计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.250100133

基于贪心策略的区块链动态分片与跨分片交易协议优化

艾　渊[1]　李家浩[1]　赵毅涛[1]　胡　凯[2,3]
1 云南电网有限责任公司计量中心　昆明 650200
2 北京航空航天大学云南创新研究院　昆明 650233
3 北京航空航天大学计算机学院　北京 100191

摘　要　针对区块链技术中分片机制所面临的挑战,包括负载不均衡、跨分片交易验证的复杂性以及跨分片交易原子性的保障问题,提出了一种优化的动态分片算法和跨分片交易协议。首先基于区块链交易数据,开发了一种基于贪心策略的动态分片算法,该算法通过权值计算动态调整分片,以实现负载均衡。进一步地,针对跨分片交易的原子性和延迟问题,结合交易锁定与回滚机制,提出了一种创新的跨分片交易协议和分片迁移策略,以确保跨分片交易的原子性。实验结果表明,该方法在降低交易延迟方面具有显著效果。

关键词:区块链;分片技术;负载均衡;动态分片算法
中图分类号　TP302.1

Optimization of Blockchain Dynamic Sharding and Cross-shard Transaction Protocol Based on Greedy Strategy

AI Yuan[1],LI Jiahao[1],ZHAO Yitao[1] and HU Kai[2,3]
1 Metrology Center,Yunnan Power Grid Co.,Ltd.,Kunming 650200,China
2 Yunnan Innovation Research Institute,Beihang University,Kunming 650233,China
3 School of Computer Science,Beihang University,Beijing 100191,China

Abstract　An optimized dynamic sharding algorithm,along with a cross-sharding transaction protocol,is proposed to tackle the challenges associated with the sharding mechanism in blockchain technology. These challenges include load imbalance,the complexity of verifying cross-sharding transactions,and ensuring the atomicity of such transactions. To address these issues,this paper develops a dynamic slicing algorithm utilizing a greedy strategy,which adjusts the slicing dynamically through weight calculations to achieve load balancing based on blockchain transaction data. Additionally,to resolve the atomicity and latency issues of cross-slicing transactions,it introduces an innovative cross-slicing transaction protocol and a slice migration strategy. This approach ensures the atomicity of cross-slicing transactions by incorporating a transaction locking and rollback mechanism. Experimental results indicate that this method significantly reduces transaction latency.

Keywords　Blockchain,Sharding technology,Load balancing,Dynamic sharding algorithm

1　引言

区块链技术作为一种分布式账本系统,通过将数据以区块的形式串联,利用密码学原理确保数据的安全性和不可篡改性。其核心创新在于独特的链式数据结构,每个区块内嵌交易信息,并以前一个区块的加密哈希链接,形成一个不断扩展的数据链。这种结构设计确保了一旦数据被记录,任何篡改尝试都将破坏整个链的完整性和一致性。区块链技术的代表性应用包括比特币[1]、以太坊[2]和 HyperLedger Fabric[3]等。

然而,随着区块链上数据量的激增,存储压力随之增大,同时吞吐量和可扩展性问题也成为其发展的主要障碍。例如,比特币由于其系统设定的平均出块时间为 10 分钟,导致其交易吞吐量仅为每秒 7 笔[4];以太坊的峰值吞吐量也仅为每秒 35 笔[5]。这种低吞吐量难以满足互联网规模的交易需求,限制了区块链技术的实际应用和推广。

为了解决区块链技术的可扩展性问题,研究者们提出了多层次的解决方案。这些技术可以分为 3 个层面:网络层、链上层和链下层。网络层的扩展技术主要涉及中继网络和分发网络的构建、OSI 模型的优化以及信息传播机制的改进[6]。链上扩展则通过优化区块链系统的基础架构来提升性能,包括区块数据结构的优化、共识机制的改进、新型链结构的开发以及分片技术的应用[7]。链下扩展则侧重于区块链的应用层和合约层,通过多链技术和状态通道等方法,在不改变区块链基础架构的前提下增强系统的处理能力[8]。

分片技术作为链上扩展技术的一种,被广泛认为是解决区块链可扩展性与性能瓶颈的有效途径。其核心理念在于将区块链网络中的节点划分为多个子网络,即分片,并将交易分

基金项目:中国南方电网有限责任公司科技项目(YNKJXM20222256)
This work was supported by the Science and Technology Project of China Southern Power Grid Co.,Ltd.(YNKJXM20222256).
通信作者:艾渊(aiyuan19890929@163.com)

散至各分片中进行并行处理。这种方法显著增强了交易处理的并发能力，从而提升了系统的整体吞吐量和交易处理效率。

然而，分片技术在将交易数据分散至不同分片处理的同时，不可避免地引发了跨分片交易的问题。以 Monoxide[9] 为例，研究显示系统的吞吐量随分片数增加而线性增长，但跨分片交易的比例也随之增加。即使在分片粒度为 5 的情况下，跨分片交易的比例已接近 80%；当分片数增至 64 时，几乎所有交易都将涉及跨分片；而当跨分片比例达到 100%时，每笔交易都将是跨分片的。由此可见，分片技术所面临的最大挑战是分区引起的跨分片交易问题。

分片技术作为链上扩展解决方案，被广泛认为能有效解决区块链的可扩展性与性能瓶颈问题。Elastico[10]，OmniLedger[11]，RapidChain[12]，LSMD[13]，Pyramid[14] 等项目均通过应用分片技术来提升区块链系统的整体吞吐量和处理能力。Elastico 作为一个无许可区块链协议，通过分片提高处理效率，但其身份认证效率低、存储资源消耗大和跨分片交易原子性保障不足的问题仍待解决。OmniLedger 利用 UTXO 模型和 Atomix 协议确保交易原子性，通过锁定-解锁机制和 blockDAG 结构实现跨分片事务处理，但存在客户端故障导致交易永久锁定的风险。RapidChain 作为一种容许高达 1/3 拜占庭节点的分片区块链协议，通过减少跨分片交易的数据交换和采用 UTXO 模型来提高效率，但隔离性问题未完全解决。LSMD 方案根据节点的交易频率和隶属度分配分片，以提高区块链的吞吐量，核心节点负责中继分片，由频繁交易的节点组成。Pyramid 作为一个分层分片区块链系统，采用账户/余额模型，通过内部分片和中间分片处理交易，利用 BFT 和 CoSi 实现分层共识。

这些项目在交易分片时未充分考虑分析交易特征，导致跨分片交易的处理效率和原子性保障存在挑战。文献[15]研究了一种基于账号的区块链分片方法，该方法通过分析发起方的交易频次进行分片，从而达到降低跨链比例的目的；但是该方法只从定性的角度判断交易是否跨分片，并不能定量地分析交易跨链的次数和交易时延等特性。文献[16-17]中的基于随机性的交易分配策略虽然简单，但会导致大量跨分片交易（如基于交易发起者地址）。文献[18]提出链下实时迁移方法来减少链上交易的数量，提高平衡过程中的效率和可用性。文献[19]提出了一种名为 TxAllo 的动态事务分配机制，将事务分配问题转化为图上的社区检测问题，并使用历史交易数据构建交易图。

针对现有研究中交易分片未充分考虑交易特征导致跨分片比例过高的问题，深入探讨了区块链分片及跨分片交易，并提出了一种基于贪心策略的动态分片与跨分片交易优化方法。以下是本文的主要贡献。

1）基于贪心策略的动态分片算法。针对跨分片交易比例高及分片间负载不平衡的问题，提出了一种基于贪心策略的动态分片算法。该算法依据各分片在每个 epoch 的交易分布情况，动态调整下一 epoch 的分片策略。通过计算分片间的权重，将交易活跃的节点集中到同一分片，实现负载均衡，并减少跨分片交易的发生。

2）跨分片交易协议与迁移策略。针对跨分片交易的原子性保障和高延迟问题，设计了一种优化的跨分片交易协议与分片迁移策略。引入状态节点分片以保存全局状态，支持跨分片交易的验证及节点状态的迁移。该协议在扩展现有状态存储结构的基础上，结合交易锁定和回滚机制，确保交易的原子性。同时，基于全局状态分片实现动态分片与节点迁移，进一步降低延迟并提高系统性能。

本文第 2 章介绍了本文方法；第 3 章介绍了跨分片交易共识协议；第 4 章对提出的方法进行了性能测试；最后总结全文。

2 基于贪心的区块链动态分片算法

为了有效地解决区块链分片技术中存在的负载不均衡和跨分片交易比例过高的问题，提出了一种基于贪心策略的动态分片算法。该算法的核心思想是通过动态调整分片结构，根据交易数据的分布情况优化分片策略，从而实现负载均衡并减少跨分片交易的发生。在具体实现中，算法需要一个明确的优化目标，以指导分片的动态调整过程。为此，首先定义了一个分片策略的优化函数，该函数综合考虑了跨分片交易数量和分片间负载均衡两个关键因素，为动态分片算法提供了理论基础和优化方向。

2.1 分片策略的优化函数

跨分片交易数量增加和工作量分布不均的原因之一是账户划分策略不合理。在账户网络中，每条边表示两个账户的交易关系，边权值统一设为 1，仅考虑交易的存在性，而不考虑其数量。图 1 中，当前账户划分策略仅产生一个跨分片交易，但分片 1 需处理 7 笔交易，而分片 2 仅需处理 1 笔，导致工作量不平衡指数为 6，该划分方法实现了工作负载平衡，但产生了 7 个跨分片交易，增加了处理负担。因此，理想的账户划分策略应将跨分片交易数量降至 1，同时将工作负载不平衡指数降至 0。分片内的节点相互联系紧密，而分片间的连接相对稀疏。

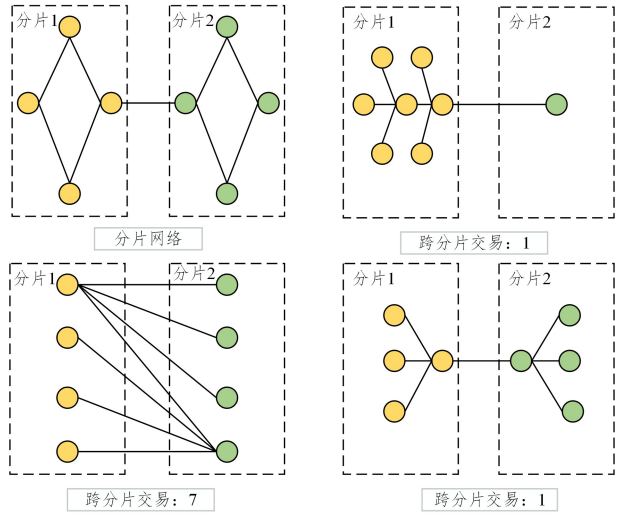

图 1 不同分片方式的对比

Fig. 1 Comparison of different sharding methods

由于区块链中的账户和交易构成复杂的网络，分片策略的重新设计会影响单个节点行为的局部属性。交易处理模式反映账户关系，有效的账户划分策略对优化跨分片交易和实现工作负载均衡至关重要。

考虑一个由 $G(V,E)$ 表示的账户网络，其中 V 代表一组账户地址，记为 $V=\{v_1,v_2,\cdots,v_N\}$，$E=\{e_1,e_2,\cdots,e_n\}$ 是与这些地址相关联的一组边。用 $S_K=\{s_1,s_2,\cdots,s_K\}$ 表示 K 个分

片的集合。对于每个账户地址v_i，定义一个二进制变量$x(v_i, s_k)$来表示一个账户的分配：

$$x(v_i, s_k)=\begin{cases}1, & v_i \in V\\ 0, & \text{else}\end{cases}$$

每个地址v_i应该被分配给一个分片，即：

$$\sum_{k=1}^{K} x(v_i, s_k)=1$$

将地址v_i，v_j分配到两个不同的分片中会导致额外的跨分片交易数，从而增加关联的工作负载。地址v_i，v_j被分配到同一个分片中时的跨分片交易的负载为0。如果地址v_i，v_j被分配到同一个分片中，则有：

$$\sum_{k=1}^{K} x(v_i, s_k)\cdot x(v_j, s_k)=1$$

跨分片交易的工作负载表示如下：

$$C(x)=\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[e_{i,j}\left(1-\sum_{k=1}^{K} x(v_i, s_k)x(v_j, s_k)\right)\right]$$

每个分片的工作量包括分片内和跨分片交易，$L_{k,l}(x)$表示分片k，l之间的跨分片交易。

$$L_{k,l}(x)=\begin{cases}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[e_{i,j}\cdot\left(1-\sum_{k=1}^{K} x(v_i, s_k)\cdot x(v_j, s_k)\cdot x(v_j, s_l)\right)\right], & k\neq l\\ \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left[e_{i,j}\cdot\left(1-\sum_{k=1}^{K} x(v_i, s_k)\cdot x(v_j, s_k)\cdot x(v_j, s_l)\right)\right], & k=l\end{cases}$$

因此，分片k的总交易工作量$L(x)$表示为：

$$L(x)=\sum_{l=1}^{K} L_{k,l}(x)$$

划分一个账户网络时，目标是找到一个设计良好的账户划分解决方法，可以产生最少的跨分片交易数量和最低的工作负载不平衡指数。现在将账户划分问题表述为如下的优化函数：

$$\min F(x)=\alpha C(x)+(1-\alpha)L(x),\ \alpha\in[0,1]$$

其中，α是一个预定义的系数，测量两个客观项$C(x)$和$L(x)$之间的权重。

2.2　分片算法

基于上述的分片策略优化函数，构建了一个区块链网络图模型，该模型通过在全局目标函数中计算顶点间边权值的代价来实现。$G(V,E)$表示一个简单的无向图，其中$|V|=n$表示顶点的数量，$|E|=m$表示边的数量。对于图G中的任意顶点子集S，$e(S,S)$表示两个顶点都在S内的边集，而$e(S,\bar{S})$表示顶点横跨子集S和其补集$\bar{S}$的边集。对于图G中给定的顶点v，定义一个顶点分区$P_v=(S_1,\cdots,S_k)$，其中每个S_i是一个顶点集合。$\partial e(P)$表示跨分区的边集，而$|\partial e(P)|$被称为边缘切割尺寸。

顶点以特定的顺序依次进入系统，考查了3种不同的顶点流入顺序。

1）随机顺序：顶点依据随机排列顺序到达，这种模型假设顶点的到达顺序是完全随机的，不考虑任何潜在的图结构特征。

2）宽度优先搜索（BFS）顺序：通过均匀随机选择一个顶点出发，执行宽度优先搜索生成的。这种顺序倾向于将图的层级结构纳入考虑，从而可能影响分片的质量和性能。

3）深度优先搜索（DFS）顺序：通过均匀随机选择一个顶点出发，执行深度优先搜索生成的。这种顺序强调图的深度连接性，可能对分片策略产生不同的影响。

进一步探讨如何确定适当的成本函数来衡量划分间和划分内的代价。划分间代价通常定义为切割边的总数，而划分内代价则是基于分片内顶点数量来定义的。

在图论中，最优k-划分问题可以表述为：给定一个图$G=(V,E)$，目标是找到一个分区P^*，它能够使得目标函数$f(P^*)\geqslant f(P)$，并且$|P^*|=k$，这样的分区P^*被称为图G的最优k-划分。通过定义一个新的函数$g(P)$，可以将寻找最优划分的问题转化为最大化$g(P)$的问题。

$$g(P)=\sum_{i=1}^{k}|e(S_i,S_i)|-c(|S_i|)$$

$$=m-\left(\sum_{i=1}^{k}|e(S_i,\bar{S_i})|-c(|S_i|)\right)$$

对每个输入节点进行遍历和求解，通过计算其在每个分片中的权重，选择权重大于阈值的分片进行分配。

输入节点v，设置分片负载的最大限度，判断该节点是否为当前轮次的最后一个输入，其中X表示最大轮次数量。接下来，遍历所有分片，检查其负载是否在预设的范围内。如果某分片的负载未超过设定的最大限度，则继续计算其权重，计算公式基于之前推导的公式。在符合负载要求的分片中，选择权重最大的分片进行节点分配。当分片负载超过系统设定的最大值时，将直接跳过该分片，不再为其分配节点。在当前轮次结束后，需清除本轮分片的状态结果。算法的详细流程如算法1所列。

算法1　分片算法求最小权值

```
Input：v，N(v)，k //节点、节点邻居、分片数
Output：partitionID //分片 ID
Begin：
1. Loadlimit=v·n/k //设置分片的负载最大限度
2. Txcount++
3. if Txcountepoch==X //判断是否是最后一个输入
4.     epoch++
5.     partitions.clear()
6. for all partitions i=1 to k do
7.     if|Pi|<Loadlimit then
8.     Pi⋂N(v)
9.     δf(vi,Pi)=|Pi⋂N(v)|·αγ|Pi|^(γ-1)//根据输入节点及其邻居节点计算权值
10.    end if
11. end for
12. for all partitions i=1 to k do
13.    ind=argmax{δf(vi,Pi)} //选取权值最大的分片加入
14. end for
15. Return End
```

2.3　分片流程

基于前述的理论框架，提出了一种动态网络分片方法，该方法能够依据前一周期的交易分布情况，动态调整分片结构。在此模型内，区块链系统的运行时被划分为多个离散的阶段，在这些阶段之间，网络的分片结构会经历随机变化。各个节点依据分片算法在不同分片间进行分配与重新调整。该机制的工作原理如图2所示。

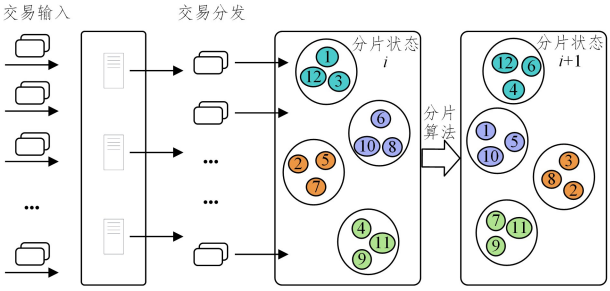

图 2 动态分片模型

Fig. 2 Dynamic sharding model

分片被划分为两种类型，即普通分片和主分片，它们各自承担不同的职责以实现高效的网络运作。普通分片主要负责存储其内部账户状态，并独立验证及处理分片内的交易，这与常规分片模型的操作相似。相对地，主分片的作用是连接多个普通分片，通过维护这些分片中的账户状态来处理跨分片交易。分片间的协作通过相互通信实现，共同完成区块的生成、同步和存储过程，其中每种类型的分片承担着特定的任务。分片流程涵盖以下 4 个主要阶段。

1)初始化阶段：在这一阶段，系统依据预定义的分层分片策略将节点分配到相应的分片中，以确保网络的安全性和性能的均衡。首先，确认委员会节点的身份，确保只有经过授权的节点能够参与管理。然后，委员会成员启动本地分片线程以处理与分片相关的任务，并最终过渡到交易状态，开始处理实际的交易。

2)数据流入阶段：在该阶段，委员会节点的分片线程解析当前轮次内的所有区块，包括本分片及其他分片委员会节点转发的区块，从而获得完整的网络交易状态信息。

3)交易共识阶段：此阶段通过分配普通分片处理内部事务，而主分片处理跨分片事务，以此提升区块链系统的整体性能。分片算法依据交易的发送者和接收者的地址、金额和类型等因素，为每条交易分配一个唯一的分片 ID，确保交易的正确分配。一旦交易被处理，所涉及的分片将更新其状态，主分片与其他相关分片协作，确保跨分片交易的一致性。所有交易处理完毕后，节点生成状态区块并发送提交消息，当超过半数节点确认后，进入共识阶段。

4)分片迁移阶段：状态区块广播完成后，节点进入重配置阶段。如果分片保持不变，节点将更新其新地址和公钥信息；如果分片发生了变化，节点则需要获取目标分片的信息，并更新网络和密钥身份信息，以确保通信的安全。数据同步完成后，节点将能够参与新分片的区块验证和生成过程。

3 跨分片交易共识协议

为了确保跨分片交易的原子性、一致性和高效性，本文设计了一种创新的跨分片交易共识协议。该协议的核心在于通过合理的存储结构和高效的通信机制，协调不同分片之间的状态更新和事务一致性。在跨分片交易中，存储结构的设计至关重要，因为它直接影响到交易的验证效率和系统的整体性能。因此，首先提出了一种新型的状态块存储结构，它不仅能够记录跨分片交易的状态，还能支持高效的交易锁定和回滚机制，从而为跨分片交易的原子性提供保障。接下来，将详细介绍这一存储结构的设计。

3.1 存储结构

该研究提出的分片协议基于账户/余额模型，将账户映射到不同的分片。以付款人地址、收款人地址和交易金额为输入，通过账户余额的分片机制确定付款人所在的分片(Network A)和收款人所在的分片(Network B)。如果两者位于同一分片，账本状态可以快速更新；若在不同分片，则需进行跨分片事务通信。

在每轮共识中，普通分片的领导者负责打包内部交易，生成称为内部块的新区块，并通过 BFT 协议(如 PBFT)提交，类似于传统分片协议中的共识过程。相比之下，主分片的领导者负责打包跨分片交易，生成与多个普通分片状态关联的新跨分片块。

如果主分片的节点通过 BFT 协议直接提交跨分片块，可能会与同轮共识中其他关联分片提交的区块发生冲突。为解决此类冲突，需要协调分片间的状态更新和事务一致性，以确保系统的稳定性和可靠性。

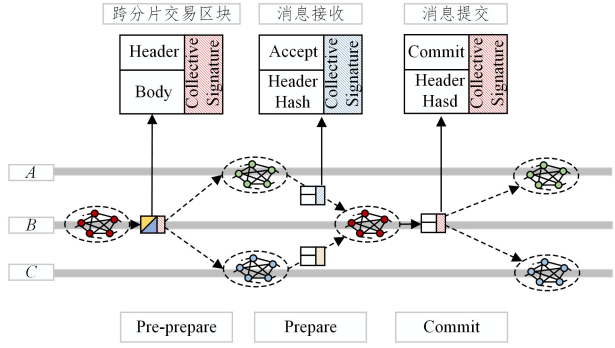

图 3 跨分片交易区块

Fig. 3 Cross-shard trading blocks

该研究提出了一种新型状态块(见图 4)，支持交易的原子性。该数据结构用于存储已确认的事务、未完成的跨分片状态迁移事务和用户间事务。在分片架构的区块链中，传统的 Merkle Patricia Tree(MPT) 结构需适当扩展，以满足需求。引入 Shard 字段、Hash of Previous 字段、Hash of C-Shard 字段和 TxcrossRoot 字段，以有效解决跨分片交易和状态同步问题。

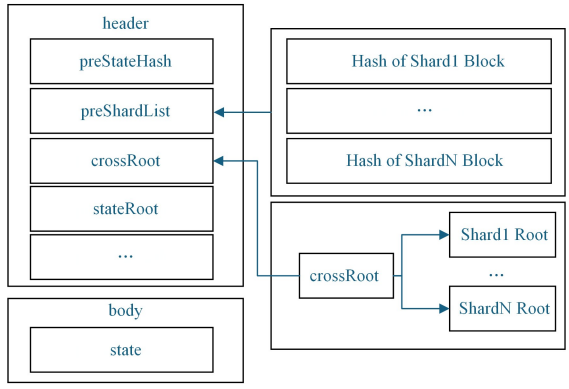

图 4 状态区块数据结构

Fig. 4 State block data structure

在分片架构下，每个节点仅需同步其所在分片的数据。新区块确认后，节点执行区块中的交易，更新本地 MPT 中的账户状态，并计算新的 MPT 根哈希值，确保与新区块头中的 state root 一致。

对于跨分片交易，节点需协调多个分片之间的通信。在prepare阶段，发起方分片将交易信息和锁定状态发送至目标分片。在commit阶段，如果所有参与分片确认交易有效，则交易将正式提交，并相应更新各分片的MPT状态。这一过程确保跨分片交易的原子性与一致性，从而提高系统的可靠性与可扩展性。

对于输入网络的交易，如果其付款人与收款人账户均在一个分片内，则进行片内交易。如果付款人和收款人账户不属于同一分片，则进行跨分片交易的处理。其流程如算法2所示。

算法2　网络中的交易流程

Input：Payer，Payee，Amount//付款人、收款人、金额

Output：Result

Begin：

1. Initialization：$S_{Account}$ = Account − basedSharding //初始化
2. $Network_a = S_{Account}[Payer]$
3. $Network_b = S_{Account}[Payee]$
4. if Account ＜ Total Amount then //如果余额小于转账金额
5.　　Return Failure，no sufficient fund
6. end if
7. if $Network_a = Network_b$//如果收款人和付款人账户在同一分片内
8.　　$Network_a$ updates the ledger by changing the account amount
9. Else //如果收款人和付款人账户在不同分片内
10.　　$Network_a$ communicates with $Network_b$ to update the ledgers together
11. end if
12. Return success

3.2　跨分片交易协议

在每个epoch中，各节点依次执行以下步骤。

1）身份确立与委员会成立：每个节点生成一个包含公钥、IP地址和工作量证明（PoW）的标识。节点需解决计算难题以生成身份，PoW解便于其他节点验证该身份。恶意节点的身份数量受到计算能力的限制，每个节点被分配至对应的委员会。

2）主委员会的确立：节点通过通信发现同一委员会内其他节点的身份。主委员会是包含所有委员会成员的全连通子图。为优化通信量，可采用少量广播（O(NC)）的方法，使节点快速识别彼此的身份。

3）内部共识：一旦委员会内节点对某一分片达成共识，该分片将提交至最终委员会。最终委员会负责处理来自其他委员会的分片，确保全系统在所有分片上达成最终共识。各委员会内运行PBFT协议，基于委员会ID处理不同的交易分片，从而避免交叉，提升系统吞吐量与可扩展性。

4）共识广播：最终委员会根据收到的所有分片值计算最终结果，委员会成员运行拜占庭容错共识协议达成一致，并将最终值广播至全网络。

5）进入下一个epoch：系统通过分布式提交方案生成具有指数偏差且有界的随机值集合，用于下一轮epoch的PoW过程，增强系统的不可预测性与安全性。

处理分片内和跨分片交易的流程如图5所示。主分片保存各个分片的元数据，能够确定交易应发送至哪个分片。当主分片接收到用户的分片内事务消息时，它会生成交易块并转发给负责处理的分片，该分片运行共识算法达成一致后，将包含签名结果的res消息返回给主分片。主委员会检查res消息，并根据共识结果确认交易，向用户反馈。

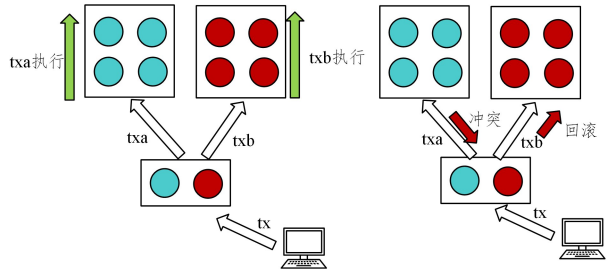

图5　节点间交易图

Fig. 5　Inter-node transaction graph

在跨分片交易中，主分片接收到交易消息后，根据两阶段提交原则将其分为准备交易（Prepare_Tx）和确认交易（Commit_Tx），以确保账本的一致性。在准备阶段，主分片向参与交易的所有分片发送Prepare_Tx消息，并接收各分片的res消息，包含Prepare_Tx的执行结果。在提交阶段，主分片依据第一阶段的结果发送Commit_Tx消息，随后收到的res消息指示提交是否成功。提交后的数据块将通知用户，相关数据在处理期间被锁定以防止其他事务访问。这种方法虽可能牺牲系统可用性，但有效确保了账本一致性。

Prepare交易类型用于锁定交易输入账户的余额。当分片节点接收到Prepare_Tx交易后，将提取输入地址，获取相应账户的公钥进行数字签名验证。验证成功后，交易继续执行；否则交易失败并被丢弃。经过验证的Prepare_Tx交易将被加入本地交易池，等待后续交易打包成区块，分片在收到Prepare_Tx后进行内部共识确认交易。

$$lockvalue_{input} = lockvalue_{input} - value$$

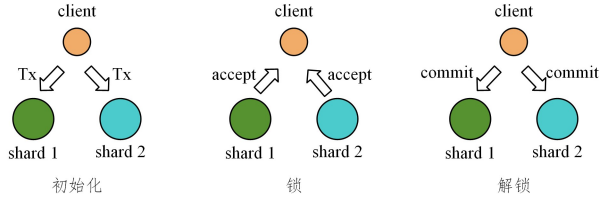

图6　交易锁定

Fig. 6　Transaction locking

当输入账户中的金额被锁定后，执行交易金额的转入操作：

1）输入账户节点所在分片内部共识完成后发送交易确认消息给主分片，其中包含交易确认签名，主分片对确认消息中分片内部签名集合进行验证，当合法签名数量大于节点数量一半时认证成功，避免双花攻击。

2）主分片打包Commit_Tx到交易池中，等待获取交易打包成区块。

3）分片收到Commit_Tx后，进行分片内部共识并交易确认。

4）完成Commit_Tx确认后，输入输出账户完成账户更新，对本地MPT中的字段进行更新：

$$value_{output} = value_{output} + value$$

$$value_{input} = lockvalue_{input}$$

输出账户所在分片执行账户更新后发送确认签名给主分片，主分片对签名集合进行验证，需要满足确认签名数大于分

片节点数量的一半才能完成认证。

当认证签名过程中出现认证失败的情况时，即合法签名数量无法满足大于节点数量一半的要求，则认为交易失败，并对交易执行回滚操作。

3.3 动态分片

在动态分片过程中，当因节点故障、网络变化或负载均衡需求而调整节点分配时，确保系统安全性和高效性至关重要。此过程中，节点配置、数据迁移和下载历史区块数据的开销巨大，因此通过状态块更新全局账户状态。与事务块不同，状态块记录了最新的账户信息，包括地址、余额和 nonce 等数据结构。重新配置初期，分片领导者会遍历最后的状态块中的所有事务块，建立用户账户状态与地址之间的映射。

当主分片的成员在第一阶段结束后，系统将生成下一时期状态区块的签名列表。主分片节点接收到普通分片领导节点的状态区块后，将其广播给其他成员。所有节点验证状态区块的签名有效性，完成后合并每个分片的状态。主分片中的每个节点对合并后的状态进行签名并广播结果，分片随后读取主分片的状态区块，并在检查签名后更新本地账本状态。验证成功后，系统进入下一个阶段，开始新的事务分片。

主分片节点将系统当前状态及最新分片调整结果广播给其他节点，其他成员验证状态正确性后进行签名并返回结果。当领导节点收集到足够的签名时，生成有效的分片状态，并向系统中的所有节点广播，调整后的节点根据最新状态动态调整。

在切换新时期时，成员节点需提交身份信息（如公钥、IP 地址等）给主分片节点进行信息同步，主分片节点检查后应用 PBFT 算法达成共识。身份信息共识验证后，认证过程完成。

在系统中，当交易集中在少数分片而大部分分片交易量较少时，吞吐量将降低。动态分片机制有助于重新平衡事务负载，提高系统性能。状态区块中记录前一个 epoch 各个分片的信息时，使用哈希列表并在区块头添加 preShardList 字段存储每个分片的哈希值，其区块结构如图 7 所示。

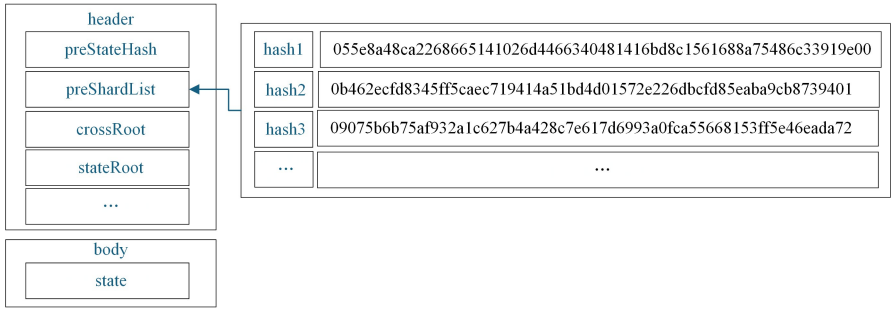

图 7 包含 preshardlist 的区块结构

Fig. 7 Block structure with preshardlist

4 分片优化算法的验证与分析

本章针对前面提出的分片优化算法进行验证与分析，介绍分片区块链系统实验所需要的硬件与软件环境，验证整体系统的功能性和正确性，分析分片优化算法的效果。通过对不同分片算法的分片效果进行对比，验证提出的分片优化算法的有效性，并根据实验数据集进行实验，直接对比分析不同算法的实验结果。

4.1 实验设置

实验环境是拜占庭容错区块链系统，最多有 f 个故障节点，网络中的节点总数需要满足 $3f+1$（f 为正整数）的条件。如果取法定最小值 $f=1$，则实验环境至少需要 4 个节点。

实验环境共有 5 台机器，其中 4 个节点用于构建最小规模的拜占庭区块链节点网络，另一个客户端节点则用于测试整个系统集群。每台服务器都具有以下特点：CPU 为 Intel（R）Xeon（R）Platinum 8255C，所有内核的时钟频率均为 2.5 GHz。服务器内存为 4 GB，支持多线程技术。

网络带宽约为 1 000 Mbps，交换机为 NETGEAR G53088 端口千兆交换机。两台计算机之间的网络延迟小于 20 ms。为减少实验中通信网络等因素的干扰，四台服务器部署在同一局域网内，使用同一网段 IP。

实验数据集来自于网站 xblock.pro 发布的 Ethereum On chain Data 数据集。

4.2 动态分片算法效果分析

跨分片交易比例主要用来衡量不同分片算法在降低跨分片交易数量上的效果，该分片算法的目标是找到一种分片方式，使得子图之间的边数最小，尽量保持降低跨分片交易比例，其计算式如下：

$$TxRatio=\frac{CrossTxNum}{TotalNum}$$

当分片数量 K 发生变化时，不同算法在跨分片交易比例上的表现也有所不同。为了评估这一影响，设置了分片数量为 2 的情境，每个分片包含 4 个节点，并对不同算法的跨分片交易比例进行了计算，结果如图 8 所示。

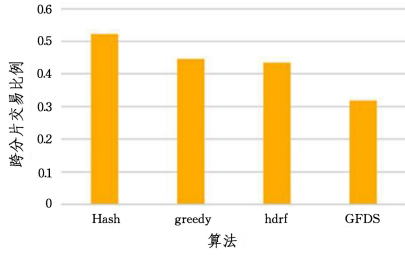

图 8 不同算法跨分片交易的比例

Fig. 8 Ratio of cross-shard transactions for different algorithms

从结果可以看出，由于 Hash 分片算法通过随机分配节点来进行分片，它并未考虑每个节点的交易数量以及不同账户之间的关联性，这种随机性使得跨分片交易比例较大，导致其性能在 4 种算法中最差。此外，较高的跨分片交易比例增加了节点间的通信延迟，进一步加剧了节点间负载的不均衡。相比之下，HDRF（High-Degree Random First）算法在进行分片时，会优先将度数较高的节点进行分区，这一策略有效地减

少了跨分片交易的数量，因此其表现优于 Hash 分片算法。而 Greedy 算法则在分片时优先选择权重较大的节点，这意味着它充分考虑了节点间的关联性，因而其效果与 HDRF 算法接近。该研究提出的算法相比上述其他 3 种算法，更加注重节点间交易的联系性，从而有效减少了跨分片交易的发生。在实验结果中，该研究算法在跨分片交易比例上表现最佳，体现出其在分片效率和负载均衡上的显著优势。

本文进一步探讨了在不同分片数量下的跨分片交易比例表现，通过实验分别选取了分片数量为 2，4，6，8 和 10 的场景，每个分片包含 4 个节点。实验结果如图 9 所示。

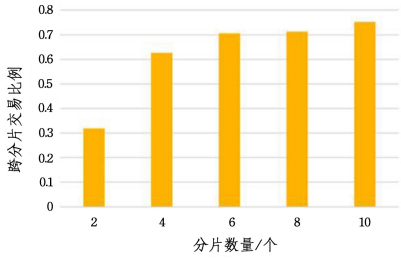

图 9　不同分片数量下跨分片交易的比例

Fig. 9　Percentage of cross-shard transactions with different number of shard

从图中可以明显看出，随着分片数量的增加，跨分片交易的比例呈现出上升的趋势。当分片数量为 2 时，跨分片交易比例显著低于其他分片数量的场景。然而，当分片数量增加至 8 和 10 时，跨分片交易比例的增加趋势趋于平缓，相较于前几种情况，差距并没有明显扩大。

这种趋势表明，尽管增加分片数量可以提升系统的扩展性，但也会导致跨分片交易比例的增加，进而可能影响系统的整体性能。然而，当分片数量达到一定规模后，跨分片交易比例的增加幅度变得较为有限，表明系统在较大分片数量下能够达到一定的稳定性。此结论对未来在实际应用中的分片数量选择提供了重要参考，建议在系统设计中权衡分片数量与跨分片交易带来的性能影响，寻找最佳平衡点。

4.3　跨分片交易共识协议分析

4.3.1　性能分析

设置分片数量分别为 2，4，6，8，10，每个分片含有 4 个节点，对系统吞吐量进行测试，结果如图 10 所示。

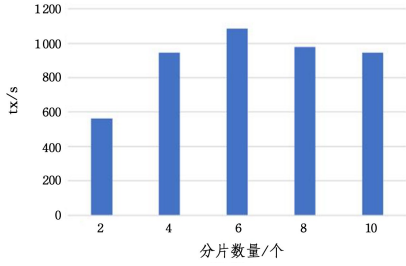

图 10　分片吞吐量数据图

Fig. 10　Shard throughput graph

可以看到，当分片数量增加时，系统吞吐量出现先增加后降低的趋势，并不是随着分片数量的增加，系统吞吐量就越大，造成这种现象的原因是，随着分片数量的增加，跨分片交易的数量也会增加，而跨分片交易的延迟相对较大，从而导致系统吞吐量反而不能增加。分片数量为 2 时吞吐量最低，分片数量为 6 时最高，分片数量为 4，6，8，10 时相差不大。

使用交易确认的时间 t 来计算交易时延，交易确认时间指一条交易从发起到最终共识确认的总时间。

$$t = t_{\text{prepare}} + t_{\text{commit}}$$

首先设置分片数量分别为 2，4，6，8，10，每个分片含有 4 个节点，对系统吞吐量进行测试，结果如图 11 所示。

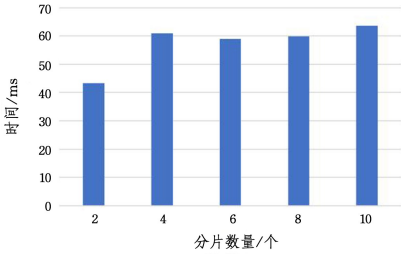

图 11　不同分片数量下的交易延迟

Fig. 11　Transaction latency with different number of slices

当分片数量增加时，交易时延增加，由于分片增加时的跨分片比例增加，导致系统整体时延增加。当分片为 2 时，系统时延最小；当分片数量为 4，6，8，10 时，相差不大，总体保持在 60s 左右。

4.3.2　安全性分析

区块链的安全性依赖于其共识算法。例如，在工作量证明（PoW）协议中，系统能够抵御恶意节点的攻击，只要恶意算力不超过总网络算力的 51%。而在实用拜占庭容错（PBFT）协议中，只要至少有 2/3 的节点是可信的，就能保证共识的绝对安全性。对于单链系统的 PBFT 而言，为了确保安全性，恶意节点的数量必须不超过总节点数量的 1/3。

在区块链系统中，主分片需要生成一个具有集体签名的有效全局状态块。主分片中的共识领导者必须从超过 2/3 的成员节点收集签名。所有诚实节点都可以使用已确认的全局消息来计算全局块的内容，并将他们的签名发送给分片领导者。当系统中恶意节点的比例小于 1/3 时，主分片的共识领导者在具有无效内容的全局块上无法获得超过验证者签名的 1/3，系统整体将处于安全状态。

状态区块中的字段 preShardList 包含前一个 epoch 最后一个通用区块的哈希值列表，这些哈希值用于确保状态区块与区块链主体的正确链接。在进行区块共识时，节点会验证 preShardList 中每个哈希值是否与实际的区块链历史相符。任何篡改操作必须同时篡改 preHash（即 preShardList 的二次哈希）和 preShardList 中至少一个哈希值。在 PBFT 共识机制下，这种篡改行为同样难以获得大多数节点的认可，因为共识过程依赖于大多数节点的正确性和可靠性。

通过默克尔树（MPT）和 preShardList 的验证，区块链系统能够在很大程度上确保区块和状态区块的完整性与一致性。即使在存在拜占庭节点的情况下，只要大多数节点是诚实的，系统便能够抵御伪造和篡改攻击。

结束语　区块链分片技术是提高系统性能的重要策略，它在维持去中心化原则的基础上，能够实现更高的交易吞吐量和减少交易延迟。尽管如此，分片技术在实际部署中仍需解决跨分片通信、数据一致性和可用性等关键挑战。该研究提出了一种基于贪心策略的动态分片算法，该算法通过动态调整分片结构和权重分配来优化负载分布，并通过实验验证了其在实现负载均衡方面的有效性。此外，针对跨分片交易的原子性和延迟问题，本研究设计了一种创新的跨分片交易

协议和分片迁移策略，该策略整合了交易锁定与回滚机制，以确保交易的原子性并减少交易延迟。未来的研究可以进一步探讨该算法在更广泛的区块链网络中的性能表现，并考虑与其他优化技术相结合，以增强分片技术的效能和稳健性。

参 考 文 献

[1] NAKAMOTO S. Bitcoin: A peer-to-peer electronic cash system [J/OL]. 2008. https://www.semanticscholar.org/paper/Bitcoin%3A-A-Peer-to-Peer-Electronic-Cash-System-Hunt/4e9ec92a90c5d571d2f1d496f8df01f0a8f38596?p2df.

[2] BUTERIN V. A next-generation smart contract and decentralized application platform[J]. White Paper, 2014, 3(37): 2-1.

[3] ANDROULAKI E, BARGER A, BORTNIKOV V, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains[C]// Proceedings of the Thirteenth EuroSys Conference. 2018: 1-15.

[4] POON J, DRYJA T. The bitcoin lightning network: Scalable off-chain instant payments[J]. 2016.

[5] FILIBA J. Ethereum breaks one million transactions in a single day[J]. Arch. from Orig, 2017, 22.

[6] IMTIAZ M A, STAROBINSKI D, TRACHTENBERG A. Empirical comparison of block relay protocols[J]. IEEE Transactions on Network and Service Management, 2022, 19(4): 3960-3974.

[7] CROMAN K, DECKER C, EYAL I, et al. On Scaling Decentralized Blockchains: (A Position Paper)[C]// International Conference on Financial Cryptography and Data Security. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016: 106-125.

[8] LERNER S D, CID-FUENTES J Á, LEN J, et al. RSK: A Bitcoin sidechain with stateful smart-contracts [J]. Cryptology ePrint Archive, 2022.

[9] WANG J, WANG H. Monoxide: Scale out blockchains with asynchronous consensus zones[C]// 16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19). 2019: 95-112.

[10] LUU L, NARAYANAN V, ZHENG C, et al. A Secure Sharding Protocol For Open Blockchains[C]// the 2016 ACM SIGSAC Conference. ACM, 2016.

[11] KOKORIS-KOGIAS E, JOVANOVIC P, GASSER L, et al. Omniledger: A secure, scale-out, decentralized ledger via sharding [C]// 2018 IEEE Symposium on Security and Privacy (SP). IEEE, 2018: 583-598.

[12] ZAMANI M, MOVAHEDI M, RAYKOVA M. Rapidchain: Scaling blockchain via full sharding[C]// Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. 2018: 931-948.

[13] HONG Z, GUO S, LI P, et al. Pyramid: A layered sharding blockchain system[C]// IEEE INFOCOM 2021—IEEE Conference on Computer Communications. IEEE, 2021: 1-10.

[14] HONG Z, GUO S, LI P, et al. Pyramid: A layered sharding blockchain system[C]// IEEE INFOCOM 2021—IEEE Conference on Computer Communications. IEEE, 2021: 1-10.

[15] Xi'an Shiyou University. A blockchain sharding method based on transaction frequency analysis: CN202210589229. 8 [P]. 2022-05-26.

[16] CROMAN K, DECKER C, EYAL I, et al. On Scaling Decentralized Blockchains: (A Position Paper)[C]// Financial Cryptography and Data Security: FC 2016 International Workshops, BITCOIN, VOTING, and WAHC. Berlin: Springer, 2016: 106-125.

[17] WANG J P, WANG H. Monoxide: Scale out Blockchains with Asynchronous Consensus Zones[C]// NSDI 2019. New York: ACM, 2019: 95-112.

[18] HONG Z, GUO S, ZHOUE, et al. Gridb: Scaling blockchain database via sharding and off-chain cross-shard mechanism[J]. arXiv: 2407.03750, 2024.

[19] ZHANG Y, PAN S, YU J. TxAllo: Dynamic Transaction Allocation in Sharded Blockchain Systems[J]. arXiv: 2022 [2025-03-18].

AI Yuan, born in 1989, bachelor, engineer. His main research interests include digitalization of power grids and blockchain.