软件学报 ISSN 1000-9825, CODEN RUXUEW
2026,37(8):3337−3369 [doi: 10.13328/j.cnki.jos.007497] [CSTR: 32375.14.jos.007497]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

中继链分片环境中两阶段自适应交易分配模型*

张佩云[1,2], 刘 颖[1,3], 陈子寒[1,2]

[1](数字取证教育部工程研究中心 (南京信息工程大学), 江苏 南京 210044)
[2](南京信息工程大学 计算机学院、网络空间安全学院, 江苏 南京 210044)
[3](南京信息工程大学 软件学院, 江苏 南京 210044)
通信作者: 张佩云, E-mail: zpy@nuist.edu.cn

摘 要: 区块链技术的广泛应用推动了多链应用的发展, 通过跨链技术可以解决不同区块链之间信息隔离的问题. 然而, 当区块链之间存在大量并发交易时, 现有跨链技术不能并行处理跨链交易, 带来可扩展性低的问题, 区块链分片技术可以有效解决该问题. 目前, 不完善的交易分配方法和跨分片交易导致分片技术对可扩展性的提升受限. 因此, 提出面向中继链分片环境的两阶段自适应交易分配模型, 该模型在第 1 阶段得到交易分配方案, 以减少跨分片交易并保证分片负载与分片性能相匹配; 在第 2 阶段, 对中继链收集节点转发后处于不稳定队列中的交易进行微调, 以解决负载激增导致的交易处理延迟增加问题. 在第 1 阶段, 设计一种交易分配预测方法, 该方法利用平行链历史跨链交易信息对交易大小和数量进行预测, 根据预测结果与分片的交易吞吐量计算负载值, 同时, 基于交易依赖性设计跨分片交易分配方法, 结合负载值和该方法得到交易分配方案; 在第 2 阶段, 中继链根据交易分配方案和跨分片交易分配方法将交易转发至对应分片进行处理, 在此过程中用户可能短时间内生成大量交易导致分片负载与分片性能不匹配. 因此, 针对交易队列中等待的交易提出一种交易队列稳定性分析方法, 该方法通过交易队列的长度变化分析交易队列稳定性并对不稳定交易队列中的交易进行分片间动态微调. 通过交易分配预测方法和交易队列稳定性分析方法进行自适应交易分配, 减少交易等待处理的时间并提高中继链的交易吞吐量. 实验结果表明, 所提模型可以并行处理大量并发跨链交易并对交易分配方法进行完善, 相较于对比方法显著提高交易吞吐量, 降低交易的处理延迟.

关键词: 区块链; 中继链; 分片; 自适应交易分配; 交易分配预测
中图法分类号: TP393

中文引用格式: 张佩云, 刘颖, 陈子寒. 中继链分片环境中两阶段自适应交易分配模型. 软件学报, 2026, 37(8): 3337–3369. http://www.jos.org.cn/1000-9825/7497.htm
英文引用格式: Zhang PY, Liu Y, Chen ZH. Two-phase Adaptive Transaction Allocation Model for Relay Chain Sharding Environment. Ruan Jian Xue Bao/Journal of Software, 2026, 37(8): 3337–3369 (in Chinese). http://www.jos.org.cn/1000-9825/7497.htm

Two-phase Adaptive Transaction Allocation Model for Relay Chain Sharding Environment

ZHANG Pei-Yun[1,2], LIU Ying[1,3], CHEN Zi-Han[1,2]

[1](Engineering Research Center of Digital Forensics (Nanjing University of Information Science and Technology), Ministry of Education, Nanjing 210044, China)
[2](School of Computer Science, School of Cyberspace Security, Nanjing University of Information Science and Technology, Nanjing 210044, China)
[3](School of Software, Nanjing University of Information Science and Technology, Nanjing 210044, China)

* 基金项目: 国家自然科学基金 (61872006)
收稿时间: 2024-12-20; 修改时间: 2025-04-28, 2025-05-29; 采用时间: 2025-06-20; jos 在线出版时间: 2025-12-17
CNKI 网络首发时间: 2025-12-25

Abstract: The widespread adoption of blockchain technology has driven the development of multi-chain applications, creating a need for cross-chain technology to address information isolation across different blockchains. However, when a large number of transactions occur concurrently across blockchains, existing cross-chain technologies are unable to process them in parallel, resulting in low scalability. Blockchain sharding offers a potential solution, but its impact on scalability is limited by inefficient transaction allocation and cross-chain transaction methods. Therefore, this study proposes a two-phase adaptive transaction allocation model for a relay chain sharding environment. In the first phase, the model generates an allocation scheme to reduce cross-shard transactions and balance shard load with performance. In the second phase, it fine-tunes transactions in unstable queues after allocation to mitigate delays caused by load surges. In the first stage, this study also includes a transaction allocation prediction method that leverages historical cross-chain data to forecast transaction size and volume, calculating load based on these predictions and transaction throughput. An inter-shard allocation method further refines transaction distribution. In the second stage, the relay chain directs transactions to specific shards based on the allocation scheme, adapting dynamically if load surges lead to a mismatch between shard load and performance. A stability analysis method assesses transaction queue changes, allowing for fine-tuning across shards to reduce waiting times and increase throughput. Experimental results show that this model significantly improves transaction throughput and reduces processing delays compared to existing methods.

Key words: blockchain; relay chain; sharding; adaptive transaction allocation; transaction allocation prediction

区块链是一种去中心化的数字技术, 以安全性、可追溯性和透明性而备受关注[1,2], 其作为一种新兴技术正在改变各个行业的运作方式, 为信息交换和价值转移提供更安全、高效和透明的解决方案. 但是区块链的可扩展性仍需提高, 研究者们为提高区块链的可扩展性已进行多次尝试, 使用的方法主要有支付通道[3,4]、有向无环图[5,6]和分片技术[7−10]等. 区块链分片技术将区块链划分为多个分片, 不同的分片并行处理交易. 分片技术作为最实用的方法之一, 在提高区块链交易吞吐量方面最有效[10], 但这些方法只适用于单链系统. 随着区块链技术的发展, 区块链技术被应用到许多场景中[11−13], 造成多链环境. 由于不同场景中的区块链具有异构性, 存在信息隔离问题, 单个区块链成为“数据孤岛”[14]. 跨链技术可以打破“数据孤岛”, 增强不同区块链之间的互操作性[15], 其中, 中继链技术作为一种可扩展的跨链技术可以基本不受限制地应用在各种场景中[16]. 现在已经有许多针对中继链技术的研究工作[17−22], 其中 Tao 等人[22]通过理论和实验得出中继链存在吞吐量瓶颈的结论, 因此, 对中继链进行可扩展性研究具有必要性, 而中继链作为一种区块链, 其可扩展性研究可以结合区块链分片技术的研究进行思考. 目前, 分片技术对中继链可扩展性的提高存在性能瓶颈, 跨分片交易和负载不均衡问题均是导致该瓶颈的主要原因, 其中, 跨分片交易的存在增加跨链交易处理过程中的通信开销, 负载不均衡问题则会导致分片负载超出分片的处理能力. 中继链分片中交易分配方法的完善可以解决分片性能瓶颈问题, 进一步发挥分片技术的作用, 提高中继链可扩展性.

中继链作为区块链之间的桥梁需要处理所有平行链产生的跨链交易, 随着平行链的增加, 交易的增多, 中继链的负载也随之增加, 因此对中继链的可扩展性进行研究是必要的. 为解决中继链的可扩展性问题, 本文将分片的思想与跨链技术相结合, 对中继链进行分片[23−25], 使中继链中节点能够并行处理平行链中的交易, 从而提高中继链处理交易的效率, 但是分片技术中存在跨分片交易和负载不均衡的问题, 影响分片技术对中继链系统可扩展性的提高[10,26]. 为解决分片负载不均衡问题, Li 等人[10]通过定期将活跃账户从交易数量较多的分片迁移到交易数量较少的分片来动态平衡不同分片上的交易负载, 但是忽略了以下问题: 该方法只关注分片的负载均衡问题, 没有考虑跨分片交易对区块链交易吞吐量的影响, 同时, 该方法没有考虑账户产生的交易突然增加, 导致分片负载激增, 进而增加交易待处理时间的问题. 为在解决分片负载不均衡的同时减少跨分片交易, Xu 等人[27]根据历史交易模式分析账户关系构建账户关系图, 将频繁交互的账户分配到同一个分片中, 同时平衡每个分片中的交易数量, 使分片负载均衡, 但该方法存在问题: 没有考虑分片负载与分片性能之间的关系, 分片性能不满足分片负载需求会导致分片交易吞吐量降低, 同时, 该减少跨分片交易的方法不适用于中继链分片, 中继链分片中跨分片交易的产生主要由于交易之间的依赖性而非账户的分配, 平行链中的跨链交易均由平行链中的收集节点打包并转发给中继链, 无法通过账户分配减少跨分片交易. 本文通过构建面向中继链分片环境的两阶段自适应交易分配模型来解决上述问题, 研究动机如下.

1) 分片负载不均衡和跨分片交易将导致中继链交易吞吐量降低, 首先, 通过交易分配均衡分片负载的方法没有考虑分片的性能, 可能会导致交易在队列中长时间等待, 增加交易处理延迟; 其次, 账户分配方法不适用于中继

© 中国科学院软件研究所 http://www.jos.org.cn

链分片中的跨分片交易问题, 中继链中一笔跨链交易需要经过平行链和中继链进行验证和确认, 该过程产生的子交易之间存在依赖关系, 导致跨分片交易产生, 增加分片之间的通信开销. 因此, 本文设计交易分配预测方法以保证分片负载与分片性能相匹配并减少跨分片交易, 提高中继链交易吞吐量.

2) 分片负载激增将导致交易等待的时延增加, 基于账户关系将账户分配到不同的分片后, 部分账户可能在短时间内产生大量交易, 分片负载迅速增加, 导致账户所属分片的性能无法满足分片负载的需求, 分片无法及时处理交易, 大量交易在分片中等待被处理, 增加交易等待处理的时延. 因此, 本文设计交易队列稳定性分析方法分析分片性能是否不满足分片负载需求, 并对分片中的交易进行分片间微调, 降低队列中交易等待处理的时延.

基于上述问题和动机, 本文贡献如下.

1) 针对中继链吞吐量降低的问题, 设计交易分配预测方法, 对交易信息进行预测并结合分片性能计算负载值, 将分片负载和分片性能转变为负载值以衡量交易分配方案的合理性, 保证分片负载与分片性能相匹配, 同时, 利用交易索引的唯一性, 使与跨链交易相关的子交易均由同一个分片处理, 减少跨分片交易, 降低交易处理延迟.

2) 针对交易等待处理的时延增加的问题, 设计交易队列稳定性分析方法, 根据分片中交易队列长度的变化分析其稳定性并对不稳定队列中的交易进行细粒度微调, 解决分片中负载激增导致的分片负载超出分片处理能力并增加交易处理延迟的问题.

本文第 1 节介绍国内外研究现状. 第 2 节设计面向中继链分片环境的两阶段自适应交易分配模型. 第 3 节从理论上分析本文工作的正确性. 第 4 节描述所提算法. 第 5 节给出实验结果. 第 6 节总结本文工作并进行展望.

1 国内外研究现状

分片技术已经被广泛应用于区块链中以提高区块链可扩展性, 然而, 跨分片交易和负载不均衡问题导致分片技术对区块链可扩展性的提高存在瓶颈, 交易分配方法的完善可以突破该瓶颈, 因此交易分配方法设计是分片技术中重要的一环, 许多使用分片技术的方法中都涉及交易分配方法的设计. 下面从单链和跨链中交易分配方法进行现状分析.

(1) 单链中交易分配方法研究

单链分片中已经有许多研究针对跨分片交易和分片负载考虑交易分配, 单链分片中交易分配问题从账户分配的角度考虑, 围绕跨分片交易和分片负载展开. Xu 等人[27]为减少跨分片交易提出一种优化跨分片交易处理的区块链系统 X-Shard, 根据历史交易模式分析账户关系构建账户关系图, 将频繁交互的账户分配到同一个分片中, 同时平衡每个分片中的交易数量, 使分片负载均衡, 但是该方法没有考虑账户交易激增的情况, 可能导致分片负载激增, 增加交易等待处理的时延. Li 等人[28]为解决实施区块链分片时存在的跨分片交易比例高和跨区块链分片的工作负载不均衡问题, 设计社区感知的账户分区算法和一个弹性分片协议, 实现对区块链中账户的分配, 但是在面对大量账户时, 账户之间的交互更频繁, 使用账户分区算法划分账户存在困难. Jia 等人[29]提出一种低跨分片区块链分片协议, 根据账户间交易的特点定义每个分片的账户归属系数, 使用分片社区重叠传播算法对账户进行分配以降低跨分片交易的比例, 平衡分片之间的工作负载, 但是区块链中账户数量多, 社区重叠传播算法效率低. Huang 等人[9]为解决热点分片和跨分片交易问题, 提出一种跨分片区块链协议 BrokerChain. 该协议利用一种账户分割机制, 使用细粒度图分区算法分割经纪人账户实现所有分片的工作负载均衡, 但是该协议在分配账户时没有考虑分片的性能. Mu 等人[26]为减少跨分片交易并且有效处理工作负载不均衡问题设计一种高效的状态分片区块链系统 EfShard, 该系统利用状态传输协议和一种基于贪婪的状态分配算法实现状态的跨分片迁移, 但是该系统只考虑跨分片交易对分片负载的影响. Li 等人[10]通过研究发现使用分片技术的区块链性能下降的主要原因是不同区块链分片上的交易负载不均衡, 为解决该问题, 该研究工作提出一种不同的分片系统 LB-Chain, 通过定期将活跃账户从负载较重的分片迁移到负载较轻的分片来动态平衡不同分片上的交易负载, 但是该系统只使用交易数量判断分片负载的轻重, 迁移交易时没有考虑分片的性能.

(2) 跨链中交易分配方法研究

目前, 跨链场景只有部分研究涉及交易分配方法, Wang 等人[24]设计动态中继分片方法, 对中继分片的工作负

© 中国科学院软件研究所 http://www.jos.org.cn

载进行评估并在发现瓶颈时进行动态中继分片, 但是, 该方法在中继分片负载太高时需要修改分配方法, 频繁且实时进行中继分片会增加资源消耗和交易处理延迟. Polkadot[30]将中继链中的节点划分为多个验证节点组, 简称验证组. 验证组并行处理跨链交易, 每个平行链对应一个验证组, 一个平行链产生的交易始终交给对应验证组进行验证. 中继链将交易从来源平行链的出口队列转移到目的平行链的入口队列. 如果平行链的入口队列超过区块处理的阈值, 中继链上就会将该入口队列标记为已满, 在队列清空之前不会再接收新的交易. 虽然 Polkadot 并行处理交易可以提高交易吞吐量, 但是该方案存在由于交易始终对应一个验证组, 交易分配方法不完善导致的新交易长时间等待的问题. Xie 等人[31]为了提高共识效率, 提出了一种最优跨链决策算法优化中继链共识方法, 在委员会去中心化、路由成本和计算资源消耗的约束下将每笔交易分配给委员会进行共识, 但是实时分配交易增加交易处理延迟. 不同交易分配方法的对比结果如表 1 所示.

表 1 不同交易分配方法对比

方法名	特点	不足
X-Shard[27]	根据历史交易模式动态分配交易	缺乏考虑账户交易激增情况
Estuary[29]	根据分片账户归属系数定期划分账户以分配交易	社区重叠传播算法效率低
BrokerChain[9]	结合账户状态图以自适应动态划分账户以分配交易	缺乏考虑分片性能
EfShard[26]	基于贪婪分配算法实现交易分配	缺乏考虑分片性能
Li等人[28]	利用社区感知的账户分区算法动态划分账户以分配交易	账户之间的频繁交互增加账户划分延迟
LB-Chain[10]	定期将活跃账户从负载较重分片迁移到负载较轻分片以均衡分配交易	迁移账户时没有考虑分片性能
Mitosis[24]	根据负载实时修改分配方法	实时进行中继分片增加交易处理延迟
Polkadot[30]	每个平行链交易由固定验证组处理	新交易可能长时间等待
Xie等人[31]	将交易分配给委员会进行共识	实时分配交易增加交易处理延迟

综合上述分析可知: 1) 单链中交易分配方法研究工作处理负载问题时只考虑交易数量对分片负载的影响, 在分配交易和账户时没有考虑交易大小对分片负载的影响以及分片负载和分片性能的匹配问题, 且单链中交易分配方法没有考虑对交易分配后分片中交易数量激增导致分片负载激增的情况; 2) 跨链中交易分配方法没有涉及跨分片交易, 单链对跨分片交易的解决方法不适用于跨链系统. 基于中继链分析其原因, 首先, 平行链和中继链针对跨链交易产生的子交易之间存在依赖关系, 导致交易跨分片, 单链则由于被分配在不同分片的账户进行交易; 其次, 跨链交易以平行链为单位打包交给中继链处理, 不涉及具体平行链中账户之间的关系, 单个区块链主要通过账户的分配和迁移改善分片的负载状态. 为此, 本文提出面向中继链分片环境的两阶段自适应交易分配模型以解决上述问题.

2 交易分配模型

2.1 相关概念

本文构建交易分配需要涉及的基本概念如下.

1) 平行链: 接入中继链的区块链称为平行链, 平行链分为源链和目标链, 源链为发起跨链交易的平行链, 目标链为接收跨链交易的平行链. 令 B_i 表示第 i 个平行链的编号并作为平行链的唯一标识.

2) 收集节点: 收集节点分为 3 种, 分别是平行链的收集节点、中继链分片的收集节点和中继链收集节点. 平行链的收集节点负责收集所属平行链生成的多个交易并打包为交易包, 将交易包转发给中继链收集节点; 中继链收集节点负责根据交易分配方案将交易包转发给对应的中继链分片; 中继链分片的收集节点负责对收到的交易包进行检查并将交易包转发给分片内的共识节点进行处理并收集共识节点的共识结果. 为避免单个收集节点单点故障的问题, 本文设置多个收集节点对交易进行收集转发, 多个收集节点组成的集合称为收集节点集.

3) 共识节点: 中继链和平行链中收集节点以外的其他节点, 对区块链中的交易进行验证和共识.

4) 中继链: 中继链连接不同的平行链, 平行链生成的跨链交易交给中继链进行验证并转发给对应的平行链.

© 中国科学院软件研究所 http://www.jos.org.cn

5) 全局分片: 负责收集平行链跨链交易信息和分片信息, 包括交易数量、大小和分片的交易吞吐量, 根据平行链的历史跨链交易信息对平行链即将生成的交易的信息进行预测, 根据预测结果计算负载值, 基于负载值得到平行链跨链交易的分配方案并存储该方案, 中继链系统中只有一个全局分片, 令 C_0 作为全局分片的编号.

6) 局部分片: 负责并行处理平行链产生的跨链交易, 中继链系统中有多个局部分片, 令 C_i 表示第 i 个局部分片的编号并作为局部分片的唯一标识.

7) 子交易: 平行链和中继链为处理一笔跨链交易产生的交易为该笔跨链交易的子交易.

8) 跨链交易包: 源链发起的跨链交易经过共识后由该源链的收集节点收集并打包为交易包, 收集节点将交易包转发至中继链进行处理, 令 $\mathbb{B}$ 表示跨链交易包的数据结构, $\mathbb{B}=<i,\mathcal{V},\mathcal{W}>$. 其中, i 表示跨链交易包的编号, $\mathcal{V}$ 表示源链中验证该交易包的共识节点的签名集合, $\mathcal{W}$ 表示跨链交易包中跨链交易索引的集合.

9) 平行链的历史跨链交易信息: 全局分片从中继链收集节点中收集平行链的历史跨链交易信息, 令 $\mathbb{F}$ 表示平行链的历史跨链交易信息的结构, $\mathbb{F}=<\mathcal{B},\mathcal{C}>$. 其中, $\mathcal{B}$ 和 $\mathcal{C}$ 分别表示不同平行链产生的跨链交易数量集合和大小集合.

10) 历史分片信息: 全局分片从局部分片的收集节点中收集历史分片信息, 令 $\mathbb{M}$ 表示历史分片信息的结构, $\mathbb{M}=<\mathcal{P},\mathcal{Q},\mathcal{R}>$. 其中, $\mathcal{P}$ 和 $\mathcal{Q}$ 分别表示不同局部分片中未处理完的跨链交易所对应的数量集合和大小集合, $\mathcal{R}$ 表示不同局部分片的交易吞吐量集合.

11) 平行链跨链交易分配方案: 全局分片根据历史平行链跨链交易信息和历史分片信息对跨链交易进行分配, 得到分配方案, 令 X 表示平行链跨链交易分配方案, $X=\{\mathbb{x}_i|i\in\{1,2,\ldots,M\}\}$. 其中, $\mathbb{x}_i$ 表示将平行链 B_i 的交易分配给局部分片 $C_{\mathbb{x}_i}$, M 表示平行链的数量.

2.2 中继链交易分配系统架构

本文所提出的中继链交易分配系统架构如图 1 所示.

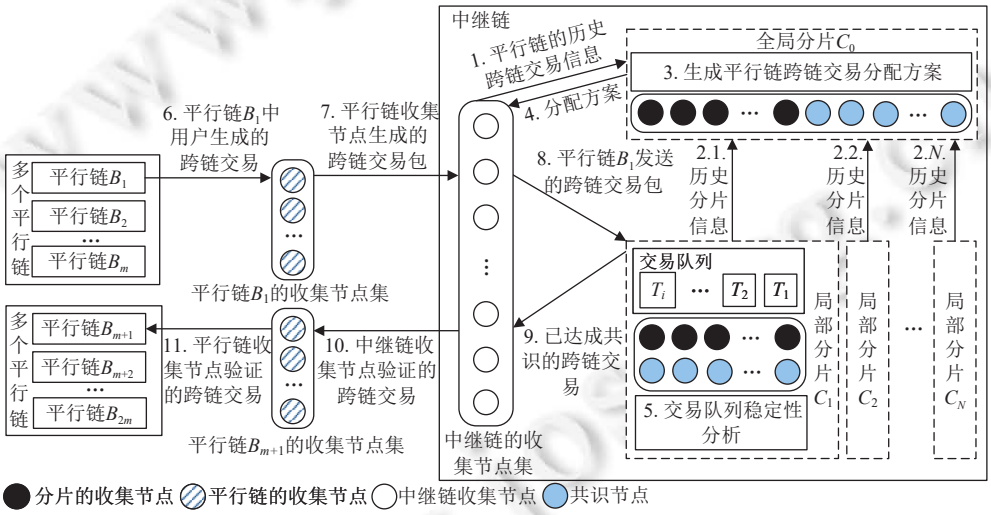


图 1 中继链交易分配系统架构图 (步骤 1–5 为本文主要工作)

图 1 中角色包括平行链 B_1、平行链 B_{m+1}、中继链收集节点和中继链分片. 平行链 B_1 中的用户发送一笔跨链交易到平行链 B_{m+1} 中的用户, 其中平行链 B_1 的交易分配给局部分片 C_1, 具体步骤如下.

(1) 全局分片收集节点从中继链收集节点收集平行链历史跨链交易, 并预测平行链将产生的交易信息.

(2) 全局分片的收集节点从局部分片的收集节点中收集局部分片的剩余交易信息和局部分片交易吞吐量信息.

(3) 全局分片利用预测的信息和分片的相关信息计算负载值, 并基于负载值和跨分片交易分配方法得到平行链跨链交易分配方案, 全局分片并对该方案进行存储.

© 中国科学院软件研究所 http://www.jos.org.cn

(4) 全局分片的收集节点将分配方案转发至中继链收集节点.

(5) 局部分片 C_1 的收集节点定期进行交易队列稳定性分析, 对不稳定交易队列进行队列交易细粒度微调.

(6) 平行链 B_1 的用户生成跨链交易并将跨链交易转发给平行链 B_1 的收集节点.

(7) 平行链 B_1 的收集节点对交易进行初步的验证并将交易打包为交易包, 转发至中继链收集节点.

(8) 中继链收集节点根据接收到的平行链跨链交易分配方案和跨分片交易分配方法, 将交易包转发到对应的局部分片 C_1 的收集节点, 局部分片 C_1 将交易存放在交易队列中, 对交易进行检查并将交易转发到共识节点进行共识.

(9) 局部分片的收集节点根据共识结果, 从交易队列中取出跨链交易并将交易转发至中继链收集节点.

(10) 中继链收集节点将跨链交易转发给平行链 B_{m+1} 的收集节点.

(11) 平行链 B_{m+1} 的收集节点对跨链交易进行验证并根据接收者的地址将跨链交易发送给对应的用户.

在图 1 的架构中, 步骤 1–4 对应两阶段自适应交易分配模型中的交易分配预测阶段 (第 2.3.1–2.3.3 节), 步骤 5 对应该模型的交易队列稳定性分析阶段 (第 2.3.4 节).

2.3 两阶段自适应交易分配模型

基于图 1, 本文设计两阶段自适应交易分配模型框架图, 如图 2 所示.

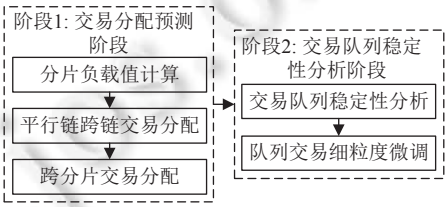


图 2 两阶段自适应交易分配模型框架图

图 2 中有两个阶段, 分析如下.

● 阶段 1: 交易分配预测阶段.

该阶段包括: 1) 分片负载值计算 (第 2.3.1 节): 全局分片结合平行链的历史跨链交易信息预测交易信息, 基于预测的交易大小和数量计算平行链跨链交易对局部分片产生的负载值, 并基于局部分片中剩余交易大小、数量和局部分片的交易吞吐量计算出局部分片剩余负载值; 2) 平行链跨链交易分配 (第 2.3.2 节): 全局分片基于局部分片负载值计算结果设计平行链跨链交易分配方法, 得到平行链跨链交易分配方案; 3) 跨分片交易分配 (第 2.3.3 节): 跨分片交易由跨链交易的子交易产生, 本文对跨分片交易分配方法进行设计, 实现对跨分片交易的分配.

● 阶段 2: 交易队列稳定性分析阶段.

该阶段包括: 1) 交易队列稳定性分析 (第 2.3.4 节): 中继链收集节点基于交易分配预测阶段得到的交易分配方案和跨分片交易分配方法将跨链交易转发至对应局部分片, 每个局部分片的收集节点根据交易队列长度进行交易队列稳定性分析; 2) 队列交易细粒度微调 (第 2.3.4 节): 局部分片收集节点根据交易队列稳定性分析的结果并对不稳定队列中的交易进行局部分片间微调.

2.3.1 分片负载值计算

本文在交易分配预测方法中收集每个平行链的历史跨链交易信息对平行链中交易数量和大小进行预测, 将预测结果作为计算平行链跨链交易对分片产生的负载值时使用的数据. 目前, 很多研究[9,10,26,28]在考虑分片负载时, 只考虑分片需要处理的交易数量, 将分片的负载与交易数量联系在一起. 但是, 本文认为分片中的负载不仅与交易数量有关并对区块链中的负载重新定义. 区块链是一种分布式数据库, 节点之间的交易和区块的传播都在网络层中处理[32]. 类比现实世界中文件的传播, 文件的大小会影响文件在网络中的传播速度. 当交易大小不同时, 在节点之间的传播速度也会不同. 所以, 本文对区块链中的负载问题有如下定义. 首先, 当交易数量较大时, 节点需要验证并达成共识的交易数量较大, 其他交易需要等待更多的时间, 节点处理完所有交易花费的时间会更长. 其次, 交易

© 中国科学院软件研究所 http://www.jos.org.cn

大小影响节点之间传播交易的时间, 交易在转发过程中的延迟影响处理交易所花的总时间.

针对交易分配问题, 本文在对分片负载值计算时不仅考虑平行链跨链交易对局部分片产生的负载, 还考虑交易队列中的交易以及局部分片对交易的处理能力, 其中, 局部分片对交易的处理能力通过分片的交易吞吐量进行量化.

本文在对局部分片负载值计算时利用多个属性, 在计算平行链跨链交易对局部分片产生的负载值时使用交易大小和交易数量这 2 个属性; 在计算局部分片的剩余负载值时使用交易大小、交易数量和交易吞吐量这 3 个属性. 为在多属性情况下进行分片负载值计算, 本文研究发现采用逼近理想解排序方法可以避免数据的主观性[33], 但是该方法常用的相对接近度量不对称, Yin 等人[34]对属性值相对度量方法进行改进, 增加该算法的合理性并对此进行证明, 因此本文使用该方法对局部分片负载进行评估. 由于不同的属性对负载值的影响不同, 本文采用熵权法分别平衡上述 5 个属性的权重, 熵值越小, 信息量越大, 该属性的权重越大, 表明该属性对最终负载值的决定性越强, 反之更弱. 根据熵权法确定权重保证权重符合本文所提区块链网络的情况, 进而计算出平行链跨链交易对局部分片产生的负载值和局部分片的剩余负载值. 利用逼近理想解排序改进方法计算局部分片剩余负载值的过程如图 3 所示.

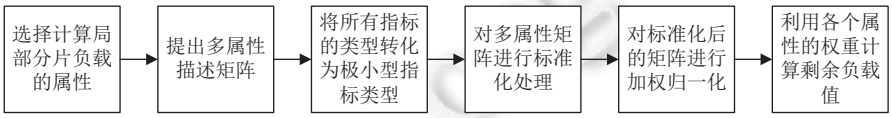


图 3 剩余负载值计算过程

为计算平行链跨链交易给局部分片带来的负载值, 本文根据交易数量和交易大小构建平行链跨链交易对局部分片造成的负载的多属性描述矩阵 $\mathcal{A}$, 矩阵的定义如公式 (1) 所示:

$$\mathcal{A}=\begin{bmatrix} c_1 & s_1 \\ c_2 & s_2 \\ \vdots & \vdots \\ c_M & s_M \end{bmatrix} \tag{1}$$

其中, 矩阵的每一列代表不同的属性, 每一行代表不同的平行链, c_i 表示 B_i 将生成的交易数量, s_i 表示平行链 B_i 中将生成的交易大小, $i\in\{1,2,\ldots,M\}$, M 表示平行链的数量. 不同的局部分片对交易的处理能力及其中剩余的交易数量和大小不同, 为计算局部分片的剩余负载值, 本文根据局部分片中剩余交易数量、交易大小和交易吞吐量构建局部分片中剩余负载的多属性描述矩阵 $\boldsymbol{A}$, 矩阵的定义如公式 (2) 所示:

$$\boldsymbol{A}=\begin{bmatrix} g_1 & d_1 & t_1 \\ g_2 & d_2 & t_2 \\ \vdots & \vdots & \vdots \\ g_N & d_N & t_N \end{bmatrix} \tag{2}$$

其中, 矩阵的每一列代表不同的属性, 每一行代表不同的局部分片, g_i 表示局部分片 C_i 中剩余的交易数量, d_i 表示局部分片 C_i 中剩余的交易大小, t_i 表示局部分片 C_i 的交易吞吐量, $i\in\{1,2,\ldots,N\}$, N 表示局部分片的总数. 两个多属性矩阵构建完成后, 开始计算负载值, 其统一步骤如下.

1) 统一指标类型. 平行链跨链交易对局部分片造成的负载由交易数量和大小两个属性组成, 两个属性值越小, 负载就越小, 对于中继链分片, 负载越小越好, 因此交易数量和大小都是极小型指标. 而局部分片的剩余负载中, 除交易数量和大小两个极小型指标外, 还有局部分片中交易吞吐量作为属性. 针对剩余负载, 吞吐量越大, 局部分片对交易的处理能力越强, 处理剩余负载的时间越短, 剩余负载对局部分片造成的影响就越小, 因此吞吐量是极大型指标, 由于多个属性的指标不同, 所以需要对指标的类型进行统一. 考虑到只有交易吞吐量一个极大型指标, 所以本文将该属性转化为极小型指标, 以简化统一指标类型运算, 其他属性不需要对指标类型进行改变, 统一指标类型后的结果用原属性值表示. 令 $\dot{t}$ 为对 t_i 统一指标后的结果, 将交易吞吐量转化为极小型指标如公式 (3) 所示:

© 中国科学院软件研究所 http://www.jos.org.cn

$$\mathcal{t} = \max\{t_i | i \in \{1,2,\ldots,N\}\} - t_i \tag{3}$$

统一指标类型后, 只有 $\boldsymbol{A}$ 中元素发生变化, 令 $\boldsymbol{D}$ 为对 $\boldsymbol{A}$ 统一指标类型后的矩阵, 得到的矩阵如公式 (4) 所示:

$$\boldsymbol{D} = \begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ \vdots & \vdots & \vdots \\ b_{(N-1)0} & b_{(N-1)1} & b_{(N-1)2} \end{bmatrix} \tag{4}$$

其中, b_{ij} 表示不同属性统一指标类型后的值. $\boldsymbol{D}$ 中的所有属性都为极小型指标.

2) 标准化处理. 由于本文使用的属性涉及多个方面, 其量纲不同, 为消除不同量纲的影响, 需要对矩阵进行标准化处理. 令 $\mathcal{D}$ 表示对 $\mathcal{A}$ 统一指标类型后的矩阵, 其中的元素值与 $\mathcal{A}$ 中元素值相同. 同时, $\mathcal{D}$ 中的所有属性都为极小型指标. 标准化 $\mathcal{D}$ 和 $\boldsymbol{D}$ 后的矩阵分别记为 $\mathcal{Z}$ 和 $\boldsymbol{Z}$, z_{ij} 表示 $\boldsymbol{Z}$ 中第 i 行第 j 列的元素, 标准化方法如公式 (5) 所示:

$$z_{ij} = \frac{b_{ij}}{\sqrt{\sum_{i=0}^{N-1} b_{ij}^2}} \tag{5}$$

令 y_{ij} 表示 $\mathcal{Z}$ 中第 i 行第 j 列的元素, 其标准化方法与公式 (5) 相同.

3) 加权处理. 本文的方案一共涉及 5 个属性, 这 5 个属性对最终的负载值会产生不同的影响, 因此本文设置不同的权重描述不同属性在计算负载值时不同的影响力并基于熵权法确定各个属性的权重. 在对平行链中交易大小和数量以及局部分片中剩余交易大小和数量对局部分片造成的负载进行计算时, 本质上都是针对局部分片进行计算的. 在设置权重时, 本文对平行链以及局部分片中交易大小和数量分配比例相同的权重, 构建权重集合 $\boldsymbol{W} = \{w_i | i \in \{0,1,\ldots,4\}\}$. 其中, w_i 表示不同属性值所占的权重. 本文使用熵权法确定各个属性的权重, 令 $\mathbb{p}_{ij}$ 表示局部分片 C_{i+1} 内剩余交易数量、大小和交易吞吐量这 3 个属性的概率值, 如公式 (6) 所示:

$$\mathbb{p}_{ij} = z_{ij} \Big/ \sum\nolimits_{i=0}^{N-1} z_{ij} \tag{6}$$

令 $\mathbb{q}_{ij}$ 表示平行链 B_{i+1} 所产生跨链交易的两个属性的概率值, 其计算方法与公式 (6) 相同. 令 $\mathbb{z}_j$ 表示局部分片内交易属性和性能属性的信息熵, 如公式 (7) 所示:

$$\mathbb{z}_j = -\frac{1}{\ln N} \sum\nolimits_{i=0}^{N-1} (\mathbb{p}_{ij} \times \ln \mathbb{p}_{ij}) \tag{7}$$

令 $\mathbb{y}_j$ 表示平行链产生交易的不同属性的信息熵, 其计算方法与公式 (7) 相同. 根据信息熵得到各个属性的最终权重, 如公式 (8)–(10) 所示:

$$w_p = (1 - \mathbb{y}_p) \Big/ \sum\nolimits_{j=0}^{1} (1 - \mathbb{y}_j) \text{且} p \in \{0,1\} \tag{8}$$

$$w_a = w_{a-2}(2 - \mathbb{z}_0 - \mathbb{z}_1) / (w_0 + w_1) \sum\nolimits_{j=0}^{2} (1 - \mathbb{z}_j) \text{且} a \in \{2,3\} \tag{9}$$

$$w_4 = (1 - \mathbb{z}_2) \Big/ \sum\nolimits_{j=0}^{2} (1 - \mathbb{z}_j) \tag{10}$$

其中, 为使 $w_0/w_1 = w_2/w_3$ 且不影响 w_4 的权重, 在使用熵权法计算 w_2 和 w_3 的基础上添加比例限制. 根据权重分别计算负载加权矩阵 $\mathcal{U}$ 中第 i 行第 j 列的元素 $u_{ij} = w_j \times y_{ij}$ 和剩余负载加权矩阵 $\boldsymbol{U}$ 中第 i 行第 j 列的元素 $f_{ij} = w_{(j+2)} \times z_{ij}$.

4) 计算负载值. 计算负载值时, 需要每种属性中的最优和最劣的属性值. 为得到最优和最劣属性值, 令 $\mathbb{C}$ 表示两个标准化矩阵中每一列元素的最优属性值集合, 令 $\mathbb{D}$ 表示两个标准化矩阵中每一列元素的最劣属性值集合. 本文定义最优属性值 $\mathbb{C} = \{\mathbb{c}_i | i \in \{0,1,\ldots,4\}\}$ 和最劣属性值 $\mathbb{D} = \{\mathbb{d}_i | i \in \{0,1,\ldots,4\}\}$. 其中, $i \in \{0,1\}$ 时, $\mathbb{c}_i$ 和 $\mathbb{d}_i$ 分别表示矩阵 $\mathcal{U}$ 中第 i+1 列元素的最大值和最小值, $i \in \{2,3,4\}$ 时, $\mathbb{c}_i$ 和 $\mathbb{d}_i$ 分别表示矩阵 $\boldsymbol{U}$ 中第 i–1 列元素的最大值和最小值. 令 $\mathbb{G}_i$ 和 $\mathcal{G}_i$ 分别表示 $\mathcal{U}$ 中属性值的乐观距离和悲观距离, 本文根据文献 [34] 对两个距离进行定义, 如公式 (11) 和 (12) 所示:

$$\mathbb{G}_i = \sqrt{\sum\nolimits_{j=0}^{1} (u_{(i-1)j} - \mathbb{c}_j)^2} \Bigg/ \left(\sqrt{\sum\nolimits_{j=0}^{1} (u_{(i-1)j} - \mathbb{d}_j)^2} + \sqrt{\sum\nolimits_{j=0}^{1} (\mathbb{c}_j - \mathbb{d}_j)^2} \right) \tag{11}$$

© 中国科学院软件研究所 http://www.jos.org.cn

$$\mathcal{G}_i = \sqrt{\sum\nolimits_{j=0}^{1}\left(u_{(i-1)j} - \mathbb{d}_j\right)^2} \Big/ \left(\sqrt{\sum\nolimits_{j=0}^{1}\left(u_{(i-1)j} - \mathbb{c}_j\right)^2} + \sqrt{\sum\nolimits_{j=0}^{1}\left(\mathbb{c}_j - \mathbb{d}_j\right)^2}\right) \tag{12}$$

其中, $i \in \{1,2,\ldots,M\}$. 令 $\mathbb{H}_i$ 和 $\mathcal{H}_i$ 分别表示 $\boldsymbol{U}$ 中属性值的乐观距离和悲观距离, 其计算方法与公式 (11)、公式 (12) 相同. 为确定计算负载值时使用乐观距离还是悲观距离, 基于 $\mathcal{K}_M$ 和 $\mathcal{K}_N$ 选择乐观距离或悲观距离计算负载值, $\mathcal{K}_M$ 和 $\mathcal{K}_N$ 的计算如公式 (13) 所示:

$$\mathcal{K}_{\mathcal{E}} = \frac{1}{\mathcal{E}}\sum_{i=1}^{\mathcal{E}} l_i,\ \mathcal{E} \in \{M,N\} \tag{13}$$

其中, 当矩阵 $\mathcal{U}$ 和 $\boldsymbol{U}$ 中每一行的属性值与对应最优属性值之间的距离大于与对应最劣属性值之间的距离时, l_i 赋值 1, 反之赋值 0. 根据公式 (13) 中的 $\mathcal{K}_M$ 得到每个平行链的交易对局部分片造成的负载值 $\mathcal{F}_i$, 如公式 (14) 所示:

$$\mathcal{F}_i = \begin{cases} \mathbb{G}_i, & \text{if}\ \mathcal{K}_M \geqslant \lambda \\ \mathcal{G}_i, & \text{otherwise} \end{cases} \tag{14}$$

其中, λ 表示选择乐观或悲观距离的阈值, 其中 $i \in \{1,2,\ldots,M\}$. 因为平行链的交易对局部分片造成的负载值和剩余负载值都是独立计算出来的, 为在对应分配交易时有一个统一标准, 因此根据计算时共有的交易大小和数量属性对剩余负载值进行放缩, 根据公式 (13) 中得到的 $\mathcal{K}_N$ 计算每个局部分片的剩余负载值 F_i, 如公式 (15) 所示:

$$F_i = \begin{cases} \dfrac{\mathbb{c}_2 - \mathbb{d}_2 + \mathbb{c}_3 - \mathbb{d}_3}{\mathbb{c}_0 - \mathbb{d}_0 + \mathbb{c}_1 - \mathbb{d}_1}\mathbb{H}_i, & \text{if}\ \mathcal{K}_N \geqslant \lambda \\ \dfrac{\mathbb{c}_2 - \mathbb{d}_2 + \mathbb{c}_3 - \mathbb{d}_3}{\mathbb{c}_0 - \mathbb{d}_0 + \mathbb{c}_1 - \mathbb{d}_1}\mathcal{H}_i, & \text{otherwise} \end{cases} \tag{15}$$

其中, $i \in \{1,2,\ldots,M\}$. 计算出负载值之后, 本文基于负载值对平行链跨链交易进行分配, 将平行链跨链交易分配给特定的局部分片进行处理.

2.3.2 平行链跨链交易分配

本文在对平行链跨链交易进行分配时需要将 M 个平行链产生的交易分配给 N 个局部分片, 每个平行链的跨链交易只能分配给一个局部分片, 且方案不仅要考虑平行链对局部分片产生的负载值, 也要考虑局部分片的剩余负载值, 因此需要一种双向的分配方法将平行链跨链交易分配给合适的局部分片, 使每个局部分片的最终负载值处于均衡状态, 即局部分片的负载与性能相互匹配. 局部分片性能满足负载需求对于局部分片十分重要, 可以减少资源浪费, 降低交易的处理延迟, 增加跨链交易模型的可扩展性并保证交易的顺序公平性[35].

本文提出的自适应交易分配方法是动态分配方法, 全局分片会根据不同负载值分配交易, 平行链交易不会一直交给同一个局部分片进行处理. 动态的方法优于静态方法, 动态方法可以根据当前的平行链和局部分片状态对平行链跨链交易进行分配[36]. 本文在设计分配方法时需要考虑分配平行链跨链交易后每个局部分片负载值的情况, 因此本文的分配问题可以转化为最优解问题进行解决, 启发式方法是一种流行的方法, 也被广泛认为是解决该类问题的有前途方法之一, 本文基于启发式方法设计平行链跨链交易分配方法, 以最小化局部分片之间的负载方差, 使局部分片之间的负载值达到均衡状态为目标, 找到分配方案中的最优解. 本文在计算负载方差之前先计算负载平均值, 由于需要在多个分配方案中找到最优解, 令 $\bar{F}_j$ 表示第 j 个分配方案的负载平均值, 如公式 (16) 所示:

$$\bar{F}_j = \frac{\sum\nolimits_{r=1}^{M}\mathcal{F}_r + \sum\nolimits_{r=1}^{N}F_r}{N} \tag{16}$$

本文根据 $\bar{F}_j$ 计算第 j 个分配方案中局部分片之间的负载方差 V_j, 由于一个局部分片可能管理多个平行链, 计算局部分片的负载方差时需要避免将局部分片的剩余负载计算多次, 如公式 (17) 所示:

$$V_j = \frac{\sum\nolimits_{r=1}^{N}\left[\left(\sum\nolimits_{k=1}^{M}\mathcal{F}_k \times x_r^k\right) + F_r - \bar{F}_j\right]^2}{N} \tag{17}$$

其中, x_r^k 用于判断平行链 B_k 的交易是否分配给局部分片 r, 通过对比 q_{jk} 和 r 来判断. 当 $q_{jk} = r$ 时, $x_r^k = 1$, 表示平行

© 中国科学院软件研究所 http://www.jos.org.cn

链 B_k 的交易分配给局部分片 r; 当 $q_{ij} \neq r$ 时, $x_r^k = 0$, 表示平行链 B_k 的交易未分配给局部分片 r. 计算出每个个体的负载方差后, 选择负载方差最小的分配方案作为最终解.

2.3.3 跨分片交易分配

中继链分片之间的跨分片交易会影响分片技术对中继链系统可扩展性的提高, 因此分析中继链分片之间跨分片交易产生的原因并有针对性地进行跨分片交易分配具有必要性. 一般区块链分片在处理交易时, 由于账户分配在不同的分片, 交易需要在不同分片之间进行转发导致产生跨分片交易. 与一般区块链不同, 本文中继链局部分片处理跨链交易时, 一个跨链交易从一个平行链中账户转发至另一个平行链中账户时只经过一个局部分片. 虽然跨链交易的转发过程不涉及多个局部分片, 但是, 处理跨链交易的过程中局部分片和平行链需要产生多笔基于跨链交易的子交易, 即用于处理跨链交易的交易, 部分子交易可能成为跨分片交易. 跨分片交易的存在会增加局部分片之间的通信成本, 增加跨链交易的等待时间, 降低多个局部分片并行处理跨链交易的效率, 增加局部分片间负载不均衡的概率, 限制局部分片对中继链交易吞吐量的提高. 因此, 本文分析模型中产生跨分片交易的原因, 设计跨分片交易分配方法, 令子交易与对应跨链交易分配至同一个局部分片, 对中继链交易吞吐量的提高具有重要意义. 跨分片交易由交易依赖性产生, 中继链在处理跨链交易的过程中产生注册交易、记录交易和完成交易这 3 个交易[22], 这 3 个交易之间存在依赖关系, 记录交易的有效性依赖于注册交易的正确性, 完成交易的有效性依赖于记录交易的正确性. 同时, 记录交易的产生依赖于平行链发送的确认交易, 完成交易的产生依赖于目标链发送的确认交易.

以平行链 B_1 转发给平行链 B_{m+1} 的跨链交易 $\mathbb{T}$ 为例, 其中, 平行链 B_1 中的跨链交易分配给中继链局部分片 C_1 处理, 该笔跨链交易 $\mathbb{T}$ 的子交易依赖路径图如图 4 所示. 图 4 中, 局部分片 C_1 的收集节点接收到跨链交易 $\mathbb{T}$ 后产生注册交易即子交易 1, 并将其转发给平行链 B_{m+1}. 平行链 B_1 完成对交易的处理后产生确认交易即子交易 2 转发给中继链, 并由中继链收集节点转发给局部分片 C_1 的收集节点. 局部分片 C_1 中节点对平行链 B_1 中收集节点生成的子交易 2 验证成功后产生记录交易即子交易 3. 平行链 B_{m+1} 的收集节点根据跨链交易 $\mathbb{T}$ 和处理结果产生确认交易即子交易 4, 并将其最终转发给局部分片 C_1. 当局部分片 C_1 对平行链 B_{m+1} 中收集节点生成的子交易 4 验证成功后产生完成交易即子交易 5, 跨链交易 $\mathbb{T}$ 处理完成. 由于平行链的跨链交易根据分配方案交给不同的局部分片进行处理, 因此, 平行链 B_1 中收集节点生成的子交易 2 和平行链 B_{m+1} 中收集节点生成的子交易 4 可能由于分配方案的变化被中继链收集节点转发给局部分片 C_1 以外的其他局部分片进行处理, 导致局部分片 C_1 中节点跨分片验证子交易 2 和子交易 4.

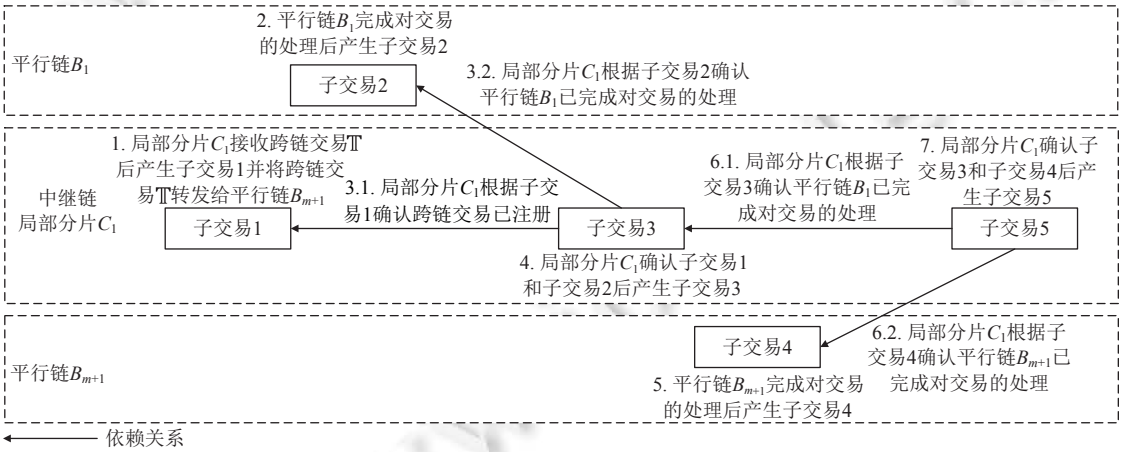


图 4 子交易依赖路径图

为减少中继链局部分片中的跨分片交易, 本文针对由于确认交易产生的跨分片交易设计跨分片交易分配方法. 每个局部分片都对应唯一的局部分片编号, 每个跨链交易都对应唯一的索引, 为区分跨链交易及其子交易, 中继链和平行链接收到跨链交易后生成的子交易的索引均以跨链交易的索引为基础形成. 在跨链交易的索引后添加后缀形成子交易的索引, 平行链收集节点和分片收集节点在分配交易时根据交易的索引确定对应局部分片并进入

© 中国科学院软件研究所 http://www.jos.org.cn

对应局部分片的交易队列中. 子交易的索引由下划线、交易的分类和对应局部分片的编号形成, 下划线可以用于判断交易是否为子交易, 定义子交易的索引 $\mathbb{R}$ 的结构为 $\mathbb{R}=\mathbb{K}_\mathbb{Y}_\mathbb{X}$. 其中, $\mathbb{K}$ 为父交易的索引; $\mathbb{Y}$ 为处理父交易的局部分片的编号, 可以用于快速确定所分配的局部分片; $\mathbb{X}$ 为子交易的类型, 使用数字表示. 注册交易、源链的确认交易、记录交易、目标链的确认交易和完成交易对应的类别编号分别为 1–5. 为确保交易索引的命名规则规范可行, 当一笔跨链交易生成时, 其索引中不得包含下划线, 同时, 当平行链和局部分片的收集节点根据一笔跨链交易生成子交易的索引时均需要满足子交易的索引结构. 基于子交易的索引定义子交易的结构, $\mathbb{P}=<\mathbb{R},\mathbb{U},\mathbb{V},\mathbb{Z}>$. 其中, $\mathbb{U}$ 为发送子交易的平行链或局部分片编号, $\mathbb{V}$ 为子交易内容, $\mathbb{Z}$ 为子交易生成时间. 本文设计跨分片交易分配方法使跨链交易及其子交易交给同一个局部分片处理, 其流程如图 5 所示, 关于该流程的详细说明如下.

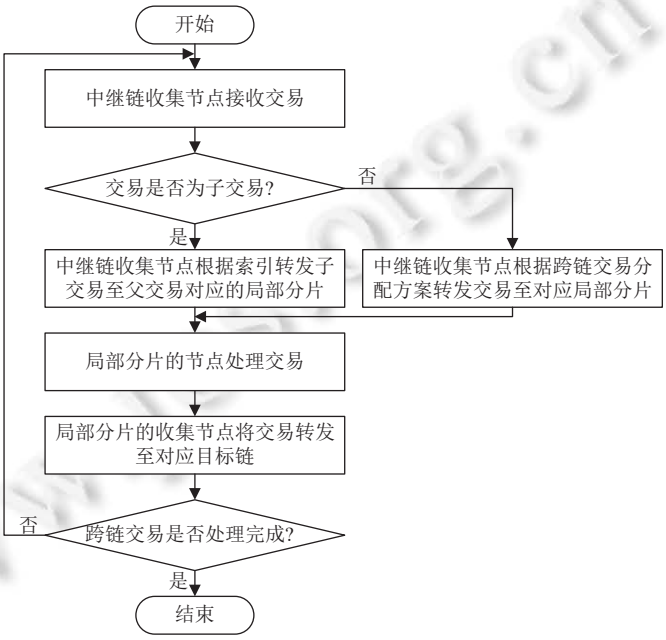


图 5 跨分片交易分配方法的流程图

1) 中继链收集节点接收来自平行链的交易.

2) 中继链收集节点判断接收的交易是否为子交易. 如果是索引添加后缀的交易, 即子交易, 则被交给处理对应跨链交易的局部分片进行处理, 处理子交易的局部分片编号与原跨链交易一致; 如果是索引未添加后缀的交易, 则根据平行链跨链交易分配方案交给对应的局部分片进行处理.

3) 局部分片的收集节点将交易转发给共识节点进行共识并发布共识结果, 收集节点收集共识结果并判断是否达成共识, 如果达成共识则将交易转发至对应的目标链. 如果目标链收集节点接收到交易完成消息即可确认跨链交易完成, 则该过程结束, 否则, 向中继链返回确认交易并重复上述步骤.

2.3.4 交易队列稳定性分析

本文将平行链跨链交易分配给合适的局部分片前, 全局分片对平行链产生的跨链交易信息进行预测, 对平行链跨链交易进行分配, 实现交易分配预测. 中继链收集节点接收到最新的分配方案后, 根据最新分配方案对交易包进行转发, 加入对应局部分片的交易队列中. 由于可能有部分用户在短时间内产生大量跨链交易进而对单个分片发动洪泛攻击, 降低单个分片对交易的处理效率. 在本文的模型中, 交易包以队列的形式进入局部分片的收集节点, 中继链分片网络可以被视为一个多队列系统. 洪泛攻击的发生将导致用户提交的跨链交易拥塞在队列中, 破坏队列稳定性, 即队列中积压的交易一直在增加, 分片负载超出局部分片的处理能力, 增加交易的处理延迟. 因此, 中继链将跨链交易分配给合适的局部分片之后仍需进行交易队列稳定性分析, 对不稳定队列中的跨链交易进行细粒

© 中国科学院软件研究所 http://www.jos.org.cn

度微调.

中继链收集节点根据负载值将平行链跨链交易分配给合适的局部分片后, 局部分片的收集节点进行交易队列稳定性分析, 对不稳定队列中的交易进行片间微调. 为保证每个局部分片中队列的稳定状态, 在交易等待的过程中, 使用 Lyapunov 理论[37]分析局部分片中交易队列的稳定性, 动态微调不稳定交易队列中的交易以缓解局部分片中出现的拥塞情况, 进一步维护局部分片负载与性能相匹配的状态. 根据 Lyapunov 理论分析交易队列稳定条件, 局部分片处理交易时, 局部分片收集节点在每个时隙开始时进行交易队列稳定性分析. 使用 $\mathbb{Q}_i^v$ 表示局部分片 C_i 在时隙 v 开始时的交易队列长度. 因此, 交易队列长度的动态变化如公式 (18) 所示:

$$\mathbb{Q}_i^{v+1} = \max\{\mathbb{Q}_i^v - \mathbb{b}_i^v + \mathbb{a}_i^v, 0\} \tag{18}$$

其中, $\mathbb{a}_i^v$ 表示在时隙 v 内到达局部分片 C_i 并在交易队列中等待的交易数量, $\mathbb{b}_i^v$ 表示在时隙 v 内被局部分片 C_i 处理的交易数量. 其中, $\mathbb{Q}_i^0 = 0$, $\mathbb{b}_i^v = 0$, $\mathbb{a}_i^v = 0$. 定义在时隙 v 内的交易队列积压向量为 $\mathbb{Q}^v$, 如公式 (19) 所示:

$$\mathbb{Q}^v = [\mathbb{Q}_1^v, \mathbb{Q}_2^v, \ldots, \mathbb{Q}_N^v] \tag{19}$$

为度量时隙 v 内中继链网络总队列积压的标量大小, 定义二次 Lyapunov 函数 $\mathbb{L}^v$ 如公式 (20) 所示:

$$\mathbb{L}^v = \frac{1}{2}\sum_{i=1}^{N} \mathbb{Q}_i^{v2} \tag{20}$$

Lyapunov 漂移 (Lyapunov drift) 为二次 Lyapunov 函数从一个时隙到另外一个时隙的变化, 可以用于表示在各个离散时间下 Lyapunov 函数的变化趋势, 是分析交易队列稳定性的工具. 定义 Lyapunov 漂移为 $\Delta\mathbb{L}^v$, $\Delta\mathbb{L}^v = \mathbb{L}^{v+1} - \mathbb{L}^v$. 结合公式 (18) 可以推出如公式 (21) 所示的不等关系:

$$\Delta\mathbb{L}^v \leqslant \frac{1}{2}\sum_{i=1}^{N}(\mathbb{a}_i^v - \mathbb{b}_i^v)^2 + \sum_{i=1}^{N}\mathbb{Q}_i^v(\mathbb{a}_i^v - \mathbb{b}_i^v) \tag{21}$$

根据 Lyapunov 定理, 队列稳定需要满足 $E(\Delta\mathbb{L}^v|\mathbb{Q}^v) \leqslant 0$. 对局部分片中交易队列的稳定性进行分析后, 本文基于分析结果确定是否进行队列交易细粒度微调. 如果中继链局部分片中交易队列不满足稳定条件, 说明存在局部分片性能与交易队列中积压交易不匹配的情况, 破坏中继链局部分片中交易队列稳定性. 为解决中继链中交易队列不稳定的问题, 本文对局部分片中积压的交易进行局部分片间微调, 减小局部分片处理交易的延迟. 首先, 确定不稳定的交易队列, 根据 Lyapunov 判断队列是否处于稳定状态; 其次, 确定微调的交易数量, 令 $\Delta\mathbb{A}_i^{v+1}$ 表示局部分片 C_i 在时隙 v+1 开始时需要微调的交易数量, 即 $\Delta\mathbb{A}_i^{v+1} = \mathbb{Q}_i^{v+1}$. 由于子交易微调至其他局部分片后仍需交给处理对应跨链交易的局部分片进行处理, 增加跨链交易处理的延迟, 因此, 子交易不进行局部分片间微调. 得到需要微调的交易数量后, 将交易微调至合适的局部分片. 将 $\mathbb{a}_i^v - \mathbb{b}_i^v < 0$ 的局部分片 C_k 作为目标分片对不稳定队列的交易进行微调. 以有 10 个平行链和 8 个局部分片的中继链为例, 假设平行链跨链交易的信息和分片的信息如表 2 和表 3 所示.

表 2 平行链交易信息

分片指标	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8	B_9	B_{10}
交易数量	900	1 100	700	800	1 200	950	650	700	700	650
交易大小 (KB)	230	275	180	200	300	240	160	180	175	160

表 3 分片信息

分片指标	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
剩余交易数量	300	250	200	400	300	400	100	500
剩余交易大小 (KB)	100	60	50	50	75	100	25	125
交易吞吐量 (tx/s)	1 000	1 300	1 300	900	1 100	1 200	1 400	800

根据表 2 和表 3 中的数据计算平行链跨链交易对局部分片产生的负载和局部分片的剩余负载. 由于平行链的数量比局部分片的数量多, 所以部分局部分片可能分配到多个平行链的交易. 经过平行链跨链交易分配算法的多次迭代, 得到适应度值最小的方案作为平行链跨链交易分配方案, 即 $\{B_1 \to C_1, B_2 \to C_6, B_3 \to C_7, B_4 \to C_4, B_5 \to C_2, B_6 \to C_5, B_7 \to C_3, B_8 \to C_8, B_9 \to C_7, B_{10} \to C_3\}$. 以该平行链跨链交易分配方案为例, 若交易队列长度集合为

© 中国科学院软件研究所 http://www.jos.org.cn

{400, 250, 200, 200, 300, 400, 100, 500}, 到达局部分片并在交易队列中等待的交易数量集合为{900, 1 200, 1 300, 800, 950, 1 100, 1 400, 700}, 被局部分片处理的交易数量集合为{1 000, 1 300, 1 300, 900, 1 100, 1 200, 1 400, 800}, 计算出 Lyapunov 漂移的集合为{−35 000, −25 000, 0, −15 000, −33 740, −35 000, −45 000}, 此时中继链系统整体的 Lyapunov 漂移为负, 处于稳定状态且各个交易队列也处于稳定状态. 如果交易队列中等待的交易数量集合的最后一个元素为 1 100 而非 700, 此时局部分片 C_8 的 Lyapunov 漂移为 195 000, 中继链整体系统处于不稳定状态且该状态由局部分片 C_8 导致, 则需要对局部分片 C_8 中的交易队列进行调整.

在对不稳定队列中的交易进行细粒度微调时, 本文需要将不稳定交易队列中的交易打包并迁移到其他局部分片的交易队列中, 此过程会产生额外的成本. 首先, 当前局部分片的收集节点需要对交易进行打包并与其他局部分片的收集节点交互以迁移交易, 导致当前局部分片的共识延迟增加; 其次, 交易迁移涉及不同分片之间的通信, 导致引入额外的网络开销, 增加通信成本. 为在一次交易队列稳定性分析后评估交易微调的代价, 本文设计交易微调的代价评估模型, 其可分为共识中断成本和中继通信成本两个部分. 本文定义一次交易微调需要的共识中断成本为 $\mathbb{W}=\mathbb{g}+\sum_{i=1}^{\mathbb{h}}\mathbb{m}_i$. 其中, $\mathbb{g}$ 表示一次交易微调导致的共识机制无法正常工作的时间, $\mathbb{m}_i$ 表示中继链系统已经用于处理第 i 笔丢失交易的时间, $\mathbb{h}$ 表示由于一次交易微调导致的丢失交易数量. 在半同步网络中, 每笔交易的处理时间都需要满足最大延迟. 如果交易的处理时间超过最大延迟, 则该交易被拒绝继续处理. 由于局部分片的收集节点处理进行交易微调时, 被微调的交易和等待继续被处理的交易的处理延迟均会增加, 可能直接导致交易处理失败. 本文定义一次交易微调中继通信成本为 $\mathbb{J}=\mathbb{e}+\mathbb{f}/\mathbb{k}$. 其中, $\mathbb{e}$ 表示局部分片的收集节点之间通信的延迟之和, $\mathbb{f}$ 表示需要微调的总的交易包大小, $\mathbb{k}$ 表示节点带宽, $\mathbb{f}/\mathbb{k}$ 表示收集节点从交易队列中取出交易包并将其发送到区块链网络的传输延迟. 本文通过通信延迟和传输延迟来衡量中继通信成本, 来反映网络延迟和带宽限制对交易微调的影响. 根据共识中断成本和中继通信成本, 交易微调的总成本为 $\mathbb{W}+\mathbb{J}$.

3 正确性分析

本文假设区块链网络为半同步网络, 跨链交易需要在最大延迟内被处理完成. 基于该假设和所提的两阶段自适应交易分配模型, 本文对该模型的安全性和活性进行分析.

3.1 安全性分析

(1) 中继链系统正常运行概率

本文中平行链与中继链使用的共识机制可能不同, 为了保证跨链交易能被正常处理, 平行链和中继链中恶意节点比例需要同时小于对应共识机制的容错比例. 本文定义中继链系统能够正常运行的概率为 $\mathbb{N}$, 其需要满足的条件如公式 (22) 所示:

$$\mathbb{N}\geqslant(1-\theta)^N\times(1-\varpi)^N\times(1-\vartheta)\times\prod_{i=1}^{M}(1-\eta_i)\times(1-\chi_i) \tag{22}$$

其中, θ 为局部分片的收集节点可以容忍的恶意节点比例, ϖ 为局部分片的共识节点可以容忍的恶意节点比例, N 为局部分片的数量, ϑ 为中继链收集节点可以容忍的恶意节点比例, M 为平行链的数量, η_i 为第 i 个平行链中收集节点可以容忍的恶意节点比例, χ_i 为第 i 个平行链中共识节点可以容忍的恶意节点比例. 本文在恶意节点比例未超过容错比例时, 设计相应方法以降低恶意收集节点对跨链交易处理过程的影响.

(2) 中继链系统安全风险分析

区块链分片可以显著提升交易吞吐量, 但是区块链系统同样面临网络安全风险[38]. 在跨链交易处理的过程中恶意节点可能发动拒绝服务攻击, 导致跨链交易长时间等待被处理. 同时, 由于子交易的处理依赖于索引完成, 恶意节点篡改子交易的索引会导致子交易冲突, 进而导致跨链交易处理失败, 具体分析如下.

1) 平行链或局部分片中恶意收集节点接收到跨链交易时, 拒绝处理该交易, 导致该交易处理过程中止, 并长时间等待, 直至处理时间超过最大延迟. 本文中平行链和局部分片中存在收集节点集, 跨链交易被转发给收集节点集中的每个收集节点, 并由选举出的一个收集节点对跨链交易进行验证和转发. 首先, 如果平行链中收集节点接收到跨链交易后拒绝将其打包并转发给中继链收集节点, 其他收集节点在该跨链交易处理超时且未收到任何反馈时,

© 中国科学院软件研究所 http://www.jos.org.cn

生成重新选举交易以替换恶意收集节点; 其次, 如果平行链中收集节点拒绝将跨链交易的处理结果返回给用户, 用户在该跨链交易处理超时且未收到任何反馈时, 将此情况以交易的形式反馈给该平行链中所有收集节点, 由其他收集节点验证并判断是否重新选举; 最后, 如果局部分片中收集节点拒绝处理跨链交易或拒绝将跨链交易的处理结果反馈给中继链收集节点, 中继链收集节点在该跨链交易处理超时且未收到任何反馈时生成重新选举交易, 以替换恶意收集节点.

2) 平行链和局部分片中恶意收集节点可能生成索引相互冲突的子交易. 本文假设父交易的索引作为其唯一标识不会重复, 子交易索引生成时可能由于节点的恶意行为导致子交易索引的后缀产生冲突. 因此, 设计子交易冲突处理机制解决该问题, 子交易冲突处理机制分以下 3 种冲突类型来说明.

a) 子交易索引中 $\mathbb{K}$ 相同, $\mathbb{Y}$ 不同, $\mathbb{X}$ 不同. 当中继链收集节点接收到相互冲突的子交易时检查 $\mathbb{U}$ 和 $\mathbb{X}$, 该子交易只能为平行链发送的确认交易. 因此, 其接收到的冲突子交易为不同平行链发送的确认交易. 如果 $\mathbb{X}$ 为 1、3 或 5, 则该子交易存在问题. 反之, 当 $\mathbb{U}$ 为平行链编号时, 中继链收集节点根据 $\mathbb{X}=2$ 的子交易中 $\mathbb{U}$ 查找平行链跨链交易分配方案, $\mathbb{Y}$ 与分配方案一致的子交易正确.

b) 子交易索引中 $\mathbb{K}$ 相同, $\mathbb{Y}$ 不同, $\mathbb{X}$ 相同. 当子交易中 $\mathbb{U}$ 为平行链编号, 首先, 中继链收集节点根据 $\mathbb{U}$ 和 $\mathbb{K}$ 区分源链和目标链, 与源链和目标链确认交易编号不一致的子交易错误; 其次, 根据分配方案判断子交易的正确性. 当局部分片的收集节点接收到相互冲突的子交易时, 如果 $\mathbb{X}$ 不同, 则 $\mathbb{Y}$ 与局部分片编号一致的子交易正确, 否则局部分片的收集节点判断子交易均错误.

c) 子交易索引中 $\mathbb{K}$ 相同, $\mathbb{Y}$ 相同, $\mathbb{X}$ 相同. 当中继链收集节点接收到相互冲突的子交易时根据 $\mathbb{U}$ 和 $\mathbb{K}$ 区分源链和目标链, 与源链和目标链确认交易编号不一致的子交易错误. 当局部分片的收集节点接收到相互冲突的子交易时, 判断子交易均错误. 这些节点判断出错误子交易后, 则返回出错信息, 并停止处理依赖于该子交易的其他交易.

3.2 活性分析

本文提出的两阶段自适应交易分配模型是基于半同步网络假设进行的, 以保障所有跨链交易最终均能被处理完成. 区块链的活性是指所有的交易都会在有限的时间内被接受或终止, 保证了处理的及时性[39]. 基于该定义, 本文区块链活性为: 当跨链交易可被正常处理, 其处理结果被正确反馈且平行链与中继链节点最终生成的区块包含该交易. 本文中跨链交易处理涵盖共识达成与动态微调两个阶段. 为保障交易最终被写入区块 (即确保中继链活性), 从共识活性与交易微调的区块链活性两个维度展开分析.

(1) 共识活性

共识活性指恶意共识节点比例不超过共识机制的容错比例时实现半同步网络的活性[5], 可以分成正常场景下共识活性分析和异常场景处理两种情况进行分析.

1) 正常场景下共识活性分析

如上述公式 (22) 所示, 当恶意共识节点和收集节点比例均小于对应共识机制的容错比例, 且跨链交易由诚实节点处理时, 首先, 一笔跨链交易可以被平行链中共识节点正确共识, 共识节点将共识结果转发给平行链的收集节点, 其在收集共识结果后可以正确判断是否达成共识. 其次, 平行链的收集节点将跨链交易转发给中继链收集节点后, 其根据交易分配方案正确地将跨链交易转发至对应的局部分片中. 最后, 局部分片的收集节点根据接收到的共识结果正确判断是否达成共识, 并将跨链交易转发给对应的目标平行链. 因此, 收集节点每次可以正确判断共识是否达成, 跨链交易的共识可以被正确执行, 保证共识结果的正确性.

2) 异常场景处理

a) 平行链崩溃时需从中继链系统中退出. 当平行链中恶意共识节点比例超过容错比例, 共识节点对跨链交易共识并将共识结果转发给平行链的收集节点, 其收集到的共识结果数量不足以判断是否达成共识, 导致共识过程长时间未停止, 直至跨链交易的等待时间超过最大延迟, 跨链交易处理失败. 由于平行链对跨链交易的共识过程无法完成, 且跨链交易无法被转发至中继链处理. 因此, 该平行链需要从中继链系统中退出.

b) 中继链中某局部分片崩溃需要被重新构建. 当局部分片中恶意共识节点比例超过容错比例, 共识节点对跨链交易共识并将共识结果转发给对应的收集节点, 其收集到的共识结果数量不足以判断是否达成共识, 导致共识

© 中国科学院软件研究所 http://www.jos.org.cn

过程长时间执行, 直至跨链交易的等待时间超过最大延迟, 跨链交易处理失败. 由于跨链交易的共识过程无法完成, 局部分片无法继续处理跨链交易. 因此, 崩溃的局部分片需要被重新构建以保证共识过程能够正确执行并终止.

c) 恶意收集节点需要被替换. 当跨链交易在被处理过程中遇到恶意收集节点, 且平行链和中继链中恶意节点比例小于对应共识机制的容错比例时, 恶意收集节点在接收共识结果时不继续完成共识过程, 导致共识过程无法终止. 在该情况下, 新的收集节点由重新选举得到. 由于恶意收集节点比例小于容错比例, 因此, 重新选举收集节点的过程可以被正确执行, 以保证被选举出的收集节点为诚实节点. 重新选举得到的诚实收集节点继续处理该跨链交易.

(2) 交易微调的区块链活性

交易微调的区块链活性指每次交易微调都会按照成功或失败的处理流程进行, 保证交易微调可以在有限的时间内终止[40], 可以分成正常场景下交易微调的区块链活性分析和异常场景处理进行分析.

1) 正常场景下交易微调的区块链活性分析: 当跨链交易被诚实节点处理, 每次跨链交易微调最终都会按照成功的处理流程或失败的处理流程进行, 则本文中继链系统可以保证微调活性, 进而实现半同步网络的活性. 本文中微调跨链交易由源局部分片打包为微调跨链交易包发送给目标局部分片, 如果微调跨链交易包的处理未超时且被目标局部分片验证成功, 只要目标局部分片没有被恶意节点破坏, 收集节点就会反馈微调成功的消息给所有源局部分片.

2) 异常场景处理: 如果目标局部分片验证微调跨链交易包时已经超时, 其不接受该微调跨链交易包; 如果源局部分片在最大延迟内没有收到任何反馈消息, 且源局部分片没有被恶意节点破坏, 则其确认微调失败并优先处理微调失败的跨链交易.

4 算法设计

基于所提的两阶段自适应交易分配模型, 本文设计分片负载值计算算法 (算法 1)、平行链跨链交易分配算法 (算法 2) 和交易队列稳定性分析算法 (算法 3), 算法流程如图 6 所示.

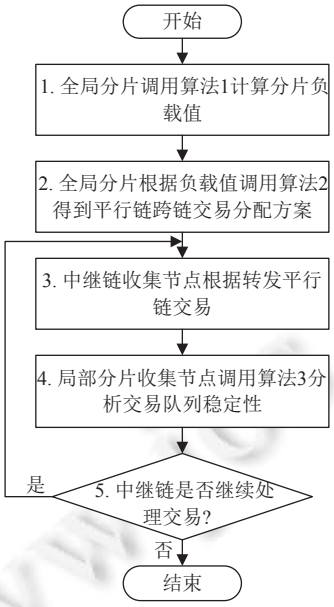


图 6 算法流程图

图 6 中被调用的 3 个算法可以解决跨链交易分配过程中存在的问题, 首先, 中继链分片的负载计算存在忽略交易大小和分片性能的问题, 算法 1 全面考虑交易大小、交易数量和吞吐量, 用于计算分片负载值; 其次, 基于算法 1 的结果, 为满足局部分片的负载与性能相互匹配的需求, 保证局部分片的负载值处于均衡状态, 本文设计算

© 中国科学院软件研究所 http://www.jos.org.cn

法 2 得到平行链跨链交易分配方案; 最后, 中继链收集节点根据算法 2 输出的结果将交易分配给对应局部分片后, 局部分片可能出现交易激增的问题, 为解决该问题, 局部分片收集节点调用算法 3 进行交易队列稳定性分析. 算法流程的说明如下.

步骤 1: 全局分片调用算法 1, 根据从中继链收集节点收集的平行链历史跨链交易信息对平行链即将产生的跨链交易数量和大小进行预测, 以预测结果作为计算负载值的数据, 利用算法 1 计算平行链跨链交易对局部分片产生的负载值; 全局分片根据从局部分片收集的局部分片信息计算局部分片的剩余负载值.

步骤 2: 全局分片调用算法 2, 以算法 1 的结果作为输入, 算法 2 的目标为最小化局部分片之间的负载方差, 选择负载方差最小的方案作为平行链跨链交易分配的最优方案.

步骤 3: 全局分片将分配方案转发至中继链收集节点, 中继链收集节点根据算法 2 的结果将平行链跨链交易转发至对应的局部分片. 中继链收集节点将平行链产生的历史跨链交易信息转发至全局分片作为算法 1 的输入.

步骤 4, 5: 局部分片的收集节点调用算法 3 进行交易队列稳定性分析, 并对不稳定队列中的交易进行片间微调. 在中继链处于工作状态时, 将一直完成步骤 3, 否则, 整个跨链交易的处理过程结束.

4.1 负载值计算算法

本文在计算平行链跨链交易对局部分片产生的负载值前利用预测算法对平行链产生的交易信息进行预测, 目前, 一些预测算法已经被应用在区块链系统中, 如预测加密货币价格[41,42]、交易数量[43]、用户声誉[44]以及区块链可靠性[45]等, 其中, LSTM (long short-term memory) 算法可有效对长时间序列进行建模, 对短期和长期时间依赖性进行捕捉, 具有鲁棒性和灵活性, 是目前时间序列预测问题最常用的方法[43]. 因此, 本文在算法 1 中使用 LSTM 算法对平行链跨链交易数量和大小进行预测, 根据平行链跨链交易的预测结果, 计算平行链跨链交易对局部分片的负载值.

算法 1. Load value calculation.

输入: $w_j, \varepsilon, \eta, \theta, \xi, \mathbb{R}$; /* w_j、ε、η、θ 和 ξ 分别表示属性权重、输入维度、隐层维度、输出维度和网络层数. $\mathbb{R}$ 表示存储历史交易信息的数组 */

输出: $\mathcal{F}$. /* $\mathcal{F}$ 表示存储所有分片负载值的数组 */

1. double $\boldsymbol{Y}$[] = NULL, $\boldsymbol{Z}$[] = NULL; /* $\boldsymbol{Y}$ and $\boldsymbol{Z}$ represent the arrays that hold attribute values, respectively */
2. double $\mathcal{A}$ [][] = $LSTM$ ($\varepsilon, \eta, \theta, \mu, \mathbb{R}$), $\mathcal{U}$ [][] = NULL; /* $\mathcal{A}$ and $\mathcal{U}$ represent the transaction information array and the
3. weighted array respectively */
4. **For** (int $j = 0; j < N; j$++) /* N represents the number of columns of array $\mathcal{A}$ */
5. $\quad \mathcal{A}[i][j] \leftarrow \mathcal{A}[i][j] \Big/ \sqrt{\sum_{i=0}^{M-1}(\mathcal{A}[i][j]\times\mathcal{A}[i][j])}$; /* M represents the number of rows of array $\mathcal{A}$ */
6. **EndFor**
7. double $\mathbb{G}$ [] = NULL, $\mathcal{G}$ [] = NULL; /* $\mathbb{G}$ and $\mathcal{G}$ represent the optimistic distance and the pessimistic distance array, respectively */
8. **For** (int $i = 0; i < M; i$++)
9. $\quad$ **For** (int $j = 0; j < N; j$++)
10. $\quad\quad \mathcal{U}[i][j] \leftarrow w_j \times \mathcal{A}[i][j]$;
11. $\quad$ **EndFor**
12. **EndFor**
13. **For** (int $j = 0; j < N; j$++)
14. $\quad$ **For** (int $i = 0; i < M; i$++)
15. $\quad\quad \boldsymbol{Y}[j] \leftarrow \max(\mathcal{U}[i][j])$; $\boldsymbol{Z}[j] \leftarrow \min(\mathcal{U}[i][j])$; /* Solve the PIS via equation (11) and solve the NIS via equation (12) */

© 中国科学院软件研究所 http://www.jos.org.cn

16. **EndFor**
17. **EndFor**
18. **For** (int $i = 0$; $i < M$; i++)
19. $\mathbb{G}[i] \leftarrow \sqrt{\sum_{j=0}^{N-1}(\mathcal{U}[i][j]-\boldsymbol{Y}[j])^2} \Big/ \left(\sqrt{\sum_{j=0}^{N-1}(\mathcal{U}[i][j]-\boldsymbol{Z}[j])^2}+\sqrt{\sum_{j=0}^{N-1}(\boldsymbol{Y}[j]-\boldsymbol{Z}[j])^2}\right)$;
20. $\mathcal{G}[i] \leftarrow \sqrt{\sum_{j=0}^{N-1}(\mathcal{U}[i][j]-\boldsymbol{Z}[j])^2} \Big/ \left(\sqrt{\sum_{j=0}^{N-1}(\mathcal{U}[i][j]-\boldsymbol{Y}[j])^2}+\sqrt{\sum_{j=0}^{N-1}(\boldsymbol{Y}[j]-\boldsymbol{Z}[j])^2}\right)$;
21. **EndFor**
22. double $\mathcal{K}$[] = NULL; /* $\mathcal{K}$ represents the array that holds the criterion for choosing a distance equation */
23. **For** (int $i = 0$; $i < M$; i++)
24. int $\boldsymbol{R}$[] = NULL;
25. **For** (int $j = 0$; $i < N$; j++)
26. **If** ($\boldsymbol{Y}[j]-\mathcal{U}[i][j] > \mathcal{U}[i][j]-\boldsymbol{Z}[j]$)
27. $\boldsymbol{R}[j] \leftarrow 1$; /* $\boldsymbol{R}$ represents the array that holds whether the values of each row are closer to the PIS */
28. **Else**
29. $\boldsymbol{R}[j] \leftarrow 0$;
30. **EndIf**
31. $\mathcal{K}[i] \leftarrow \mathcal{K}[i] + \boldsymbol{R}[j]$;
32. **EndFor**
33. $\mathcal{K}[i] \leftarrow \mathcal{K}[i]/M$;
34. **EndFor**
35. double $\mathcal{F}$[] = NULL;
36. **For** (int $i = 0$; $i < M$; i++)
37. **If** ($\mathcal{K}[i] >= \lambda$) /* λ represents the threshold for selecting optimistic or pessimistic distances */
38. $\mathcal{F}[i] \leftarrow \mathbb{G}[i]$;
39. **Else**
40. $\mathcal{F}[i] \leftarrow \mathcal{G}[i]$;
41. **EndIf**
42. **EndFor**
43. **Return** $\mathcal{F}$;

算法 1 被用于计算平行链跨链交易对局部分片产生的负载值, 其相关说明如下.

1) 算法 1 旨在计算平行链跨链交易对局部分片产生的负载值, 针对该负载值选择平行链跨链交易大小和数量作为计算负载的属性, 基于该属性平行链跨链交易对局部分片造成的负载的多属性描述矩阵 $\mathcal{A}$, 将属性的权重以及平行链历史跨链交易信息作为输入.

2) 算法 1 使用矩阵中每一个元素与该元素所在列的元素的平方和相除得到标准化后的结果, 消除属性量纲对负载值计算的影响 (第 4–6 行).

3) 矩阵 $\mathcal{A}$ 中的所有数据全部是标准化后的极小型数据, 其行数和列数分别用 M 和 N 表示. 由于矩阵 $\mathcal{A}$ 中属性的权重不同, 算法 1 使用权重与矩阵 $\mathcal{A}$ 中对应属性值进行相乘获得加权后的矩阵 $\mathcal{U}$, 利用 w_j 表示第 j 个属性的权重值 (第 8–12 行).

4) 算法 1 取出矩阵 $\mathcal{A}$ 每一列中最大的数构成正理想解 (positive ideal solution, PIS), 取出每一列中最小的数构成负理想解 (negative ideal solution, NIS), 分别使用数组 $\boldsymbol{Y}$ 和 $\boldsymbol{Z}$ 存储正理想解和负理想解 (第 13–17 行).

5) 本文在算法 1 中使用欧几里得距离计算矩阵 $\mathcal{U}$ 中每行元素到 PIS 和 NIS 的乐观距离和悲观距离, 计算得

© 中国科学院软件研究所 http://www.jos.org.cn

到的乐观距离和悲观距离分别存储在数组 $\mathbb{G}$ 和 $\mathcal{G}$ 中 (第 18–21 行).

6) 算法 1 判断矩阵 $\mathcal{U}$ 中每行元素更接近对应最优属性值还是最劣属性值, 将判断结果存入数组 $\boldsymbol{R}$, 更接近最优属性值判断结果为 1, 否则为 0, 并使用数组 $\mathcal{K}$ 计算每行元素判断结果的均值 (第 22–34 行).

7) 通过公式 (14) 选择负载值的计算公式, 即选择使用哪种距离作为计算剩余负载值的标准, 乐观或悲观距离越大, 剩余负载值越大, 否则越小, 将负载值存储在数组 $\mathcal{F}$ 中并输出该数组 (第 36–43 行).

8) 该算法存在双层 for 循环, 循环的次数分别为 M 和 N, 时间复杂度为 O(MN).

4.2 平行链跨链交易分配算法

本文利用 SWO (spider wasp optimizer) 算法实现算法 2, 获得分配方案的最优解, SWO 算法适用于不同探索要求的优化问题, 可以避免局部最优问题[46]. 本文实现算法 2 时, 蜘蛛黄蜂以编码的形式记录在向量中, 每个蜘蛛黄蜂为一种平行链跨链交易分配方案, 本文模型中每个局部分片和平行链都有唯一的编号, 使用十进制编码方式, 向量的维度为平行链的数量, 令 $\boldsymbol{Q}_{hi}$ 代表第 h 代中第 i 种平行链跨链交易的分配方案, 如公式 (23) 所示:

$$\boldsymbol{Q}_{hi} = [q_{i1}, q_{i2}, \ldots, q_{iM}] \tag{23}$$

其中, q_{ij} 表示第 i 个雌性黄蜂个体选择的第 j 个蜘蛛, $j \in \{1, 2, \ldots, M\}$.

算法 2. Parachain cross-chain transactions allocation.

输入: ϕ, μ, P_c, P_t, $\boldsymbol{F}$, $\mathcal{F}$, x_j^r; /* ϕ 和 μ 分别表示种群大小和迭代次数. P_c 和 P_t 分别表示交叉率和权衡率. $\boldsymbol{F}$ 和 $\mathcal{F}$ 分别表示存储负载值的数组 (预测的平行链分片交易) 和分片剩余的负载值. x_j^r 表示平行链 B_{k+1} 的交易是否被分配到分片 C_{j+1} */

输出: $\boldsymbol{X}$. /* $\boldsymbol{X}$ 表示最佳分配方案 */

1. double $\boldsymbol{S}$[] = NULL, $\boldsymbol{V}$[] = NULL; /* $\boldsymbol{S}$ and $\boldsymbol{V}$ represent arrays that hold the average load and the load variance, respectively */
2. int $\boldsymbol{X}$[] = NULL, $\mathcal{Q}$[] = NULL; /* $\mathcal{Q}$ represents initial allocation plan of different individuals */
3. **For** (int $i = 0$; $i < \phi$; i++)
4. **For** (int $j = 0$; $j < M$; j++)
5. $\mathcal{Q}[i][j] \leftarrow 1 + rand(0, 1) \times (N - 1)$; /* N and M represent the number of shards and the number of parachains, respectively */
6. **EndFor**
7. **EndFor**
8. **For** (int $i = 0$; $i < \phi$; i++)
9. $\boldsymbol{S}[i] \leftarrow \dfrac{\sum_{j=0}^{M-1} \mathcal{F}[j] + \sum_{j=0}^{N-1} \boldsymbol{F}[j]}{N}$; /* Calculate average load via equation (16) */
10. $\boldsymbol{V}[i] \leftarrow \dfrac{\sum_{j=0}^{N-1} \left[\left(\sum_{k=0}^{M-1} \mathcal{F}[k] \times x_j^k\right) + \boldsymbol{F}[j] - \boldsymbol{S}[i]\right]^2}{N}$; /* Calculate load variance via equation (17) */
11. **EndFor**
12. $\boldsymbol{X} \leftarrow SWO(P_c, P_t, \phi, \mu, \mathcal{Q}, M, \boldsymbol{S}, \boldsymbol{V}, N, \mathcal{F})$; /* Call SWO algorithm to obtain the scheme when $\boldsymbol{V}[j]$ is minimum */
13. **Return** $\boldsymbol{X}$;

算法 2 随机生成多个蜘蛛黄蜂初始化种群, 计算每一个分配方案的负载方差, 迭代地将负载值高的交易动态分配给剩余负载值低的局部分片以改善局部分片负载, 其相关说明如下.

1) 根据算法 1 计算出的负载值初始化蜘蛛黄蜂种群, 算法 2 利用局部分片的编号, 采用随机的方法将局部分片的编号与平行链的编号对应, 分片和平行链的数量分别为 N 和 M, 对种群中每一个蜘蛛黄蜂进行初始化, 种群

© 中国科学院软件研究所 http://www.jos.org.cn

大小定义为 ϕ, 将分配方案映射为向量 $\mathcal{Q}$ 使每个方案转变为 SWO 算法中的表现形式 (第 3–7 行).

2) 适应度可以表明个体或解的优劣性. 对于不同的问题, 适应度的定义方式不同, 本文使用局部分片负载值之间的方差作为适应度, 计算每个个体的适应度时, 将负载均值存储在数组 $\boldsymbol{S}$ 中并将方差存储在数组 $\boldsymbol{V}$ 中. 适应度越小, 局部分片负载值之间的方差越小, 所有局部分片中负载与性能越匹配 (第 8–11 行).

3) 调用 SWO 算法选择每个种群中适应度最小的个体, 将该个体的基因存放在数组 $\boldsymbol{X}$ 中, 将 $\boldsymbol{X}$ 作为最终分配方案输出 (第 12、13 行).

4) 算法 2 实现的过程需要在最大迭代次数之内, 每一个蜘蛛黄蜂种群都进行适应度计算、狩猎、筑巢、交配操作, 算法 2 中存在 3 层循环, 分别是循环生成每个蜘蛛黄蜂, 循环生成每个种群和循环迭代, 时间复杂度为 $O(\phi\mu M)$.

4.3 交易队列稳定性分析算法

本文设计一种交易队列稳定性分析算法, 通过分析交易队列稳定性微调不稳定交易队列中的交易以缓解局部分片中出现的拥塞问题, 如算法 3 所示.

算法 3. Queue trading adjustment algorithm.

输入: $\mathbb{Q}$, $\mathbb{E}$, $\mathbb{F}$; /* $\mathbb{Q}$、$\mathbb{E}$ 和 $\mathbb{F}$ 分别表示存储队列长度的数组，交易到达率和交易处理率 */
输出: $\mathbb{A}$. /* $\mathbb{A}$ 表示需要存储被调整交易的数组 */

```
double 𝕃[] = NULL, L[] = NULL; /* 𝕃 and L represent the backlog size of queue at different time slots */
int 𝕊[] = NULL; /* 𝕊 represents the array that holds the changed queue length */
For (int i = 0; i < N; i++)
    𝕊[i] ← ℚ[i] − 𝔽[i] + 𝔼[i];
EndFor
𝕃[0] ← 0;
For (int i = 0; i < N; i++)
    𝕃[i] ← 𝕃[i] + 𝕊[i] × 𝕊[i];
    L[i] ← L[i] + ℚ[i] × ℚ[i];
EndFor
For (int i = 0; i < N; i++)
    𝕃[i] ← 𝕃[i]/2;
    L[i] ← L[i]/2;
EndFor
int 𝒱[] ← NULL; /* 𝒱 represents the array that holds evaluation results of queues */
For (int i = 0; i < N; i++)
    𝒱[i] ← 𝕃[i]–L[i];
EndFor
int 𝔸[] = NULL;
For (int i = 0; i < N; i++)
    If (𝒱[i] < 0)
        𝔸[i] ← 𝕊[i];
        int j = 0;
        While (j < 𝔸[i] && j < ℚ[i])
            If (A transaction is not a sub-transaction)
                Collect nodes add them to the transaction package that needs to be fine-tuned;
```

© 中国科学院软件研究所 http://www.jos.org.cn

```
27.            j++;
28.         EndIf
29.      EndWhile
30.    EndIf
31. EndFor
32. Return 𝔸;
```

算法 3 的相关说明如下.

1) 局部分片收集节点在每个时隙开始时进行交易队列稳定性分析, 算法 3 根据队列中等待的交易数量、交易到达率和交易处理率计算得到队列中交易数量 (第 3–5 行).

2) 根据队列积压量计算每个队列在相邻时隙中的队列积压标量大小 (第 7–14 行).

3) 记录队列积压标量从一个时隙到另一个时隙的动态变化量, 使用 Lyapunov 漂移表示该动态变化, 用于分析队列稳定性 (第 15–18 行).

4) 本文根据 Lyapunov 定理对队列稳定性进行分析并根据队列中等待被处理的交易数量得到队列中需要微调的交易数量. 局部分片的收集节点根据需要微调的交易数量从交易队列中取出父交易顺序打包, 并在交易微调时以交易包的形式将其转移到其他局部分片的交易队列中. 将需要转移的交易数量数组输出 (第 19–32 行).

5) 算法 3 遍历局部分片, 进行交易队列稳定性分析并计算每个交易队列中需要微调的交易数量, 需要经过双层循环, 由于内层循环的次数为一个常量, 因此, 时间复杂度为 O(N).

5 实 验

5.1 实验设置环境和参数设置

5.1.1 实验环境和工具

本文工作重点关注区块链中协议的优化, 主要考虑平行链跨链交易分配的影响因素. 当前, 一些工具已经被广泛使用在区块链实验中, 如 OMNeT++[47]、Python[9]. 其中 OMNeT++可以对区块链网络进行仿真, 测试其性能[47], Java 和 Python 可以用于开发模拟器, 对区块链网络进行模拟. 本文方案设计的目的是提高跨链系统处理交易的性能, 需要观测交易在区块链网络中处理的性能, OMNeT++符合本文的需求. 因此, 本文使用 OMNeT++ 5.4.1 作为仿真工具进行实验, 用于模拟跨链网络中的交易处理过程. 同时, 本文中使用 LSTM 和 SWO 算法, 为测试 LSTM 算法在区块链环境中的可行性, 使用 Python 对其在区块链数据集中的预测精度进行实验; 为确定 SWO 算法中的参数, 使用 Matlab 2020a 对其进行不同的参数实验. 实验环境: Windows 11 64 位操作系统, Intel i7-12700H 处理器和 16 GB 内存.

5.1.2 数据集和实验参数

本文工作使用 XBlock 中的以太坊数据集. XBlock 是一个旨在助力区块链良性发展和数据研究的数据集共享平台, 不仅包括比特币、以太坊以及 EOS 等多种加密货币的数据, 而且数据集的内容有着丰富的类型, 如 XBlock 的以太坊数据集包括以太坊链上数据、智能合约属性数据以及交易数据, 链上数据可以帮助用户追溯以太坊的详细区块信息, 用户也可以使用交易数据探索以太坊中的详细交易信息, 文献 [24] 使用该数据集中的数据进行实验. 本文使用的以太坊数据集包括 100 000 条链上数据和交易数据, 其中包括以太坊区块的大小、交易数量、交易大小和产生区块的时间戳等, 本文从中选取交易大小、交易数量和区块大小数据进行实验. 实验参数如表 4 所示.

表 4 中, 局部分片数量根据文献 [7,10] 设置, 局部分片中节点数量根据文献 [48] 设置, 节点带宽大小根据文献 [7,8] 设置, 节点的带宽设置为 1–3 Mb/s 之间的随机值. 区块的创建间隔根据以太坊的数据、波卡的参数以及文献 [7,10] 设置为 10 s. 依据文献 [17] 中区块的大小, 结合每个交易大小设置平行链中每秒产生的跨链交易数量. 为在计算负载值时保证选择乐观距离和悲观距离的公平性, 依据文献 [34] 设置选择阈值为 0.5. 在局部分片数量为 8 的情况

© 中国科学院软件研究所 http://www.jos.org.cn

下, 如果平行链的数量小于 8, 则在本文方案中一定会产生局部分片资源浪费, 当平行链的数量大于或等于 8 时, 才能测试本文分配方案的合理性, 本文将平行链的数量设置为 10 进行测试.

表 4 实验参数设置

参数	描述	取值/取值范围
N	局部分片数量	8
ω	分片中节点数量	60, 80, 100, 120, 140, 160
α	节点带宽	[1, 3] Mb/s
β	每个交易大小	[256, 512] B
γ	区块创建间隔	10 s
δ	每秒产生跨链交易数量	1 000, 1 500, 2 000, 2 500
ζ	平行链数量	10
P_c	交叉率	0.4, 0.6, 0.8, 0.9
P_t	权衡率	0.2, 0.3, 0.4, 0.5
ϕ	黄蜂数量	20, 40, 60, 80
ψ	蜘蛛数量	100, 200, 300, 400
λ	选择乐观或悲观距离的阈值	0.5

5.1.3 实验指标

为将本文方法与其他方法进行对比, 使用以下指标: 交易处理延迟、交易吞吐量和交易处理成功率.

1) 交易处理延迟 ($\mathcal{L}$)

$$\mathcal{L}=\left(\sum_{n=1}^{\mathcal{N}}(p_n-q_n)\right)\Big/\mathcal{N} \tag{24}$$

其中, p_n 和 q_n 分别是交易 n 的生成时间和其被确认接收的时间, 生成时间记录在跨链交易数据结构的时间戳中, 假设局部分片处理 $\mathcal{N}$ 个交易, 本文使用交易处理延迟的平均值作为实验指标.

2) 交易吞吐量 ($\mathcal{T}$)

$$\mathcal{T}=\sum_{i=1}^{N}(\mathcal{N}/\mathbb{T}) \tag{25}$$

其中, $\mathcal{N}$ 是局部分片 C_i 在时间 $\mathbb{T}$ 内处理完成的交易数量.

3) 交易处理成功率 ($\mathcal{S}$)

$$\mathcal{S}=\mathcal{N}/\mathcal{X} \tag{26}$$

其中, $\mathcal{X}$ 是中继链在时间 $\mathbb{T}$ 内需要处理的交易总量, PBFT (practical Byzantine fault tolerance) 算法具有高效性且可以在分布式环境下保持集群状态的一致性[49], 本文中继链分片中节点数量较少, 满足 PBFT 的使用条件, 因此本文在中继链分片中使用 PBFT 算法进行共识. PBFT 共识算法可以容忍 1/3 的恶意节点, 如果当前分片中的恶意节点超过节点总数的 1/3, 分片就将失效, 恶意节点的存在会降低分片处理交易的成功率.

5.1.4 对比方法

为分析本文所用交易分配方法可以均衡不同局部分片的交易处理时间, 保证局部分片负载和性能相匹配, 本文选取 LB-Chain[10]和 EfShard[26]作为对比方法, 该两种方法均在分片中研究分片负载, 与本文研究对象一致. 为分析本文方法在提高跨链系统可扩展性方面优于其他方法, 选取最新的在不同跨链技术中提高交易处理效率的方法进行对比, 包括 CrossChannel[50]、Notary[51]和 RAC-Chain[52].

LB-Chain[10]: 该模型是一种新颖的分片系统, 其分析分片性能受限的原因是不同分片上的负载不均衡, 提出一种高效安全的账户和交易迁移方案和一种账户分配算法, 通过定期将活动账户从负载较重的分片迁移到负载较轻的分片来动态平衡不同分片上的负载.

EfShard[26]: 该模型是一种高效的状态分片区块链系统, 该系统实时将交易分配至负载较低的分片中, 保证不

© 中国科学院软件研究所 http://www.jos.org.cn

同分片的负载均衡.

CrossChannel[50]: 该模型利用链下支付渠道促进跨链交易, 引入链中继协议和资产管理机制以解决区块链之间分离带来的安全性和可用性挑战. 同时, 该模型设计通道快照机制保证链中继的效率, 提高可扩展性.

Notary[51]: 该模型是一种基于可验证随机数、增强的 EigenTrust 算法和累积概率的新型声誉值计算和节点选择模型, 针对节点维护跨链价值不积极进而影响跨链机制效率和安全性的问题提出, 以确保节点参与度, 提升跨链机制的运行效率和安全性.

RAC-Chain[52]: 该模型是一种基于中继链和异步共识的联盟链跨链模型, 解决单个区块链难以支持大规模数据的可扩展性限制, 利用异步 BFT 支持更多应用链接入中继链保证跨链双方交易状态的一致性.

5.2 实验结果与分析

(1) 预测算法结果

本文使用预测算法对跨链交易数量和大小进行预测以调整预测算法的参数并证明本文预测算法的可行性. 本文使用以太坊数据集, 图 7 和图 8 分别显示对以太坊 10000 个区块中交易数量和交易大小的测试结果, 使用 8000 个区块中交易的信息对模型进行训练. 多次训练后, 对交易数量和交易大小进行训练, 平均训练时长分别为 1.063 s 和 1.067 s, 对交易数量和交易大小进行测试的平均测试时长都为 0.01 s. 从图 7 和图 8 可见对 1000 个区块中交易信息的预测值与真实值较接近, 平均绝对误差 (mean absolute error, MAE) 分别为 90.82 和 65.32, 均方根误差 (root mean square error, RMSE) 分别为 107.83 和 76.59. 在本文对交易数量和交易大小的预测中, RMSE 均大于 MAE, 表明本文模型在预测较大交易数量时存在更大的偏差. 同时, 相较于 MAE, RMSE 对异常值更敏感. 因此, 交易数量和交易大小存在极端情况. 同时, 图 7 和图 8 中交易数量的 MAE 占每次预测时交易数量平均值的比例为 30.48%, 交易大小的 MAE 占每次预测时交易大小平均值的比例为 32.74%, 表明本文中预测方法可以反映交易活动的变化趋势. 根据对 MAE、RMSE 和 MAE 占平均值比例的情况分析可知, 由于区块链交易具有突发性, 交易数量和大小波动幅度较大, 导致负载激增问题. 因此, 本文使用交易微调方法缓解该问题.

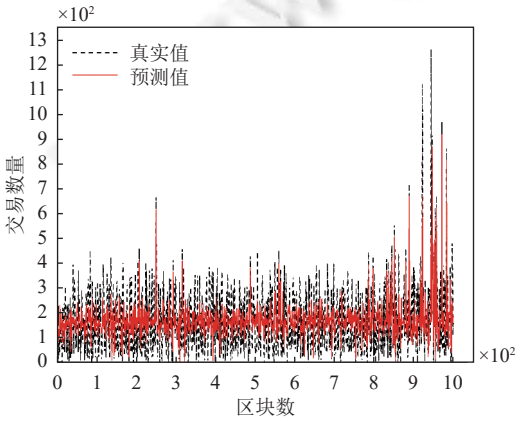


图 7 交易数量预测结果

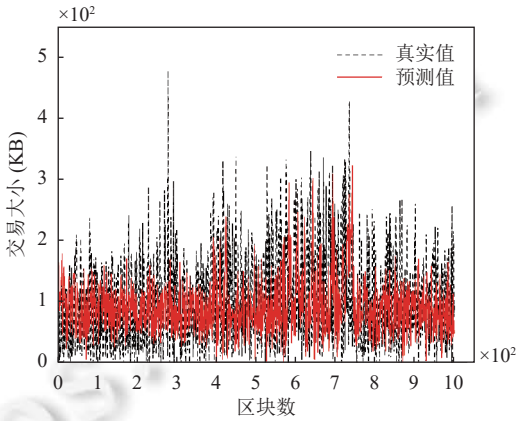


图 8 交易大小预测结果

(2) 分配算法中参数确定实验

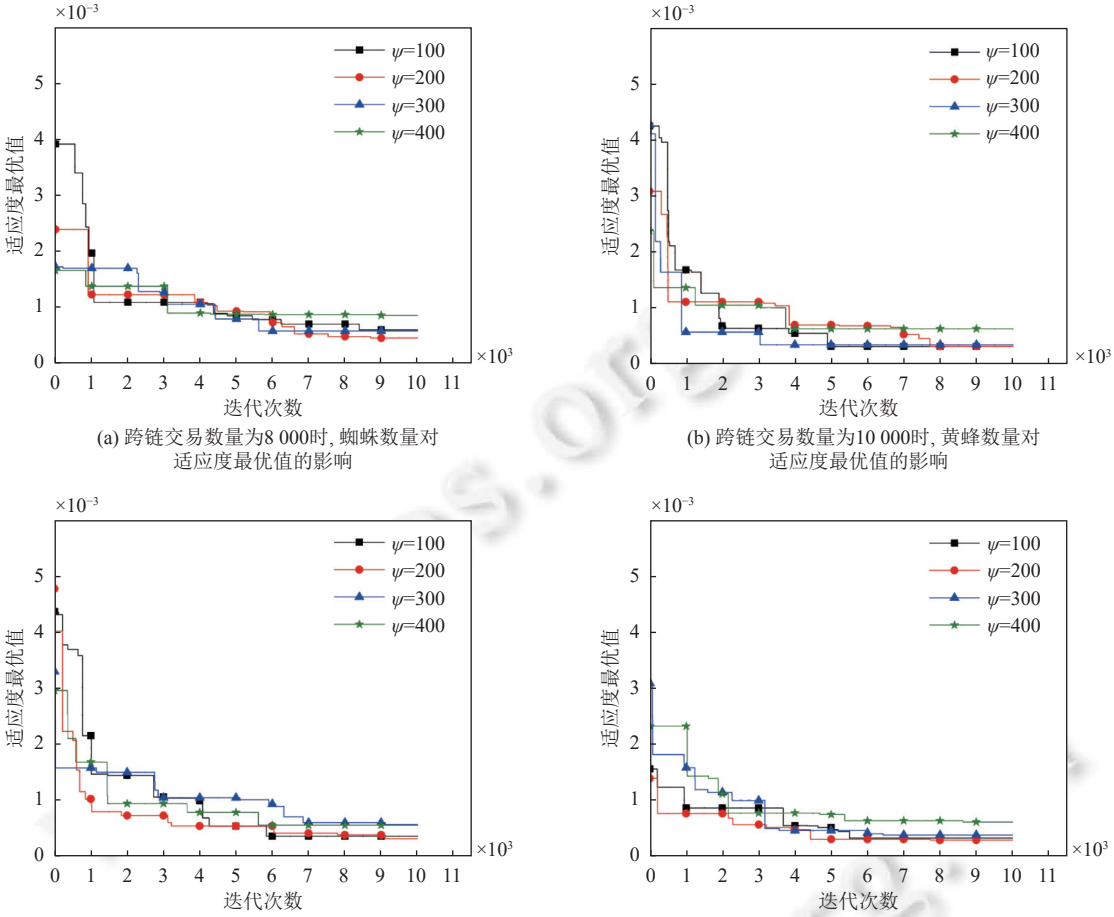
本节使用适应度最优值确定分配算法 SWO 中的各个参数, 包括蜘蛛的数量、黄蜂的数量、狩猎和交配之间的权衡率以及交叉率. 使用适应度最优值可以直观展现各个参数的设置对分配方案的影响.

1) 根据适应度最优值变化确定蜘蛛数量

在 SWO 算法中, 蜘蛛主要用于对解方案的局部搜索, 蜘蛛的数量表示局部搜索能力对适应度的影响. 本文在实验中改变蜘蛛的数量, 分别设置蜘蛛的数量 ψ 为 100、200、300 和 400, 迭代的次数为 10000, 设置中继链处理的跨链交易数量分别为 8000、10000、12000 和 14000, 不同蜘蛛数量对适应度最优值变化的影响如图 9 所示. 图 9 中, 纵坐标表示的是在给定迭代次数下适应度达到的最优值. 实验结果表明, 最小适应度最优值随着 ψ 的增加

© 中国科学院软件研究所 http://www.jos.org.cn

呈现先减小后增加的趋势, ψ=400 时的适应度最优值始终为较高值, 而 ψ=200 时的适应度最优值始终为较低值且适应度最优值在 ψ=200 时达到最小值. 同时, 跨链交易数量不同时, 适应度都可以在迭代后达到较低值. 因此, 在后续实验中, 蜘蛛的数量设为 200.



(a) 跨链交易数量为8 000时, 蜘蛛数量对适应度最优值的影响

(b) 跨链交易数量为10 000时, 黄蜂数量对适应度最优值的影响

(c) 跨链交易数量为12 000时, 蜘蛛数量对适应度最优值的影响

(d) 跨链交易数量为14 000时, 蜘蛛数量对适应度最优值的影响

图 9 跨链交易数量不同时, 不同蜘蛛数量对适应度最优值的影响

2) 根据适应度最优值变化确定黄蜂数量

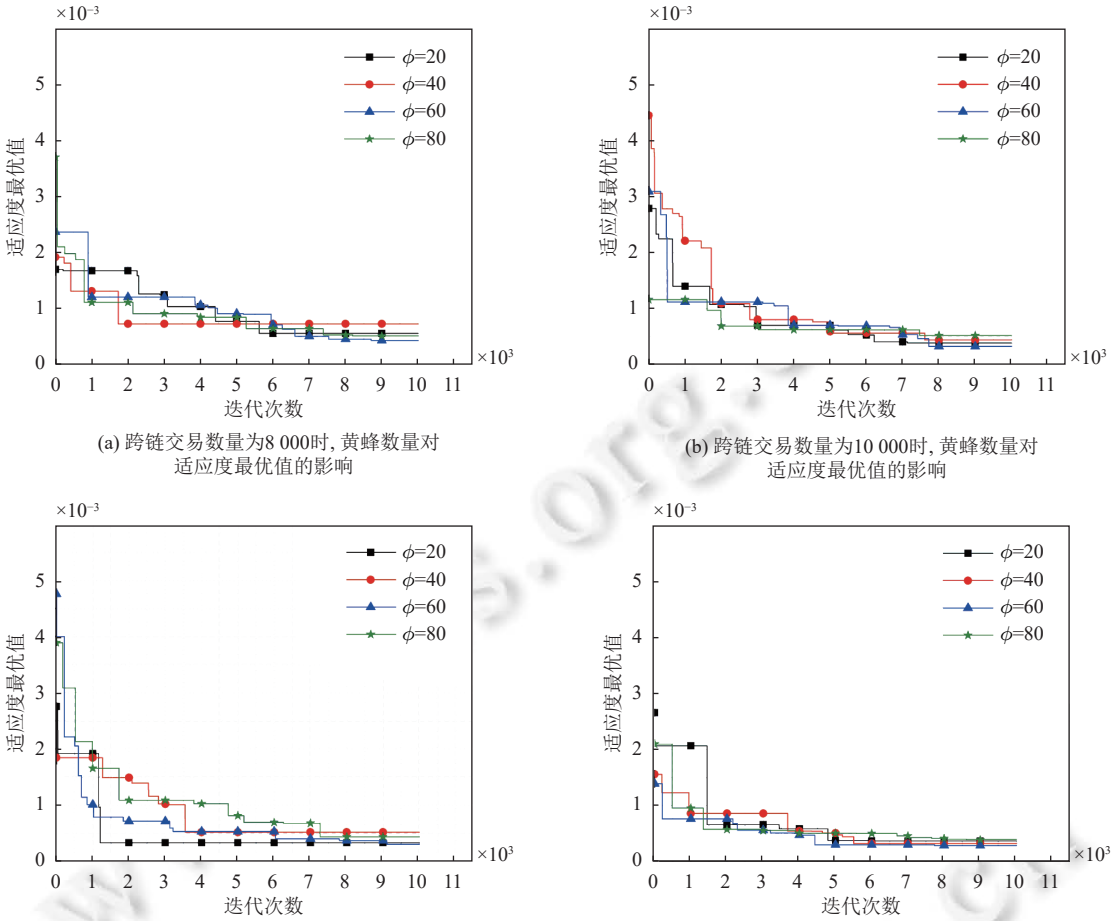
在 SWO 算法中, 黄蜂主要用于对解方案的全局搜索, 黄蜂的数量表示全局搜索能力对适应度的影响. 本文在实验中改变黄蜂的数量, 分别设置黄蜂的数量 ϕ 为 20、40、60 和 80, 迭代的次数为 10 000, 设置中继链处理的跨链交易数量分别为 8 000、10 000、12 000 和 14 000, 不同黄蜂数量对适应度最优值变化的影响如图 10 所示. 图 10 中, 跨链交易数量不同时, 适应度都可以在迭代后达到较低值. 随着黄蜂数量的变化, 适应度最优值在 ϕ=60 时达到最小值. 因此, 在 SWO 算法的后续实验中, 黄蜂的数量设为 60.

3) 根据适应度最优值变化确定狩猎和交配之间的权衡率

狩猎和交配之间的权衡率表示黄蜂采用全局搜索和交配行为以产生新一代种群之间的平衡, 为在算法运行过程中确定产生下一代种群的方法, 本文在实验中改变权衡率, 分别设置权衡率 P_t 为 0.2、0.3、0.4 和 0.5, 设置中继链处理的跨链交易数量分别为 8 000、10 000、12 000 和 14 000, 不同权衡率对适应度最优值变化的影响如图 11 所示. 图 11 中, 在不同跨链交易数量情况下, P_t=0.3 时, 适应度最优值的最终结果均最小, 表示此时黄蜂种群不断

© 中国科学院软件研究所 http://www.jos.org.cn

迭代产生新一代种群可以得到最优的解方案. 因此, 在 SWO 算法的后续实验中, 权衡率设为 0.3.



(a) 跨链交易数量为8 000时, 黄蜂数量对适应度最优值的影响

(b) 跨链交易数量为10 000时, 黄蜂数量对适应度最优值的影响

(c) 跨链交易数量为12 000时, 黄蜂数量对适应度最优值的影响

(d) 跨链交易数量为14 000时, 黄蜂数量对适应度最优值的影响

图 10 跨链交易数量不同时, 不同黄蜂数量对适应度最优值的影响

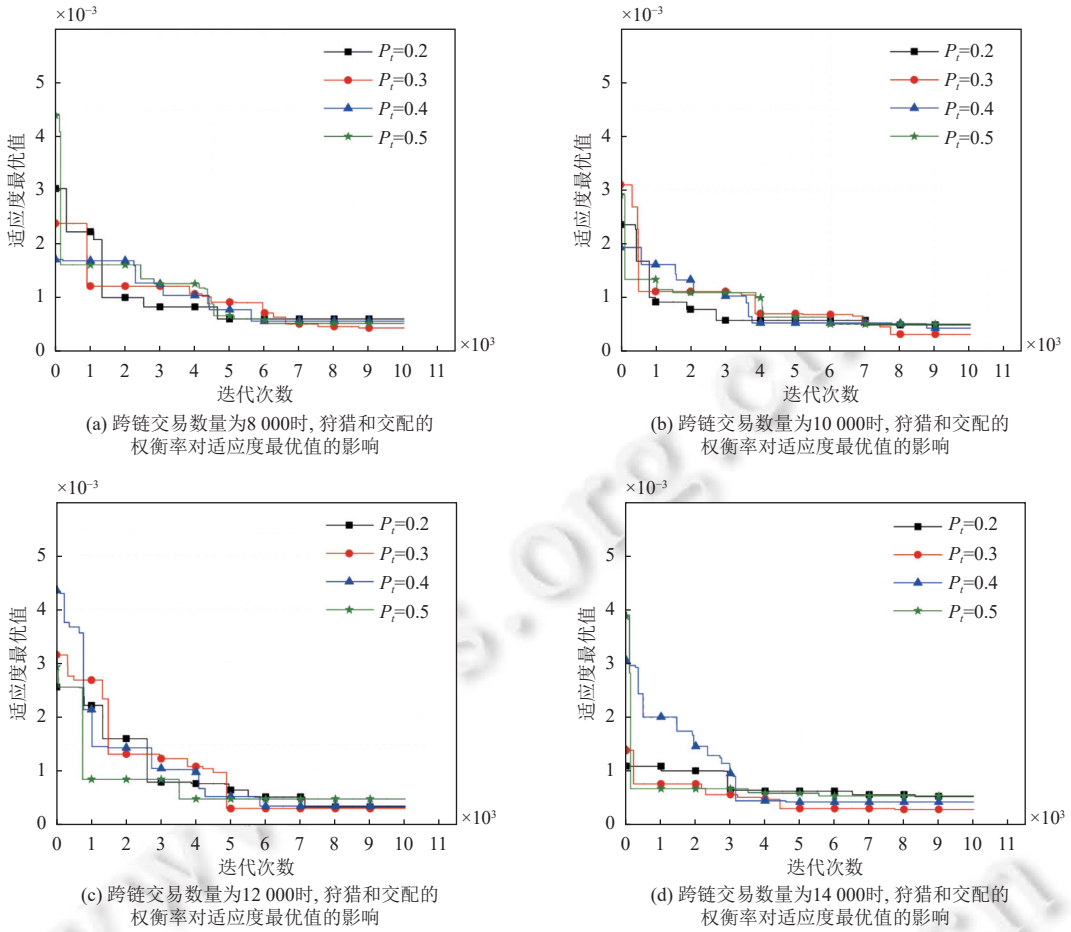
4) 根据适应度最优值变化确定交叉率

雄黄蜂与雌黄蜂之间使用均匀交叉算子产生后代, 从而形成两个新的解方案. 本文对交叉率和适应度之间的关系进行实验, 分别设置交叉率 P_c 为 0.2、0.4、0.6 和 0.8, 设置中继链处理的跨链交易数量分别为 8 000、10 000、12 000 和 14 000, 不同交叉率对适应度最优值变化的影响如图 12 所示. 图 12 中, P_c=0.2 和 0.8 时, 适应度最优值均曾达到最小值. 然而, 随着跨链交易数量的变化, 大部分情况下, P_c=0.2 时适应度最优值达到最小值, 即使 P_c=0.2 时适应度最优值未达到最小值, 其与最小值接近. 因此, 在 SWO 算法的后续实验中, 交叉率设为 0.2.

5) 蜘蛛数量与交叉率对适应度最优值变化的影响

根据图 9 和图 12, 蜘蛛数量为 100 和 200, 交叉率为 0.2 和 0.4 时, 适应度最优值在不同跨链交易数量下始终保持较低值. 为进一步确定蜘蛛数量和交叉率, 本文分别设置 ψ=100 且 P_c=0.2、ψ=100 且 P_c=0.4、ψ=200 且 P_c=0.2、ψ=200 且 P_c=0.4, 迭代的次数为 10 000. 在跨链交易数量分别为 8 000、10 000、12 000 和 14 000 时, 适应度最优值随迭代次数的变化如图 13 所示. 图 13 中, 跨链交易数量不同时, 适应度都可以在迭代后达到较低值. 适应度最优值在 ψ=200 且 P_c=0.2 时达到最小值. 因此, 进一步确定在后续实验中, 蜘蛛数量设为 200, 黄蜂数量设为 60, 权衡率设为 0.3, 交叉率设为 0.2, 本文通过 10 000 次迭代, 消耗时间成本 4.43 s, 使用 SWO 算法获得分配方案.

© 中国科学院软件研究所 http://www.jos.org.cn



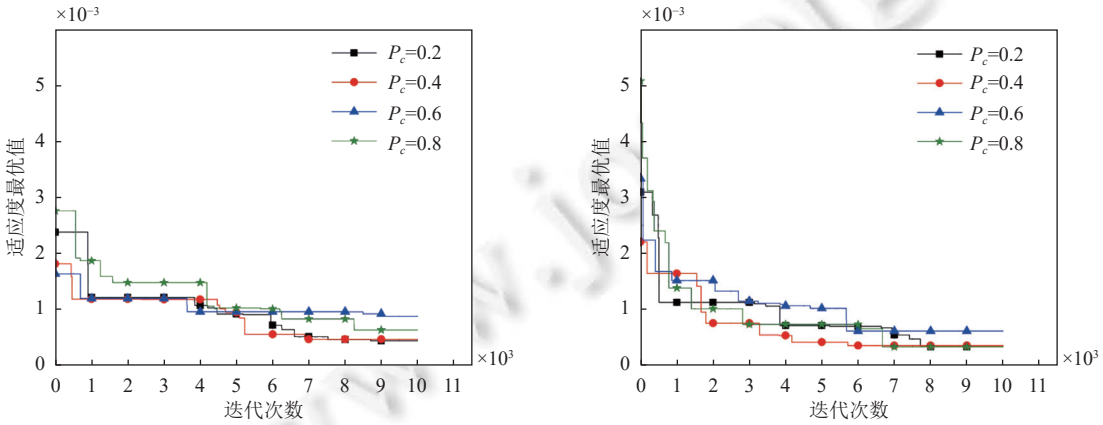
(a) 跨链交易数量为8 000时, 狩猎和交配的权衡率对适应度最优值的影响

(b) 跨链交易数量为10 000时, 狩猎和交配的权衡率对适应度最优值的影响

(c) 跨链交易数量为12 000时, 狩猎和交配的权衡率对适应度最优值的影响

(d) 跨链交易数量为14 000时, 狩猎和交配的权衡率对适应度最优值的影响

图 11　跨链交易数量不同时, 狩猎和交配的权衡率对适应度最优值的影响

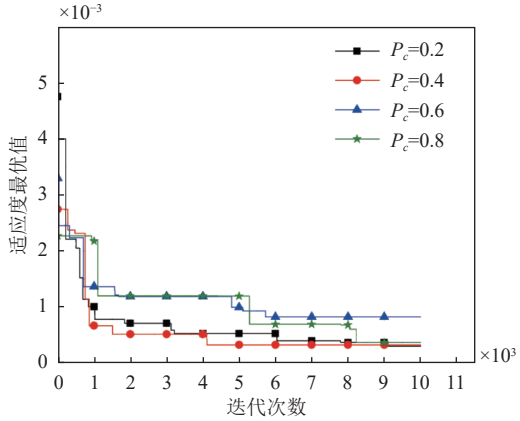


(a) 跨链交易数量为8 000时, 交叉率对适应度最优值的影响

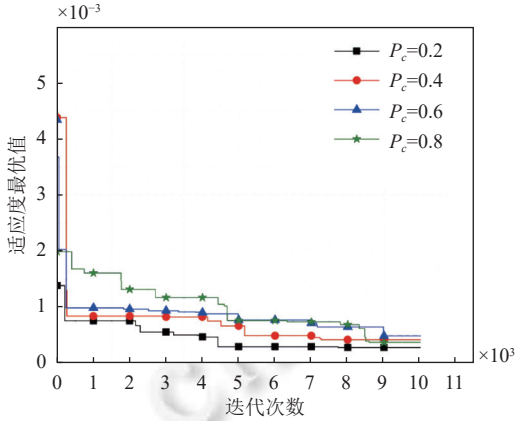
(b) 跨链交易数量为10 000时, 交叉率对适应度最优值的影响

图 12　跨链交易数量不同时, 交叉率对适应度最优值的影响

© 中国科学院软件研究所 http://www.jos.org.cn

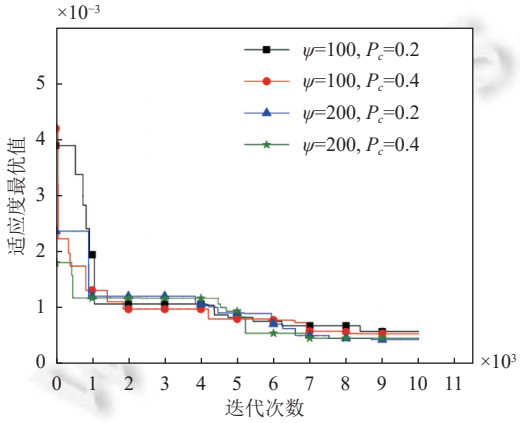


(c) 跨链交易数量为12 000时, 交叉率对适应度最优值的影响

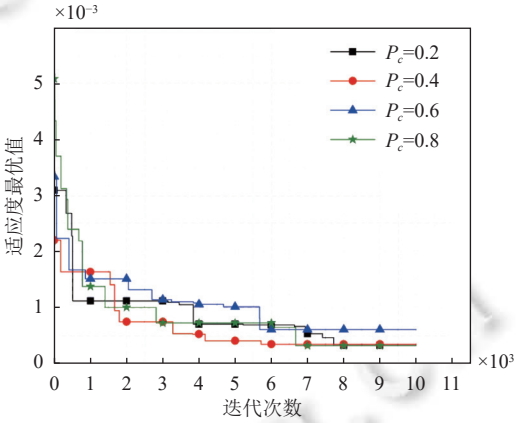


(d) 跨链交易数量为14 000时, 交叉率对适应度最优值的影响

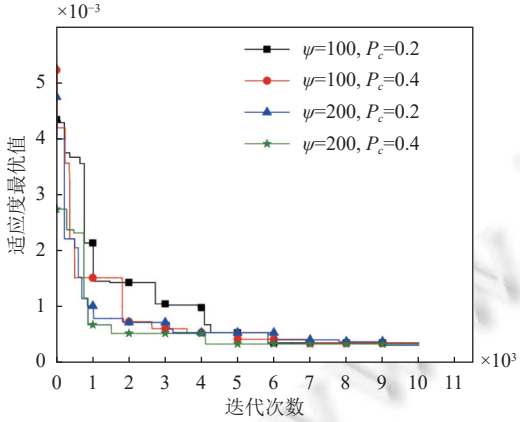
图 12 跨链交易数量不同时, 交叉率对适应度最优值的影响 (续)



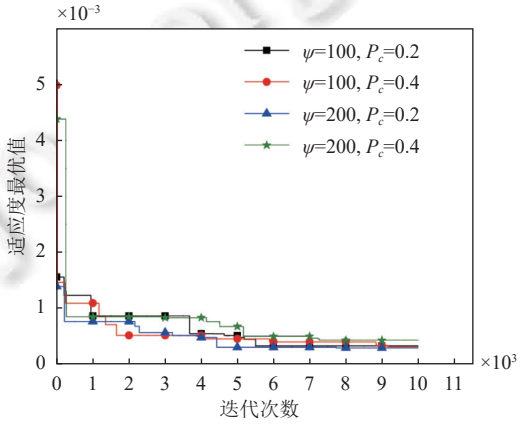
(a) 跨链交易数量为8 000时, 蜘蛛数量与交叉率对适应度最优值的影响



(b) 跨链交易数量为10 000时, 蜘蛛数量与交叉率对适应度最优值的影响



(c) 跨链交易数量为12 000时, 蜘蛛数量与交叉率对适应度最优值的影响



(d) 跨链交易数量为14 000时, 蜘蛛数量与交叉率对适应度最优值的影响

图 13 跨链交易数量不同时, 蜘蛛数量与交叉率对适应度最优值的影响

© 中国科学院软件研究所 http://www.jos.org.cn

(3) 不同乐观距离和悲观距离的选择阈值 λ 对交易处理时间的影响

在负载值计算方法中, 阈值 λ 的选择会影响对乐观距离和悲观距离的选择, 进而影响跨链交易的分配结果. 为深入研究阈值 λ 对跨链交易分配结果的敏感性, 本文将阈值 λ 分别设置为 0.3、0.5、0.7 和 0.9, 比较不同局部分片处理完所分配交易花费的时间, 实验结果如图 14 所示. 在图 14 中, 阈值 λ 不同, 不同局部分片的交易处理时间波动也不同, 即跨链交易的分配结果不同. λ=0.5 时, 不同局部分片的交易处理时间波动较小, 说明不同局部分片的负载较均衡, 跨链交易的分配结果最满足本文的需求. 因为平行链跨链交易分配涉及两种负载值的匹配, λ=0.5 时, 分片负载值计算方法中乐观距离和悲观距离被选择的可能性相同, 两种负载值的计算都能以相同的可能性根据实际情况得到最终负载值. λ=0.9 时, 不同局部分片的交易处理时间波动较大, 因为此时悲观距离被选择的可能性极小, 最终负载值受最优属性值的影响更大, 即依赖于乐观距离的计算, 导致部分最终负载值偏大或偏小, 进而影响跨链交易的分配结果. 阈值 λ 的敏感性体现在其变化对交易处理时间和系统负载均衡程度具有显著影响. 实验结果表明, λ=0.5 时, 系统在处理跨链交易时能够实现负载均衡, 交易处理时间波动最小. 因此, 本文在后续的实验中设置阈值 λ 为 0.5.

(4) 不同节点数量对交易吞吐量和通信延迟的影响

为研究中继链分片中节点数量对中继链中交易吞吐量的影响, 本文将中继链分片中的节点数量分别设置为 60、80、100、120、140 和 160, 实验结果如图 15 所示. 在图 15 中, 随着每个局部分片中节点数量的增加, 每个节点需要与更多的节点进行通信, 同时, 交易在节点之间传播可能产生更多的冗余, 导致通信延迟增加. 与之相反, 增加节点数量可以提升局部分片的整体带宽, 在同一时间可以有更多的交易被同时处理, 缩短交易的整体处理时间. 虽然中继链整体吞吐量受到通信延迟和交易处理时间的双重影响, 但是从图 15 可见, 随着节点数量的增加, 通信延迟呈现递增趋势, 吞吐量呈现递减趋势, 表明在节点数量增加的情况下, 通信延迟对交易吞吐量的影响更明显, 因此局部分片中的节点数量并非越多越好.

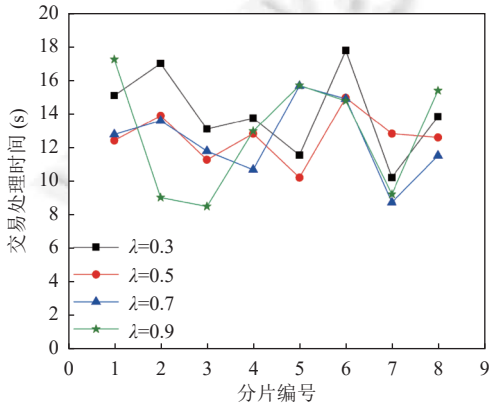


图 14 乐观距离和悲观距离的选择阈值 λ 对交易处理时间的影响

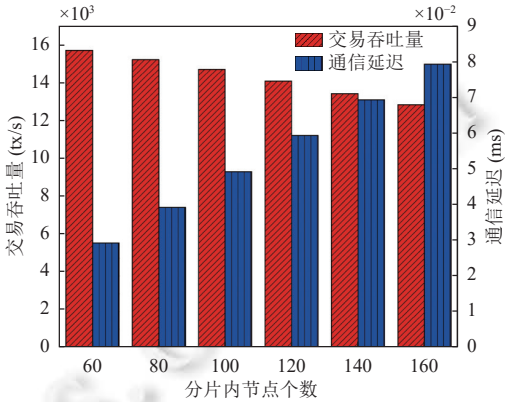


图 15 节点数量对吞吐量和通信延迟的影响

(5) 中继链区块大小对交易吞吐量和延迟的影响

在中继链中, 区块大小越大, 所能包含的交易数量越多, 区块在节点之间的传播延迟就越大, 传播延迟增加对交易吞吐量会产生影响. 因此, 需要分析区块大小与交易吞吐量的关系, 以确定达到最优吞吐量时区块的大小. 本文将区块的大小分别设置为 125、250、375、500、625、750、875 和 1 000, 对节点数不同的情况进行实验分析, 实验结果如图 16 所示. 图 16 中, 在不同节点数量下, 交易吞吐量均随着区块大小的增加, 先提高后降低. 当区块大小小于 500 KB 时, 随着区块中交易数量增加, 单位时间内有更多的交易转发至中继链, 中继链可以一次处理更多的交易从而提高中继链的处理效率, 因此交易吞吐量随着区块大小的增加而提高. 在区块大小大于 500 KB 时, 节点之间传播区块的延迟增加且节点带宽受限, 导致交易吞吐量随着区块大小的增加而降低. 因此, 选择合适的区块

© 中国科学院软件研究所 http://www.jos.org.cn

大小可以优化中继链的交易吞吐量, 本文根据交易吞吐量变化设置区块大小为 500 KB.

(6) 节点数量对交易处理成功率的影响

由于对中继链进行分片后, 恶意节点被划分到多个分片中, 因此中继链的容错能力由分片的容错能力决定. 本文交易分配模型使用 PBFT 共识算法对交易进行共识, PBFT 共识算法的容错能力与节点数量密切相关, 只能容忍 1/3 的恶意节点. 每个分片内的节点数量决定系统可以容忍的恶意节点数量, 因此, 需要通过实验分析节点数量对交易处理成功率的影响, 为在不同情况下分析该影响, 设置中继链中恶意节点比例为 0.15、0.2、0.25、0.3、0.35 和 0.4, 实验结果如图 17 所示. 图 17 给出不同恶意节点比例和不同节点数情况下, 交易处理成功率的变化, 将恶意节点随机分配到分片中, 随着中继链中恶意节点比例的增加, 虽然存在较小的波动, 但是交易处理成功率整体呈现下降的趋势. 恶意节点比例小于 1/3 时, 节点数越大, 交易成功率越大, 因为节点数大时, 分片内超过 2/3 节点达成共识的可能性越高; 节点比例大于 1/3 时, 更多的恶意节点可能被分配到同一个分片, 使分片崩溃. 图中, 节点数为 120 且恶意节点比例小于 0.35 时, 交易处理成功率相较于其他情况处于较高水平. 但是节点数为 120 时, 交易处理成功率的优势不足以弥补吞吐量缺陷. 节点数为 60 时, 交易处理成功率在大部分情况下低于节点数为 80 和 100 时的成功率. 节点数为 80 时, 交易处理成功率总体大于节点数为 100 时的成功率, 且节点数为 80 时, 交易吞吐量处于较高水平. 因此确定每个分片内节点数为 80, 即中继链中节点数为 640, 并在后续实验中将节点数设置为 640.

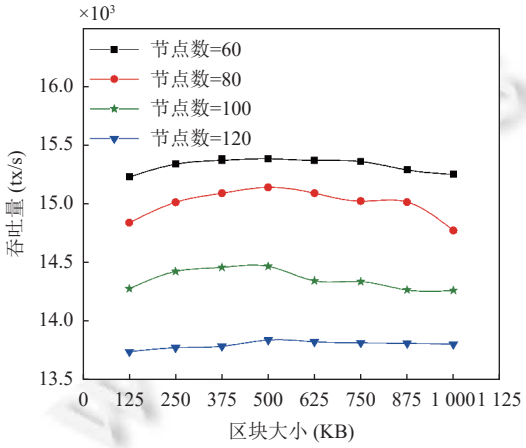


图 16 区块大小与交易吞吐量的关系

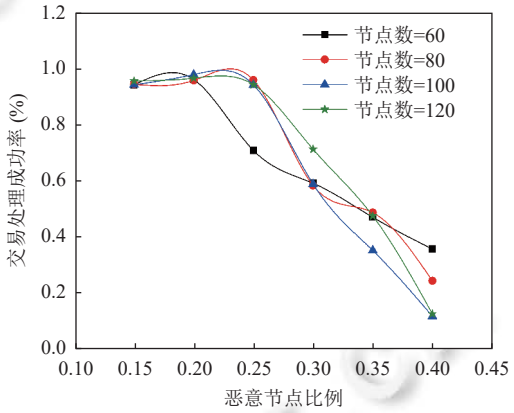


图 17 节点数量对交易处理成功率的影响

(7) Lyapunov 微调对交易队列长度的影响

为评估 Lyapunov 在交易处理过程中保证交易队列稳定的作用, 本文对未使用 Lyapunov 和使用 Lyapunov 微调交易的两种情况进行实验. 实验中平行链产生 2 000 000 个交易并以每秒 20 000 个交易的到达率发送给中继链, 每 200 s 使用 Lyapunov 分析交易队列稳定性并对不稳定交易队列中的交易进行微调. 交易队列长度表现为队列中交易数量, 记录队列中交易数量的变化情况, 计算出不同交易队列长度之间的标准差, 使用该标准差展现 Lyapunov 对交易队列的微调效果, 实验结果如图 18 所示. 图 18 中, 未使用 Lyapunov 对交易进行局部分片间微调时, 中继链处理完所有交易共使用约 1 335 s, 而使用 Lyapunov 对交易进行动态微调后, 中继链处理完所有交易只使用约 1 001 s. 相比之下, 使用 Lyapunov 微调交易后处理时间减少 25%. 同时, 从是否使用 Lyapunov 的两种情况看, 使用 Lyapunov 时交易队列长度之间标准差的波动幅度远小于未使用 Lyapunov 时的波动幅度. 从使用 Lyapunov 的情况看, 每 200 s 微调后, 交易队列长度之间的标准差都出现明显下降. 因此, 本文使用 Lyapunov 动态微调交易可以维持不同局部分片处理交易时间的均衡.

(8) 不同局部分片的交易处理时间对比实验

为验证本文模型中自适应交易分配方法在交易处理时间方面的优势, 本文在实验中选取随机分配策略、LB-Chain 和 EfShard 作为对比方法, 在相同交易到达率和交易数量情况下比较不同局部分片处理完所分配交易花费

© 中国科学院软件研究所 http://www.jos.org.cn

的时间, 实验结果如图 19 所示. 图 19 中, 由于不同局部分片性能不同, 随机分配方法在不考虑负载和分片性能的情况下将交易随机分配到不同的局部分片, 导致局部分片负载与其处理能力不匹配, 进而导致局部分片之间的交易处理时间存在很大差异, 局部分片的资源不能得到合理利用. LB-Chain 主要考虑不同分片之间交易数量的差异以对交易进行分配, 虽然交易处理时间均衡效果相较于随机分配方法有较大提升, 但是没有考虑不同分片之间性能的区别以及交易大小对处理时间的影响, 分片的性能不同, 处理相同数量的交易花费的时间不同, 同时, 交易大小不同, 其在节点之间的传播延迟也不同. 因此, 该方法中不同分片的交易处理时间依然存在较大差异. EfShard 采用实时分配交易的方式, 可以实时反映分片处理交易的情况以均衡不同分片的负载. 因此, 该方法中不同分片的交易处理时间较随机分配策略和 LB-Chain 方法更均衡, 但是由于实时分配交易增加交易分配时的延迟, 交易处理时间始终高于本文方法. 本文首先提出自适应交易分配方法, 该方法在计算负载值时同时考虑交易大小、数量和分片的性能, 可以均衡局部分片处理交易的时间; 其次, 根据源链编号将跨链交易分配给对应局部分片, 交易和账户状态涉及多个平行链而不涉及多个分片. 同时, 本文利用索引将可能跨分片的子交易交给同一个局部分片处理从而避免子交易成为跨分片交易, 减少跨分片交易对交易处理时间的影响. 从图 19 中可见不同局部分片的交易处理时间接近, 波动较小. 因此, 本文的自适应交易分配方法可行.

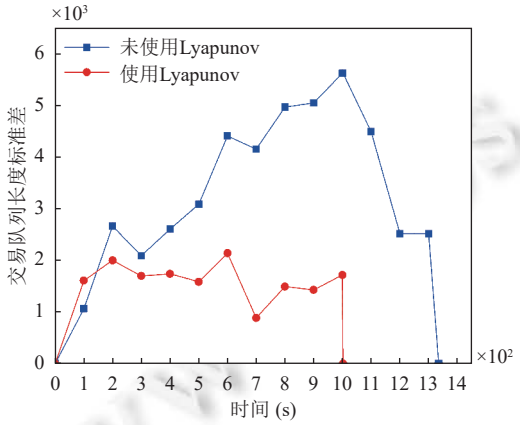


图 18 Lyapunov 对交易队列长度标准差的影响

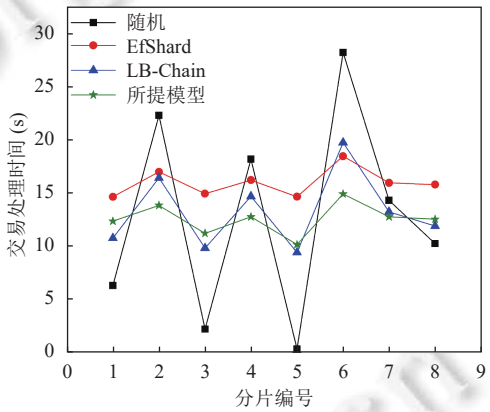


图 19 不同局部分片的交易处理时间对比

(9) 交易吞吐量和处理延迟对比实验

为验证本文所提模型在降低交易处理延迟和提高跨链交易吞吐量方面具有优势, 本文在上述最佳区块大小和节点数量下, 针对交易处理延迟和交易吞吐量进行统计, 并选取 CrossChannel、Notary 和 RAC-Chain 作为对比方法, 如图 20 所示.

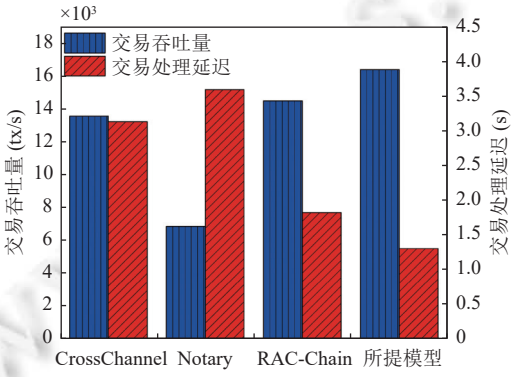


图 20 不同模型交易吞吐量和处理延迟对比

© 中国科学院软件研究所 http://www.jos.org.cn

图 20 中, 本文所提模型的交易处理延迟低于 CrossChannel、Notary 和 RAC-Chain 且交易吞吐量高于这 3 个对比方法. CrossChannel 采用链外小额支付通道实现跨链资产交换, 并使用通道快照机制将所有与通道相关的信息包含在短内容中进行中继, 以提高链中继的可扩展性. 但是, 创建和关闭链下通道时间较长, 在链下通道创建和关闭过程中, 交易被锁定, 进而增加交易处理延迟. Notary 为提高节点积极性, 提高处理跨链交易效率, 设计一种新型声誉值计算和节点选择模型, 但是该模型主要针对节点的声誉, 通过减少恶意行为的出现, 提高交易的吞吐量. 但是在面对大量交易时, 节点串行处理跨链交易增加交易的堵塞情况, 导致交易队列中的交易长时间等待, 进而增加交易处理时间, 同时, 该方法没有处理恶意行为的解决方法, 可能导致交易的处理过程停滞, 进而降低跨链交易吞吐量. 因此, 该方法的交易吞吐量最低且交易处理延迟最高. RAC-Chain 是一种基于中继链和异步共识的跨链模型, 异步共识可以容忍极端延迟, 有效处理存在极端延迟的交易, 同时, 通过中继链和多个跨链网关有效提高跨链系统可扩展性, 可以实现跨链交易的快速传输. 该方法中的异步共识算法可以更好地适应异步网络环境, 降低节点之间的通信延迟, 提高交易的共识效率, 从传输和共识方面提高交易吞吐量. 因此, 其吞吐量高于 CrossChannel 和 Notary, 但是 RAC-Chain 存在交易长时间等待的情况且不能并行处理交易.

本文所提模型对中继链进行分片以并行处理跨链交易, 完善交易分配方法, 在分配交易时充分考虑分片的性能和交易造成的负载, 考虑交易大小和交易数量对于通信延迟的影响, 将交易分配到合适的分片, 充分利用每个分片的资源. 同时, 本文在交易处理过程中进行交易队列稳定性分析, 细粒度微调不稳定队列中的交易, 解决分片中交易激增导致的其负载与分片处理能力不匹配问题, 避免部分交易长时间等待. 本文模型通过充分利用分片资源和减少交易等待时间, 降低交易的处理延迟. 因此, 本文提出的模型可以并行处理大量并发交易, 降低交易处理延迟并提高系统的交易吞吐量. 实验分析总结如下: 1) 交易吞吐量分析: 本文所提模型的交易吞吐量高于 CrossChannel、Notary 和 RAC-Chain, 同时由于通信延迟随着节点数量的增加而增加, 本文所提模型交易吞吐量随着局部分片中节点数量的增加而降低. 2) 交易处理延迟分析: 本文所提模型完善交易分配方法, 充分利用分片资源, 其交易处理延迟低于 CrossChannel、Notary 和 RAC-Chain. 3) 局部分片的交易处理时间均衡分析: 本文所提模型中每个局部分片的交易处理时间相较于随机方法、LB-Chain 和 EfShard 更均衡, 且整体交易处理时间较低.

6 结论和未来工作

本文设计面向中继链分片环境的两阶段自适应交易分配模型, 提出交易分配预测方法和交易队列稳定性分析方法, 从两阶段进行自适应交易分配. 其中, 交易分配预测方法利用预测算法预测交易信息并计算负载值, 基于负载值动态分配平行链跨链交易, 同时, 对中继链分片中跨分片交易产生的原因进行分析, 利用交易的数据结构中交易索引的唯一性和关联性对子交易进行分配, 减少跨分片交易和跨链交易的处理延迟; 交易队列稳定性分析方法根据交易队列长度变化分析局部分片中交易队列的稳定性, 对不稳定队列中的交易进行局部分片间动态微调, 以减少交易队列中交易的等待时间. 实验结果显示, 本文所提模型相较于其他方法可以保证每个局部分片的交易处理时间均衡, 充分利用局部分片的资源, 同时, 在交易吞吐量和交易处理延迟方面也表现出更好的性能, 可以有效提高中继链的可扩展性. 本文工作关注中继链分片环境中的自适应交易分配, 中继链分片环境中跨链交易往往涉及多个区块链, 交易状态不一致可能导致不同区块链之间的冲突以及分叉问题, 影响整个跨链系统的可靠性和安全性. 同时, 该模型可能面临如女巫攻击等安全性威胁, 进而导致跨链交易处理失败. 因此, 未来工作可以重点研究如何保证跨链交易在处理过程中跨链交易状态的全局一致性以及如何选举诚实节点作为收集节点, 完善信誉机制以应对女巫攻击等由恶意节点发动的攻击.

References

[1] Xue L, Liu DX, Ni JB, Lin XD, Shen XS. Enabling regulatory compliance and enforcement in decentralized anonymous payment. IEEE Trans. on Dependable and Secure Computing, 2023, 20(2): 931–943. [doi: 10.1109/TDSC.2022.3144991]

[2] Chen XH, Yang AJ, Weng J, Tong Y, Huang C, Li T. A blockchain-based copyright protection scheme with proactive defense. IEEE Trans. on Services Computing, 2023, 16(4): 2316–2329. [doi: 10.1109/TSC.2023.3246476]

[3] Zhang JJ, Ye YJ, Wu WG, Luo XP. Boros: Secure and efficient off-blockchain transactions via payment channel hub. IEEE Trans. on

© 中国科学院软件研究所 http://www.jos.org.cn

Dependable and Secure Computing, 2023, 20(1): 407–421. [doi: 10.1109/TDSC.2021.3135076]

[4] Mazumdar S, Ruj S. CryptoMaze: Privacy-preserving splitting of off-chain payments. IEEE Trans. on Dependable and Secure Computing, 2023, 20(2): 1060–1073. [doi: 10.1109/TDSC.2022.3148476]

[5] Wan ZG, Liu W, Cui H. HIBEChain: A hierarchical identity-based blockchain system for large-scale IoT. IEEE Trans. on Dependable and Secure Computing, 2023, 20(2): 1286–1301. [doi: 10.1109/TDSC.2022.3152797]

[6] Ni JP, Xiao J, Zhang SJ, Li B, Li BC, Jin H. FLUID: Towards efficient continuous transaction processing in DAG-based blockchains. IEEE Trans. on Knowledge and Data Engineering, 2023, 35(12): 12679–12692. [doi: 10.1109/TKDE.2023.3272312]

[7] Liu YZ, Xing XX, Cheng HS, Li DW, Guan ZY, Liu JW, Wu QH. A flexible sharding blockchain protocol based on cross-shard byzantine fault tolerance. IEEE Trans. on Information Forensics and Security, 2023, 18: 2276–2291. [doi: 10.1109/TIFS.2023.3266628]

[8] Cai ZT, Liang JY, Chen WH, Hong ZC, Dai HN, Zhang JT, Zheng ZB. Benzene: Scaling blockchain with cooperation-based sharding. IEEE Trans. on Parallel and Distributed Systems, 2023, 34(2): 639–654. [doi: 10.1109/TPDS.2022.3227198]

[9] Huang HW, Peng XW, Zhan JZ, Zhang SY, Liu Y, Zheng ZB, Guo S. BrokerChain: A cross-shard blockchain protocol for account/balance-based state sharding. In: IEEE Conf. on Computer Communications. London: IEEE, 2022. 1968–1977. [doi: 10.1109/INFOCOM48880.2022.9796859]

[10] Li MZ, Wang W, Zhang J. LB-Chain: Load-balanced and low-latency blockchain sharding via account migration. IEEE Trans. on Parallel and Distributed Systems, 2023, 34(10): 2797–2810. [doi: 10.1109/TPDS.2023.3238343]

[11] Zhao Y, Qu YY, Xiang Y, Zhang YS, Gao LX. A lightweight model-based evolutionary consensus protocol in blockchain as a service for IoT. IEEE Trans. on Services Computing, 2023, 16(4): 2343–2358. [doi: 10.1109/TSC.2023.3238690]

[12] Yue JT, Xiao J, Zhang SJ, Cheng F, Chen HH, Jin H. ElasticDAG: Elastic DAG-based blockchain. Ruan Jian Xue Bao/Journal of Software, 2024, 35(11): 5279–5305 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/7050.htm [doi: 10.13328/j.cnki.jos.007050]

[13] Wu ZY, Liu JL, WU JJ, Zheng ZB, Chen T. TRacer: Scalable graph-based transaction tracing for account-based blockchain trading systems. IEEE Trans. on Information Forensics and Security, 2023, 18: 2609–2621. [doi: 10.1109/TIFS.2023.3266162]

[14] Duan TT, Zhang HW, Li B, Song ZX, Li ZC, Zhang J, Sun Y. Survey on blockchain interoperability. Ruan Jian Xue Bao/Journal of Software, 2024, 35(2): 800–827 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6950.htm [doi: 10.13328/j.cnki.jos.006950]

[15] Zhang PY, Hua XQ, Zhu HB. Cross-chain digital asset system for secure trading and payment. IEEE Trans. on Computational Social Systems, 2024, 11(2): 1654–1666. [doi: 10.1109/TCSS.2023.3241065]

[16] Chen J, Yang H, He K, Li K, Jia M, Du RY. Current situation and prospect of blockchain scaling technology. Ruan Jian Xue Bao/Journal of Software, 2024, 35(2): 828–851 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/6954.htm [doi: 10.13328/j.cnki.jos.006954]

[17] Sun Z, Zhao P, Wang CP, Zhang XL, Cheng HB. An efficient and secure trading framework for shared charging service based on multiple consortium blockchains. IEEE Trans. on Services Computing, 2023, 16(4): 2437–2450. [doi: 10.1109/TSC.2022.3216659]

[18] Yang F, Qiao YN, Bo JG, Ye LY, Abedin MZ. Blockchain and digital asset transactions-based carbon emissions trading scheme for industrial Internet of Things. IEEE Trans. on Industrial Informatics, 2024, 20(4): 6963–6973. [doi: 10.1109/TII.2024.3354338]

[19] Tao YC, Li B, Li BC. On atomicity and confidentiality across blockchains under failures. IEEE Trans. on Knowledge and Data Engineering, 2024, 36(2): 766–780. [doi: 10.1109/TKDE.2023.3255842]

[20] Yang YH, Bai FH, Yu Z, Shen T, Liu YL, Gong B. An anonymous and supervisory cross-chain privacy protection protocol for zero-trust IoT application. ACM Trans. on Sensor Networks, 2024, 20(2): 32. [doi: 10.1145/3583073]

[21] Ma ZF, Wang JY, Gai KK, Duan PF, Zhang YQ, Luo SS. Fully homomorphic encryption-based privacy-preserving scheme for cross edge blockchain network. Journal of Systems Architecture, 2023, 134: 102782. [doi: 10.1016/j.sysarc.2022.102782]

[22] Tao YC, Li B, Li BC. On sharding across heterogeneous blockchains. In: Proc. of the 39th IEEE Int'l Conf. on Data Engineering. Anaheim: IEEE, 2023. 477–489. [doi: 10.1109/ICDE55515.2023.00043]

[23] Polkadot. Polkadot—LightPaper. 2020. https://wiki.polkadot.network/general/getting-started/

[24] Wang KY, Jia LP, Song ZX, Sun Y. Mitosis: A scalable sharding system featuring multiple dynamic relay chains. IEEE Trans. on Parallel and Distributed Systems, 2024, 35(12): 2497–2512. [doi: 10.1109/TPDS.2024.3480223]

[25] Sha WN, Luo TY, Leng JW, Lin ZS. Heterogeneous multi-blockchain model-based intellectual property protection in social manufacturing paradigm. In: Proc. of the 25th IEEE Int'l Conf. on Computer Supported Cooperative Work in Design. Hangzhou: IEEE, 2022. 891–896. [doi: 10.1109/CSCWD54268.2022.9776286]

[26] Mu K, Wei XT. EfShard: Toward efficient state sharding blockchain via flexible and timely state allocation. IEEE Trans. on Network and

© 中国科学院软件研究所 http://www.jos.org.cn

Service Management, 2023, 20(3): 2817–2829. [doi: 10.1109/TNSM.2023.3236433]

[27] Xu J, Ming YL, Wu ZH, Wang C, Jia XH. X-Shard: Optimistic cross-shard transaction processing for sharding-based blockchains. IEEE Trans. on Parallel and Distributed Systems, 2024, 35(4): 548–559. [doi: 10.1109/TPDS.2024.3361180]

[28] Li CL, Huang HW, Zhao YT, Peng XW, Yang RJ, Zheng ZB, Guo S. Achieving scalability and load balance across blockchain shards for state sharding. In: Proc. of the 41st Int'l Symp. on Reliable Distributed Systems. Vienna: IEEE, 2022. 284–294. [doi: 10.1109/SRDS55811.2022.00034]

[29] Jia LP, Liu YX, Wang KY, Sun Y. Estuary: A low cross-shard blockchain sharding protocol based on state splitting. IEEE Trans. on Parallel and Distributed Systems, 2024, 35(3): 405–420. [doi: 10.1109/TPDS.2024.3351632]

[30] Polkadot. PolkadotPaper. 2016. https://polkadot.network

[31] Xie TX, Gai KK, Zhu LH, Guo YW, Choo KKR. Cross-chain-based trustworthy node identity governance in Internet of Things. IEEE Internet of Things Journal, 2023, 10(24): 21580–21594. [doi: 10.1109/JIOT.2023.3308130]

[32] Zhang PY, Li CX, Zhou MC, Huang WJ, Abusorrah A, Bamasag OO. Transaction transmission model for blockchain channels based on non-cooperative games. Science China Information Sciences, 2023, 66(1): 112105. [doi: 10.1007/s11432-021-3362-9]

[33] Wang ZC, Cui B, Hou WH. A dynamic load balancing scheme based on network sharding in private Ethereum blockchain. In: Proc. of the 46th IEEE Annual Computers, Software, and Applications Conf. Los Alamitos: IEEE, 2022. 362–367. [doi: 10.1109/COMPSAC54236.2022.00057]

[34] Yin HL, Li XR, Gao YX. Relative euclidean distance with application to TOPSIS and estimation performance ranking. IEEE Trans. on Systems, Man, and Cybernetics: Systems, 2022, 52(2): 1052–1064. [doi: 10.1109/TSMC.2020.3017814]

[35] Zhang YZ, Pan SR, Yu JS. TxAllo: Dynamic transaction allocation in sharded blockchain systems. In: Proc. of the 39th IEEE Int'l Conf. on Data Engineering. Anaheim: IEEE, 2023. 721–733. [doi: 10.1109/ICDE55515.2023.00390]

[36] Kashani MH, Mahdipour E. Load balancing algorithms in fog computing. IEEE Trans. on Services Computing, 2023, 16(2): 1505–1521. [doi: 10.1109/TSC.2022.3174475]

[37] Huang HW, Yue ZY, Peng XW, He LD, Chen WH, Dai HN, Zheng ZB, Guo S. Elastic resource allocation against imbalanced transaction assignments in sharding-based permissioned blockchains. IEEE Trans. on Parallel and Distributed Systems, 2022, 33(10): 2372–2385. [doi: 10.1109/TPDS.2022.3141737]

[38] Nguyen CT, Hoang DT, Nguyen DN, Xiao Y, Niyato D, Dutkiewicz E. MetaShard: A novel sharding blockchain platform for metaverse applications. IEEE Trans. on Mobile Computing, 2024, 23(5): 4348–4361. [doi: 10.1109/TMC.2023.3290955]

[39] Zhou K, Zhang XL, Wang CP, Cheng HB. Accelerating cross-shard blockchain consensus via decentralized coordinators service with verifiable global states. IEEE Trans. on Services Computing, 2024, 17(4): 1340–1353. [doi: 10.1109/TSC.2024.3349539]

[40] Huang HW, Lin Y, Zheng ZB. Account migration across blockchain shards using fine-tuned lock mechanism. In: Proc. of the 2024 IEEE Conf. on Computer Communications. Vancouver: IEEE, 2024. 271–280. [doi: 10.1109/INFOCOM52122.2024.10621244]

[41] Nasirtafreshi I. Forecasting cryptocurrency prices using recurrent neural network and long short-term memory. Data & Knowledge Engineering, 2022, 139: 102009. [doi: 10.1016/j.datak.2022.102009]

[42] Rathore RK, Mishra D, Mehra PS, Pal O, Hashim AS, Shapi'i A, Ciano T, Shutaywi M. Real-world model for bitcoin price prediction. Information Processing & Management, 2022, 59(4): 102968. [doi: 10.1016/j.ipm.2022.102968]

[43] Wang JS, Zhu C, Miao C, Zhu R, Zhang X, Tang YH, Huang HX, Gao C. BPR: Blockchain-enabled efficient and secure parking reservation framework with block size dynamic adjustment method. IEEE Trans. on Intelligent Transportation Systems, 2023, 24(3): 3555–3570. [doi: 10.1109/TITS.2022.3222960]

[44] Zhang C, Zhao MY, Zhu LH, Zhang WT, Wu T, Ni JB. FRUIT: A blockchain-based efficient and privacy-preserving quality-aware incentive scheme. IEEE Journal on Selected Areas in Communications, 2022, 40(12): 3343–3357. [doi: 10.1109/JSAC.2022.3213341]

[45] Zheng PL, Zheng ZB, Chen L. Selecting reliable blockchain peers via hybrid blockchain reliability prediction. IET Software, 2023, 17(4): 362–377. [doi: 10.1049/sfw2.12118]

[46] Abdel-Basset M, Mohamed R, Jameel M, Abouhawwash M. Spider wasp optimizer: A novel meta-heuristic optimization algorithm. Artificial Intelligence Review, 2023, 56(10): 11675–11738. [doi: 10.1007/s10462-023-10446-y]

[47] Li XC, Yin XC, Ning JT. Trustworthy announcement dissemination scheme with blockchain-assisted vehicular cloud. IEEE Trans. on Intelligent Transportation Systems, 2023, 24(2): 1786–1800. [doi: 10.1109/TITS.2022.3220580]

[48] Guo YH, Xu MH, Yu DX, Yu Y, Ranjan R, Cheng XZ. Cross-channel: Scalable off-chain channels supporting fair and atomic cross-chain operations. IEEE Trans. on Computers, 2023, 72(11): 3231–3244. [doi: 10.1109/TC.2023.3288765]

[49] Hao XH, Ren W, Fei YY, Zhu TQ, Choo KKR. A blockchain-based cross-domain and autonomous access control scheme for Internet of Things. IEEE Trans. on Services Computing, 2023, 16(2): 773–786. [doi: 10.1109/TSC.2022.3179727]

© 中国科学院软件研究所 http://www.jos.org.cn

[50] Luo XY, Xue KP, Sun QB, Lu J. CrossChannel: Efficient and scalable cross-chain transactions through cross-and-off-blockchain micropayment channel. IEEE Trans. on Dependable and Secure Computing, 2025, 22(1): 649–663. [doi: 10.1109/TDSC.2024.3411820]

[51] Guo ZH, Hu XM. Calculation and selection scheme of node reputation values for notary mechanism in cross-chain. The Journal of Supercomputing, 2024, 80(12): 18177–18198. [doi: 10.1007/s11227-024-06152-3]

[52] Xie TX, Gai KK, Zhu LH, Wang S, Zhang ZJ. RAC-Chain: An asynchronous consensus-based cross-chain approach to scalable blockchain for metaverse. ACM Trans. on Multimedia Computing, Communications, and Applications, 2024, 20(7): 187. [doi: 10.1145/3586011]

附中文参考文献

[12] 岳镜涛, 肖江, 张世桀, 程凤, 陈汉华, 金海. ElasticDAG: 弹性图式区块链. 软件学报, 2024, 35(11): 5279–5305. http://www.jos.org.cn/1000-9825/7050.htm [doi: 10.13328/j.cnki.jos.007050]

[14] 段田田, 张瀚文, 李博, 宋兆雄, 李忠诚, 张珺, 孙毅. 区块链互操作技术综述. 软件学报, 2024, 35(2): 800–827. http://www.jos.org.cn/1000-9825/6950.htm [doi: 10.13328/j.cnki.jos.006950]

[16] 陈晶, 杨浩, 何琨, 李凯, 加梦, 杜瑞颖. 区块链扩展技术现状与展望. 软件学报, 2024, 35(2): 828–851. http://www.jos.org.cn/1000-9825/6954.htm [doi: 10.13328/j.cnki.jos.006954]

作者简介

张佩云, 博士, 教授, CCF 高级会员, 主要研究领域为区块链, 云计算, 边缘计算, 可信计算, 服务计算, 智能信息处理.

刘颖, 硕士, 主要研究领域为区块链.

陈子寒, 硕士生, 主要研究领域为区块链.

www.jos.org.cn

www.jos.org.cn