$，则先验证视图$v$是否相同，然后验证时间戳$T$是否大于旧授权的$T'$，如果是则更换记账节点并回归准备状态。

双链的通信可以和基于贡献值的场景巧妙适配，可以考虑的一个做法是记录下前$2f+1$条消息的发送节点 ID，因为收到的消息越早，可能代表对应的节点处理消息的行为更积极，可以将其视为“共识行为积极的节点”进行贡献值的奖励。而举报消息可以反映某个节点有恶意行为或者错误处理，可以将它们视为“犯错节点”进行贡献值的处罚。具体的方案可以根据应用场景的不同灵活实现。

3 分析

3.1 安全性分析

在本文提出的共识机制中，对于系统的安全性主要有 2 个设计：

1)基于系统链节点对业务链共识进程的监督。一方面，业务链的共识参与情况将转换为贡献值的奖惩情况，由贡献值的管理进行对系统的软保护，甚至可以设计对恶意节点进行硬处理；另一方面，如果业务链共识出现异常，则系统链节点即可马上发送新的授权消息指定新的记账节点，将共识流程回归到正常的准备阶段。

2)引入节点随机化与双链设计。许多基于领导者的区块链共识机制都面临一个问题，即领导者的 IP 如果被敌人知晓，很容易遭受恶意攻击，导致需要更换领导者才能继续进行共识。本文通过领导者随机选择的方法，采用双链共识的设计，一是具体的领导者身份在一定程度上被隐藏，二是贡献值的情况只有系统链节点才能获取，提高了系统的安全性。

本文共识机制针对各种常见的攻击都能进行良好的防范：

1)双花攻击。在比特币系统中，攻击者往往将一笔钱在不同的分叉中转账给不同的接收方。在某个分叉的区块被确认之前获取支付的利益反馈，然后该分叉因为没有成为主分叉被丢弃，从而回溯收回支付的金额。因此区块链建议 6 个区块的等待确认时间。此种攻击方式是因分叉问题产生的，而在本文共识机制中，采用基于 leader 选举的 BFT 共识实现了确定性共识，因此不会产生分叉问题，也避免了双花攻击。

2)Sybil 攻击。恶意方通过在区块链中注册和控制大量节点，从而试图获得区块链中的大多数话语权。在基于 PoX 的共识算法实现的共识系统中，都要求节点付出一定量的资源才能成为 leader 或者获得区块话语权，有效避免了 Sybil 攻击。在本文共识机制中，节点通过对系统做出一定贡献，获取贡献值才能成为 leader，即使恶意方拥有大量的虚假节点，也无法获得足够的贡献值对区块链造成侵害。

3)合谋攻击。恶意节点之间通过某种协作共同获得区块链的话语权，从而获得非法利益。在 PoW 系统中，合谋攻击可以具体为 51%的算力攻击，在 PoW 系统中，诚实节点的算力占比超过 50%是基本底线和假设。在本文的共识机制中，在业务链节点中各节点是竞争关系，只有获得最高贡献值的节点才能控制区块数据，没有合谋的方式；而在系统链节点中存在合谋的方式，系统链节点之间可能通过合谋控制业务链节点的选举，但是在本文系统中，系统链节点是由业务链节点在一轮共识中获得最高贡献值并出块后进行身份转换产生的。在这种设计下，可以合理认为大部分系统链节点是诚信的。因此，只要系统链节点中恶意节点的个数少于 $1/3n$，就无法对系统造成安全性的侵害。

3.2 实验分析

本文实验的实验环境为 AMD R7-5800H CPU 3.20 GHz 和 32 GB 内存，采用的操作系统为 Windows 10，测试方法为使用 Golang 多线程编程实现节点模拟。

3.2.1 出块节点概率分析

实验 1 实验将节点组别作为变量，对比本文共识机制中基于二叉树分组和随机数的随机出块算法，以及 PoC+PoW 机制中基于斐波那契分组和难度值的随机出块算法在贡献值排名靠前的几组节点的出块概率。实验完成对 10 000 个区块的共识，记录各组节点成为出块节点的频次。实验结果如图 6 所示。

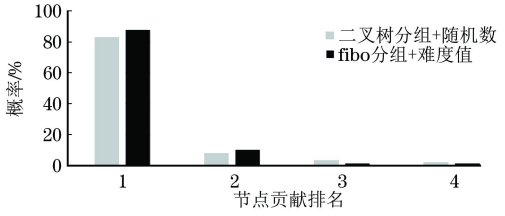

图 6 节点随机选择算法概率比较

Fig.6 Probabilistic comparison of node random selection algorithms

在本文提出的共识机制的随机出块算法下，将贡献值前两名的组记账概率降低，将后几组的记账概率提高。对比 PoC+PoW 共识机制中节点前三名的记账概率超过 98%，这样的概率分布较为极端，本文的算法概率分布更加平缓，在保持了“贡献值=记账权”的理念下，降低了记账节点遭受恶意攻击的风险，进一步提高了系统的可靠性。

3.2.2 共识性能分析

性能评估：假设每条贡献值数据处理内存占用为 m，磁盘占用为 d，则贡献值计算排序时间为 $O(n\log_a n)$，节点数为 n。

PoW+PoC 共识机制：总内存占用为 $M1=m\times n\times n=mn^2$，总磁盘占用为 $D1=d\times n\times n=dn^2$，节点通信量为 $O(n)$，PoW 模块耗时为 P，故障转移开销为 $P+O(n)$，出块节点发生单点故障的概率为 p，综合共识时延为 $O(n\log_a n)+O(n)+p\times$

$(O(n)+P)$。

CON_DC_PBFT 共识机制：总内存占用为 $M2=m\times n/2\times n/2=mn^2/4$，总磁盘占用为 $D2=d\times n/2\times n/2=dn^2/4$，节点通信量为 $O((c\times n/2)^2)$，其中 c 为贡献值数据外置导致的区块传输通信量减少系数，故障转移开销为 $O(1)$，出块节点发生单点故障的概率为 p，综合共识时延为 $O((c\times n/2)^2)+p\times O(1)$。

结合实验证明 CON_DC_PBFT 在内存、存储资源占用上节省了 50%以上，并且无须浪费 CPU 进行无意义的挖矿，在综合共识时延上对比 PoW+PoC 具有 30%的改善。

实验 2 对比不同节点数下 CON_DC_PBFT、PoC+PoW 和 PBFT 的平均出块时延。

实验设置：以不同共识节点数为变量设置不同实验组，运行不同的共识算法对 100 个区块进行共识处理。实验中引入了不同共识机制的可靠性对性能造成的影响，设定贡献值最高节点遭受攻击导致节点暂时下线的概率为 10%，且由于 CON_DC_PBFT 双链结构的特点，考虑到区块大小导致的传输时延的因素，将业务链的区块大小适当降低 20%。

如图 7 所示，在节点数较少时，基于 BFT 通信的 PBFT 和 CON_DC_PBFT 算法都具有较好的性能，但 PoC+PoW 由于需要计算 Nounce 值的过程，共识时延具有较高的下限。在节点数增多后，由于 BFT 通信的通信量大幅增加，因此 PBFT 的共识时延开始剧增，而 CON_DC_PBFT 由于使用了双链结构，仍具有较好的性能。

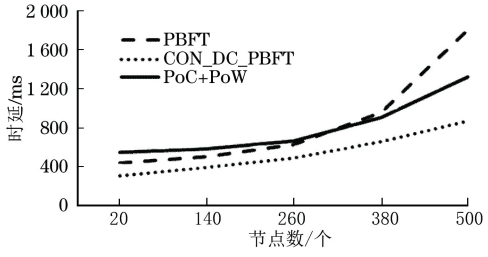

图 7 不同节点数下的共识时延比较

Fig.7 Comparison of consensus latency under different node numbers

实验 3 CON_DC_PBFT、PoC+PoW 和 PBFT 的可靠性对共识性能的影响。

实验设置：以贡献值最高节点遭受攻击的概率作为变量设置对照组，设置节点数 N 为 500，运行不同的共识算法对 100 个区块进行共识处理，将贡献值最高节点的下线概率分别设置为 10%、20%、30%、40%，对比不同条件下的共识时延。

如图 8 所示，3 种算法在应对出块节点故障时采取不同的策略，PBFT 是进入视图转换机制，由于多轮通信，一般会有秒级的额外延迟，PoC+PoW 则是通过节点随机的方法，等待贡献值排名第二、第三的节点成功挖矿更换出块节点，但是由于其他节点的挖矿成功概率相对于主节点只有 10%左右，因此系统恢复的额外延迟也是秒级，而 CON_DC_PBFT 则是通过节点随机与系统链监督相结合的方法，在最坏情况下也能通过几个消息传输延迟的时间恢复共识流程，具有非常低的共识进程恢复延迟，只有平均几百毫秒的额外延迟。

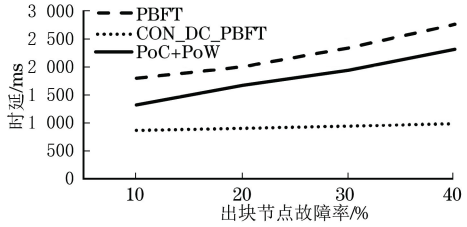

图 8 出块节点故障率与共识时延的关系

Fig.8 The relationship between the failure rate of block nodes and consensus delay

实验 4 对 CON_DC_PBFT 的贡献值数据分链处理来进行共识性能的提升测试。

实验设置：以贡献值数据的相对大小作为变量设置对照组，设置节点数 N 为 500，分别运行 CON_DC_PBFT 和 PoC+PoW 算法，测试产生 100 个区块的共识时延，假设业务数据量不变，在 PoC+PoW 中，贡献值在区块中的比重分别设置为 10%、20%、30%和 40%，对应地，在 CON_DC_PBFT 中，区块容量就可以被压缩至 90%、80%、70%和 60%。

如图 9 所示，由于在固定的上传下行速度下，区块大小和区块在网络中传输的延迟成反比，CON_DC_PBFT 共识机制通过将贡献值数据分链处理，有利于减小区块的传输体积，降低区块在网络中的传输延迟，从而提升共识的性能。

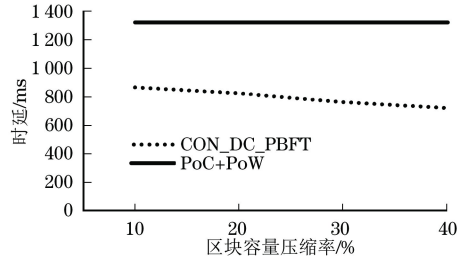

图 9 区块容量压缩率对共识时延的影响

Fig.9 Impact of block capacity compression rate on consensus latency

3.2.3 资源消耗分析

实验 5 对比 CON_DC_PBFT 和 PoC+PoW 在运行时的 CPU 使用率。

实验设置：在运行 100 个节点分别使用 CON_DC_PBFT 和 PoC+PoW 算法时，进行 1 min 的 CPU 资源的监控。

如图 10 所示，PoC+PoW 需要进行随机数的计算来出块，占用了较大的计算资源，而 CON_DC_PBFT 只需要进行对消息的监听和处理，因此无须花费额外的算力。

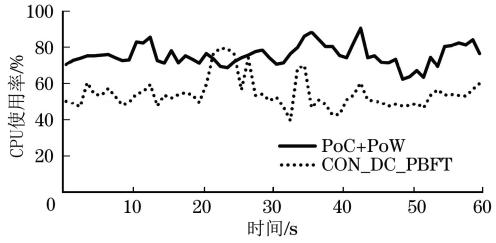

图 10 CPU 资源消耗对比

Fig. 10 Comparison of CPU resource consumption

4 结束语

针对“非币”区块链系统的 PoC+PoW 共识机制中存在的贡献值数据处理效率低、安全性差和资源浪费等问题，提出一种 CON_DC_PBFT 共识机制。该机制基于贡献值设定，使用半独立的双链结构对贡献值与业务数据进行分链处理，并在双链的共识过程中实现并行化和流水化，从而提高共识效率。同时，通过基于双链优化的拜占庭容错协议和节点随机选择算法，保证贡献值数据的安全性，进而增强共识系统的备灾能力。此外，本文提出的共识机制不再需要消耗计算资源进行 PoW 挖矿，可以节省大量资源。实验结果表明，CON_DC_PBFT 较已有的共识机制具有更好的共识性能和可靠性。下一步将对共识的消息流进行优化，以达到降低消息传输成本并提高通信性能的效果。

参考文献

[1] ZHENG Z B, XIE S A, DAI H N, et al. Blockchain challenges and opportunities: a survey[J]. International Journal of Web and Grid Services, 2018, 14(4): 352-375.

[2] YLI-HUUMO J, KO D, CHOI S, et al. Where is current research on blockchain technology: a systematic review[J]. PLoS One, 2016, 11(10): e0163477.

[3] DAI H N, ZHENG Z B, ZHANG Y. Blockchain for Internet of things: a survey[J]. IEEE Internet of Things Journal, 2019, 6(5): 8076-8094.

[4] 张亮，刘百祥，张如意，等. 区块链技术综述[J]. 计算机工程，2019，45(5)：1-12.
ZHANG L, LIU B X, ZHANG R Y, et al. Overview of blockchain technology[J]. Computer Engineering, 2019, 45(5): 1-12. (in Chinese)

[5] 刘敖迪，杜学绘，王娜，等. 区块链技术及其在信息安全领域的研究进展[J]. 软件学报，2018，29(7)：2092-2115.
LIU A D, DU X H, WANG N, et al. Research progress of blockchain technology and its application in information security[J]. Journal of Software, 2018, 29(7): 2092-2115. (in Chinese)

[6] 李鸣，张亮，宋文鹏，等. 区块链：元宇宙的核心基础设施[J]. 计算机工程，2022，48(6)：24-32，41.
LI M, ZHANG L, SONG W P, et al. Blockchain: core metaverse infrastructure[J]. Computer Engineering, 2022, 48(6): 24-32, 41. (in Chinese)

[7] WANG Y T, SU Z, ZHANG N, et al. A survey on metaverse: fundamentals, security, and privacy[J]. IEEE Communications Surveys & Tutorials, 2023, 25(1): 319-352.

[8] BAMAKAN S M H, MOTAVALI A, BABAEI BONDARTI A. A survey of blockchain consensus algorithms performance evaluation criteria[J]. Expert Systems with Applications, 2020, 154: 113385.

[9] CHAUDHRY N, YOUSAF M M. Consensus algorithms in blockchain: comparative analysis, challenges and opportunities[C]// Proceedings of the 12th International Conference on Open Source Systems and Technologies. Washington D. C., USA: IEEE Press, 2018: 54-63.

[10] LASHKARI B, MUSILEK P. A comprehensive review of blockchain consensus mechanisms[J]. IEEE Access, 2021, 9: 43620-43652.

[11] KAUR S, CHATURVEDI S, SHARMA A, et al. A research survey on applications of consensus protocols in blockchain[J]. Security and Communication Networks, 2021, 2021: 6693731.

[12] FU X, WANG H M, SHI P C. A survey of blockchain consensus algorithms: mechanism, design and applications[J]. Science China Information Sciences, 2020, 64(2): 121101.

[13] NAKAMOTO S. Bitcoin: a peer-to-peer electronic cash system[EB/OL]. [2023-05-10]. https://bitcoin.org/bitcoin.pdf.

[14] KING S, NADAL S. PPcoin: peer-to-peer crypto currency with proof-of-stake[EB/OL]. [2023-05-10]. https://bitcoin.peryaudo.org/vendor/peercoin-paper.pdf.

[15] CASTRO M, LISKOV B. Practical Byzantine fault tolerance and proactive recovery[J]. ACM Transactions on Computer Systems, 2022, 20(4): 398-461.

[16] YIN M F, MALKHI D, REITER M K, et al. HotStuff: BFT consensus with linearity and responsiveness[C]// Proceedings of 2019 ACM Symposium on Principles of Distributed Computing. New York, USA: ACM Press, 2019: 347-356.

[17] 何泾沙，张琨，薛瑞昕，等. 基于贡献值和难度值的高可靠性区块链共识机制[J]. 计算机学报，2021，44(1)：162-176.
HE J S, ZHANG K, XUE R X, et al. A highly reliable consensus mechanism for blockchain based on contribution and difficulty values[J]. Chinese Journal of Computers, 2021, 44(1): 162-176. (in Chinese)

[18] GRAMOLI V. From blockchain consensus back to Byzantine consensus[J]. Future Generation Computer Systems, 2020, 107: 760-769.

[19] 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. Development status and prospects of blockchain consensus algorithms[J]. Acta Automatica Sinica, 2018, 44(11): 2011-2022. (in Chinese)

[20] ANDROULAKI E, BARGER A, BORTNIKOV V, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains[C]//Proceedings of the 13th EuroSys Conference. New York, USA: ACM Press, 2018: 1-15.

[21] 邵奇峰,张召,朱燕超,等. 企业级区块链技术综述[J]. 软件学报,2019,30(9):2569-2592.
SHAO Q F, ZHANG Z, ZHU Y C, et al. Survey of enterprise blockchains[J]. Journal of Software, 2019, 30(9): 2569-2592. (in Chinese)
[22] GILAD Y, HEMO R, MICALI S, et al. Algorand: scaling Byzantine agreements for cryptocurrencies[C]//Proceedings of the 26th Symposium on Operating Systems Principles. New York, USA: ACM Press, 2017: 51-68.
[23] 刘泽坤,王峰,贾海蓉. 结合动态信用机制的 PBFT 算法优化方案[J]. 计算机工程,2023,49(2):191-198.
LIU Z K, WANG F, JIA H R. Optimization scheme of PBFT algorithm combining dynamic credit mechanism[J]. Computer Engineering, 2023, 49(2): 191-198. (in Chinese)
[24] 陈润宇,王伦文,朱然刚. 基于信誉值投票与随机数选举的 PBFT 共识算法[J]. 计算机工程,2022,48(6):42-49,56.
CHEN R Y, WANG L W, ZHU R G. PBFT consensus algorithm based on reputation value voting and random number election[J]. Computer Engineering, 2022, 48(6): 42-49, 56. (in Chinese)
[25] HOU R M, YU H F, SAXENA P. Using throughput-centric Byzantine broadcast to tolerate malicious majority in blockchains[C]//Proceedings of IEEE Symposium on Security and Privacy. San Francisco, USA: IEEE Press, 2022: 1263-1280.

编辑 索书志