自适应节点规模的区块链分片可扩展模型

李宝莹,李志淮,王成爱,杨锋
(大连海事大学信息科学技术学院,辽宁 大连 116026)

摘 要: 分片是解决区块链可扩展性问题的核心技术,然而现有分片方案普遍采用预定分片规模的静态分片方式,这与公链开放低门槛的分布式环境不匹配。当网络中的节点数大幅增加时静态分片方式难以及时充分地发挥全部节点的性能,当网络中的节点数大幅减少时又会增加分片内的安全隐患。为此,构建一种自适应节点规模变化的动态分片可扩展模型(DSSM)。在基础分片上建立分层的逻辑分片,通过支持状态归约允许节点在不同层级的分片上进行状态同步。在逻辑与基础分片间建立满二叉树的逻辑关系,通过分片的动态分裂和合并来扩张和收缩分片规模,实现分片规模的自适应调整。实验结果表明,DSSM在节点数量大幅增加时通过自适应扩展分片规模使网络吞吐量得到了近乎翻倍的提升,在节点数量大幅减少时通过自适应收缩分片规模保证了网络的最低安全要求。

关键词: 区块链;自适应节点规模;状态分片;归约节点;状态冗余
中图分类号: TP311 **文献标志码:** A **DOI:** 10.19678/j.issn.1000-3428.0066930

Blockchain Sharding Scalable Model of Adaptive Node Scale

LI Baoying, LI Zhihuai, WANG Chengai, YANG Feng
(School of Information Science and Technology, Dalian Maritime University, Dalian 116026, Liaoning, China)

【Abstract】 Sharding is one of the core technologies to address the scalability of blockchains. However, the existing sharding schemes generally adopt a static sharding method with a predetermined shards' scale, which does not match the open and low-threshold distributed environment of the public blockchain. When the number of nodes in the network increases significantly, it is difficult for the static sharding method to fully utilize the performance of all nodes in a timely manner, and when the number of nodes in the network is significantly reduced, it will increase the security risks in the shards. Accordingly, a Dynamic Sharding Scalable Model (DSSM) with an adaptive change of nodes' scale is constructed. On the basis of basic shards, this model establishes layered logical shards, supports state reduction, and allows nodes to synchronize the state of shards at different levels. A logical relationship of a full binary tree is established between logical and basic shards. To achieve adaptive adjustment, the scale of the shards is expanded and shrunk through dynamic splitting and merging of the shards. The experiments have demonstrated that when the number of nodes increases significantly, the DSSM can nearly double the throughput by adaptively expanding the scale of shards, and when the number of nodes is significantly reduced, the DSSM guarantees the minimum security requirements of the network by adaptively shrinking the scale of shards.

【Key words】 blockchain; adaptive node scale; state sharding; reduction node; state redundancy

0 引言

自从2008年中本聪发布比特币白皮书[1]以来,越来越多的研究者开始关注比特币背后的区块链技术。然而,随着区块链网络中交易数量的增多与去中心化应用的发展,区块链出现了严重的性能瓶颈,已无法满足实际应用的需求。扩容是区块链的核心任务和前沿技术难题。研究者们提出了许多扩容方案,例如,扩块[2]、有向无环图(DAG)[3-5]、分片[6-7]、侧链[8]、状态通道[9]等,其中分片是目前最可能实现区块链扩容的技术之一,状态分片能够从本质上解决公链的可扩展性问题。在状态分片的情况下,由于每个分片只保留了状态的一部分,因此一旦一个新节点加入一个分片中,系统就必须确保该节点有足够的时间进行分片状态的同步,否则新节点将无法参与交易的验证。

分片方案通常被视为安全地适应开放网络中节点数变化的可持续扩展方案,但现有分片方案大多预先设定了分片规模。公链处于一个开放低门槛的分布式环境中,网络中的节点状态和规模随时会发生变化,当网络中的节点数量在短时间内发生较大变化时,现有利用时间片在分片内更新节点的方式将相对滞后,并且不适用于节点数量明显波动的情况。一方面,预先设定分片的规模只是为了暂时的方便,偏离了更多的节点提供更高性能的设计目的,同时也会使网络在短时间内面对更多节点时无法及

收稿日期: 2023-02-13 **修回日期:** 2023-05-15
基金项目: 国家科技支撑计划(2008BAH37B05)。
通信作者 E-mail: lbydlhsh@163.com

时发挥节点性能,从而导致网络无法充分利用节点资源。另一方面,在区块链网络中的节点数在短时间内大幅减少的情况下,预先设定分片规模也会增加分片内的安全隐患。除此之外,预先设定分片规模的静态分片方案未来即使需要扩大分片规模,也只能采用时延较长、成本较高的硬分叉方式,这使得系统更新更困难、代价更大、速度更慢。

在分片后要面临的是安全性降低的问题:分片后单个分片内节点数量远低于分片前整个区块链网络的节点数量,导致攻击者对单个分片发起攻击的成本极大降低。假设系统包含S个分片,在节点被均匀地分配到各个分片的情况下,每个分片的安全性变为原先整条链的1/S。若分片前后区块链都采用工作量证明(POW)共识(假设各个节点的算力相当),则分片前攻击者需要控制超过50%的节点才能发动51%攻击,而分片后攻击者若想攻击单个分片,则只需要控制分片内超过50%/S的节点。

在网络中节点数一定的情况下,扩展分片的数量意味着分片内节点数的减少,也意味着攻击者攻击单个分片的代价更小,分片的安全性被削弱。因此,从安全方面而言,分片内要有足够的节点来保证安全性和去中心化,分片的节点数应该有一个最小值,即下限阈值。当分片的节点数低于下限阈值时,此时的分片不满足基本安全要求,节点数量需要增加,给分片补充节点。但由于在状态分片下,补充新的节点存在同步分片状态的滞后问题,因此应采取缩减分片规模的方式来确保分片的节点数不低于下限阈值。

在满足基本的安全要求后,若分片的节点数过多,则会影响网络的吞吐量。在网络中节点数一定的情况下,增加分片内的节点数而减少分片的数量会使整个网络的并行度降低,进而影响网络的吞吐量。同时,若分片内的节点过多,则会增加节点间的通信开销,节点需要存储和维护的区块信息和状态数据也会随之增多,而通信量的增大也会对网络带宽造成影响。因此,分片的节点数也应该有一个最大值,即上限阈值。当分片节点数高于上限阈值时,此时的分片可以通过向外扩展以提高吞吐能力。

本文建立基于状态归约的自适应节点规模的动态分片可扩展模型(DSSM),旨在使分片规模能够自适应节点环境并且动态可扩展,同时不削弱去中心化程度和安全性。设计基于可验证延迟函数(VDF)+可验证随机函数(VRF)的节点随机数生成算法,可以对抗Sybil攻击以及避免可能存在的恶意节点相互串通、提前计算符合它们要求的随机数的情况。

1 相关工作

1.1 符号与定义

本节简要介绍了使用的相关符号及定义,如表1所示。

表1 符号及定义

Table 1 Symbols and definitions

符号	意义
O	分片节点数的上限阈值
T	分片节点数的下限阈值
N_p	分裂过程中父级分片的节点数量
N_r	复用节点集合中的节点数量
N_u	非复用节点集合中的节点数量

1.2 安全的去中心化随机数生成器

1.2.1 可验证随机函数

VRF是一种将输入映射为可验证的伪随机输出的加密函数[10],已被广泛应用于Algorand[11]、本体的VBFT算法等加密场景,一般由Keygen、Evaluate、Verify算法组成。

对于相同的密钥对以及输入X,VRF只会产生唯一的伪随机字符串Y和证明ρ。因此,VRF满足以下性质:1)伪随机性,对于不知道证明ρ的第三方而言,它看起来输出值是随机的;2)可验证性,输出结果能够被验证;3)唯一性,输出的随机值是唯一的,具有不可伪造或抵赖性,不可能通过给定的密钥对(VK,SK)和消息X产生不同的输出Y和证明ρ。

1.2.2 可验证延迟函数

VDF是一类数学函数[12],即使在同时使用不同CPU并行计算的情况下计算出结果也至少需要一段已知的时间,一般由Setup、Eval和Verify算法组成。

VDF满足以下性质:1)验证高效;2)唯一性,对于任意一个VDF的输入,应有唯一的输出结果能够通过检验;3)串行性,攻击者即使拥有很多算力,能够并行计算,但是其以少于某个固定时间计算出VDF结果的概率小到可忽略不计。

1.3 其他区块链分片方案

1.3.1 以太坊

2022年9月,以太坊进行了合并,实现了从POW到权益证明(POS)的过渡。同时,以太坊官方明确了扩展以太坊性能的方向,即放弃最初的分片方案,启动以Rollup为中心的扩容方案,也就是Danksharding[13]。

Danksharding的主要特点包括:1)Blob数据分片传输;2)数据可用性抽样实现区块验证;3)区块提议者与构建者分离(PBS)。

1.3.2 Monoxide

Monoxide[14]的主要思想是通过异步共识组来实现公链扩容,主要特点包括:1)将区块链网络划分为多个独立和并行的Zone;2)高效处理跨越不同Zone的交易;3)通过“连弩挖矿”的方式解决分组后挖矿算力稀释的问题。

Monoxide在完全去中心化的前提下实现了全方位的分片,保证了可以正确高效地完成其上的跨分片交易,并保证了攻击单个共识组的代价与攻击整个网络的代价相当。

1.3.3 NEAR

NEAR[15]将系统建模成一个单独的区块链，其中每个块逻辑上包含所有分片的全部交易以及改变所有分片的整个状态。然而，在物理上任何参与者都不会下载完整的状态或完整的逻辑区块，网络的每个参与者只是维护为之验证交易的分片上对应的状态，区块中的所有交易的列表被分割成物理上的段，一个分片为一个段。

1.4 多轮验证思想与状态分片基础上的共识组

实用拜占庭容错（PBFT）[16]算法具有“即时敲定”的特性，即被 PBFT 共识验证通过并上链的交易会被永久确认，没有被篡改的可能性，这使得它天然具有避免分叉的特性。然而，PBFT 只有 1/3 的拜占庭容错率，这就使得它的实际应用会受到很多限制。

针对在分片内使用 PBFT 共识所面临的因拜占庭节点比例超过 1/3 而导致共识失败的问题，研究者们提出了多轮验证的思想。多轮验证的主要思路为：若交易验证失败，则不是立刻丢弃该交易，而是更新分片内的节点，对交易进行新一轮验证，以此类推，直到交易验证成功或者达到验证轮数上限而放弃该交易。

然而，普通的多轮验证方案对于交易验证采用的是分片内的全部节点均参与的方式。这会带来一些问题：PBFT 的通信复杂度是 $O(n^2)$级别，若是分片内全部节点均参与 PBFT 共识，则会引起通信量过大的问题。另外，每轮都更新分片内的节点，会造成许多额外开销。

针对以上问题，研究者们提出分片共识组[17]的概念，其主要思想是筛选分片内的部分节点，构建共识组对交易进行验证，这能有效降低通信开销，同时启用积分机制对节点表现进行评价，从而能够高效甄别拜占庭节点，提高了系统安全性。

1.5 动态分片

自适应节点规模的区块链分片可扩展模型是动态分片技术的优化研究。针对传统分片预先设定分片规模的静态分片方式的局限性：文献[18]提出一种在联盟链中的基于随机值的动态分片协议，在联盟链中降低了静态分片的安全风险，但并未给出具体的应用场景；文献[19]提出一种基于深度强化学习的动态分片区块链框架，采用深度强化学习来调整分片策略，以解决节点加入、退出、恶意攻击等动态环境下的问题；文献[20]为了解决跨分片交易问题提出一种方案，首先在划分分片时考虑其数据相关性以减少跨分片交易的概率，然后通过考虑分片的利用率进行合并和分割，并通过选举领导者来减少共识延迟；文献[21]通过动态分片减少各个节点存储的旧区块副本来减少存储负载，但并未规定节点间或者分片间的通信规则，在验证交易时如果需要的数据不在本地，则需要从其他节点处获取，这就存在数据一致性问题。

2 环境假设

2.1 网络模型

对于底层网络，假设与其他方案一致[22-24]，并假设：1）诚实节点的网络是良好连接的；2）诚实节点的通信通道是同步的，即一个诚实节点广播一个消息，那么所有节点都能在一个已知的最大延迟 Δ[25]内接收该消息。

2.2 威胁模型

网络中存在诚实和拜占庭两种节点。假设网络中拜占庭节点的比例 $f \leqslant 1/3$，也就是网络中最多有 1/3 的验证者可能是恶意的。诚实节点遵守所有协议，恶意节点可以进行任意操作，它们可以以任意方式违反协议，如拒绝发送、篡改和伪造消息等。它们之间还可能相互串通来共同作恶。假设拜占庭对手是慢适应的，即拜占庭节点和诚实节点的集合在每个时隙内都是固定的，只有在不同的时隙之间才能变化。

3 DSSM 模型

针对传统分片方案预定分片规模的静态分片方式产生的性能缺陷与安全隐患，本文力图实现一种树型分层的自适应节点规模的分片可扩展模型，以提高分片区块链系统在向外扩展时的鲁棒性与自适应性。

3.1 状态归约思想

传统分片方案中分片内的节点只会维护本分片的状态，而状态归约思想鼓励高性能的节点同步其他分片的状态。当某个节点具有两个及以上的分片状态时，它就成了归约节点。状态同步且具有相同分片（至少两个）状态的节点集合构成了新的分片，称为归约分片。归约分片与被同步状态的分片之间构成了树结构，如图 1 所示。

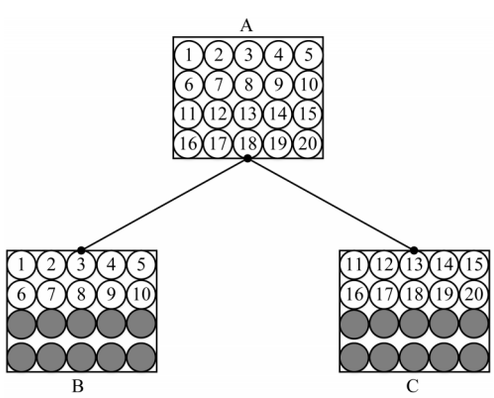

图 1 分片间的逻辑关系

Fig.1 Logical relationship between shards

根据状态归约思想，处于父节点位置的分片（父级分片）中的节点将会存储处于其孩子节点位置的分片（子分片）的所有状态信息。在图 1 中，分片 A 处于分片 B 与分片 C 的父节点的位置上，分片 B 和 C 分别处于两个孩子节点的位置上。分片 B 和 C 中的

白色节点分别同步了其兄弟分片的状态而成了归约节点，这两部分白色节点同时具有分片B和C的状态，它们共同组成了归约分片A。对于分片B和C中的白色节点，它们的逻辑级别属于分片A这一级别。

3.2 DSSM模型构建

在DSSM中分片可以分为两种：一种是基础分片，它们是物理上的实际分片，与其他方案中的分片相似，每个基础分片独立验证分片的事务；另一种是逻辑分片，又称为归约分片，它们是逻辑上的分片，不是实际的分片。在归约分片中的节点，要在分片的动态调整中发挥作用，例如为节点的状态同步提供相应数据、在分片动态调整过程中传递相应的消息等。

DSSM模型使用分片归约树这样的逻辑结构来控制分片的动态扩展。分片归约树是一个二叉树的结构，如图2所示（彩色效果见《计算机工程》官网HTML版），其中每个叶子节点代表的是一个基础分片，而处于非叶子节点位置上的节点是逻辑分片（归约分片），它们维护孩子节点所对应分片的所有状态，并在分片动态调整时保证分片状态的平稳过渡。

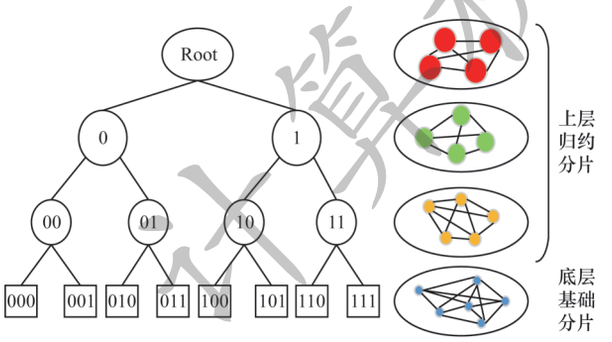

图2 DSSM归约模型

Fig.2 Reduction model of DSSM

DSSM考虑到节点的性能存在差异，因此通过激励机制，鼓励高性能的节点尽可能地同步兄弟分片的状态，成为上层归约分片中的节点，从而使得在分片动态调整时，相应的状态信息不会丢失，分片也不至于因为状态信息的不一致而导致瘫痪。

在整个网络中，性能较好的节点处于DSSM模型较上层位置，性能较差的节点处于DSSM模型较下层位置。DSSM还有一个基本原则就是所有归约分片中的节点即使逻辑级别和状态存储各有不同，但在物理上它们都处于各自的基础分片中，它们仍然需要承担所属基础分片的交易验证任务，这符合性能越好需要承担的任务越多的设计理念。这样的设计能够较为充分地发挥节点的性能，从而提高整个区块链网络的性能。

3.3 DSSM消息传递规则

由于DSSM所有节点实际上都处于基础分片中，节点同步更多的状态只是为了在分片的动态调整中发挥更多的作用。归约分片并不是真分片，是逻辑上的分片，是虚分片。同时，为了动态扩展的需要，每个基础分片中也必然存在这样的节点，它们处于各级的祖先分片中。

在DSSM中的消息传递规则是所有的消息都在本分片内传递，这里的本分片既包括物理上的实际分片，也包括逻辑上的虚分片。

以图3为例，假设节点4在分片C中广播消息，那么节点1、2和3就可以接收到消息，并分别将消息在分片Root、A和B中广播。此时，消息在分片C内部广播的同时，也将近乎并行地在分片Root、A和B中广播。同样地，节点7、10和13也会因为分片Root、A和B中的消息广播而捕获此消息，那么分片E、D和F也会接收到此消息。以此类推，父级分片的节点在广播消息的同时，其子分片的节点也在近乎并行地广播消息。因此，当某一节点在自己的分片中广播消息时，全网的所有分片（包括基础分片和归约分片）也在近乎并行地广播消息，这大大提高了消息的传递效率。

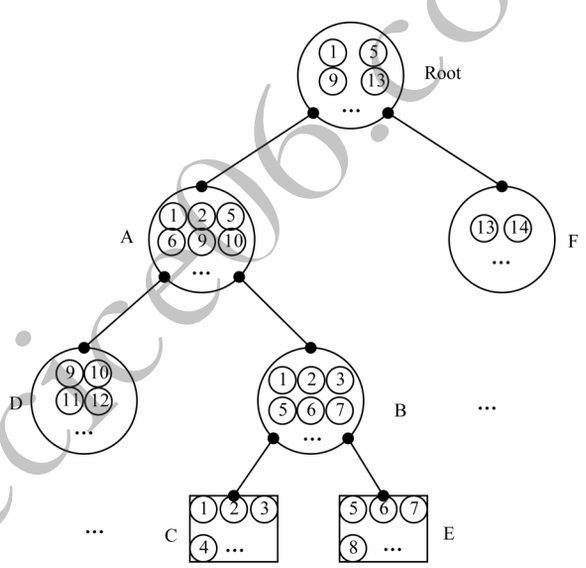

图3 消息传递规则

Fig.3 Messaging rules

4 DSSM分片自适应扩展设计

DSSM实行状态分片，并以存储状态为依据将节点进行逻辑分层。

4.1 归约树逻辑结构初始化

一般地，为了更好地兼容未实施分片的区块链系统，DSSM的分片初始化由单个分片（称为根分片Root）开始。未分片的区块链网络可以视为网络中只有单个分片。

1）当Root（对于未实施分片的区块链，Root包括其所有节点）中的节点数超过上限阈值O时，便可以开始分片的分裂。

2）由于存在分片分裂之后产生的两个子分片的节点数依然超过上限阈值O的情况，因此每次分裂

后形成的子分片的节点继续按照步骤 1)中的标准判断所在子分片是否满足分片分裂条件,若满足,则可以执行相应的分片分裂过程。

3)执行步骤 2)中的过程,直到节点检测到所在子分片的节点数不超过 O 为止。

4.2 分片的全网分裂

4.2.1 分裂条件

1)节点发起分裂请求提案的条件

当基础分片内任一节点发现分片的节点数已经超过上限阈值 O,并且它能够提供其所处基础分片满足分片分裂条件的证明时,就可以发起分裂请求提案。

2)全网分片分裂条件

当全网分片分裂后不会导致某些新分片的节点数低于 T 且超过 1/2 的分片响应分裂时,可以开始全网分片的分裂。

3)单个分片分裂条件

当基础分片中的节点就是否响应分裂进行 PBFT 共识时,每个节点除了需要在消息中注明是否响应,同时需要注明是否愿意留在当前分片内成为归约节点,即是否愿意同步两个基础分片的状态(在分裂完成后,当前分片处于归约树中非叶子节点的位置,它在逻辑上已经属于归约分片,分裂产生的新分片处于新的叶子节点位置上,它们是新的基础分片)。在对响应分裂达成共识的情况下,只有愿意留在当前分片内的节点数大于等于 T,基础分片才可以将响应消息向上传递。

4.2.2 分片的分裂过程

4.2.2.1 请求阶段

基础分片内某一节点发现分片的节点数已经超过了 O,它便发起分裂请求提案。该提案包括其所属基础分片满足分裂条件的证明、请求编号以及它的数字签名。

4.2.2.2 预准备阶段

分裂请求提案一经提出,便会在发起提案的节点所属的基础分片内广播,该基础分片的祖先分片中的节点便可以捕获这个提案,然后捕获提案的节点作为当前其所在的归约树中最高级别分片 PBFT 共识的主节点,执行 PBFT 共识,目的是就该提案是否可以执行达成一致。以图 3 为例,假设分片 C 中的节点发起分裂请求提案,分片 Root、A 和 B 中的节点 1、2 和 3 若是捕获到该提案,则其便作为主节点在相应分片内开始 PBFT 共识。

若是达成一致性共识,则先向另一个非提案来源的子分片广播一致性结果和提案,再由该子分片向它的所有子分片传递一致性结果和提案。以此类推,直到传递至基础分片。以图 3 为例,对于分片 A 中的节点,它们接收的提案来自分片 C,也就是分片 B 这一分支,则在分片 A 达成一致性共识的情况下,分片 A 中的节点要主动将一致性结果和提案传递给另一分支即分片 D,再由分片 D 继续向下层扩散传递。同样地,对应图 4 中的分片 Root 和分片 B,它们要分别传递一致性结果和提案给分片 F 和分片 E,再由 F 和 E 继续向下传递,其中箭头上的标注代表传递过程中一致性结果的来源。

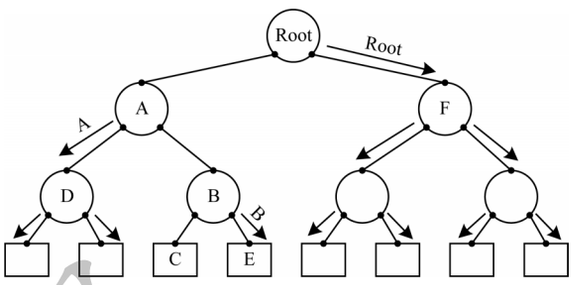

图 4 预准备阶段的消息传递路线

Fig.4 Messaging routes of pre-prepare phase

若是在某一级分片达不成一致性共识,则该分片的主节点要向其子分片传递拒绝请求消息。拒绝请求消息传递路线如图 5 所示。

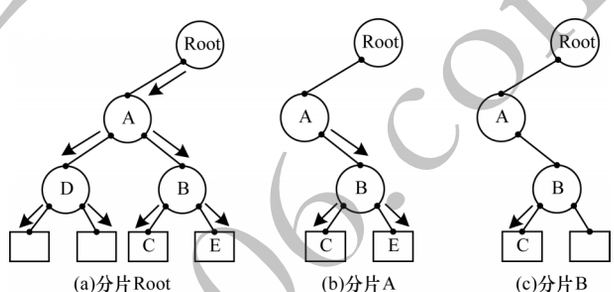

图 5 拒绝请求消息传递路线

Fig.5 Reject request messaging routes

仍以图 3 为例,若是分片 Root 没有达成一致性共识,则主节点会向来源分片(分片 A)传递拒绝请求消息,消息的传递路线如图 5(a)所示。若分片 A 没有达成一致性共识,则拒绝请求消息的传递路线如图 5(b)所示。对于分片 B,拒绝请求消息的传递路线如图 5(c)所示。对于来源分片,例如图 5(a)中的分片 A,则需要向其所有子分片传递拒绝请求消息,以此类推,直到基础分片。这是因为拒绝请求的分片在发现请求需要被否决时,来源分片可能已经达成一致性共识并将结果传递给下级分片。

4.2.2.3 准备阶段

预准备阶段通过消息的并行传递可以使一致性结果和分裂请求提案传递到各个基础分片。此时,若是某基础分片中不属于归约分片的节点捕获到传递来的一致性结果,它便作为该基础分片中 PBFT 共识的主节点执行 PBFT 共识,其目的是就是否响应分片分裂达成一致性共识。在基础分片达成共识后,将结果向上传递给根分片 Root。

与此同时,Root 在预准备阶段达成一致性共识后,便开始使用 VRF 来确定分裂的启动者。在启动者确定后,设置超时时间,等待并收集基础分片的响应消息。若收集到超过 1/2 的来自基础分片的同意响应的消息,则开始分裂的执行阶段。

4.2.2.4 执行阶段

1)分裂过程相关参数的确定

分裂启动者一旦收集到超过1/2的来自基础分片的同意响应的消息,并且全网分片分裂后不会导致某些新分片的节点数低于分片节点数的下限阈值T,它就作为分裂决策共识(共识节点集合是Root中的全部节点)的主节点发起分裂提案,提案中包括分裂过程中相关参数的一些设定,包含VDF所需的时间参数t、安全参数λ和启动者产生提案的当前时隙C以及参数z(将C时隙之后第z个区块的哈希值作为VDF的随机性输入x)。

若根分片Root没有对分裂提案达成共识,则依据多轮验证的思想,重新通过VRF算法选择新的启动者进行新一轮的分裂决策共识,直到达到共识轮数上限仍未达成共识才放弃此次分裂过程。

若在分裂决策共识中分裂提案被同意,则分片Root中的节点便将一致性结果向下传递,直至传递到所有基础分片。

2)节点随机数的生成

在基础分片中的节点接收到分裂决策共识的一致性结果后,便可以按照相关参数生成随机数。DSSM为了对抗Sybil攻击以及避免可能存在的恶意节点相互串通而提前计算符合它们要求的随机数的情况,采用VDF+VRF的方式来生成各个节点的随机数。

算法1 节点随机数生成算法

输入 VDF的难度参数λ,VDF的时间参数t,提案产生的时隙C,分裂启动者给定的参数z,超时时间T_o

输出 VDF的结果v_1,VDF的证明p_1,VRF的结果v_2,VRF的证明p_2

1.(VK_i,SK_i) = VRF_Keygen()//节点获得非对称密钥对//(公钥VK_i,私钥SK_i)

2.x = Hash($block_{C+z}$)//获取VDF随机性输入(将C时隙//之后第z个区块的哈希值作为VDF的随机性输入x)

3.ek,vk = VDF_Setup(λ,t)//计算VDF的公共参数

4.v_1,p_1 = VDF_Evaluate(ek,x)

5.v_2,p_2 = VRF_Evaluate(SK_i,v_1)//使用自身私钥和VDF//的结果v_1作为参数执行VRF

6.if Time.now() ≤ T_o then

7.SendValue(VK_i,v_1,p_1,v_2,p_2)//v_2是生成的随机数

8.end if

3)节点分配

每个节点以产生的随机数与分裂后的子分片数量(这里取2)为参数,通过跳跃一致性Hash算法得到其分裂之后所在的子分片编号。

4.2.3 分裂过程中的特殊情况

在分片的分裂过程中,有可能会出现单个分片因为某种因素不再满足分裂条件的情况,即分片分裂后新分片的节点数低于T或分裂前分片中的节点数就已经小于T,这两种情况都对分片的安全性造成了威胁。对此,提出复用和降级两种相应的处理策略。

1)复用

针对分片分裂后新分片的节点数低于T的情况,选择在分片内复用节点的策略。当出现上述情况时,如果有归约节点愿意同时作为两个新的子分片的节点存在并且同时负责处理两个子分片的事务,则该节点被定义为复用归约节点,如图6(a)中的节点1和2所示。

节点复用的条件如式(1)所示:

$$T \leqslant N_p = N_r + N_u \leqslant 2T \tag{1}$$

当分片内的节点发现上述情况时,开始在分片内广播复用询问消息,此时该节点成为分片中的源节点。当分片内的其他节点接收到此消息时,若是同意作为复用归约节点而存在,则广播发送复用确认消息。

在源节点发送复用询问消息后,开始准备收集确认消息。在源节点收集到足够多的确认消息后,便在本分片(分裂前的分片)内广播复用消息,此消息内包含已确认的复用节点名单的信息。

普通节点继续执行分片分裂过程,复用归约节点不必继续执行此过程,属于它们的分裂过程已经结束,生成的随机数也随之作废。

2)降级

节点降级的条件如式(2)所示,待分裂的分片中的节点数量已低于T:

$$N_p = N_r + N_u < T \tag{2}$$

此时,即使该分片中所有的节点都成了复用归约节点,也无法满足两个子分片中基本节点数的要求。在这种情况下,采取此分片分裂中止并降级的策略。当分片中的节点发现上述情况时,便按照消息的传递规则将降级证明消息传递给全网分片,此消息包含本分片满足降级条件的证明。同时,接收到该消息的本分片节点停止分裂过程,整个分片整体降级。

如图6(b)所示,分片A中的节点数量小于T,那么在分裂过程中采取的策略是分片A中的节点在物理上整体下移至分片B中,具体下移至左子分片还是右子分片,可根据实际情况而制定相应的规则。分片B作为新的基础分片,与其他分裂后产生的新分片同处于归约树叶子节点的位置。分片B的兄弟节点的位置原本也应该有一个基础分片C,只是因为分片降级的问题导致该分片是空分片。此时,分片A与分片B实际上是一个分片,它们的节点集合是一样的,但是它们在归约树中占据两个节点的位置。

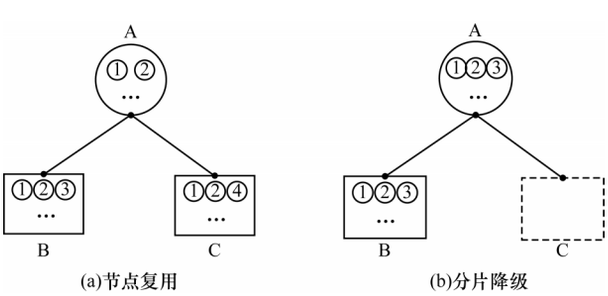

图6 节点复用与分片降级

Fig.6 Node reuse and sharding degradation

4.3 分片的全网合并

分片在正常情况下可能会因为某些原因而导致大批节点宕机,造成分片内节点数过少,从而使得分片的安全性和去中心化程度大大降低。为此,采用分片合并策略。

4.3.1 合并条件

1)节点发起合并请求提案的条件

当基础分片内任一节点发现分片的节点数已经低于下限阈值T并且能够提供所处基础分片满足分片合并条件的证明时,便可以发起合并请求提案。

2)全网分片合并条件

当超过半数的分片响应合并时,开始全网分片的合并。

4.3.2 分片的合并过程

4.3.2.1 请求阶段

类似于分片分裂的请求阶段,在满足相应条件后,节点可以发起合并请求提案。与分裂请求提案不同的是,该提案中要含有该节点所属的基础分片满足合并条件的证明以及合并请求编号。

4.3.2.2 预准备阶段

合并请求提案一经提出便在发起提案的节点所属的基础分片内广播,该基础分片的祖先分片中的节点便可以捕获这个提案。与分裂过程类似,捕获提案的节点作为当前其所在的分片归约树中最高级别分片PBFT共识的主节点,执行PBFT共识。

若是达成一致性共识,则向另一个非提案来源的子分片广播一致性结果和合并请求提案,消息的传递规则与分裂时的预准备阶段一致,传递路径如图4所示。

若是某一级分片达不成一致性共识,则由该分片的主节点向其子分片传递拒绝请求消息。消息的传递路径与分裂时的预准备阶段一致,传递路径如图5所示。

4.3.2.3 准备阶段

类似于分裂的准备阶段,若是基础分片中不属于归约分片的节点捕获到父级分片传递的一致性结果和合并请求提案,它便作为该基础分片中PBFT共识的主节点执行PBFT共识,就是否响应分片合并进行共识。在基础分片达成共识后,将结果向上传递给根分片Root。

根分片Root在预准备阶段达成一致性共识后,便开始使用VRF来确定合并过程的启动者。在启动者确定后,设置超时时间,等待并收集基础分片的响应消息。若是启动者收集到超过1/2的来自基础分片的同意响应的消息,则开始合并的执行阶段。

4.3.2.4 执行阶段

1)合并决策共识

合并启动者一旦收集到超过1/2的来自基础分片的同意响应的消息,就作为合并决策共识(共识节点集合是分片Root中的全部节点)的主节点发起合并提案,就是否进行全网合并进行共识。合并启动者发起的提案中必须包含其所收集的超过1/2的来自基础分片的同意响应的消息。

若分片Root没有对合并提案达成共识,则与分裂时一致,采用多轮验证思想继续选择启动者进行流程。

若在合并决策共识中提案被通过,则分片Root中的节点便将达成的一致性结果向下传递,直至传递到所有基础分片。

2)合并执行

当处于归约树倒数第2层的分片中的节点收到分片Root传递的一致性结果时便开始状态同步过程,将自己目前所保存的两个基础分片的状态同步给自己所处的基础分片的非归约节点(也就是未同步其他分片状态的节点)。此时,原来的只在基础分片中的节点便成为其父级分片中的待同步节点。原来的归约树倒数第2层的分片变成了新的基础分片。

4.4 节点分配算法

在父级分片分裂为两个子分片的过程中,对于其中的节点,采用跳跃一致性Hash算法[26]来决定其将进入哪个子分片。跳跃一致性Hash算法具有均匀性与一致性两个特点。

节点分配算法具体如下:

```
int ch( int k,int n) {
random.seed( k);
int b = –1;
int j = 0;
while ( j < n) {
b = j;
r = random.next( );
j = floor( ( b + 1) / r);
}
return b;
}
```

在DSSM中,由于只需要将父级分片分裂为两个子分片,对应于节点分配算法,则相当于只需要增加一个新的槽位,因此n的取值为2即可。当节点提供的随机数为参数k、分片数目为参数n时,ch(k,n)的返回值为节点所划分到的子分片。本文规定:返回值为0的节点属于左子分片,返回值为1的节点属于右子分片(当n为2时,只有0和1两个返回值)。

4.5 激励机制

DSSM鼓励性能较高的节点同步其所在分片的兄弟分片的状态,从而使高性能的节点成为更高级别的归约节点。若想促使节点积极地参与状态归约,则需要激励机制驱动。

适应DSSM激励机制的原则为:1)对于归约树的不同层级上的分片,在其上执行事务所获得的激励必不相同,处理上层分片的事务所获得的激励应该比下层高,只有这样才能使节点积极地参与到归约过程中;2)分片的合并和分裂过程中承担着相应角色(例如分裂和合并的启动者)的节点也应获得相

应的激励;3)对于恶意行为的检举及证明也会有相应的激励。

具体的激励机制需要匹配具体的网络环境和相应的项目,在此不再赘述。

5 安全性分析与参数设置

5.1 安全性分析

由于DSSM结合多轮共识模型进行验证,因此需要结合多轮共识模型进行分析。

5.1.1 共识的安全性分析

对于采用PBFT算法作为共识机制的区块链网络而言,一旦拜占庭节点的比例超过2/3,根据威胁模型:若拜占庭对手没有获得发布区块的主节点的身份,则它会选择遵守协议不作恶;若拜占庭对手获得了发布区块的主节点的身份,则它就可以轻易地发起双花攻击或发布错误的铸币交易,最终导致数据的不一致。考虑最坏的情况,即一旦拜占庭对手控制了共识节点中超过2/3的节点,那么它就可以发起合谋攻击,在区块中发布错误事务。

为了最大化共识的安全性,从分片的节点数入手降低合谋攻击的概率。在多轮共识模型的基础上,按照最低轮数下限的要求,计算在不同分片节点数的情况下,攻击者若想使合谋攻击的概率高于百万分之一,则需要付出的代价如图7所示,其中代价用攻击者需要控制的最少节点数量表示。

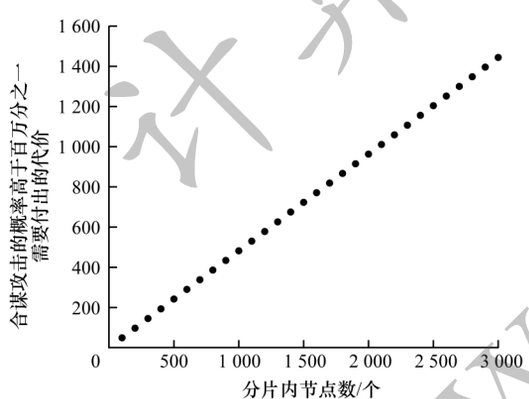

图7 合谋攻击代价

Fig.7 Cost of collusion attack

由图7可以看出,随着分片内节点数的增多,攻击者要想使其发动合谋攻击的概率超过百万分之一,需要控制的节点数也会随之增加,并且近似于线性增加。分片内节点数越多,攻击者发动合谋攻击所付出的代价越大,分片也越安全。

由数据计算而得,若合谋攻击的概率超过百万分之一,则需要攻击者至少控制分片的将近1/2的验证节点。这也意味着:若分片内节点数翻倍,则攻击者进行攻击的代价至少也要翻倍。

5.1.2 节点随机数的可靠性分析

在节点随机数生成算法中,由于参数t和λ是根据分裂决策共识的一致性结果获得的,x是由一致性结果中的信息求得的,那么同一分片的验证节点生成的VDF结果必然是一样的,因此获得了可靠的公共随机数。

所有验证节点以VDF的结果作为VRF的Evaluate(SK,X)函数的随机性输入X,再使用自己的私钥作为输入SK,那么通过计算便可获得属于节点自己的可验证的可靠的随机数。

5.1.3 分片的安全性分析

根据节点随机数的可靠性以及节点分配过程的随机性和不可预测性,拜占庭对手很难再主动地将拜占庭节点聚集到某些固定分片内。

对于单个分片,考虑最坏情况,某个基础分片超过50%的节点被拜占庭对手控制,并且这些恶意节点全部向上归约。由均匀性可得,该分片的父级分片会有将近1/2的节点被拜占庭对手控制,而再上一级的归约分片只有1/4的节点被控制。以此类推,越往上层,拜占庭节点的比例被稀释得越低。

本文规定:DSSM归约模型的最底层,也就是基础分片(由叶子节点代表的分片)所在的那层称为归约模型的第0层,往上1层也就是归约模型的倒数第2层,被称为归约模型的第1层,以此类推。那么,随着层数的递增,某层分片中拜占庭节点的最高比例如式(3)所示:

$$P_j=\left(\frac{1}{2}\right)^j \tag{3}$$

由式(3)可以得出,在分片归约树的第2层分片中的拜占庭节点比例只有1/4,已经可以满足大多数分片协议的容错率,越往上层,比例越低。另外,由于多轮共识模型中多轮验证与欺诈证明的存在,因此拜占庭对手在第0和1层分片作恶成功的概率极低。

在DSSM中,每个基础分片的祖先分片中的节点都会存储该基础分片的状态,并能对该基础分片中的区块信息进行验证。同时,基础分片的各级祖先分片中几乎包含所有基础分片的节点,因此拜占庭对手想要发起攻击而不被发现无异于要控制整个系统。

状态归约与冗余可以使拜占庭对手攻击单个分片,作恶成功的代价无异于攻击整个区块链网络。同时,由于各级归约节点的存在,拜占庭对手在分片动态调整过程中的恶意行为也很容易被检举,因此分片的安全性得到了保障。

5.1.4 数据的一致性分析

状态归约的存在使得上层分片的节点需要存储其孩子分片的状态。一个分片的状态不仅被本分片内的节点存储,而且还会被其所有的祖先分片的节点存储。一个分片的状态会有多个备份。

当有新的节点加入分片时(下层节点通过状态归约加入归约分片或者新加入网络的节点进入基础分片),原分片中的节点与该分片的祖先分片中的节点都可以向新节点同步状态,并且由于在该分片或其祖先分片中存在多个该分片的状态备份,新节点可以快速同步到该分片的最新状态。

参与共识的验证节点必须同步到最新的分片状态,这样才能保证共识正常进行,确保一致性。若因为性能或网络原因,导致节点总是无法及时同步到最新状态,该节点可以选择自行降级,在下一级分片中工作,这也符合DSSM的设计理念。

5.2 多轮共识模型下阈值 *T* 和 *O* 的确定

本节结合多轮共识模型探讨DSSM中参数 *T* 和 *O* 的取值。

5.2.1 分片节点数下限阈值 *T* 的确定

在多轮共识模型下,考虑一种极端情况,每轮选择的共识组的节点与前面轮次的共识组的节点均不同,即每轮共识组内的节点相较于之前都是新的节点。那么,根据多轮共识模型所确定的轮数上限,估算得到分片内至少需要150个节点才能满足需求。

根据上文分析,若拜占庭对手想对有150个节点的分片发动合谋攻击,要使合谋攻击的概率超过百万分之一,则至少需要控制约1/2的节点,但其收益远远低于其所付出的代价(即使控制了1/2的节点也有很大概率攻击失败)。因此,考虑设定下限阈值 *T* 为150。

5.2.2 分片节点数上限阈值 *O* 的确定

由于分裂时子分片是由父级分片一分为二得到的,则父级分片节点数的上限阈值 *O* 需要满足:父级分片分裂之后,其子分片内的节点数仍需满足最低运行标准,即子分片内的节点数仍然大于等于 *T*,那么 *O* 至少要满足的条件是 $O \geqslant 2T$。

在多轮共识模型下,即使分片内节点数达到了600,共识仍然可用。另外,考虑到多分片并行可以提高吞吐量。为兼顾网络的整体性能,分片内的节点数不宜过多。因此,在多轮共识模型下设定上限阈值 *O* 为600。

6 模拟验证

模拟实验将验证DSSM可以随着节点数量的大幅增多实现分片规模的动态扩充,以提供更好的性能,同时在节点显著减少时,DSSM可以为了分片的安全,动态地收缩分片的规模。选用Linux系统作为开发平台,以Go语言作为开发语言,以VS Code和Docker作为开发工具,进行模拟实验。

6.1 实验假设

1)测试网络中节点间的通信状况良好,延迟在可容忍范围内。

2)不考虑由于交易分片导致的分片负载不均衡的问题,即每个基础分片在同一个时间段内需要处理的交易是相对均匀的。

3)系统的拜占庭节点比例为0.33。

4)随着模拟时间的推移,节点的表现可以发生变化,即原本的正常节点可以表现为拜占庭行为,但是系统中表现拜占庭行为的节点的比例不变。

5)节点可以动态加入或退出系统。

6)拜占庭节点秉承对自己最为有利的方式执行操作。

6.2 DSSM 可行性测试

从测试系统的一个初始状态开始,随着时间的推移,逐步往系统中添加或减少节点来模拟节点加入或退出系统的情况,同时逐步往系统中注入交易,观察系统的吞吐能力随着时间的变化情况。在其他外部条件不变的情况下,网络中分片规模的变化可以通过吞吐量的变化体现出来。通过吞吐能力的变化来验证DSSM能够根据节点环境自适应地调整分片的规模。

为能够模拟DSSM中分片分裂过程的各种情况,进行了如下设置:

1)初始状态的系统内有600个节点,正好是分片的上限阈值 *O* 的值。

2)在第10个时隙时,在系统中加入400个节点,满足分片分裂条件,观察系统正常分裂情况下的吞吐能力的变化。

3)在第45个时隙时,在系统中再加入400个节点,每个分片内有700个节点。然后,在第57个时隙时减少某个分片的节点数至200来模拟节点复用情况。

4)在第75个时隙时,在系统中加入1 100个节点,使得每个分片的节点数都达到500个。在第90个时隙时,在系统中加入1 200个节点,此时每个分片的节点数都达到了800。在第103个时隙时,减少某个分片的节点数至100来模拟分片降级情况。在第120个时隙时,将系统恢复至3 200个节点的状态。

经过以上的模拟步骤,得到系统吞吐能力在分片分裂过程中随着时间变化的情况如图8所示。

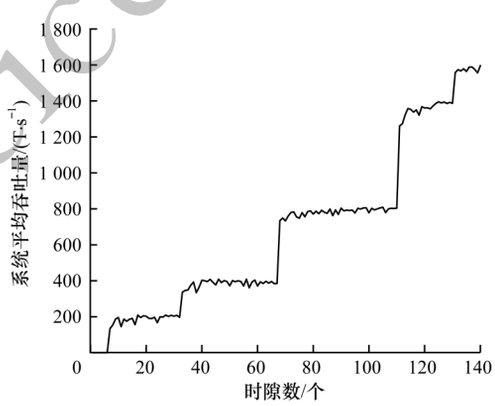

图8 吞吐能力在分裂过程中的变化

Fig.8 Changes of throughput capacity during splitting process

为了能够模拟DSSM中分片合并过程的情况,进行如下设置:初始状态的测试系统内有2 400个节点,分为8个分片,每个分片有300个节点;在第20个时隙时,系统中每个分片都减少100个节点,此时每个分片有200个节点;在第40个时隙时,系统减少600个节点,此时每个分片内有125个节点,满足合并条件。经过以上的模拟步骤,得到系统吞吐能力在分片合并过程中随着时间变化的情况如图9所示。

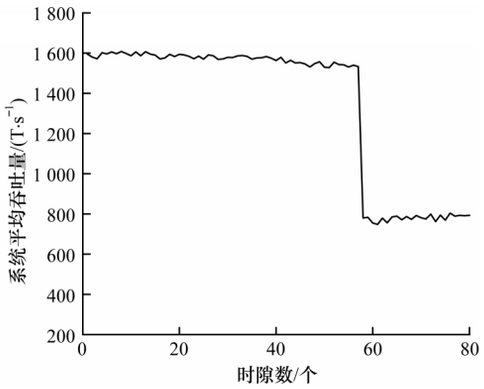

图9 吞吐能力在合并过程中的变化

Fig.9 Changes of throughput capacity during merging process

由图8可以看出,系统吞吐能力随着节点数量的增多得到了增强,而且是近乎翻倍的增强。由图9可以看出,当节点数量大幅减少时,系统吞吐能力有所下降。结合图8与图9验证了DSSM可以根据不同的节点环境,动态快速地调整网络中分片的规模,从而实现了在保证基本安全要求的情况下,自适应快速地进行分片规模的调整,满足系统性能更好的目标。

6.3 可扩展性对比实验

设置结合DSSM的多轮共识模型与未结合DSSM的多轮共识模型的吞吐量对比实验,以验证DSSM能够在节点数量满足条件时具有更好的可扩展性。

在初始状态下,两种模型各有两个分片,每个分片的节点数均为500。然后在第5个时隙时向环境中加入节点,使得每个分片的节点数达到700,观察系统的吞吐能力随时间变化的情况,如图10所示。

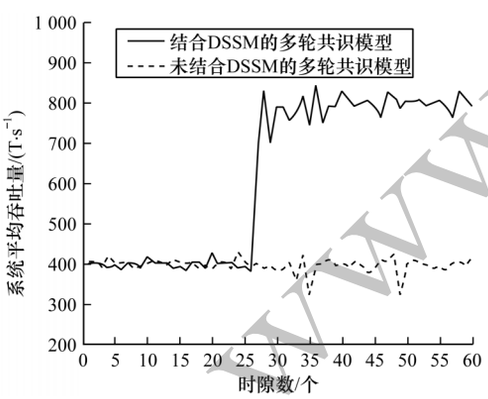

图10 可扩展性对比

Fig.10 Scalability comparison

由图10可以看出,随着节点数量的增多,在一段时间后,采用DSSM的系统吞吐量有明显的提升,而未采用DSSM并沿用传统静态分片方式的系统吞吐量却并未随着节点数量的增多而得到较大改善。

因此,在区块链运行过程中,DSSM可以随着节点数量的增多,动态自适应地扩展分片的规模,以提高网络的性能和可扩展性。

6.4 时延对比实验

设置结合DSSM的多轮共识模型与未结合DSSM的多轮共识模型的交易时延对比实验,以验证DSSM能够在节点数量大幅减少时,通过自适应地收缩分片的规模来保证网络的活性与安全性。

在初始状态下两种模型各有两个分片,每个分片的节点数是150,然后在第5个时隙时在环境中减少节点,使得每个分片的节点数达到80,观察系统的交易时延随时间变化的情况,如图11所示。

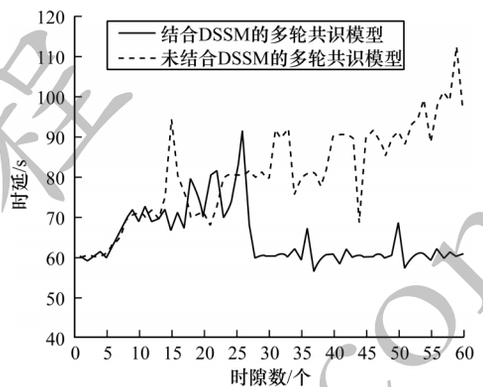

图11 交易时延对比

Fig.11 Transaction latency comparison

由图11可以看出,当节点数量大幅减少时,采用DSSM的系统和未采用DSSM的系统都出现了时延增加且不稳定的情况。这是因为分片内的节点数量变少,拜占庭节点进入共识组的概率会大大增加,导致交易的验证需要更多轮次的共识,所以时延也就升高了。采用DSSM的系统在一段时间后,时延会慢慢降低,逐步恢复到节点减少之前的状态并稳定下来,而未采用DSSM并沿用传统静态分片方式的系统交易时延依旧偏高且很不稳定。

因此,在区块链运行过程中,当节点数量大幅减少时,DSSM可以通过动态自适应地收缩分片的规模来保障网络的活性与安全性。

7 结束语

针对现有分片方案不能及时调整分片规模以适应公链环境下分片内节点数量在短时间内的大幅变化以及由此导致的安全隐患和性能提升困难问题,本文提出一种基于状态归约的自适应节点规模的动态分片可扩展模型。考虑到验证节点的性能和表现差异,在基础和逻辑分片的组合下,通过支持状态归约与状态冗余允许节点在不同层级的分片上进行状态同步,同时分片的动态分裂能够充分利用节点资源以实现更高的性能,分片的动态合并可以更好地保障分片安全,由此使得整个系统具有较好的规模弹性。模拟实验结果证明,该模型能够快速、低成本、自适应地针对节点环境做出相应变化,从而实现分片系统的动态自适应调整。

本文提出的基于状态归约的自适应节点规模的动态分片可扩展模型还有一些地方需要进一步完善。例如,模型中的参数在面对不同的环境和不同的共识机制是不一样的,这与系统的安全要求有关。另外,本文考虑的是一个整体系统的扩展与收缩,尚未针对单个分片的突发情况进行详尽讨论。这些都将是下一步研究的重点内容。

参考文献

[1] NAKAMOTO S. Bitcoin: a peer-to-peer electronic cash system[EB/OL]. [2023-01-11]. http://bitcoin. org/bitcoin. pdf.

[2] JAVARONE M A, WRIGHT C S. From bitcoin to bitcoin cash: a network analysis [C]//Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems. New York, USA: ACM Press, 2018: 77-81.

[3] WANG Q. Improving the scalability of blockchain through DAG[C]//Proceedings of the 20th International Middleware Conference. New York, USA: ACM Press, 2019: 34-35.

[4] 高政风,郑继来,汤舒扬,等. 基于DAG的分布式账本共识机制研究[J]. 软件学报, 2020, 31(4): 1124-1142.
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, 2020, 31(4): 1124-1142. (in Chinese)

[5] SILVANO W F, MARCELINO R. Iota Tangle: a cryptocurrency to communicate Internet-of-Things data[J]. Future Generation Computer Systems, 2020, 112: 307-319.

[6] DANG H, DINH T T A, LOGHIN D, et al. Towards scaling blockchain systems via sharding[C]//Proceedings of 2019 International Conference on Management of Data. New York, USA: ACM Press, 2019: 123-140.

[7] YU G S, WANG X, YU K, et al. Survey: sharding in blockchains[J]. IEEE Access, 2020, 8: 14155-14181.

[8] POON J, BUTERIN V. Plasma: scalable autonomous smart contracts[EB/OL]. [2023-01-11]. https://www. plasma. io/plasma-deprecated. pdf.

[9] DZIEMBOWSKI S, FAUST S, HOSTÁKOVÁ K. General state channel networks[C]//Proceedings of 2018 ACM SIGSAC Conference on Computer and Communications Security. New York, USA: ACM Press, 2018: 949-966.

[10] MICALI S, RABIN M, VADHAN S. Verifiable random functions[C]//Proceedings of the 40th Annual Symposium on Foundations of Computer Science. Washington D. C., USA: IEEE Press, 1999: 120-130.

[11] 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.

[12] BONEH D, BONNEAU J, BÜNZ B, et al. Verifiable delay functions[C]//Proceedings of the 38th Annual International Cryptology Conference. New York, USA: ACM Press, 2018: 757-788.

[13] HackMD. New sharding design with tight beacon and shard block integration[EB/OL]. [2023-01-11]. https://notes. ethereum. org/@dankrad/new_sharding.

[14] WANG J, WANG H. Monoxide: scale out blockchains with asynchronous consensus zones[C]//Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation. New York, USA: ACM Press, 2019: 95-112.

[15] SKIDANOV A, POLOSUKHIN I. Nightshade: near protocol sharding design[EB/OL]. [2023-01-11]. https://near. org/papers/nightshade.

[16] CASTRO M, LISKOV B. Practical Byzantine fault tolerance [C]//Proceedings of the 3rd Symposium on Operating Systems Design and Implementation. New York, USA: ACM Press, 1999: 173-186.

[17] 厚香莹,李志淮,李宝莹. 容忍高比例拜占庭节点的区块链分片扩容算法[EB/OL]. [2023-01-11]. http://www. paper. edu. cn/releasepaper/content/202110-29.
HOU X Y, LI Z H, LI B Y. Blockchain fragment expansion algorithm tolerant of high proportion of Byzantine nodes [EB/OL]. [2023-01-11]. http://www. paper. edu. cn/releasepaper/content/202110-29. (in Chinese)

[18] ZHOU Z X, QIU Z J, YU Q, et al. A dynamic sharding protocol design for consortium blockchains [C]// Proceedings of IEEE International Conference on Big Data. Washington D. C., USA: IEEE Press, 2020: 2590-2595.

[19] ZHANG J T, HONG Z C, QIU X Y, et al. SkyChain: a deep reinforcement learning-empowered dynamic blockchain sharding system[C]//Proceedings of the 49th International Conference on Parallel Processing. New York, USA: ACM Press, 2020: 1-11.

[20] SET S K, PARK G S. Service-aware dynamic sharding approach for scalable blockchain[J]. IEEE Transactions on Services Computing, 2023, 16(4): 2954-2969.

[21] KHACEF K, BENBERNOU S, OUZIRI M, et al. Trade-off between security and scalability in blockchain design: a dynamic sharding approach[C]//Proceedings of International Conference on Deep Learning, Big Data and Blockchain. Berlin, Germany: Springer, 2022: 77-90.

[22] KIAYIAS A, RUSSELL A, DAVID B, et al. Ouroboros: a provably secure proof-of-stake blockchain protocol[C]// Proceedings of Annual International Cryptology Conference. Berlin, Germany: Springer, 2017: 357-388.

[23] KOKORIS-KOGIAS E, JOVANOVIC P, GASSER L, et al. OmniLedger: a secure, scale-out, decentralized ledger via sharding[C]//Proceedings of IEEE Symposium on Security and Privacy. Washington D. C., USA: IEEE Press, 2018: 583-598.

[24] LUU L, NARAYANAN V, ZHENG C D, et al. A secure sharding protocol for open blockchains[C]//Proceedings of 2016 ACM SIGSAC Conference on Computer and Communications Security. New York, USA: ACM Press, 2016: 17-30.

[25] PASS R, SEEMAN L, SHELAT A. Analysis of the blockchain protocol in asynchronous networks [C]// Proceedings of Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin, Germany: Springer, 2017: 643-673.

[26] LEE B, JEONG Y, SONG H, et al. A scalable and highly available network management architecture on consistent hashing[C]//Proceedings of IEEE Global Telecommunications Conference. Washington D. C., USA: IEEE Press, 2010: 1-6.

编辑 陆燕菲