计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.250200001

基于区块链的轻量级可验证数据管理方法

武沫旬[1] 彭泽顺[1] 于明鹤[2] 李晓华[1] 董晓梅[1] 聂铁铮[1] 于 戈[1]
1 东北大学计算机科学与工程学院 沈阳 110000
2 东北大学软件学院 沈阳 110000
(wumoxun@stumail.neu.edu.cn)

摘 要 结合区块链和数据库两种技术的优势,提出了一种基于区块链的可验证数据管理方法。与现有工作不同,该方法不需要改变原有的数据库存储模式,即可在实现可验证查询和数据溯源功能的同时不影响原有数据库的性能和结构,从而更好地适用于各种应用场景。首先,基于事务的两阶段提交协议设计了一种用户、数据库和区块链的交互模式,以及高效的数据同步机制,以保证区块链与数据库事务执行的原子性。其次,提出了一种基于数据属性编码的索引树结构,以保证区块链进行准确、完整且高效的可验证查询。最后,进行了充分的理论分析和性能测试。所提方法在安全性方面实现了数据的防篡改以及用户隐私的防泄露;在系统有效性方面保证了区块链和数据库的数据一致性;在系统效率方面,与其他典型方案相比,减少了约90%的存储空间,提升了约77%的查询验证效率。

关键词:区块链;可验证查询;数据溯源;分布式数据库
中图分类号 TP393

Approach for Lightweight Verifiable Data Management Based on Blockchains

WU Moxun[1],PENG Zeshun[1],YU Minghe[2],LI Xiaohua[1],DONG Xiaomei[1],NIE Tiezheng[1] and YU Ge[1]
1 School of Computer Science and Engineering,Northeastern University,Shenyang 110000,China
2 Software College of Northeastern University,Shenyang 110000,China

Abstract Combining the advantages of both blockchain and database technologies,this paper proposes a blockchain-based verifiable data management approach. Unlike existing works,this approach does not require changes to the existing database storage mode,allowing for verifiable queries and data traceability without changing the performance and structure of the original database,thus better suiting various application scenarios. Firstly,a two-phase commit protocol is adopted to design an interaction model among users,the database,and the blockchain,along with an efficient data synchronization mechanism to ensure the atomicity of transactions between the blockchain and the database. Secondly,an index tree structure based on data attribute encoding is introduced to support accurate,complete,and efficient verifiable queries on the blockchain. Finally,comprehensive theoretical analysis and performance evaluations are conducted. In terms of security,it achieves data tamper-resistance and user privacy protection. In terms of system effectiveness,it ensures consistency between blockchain and database data. In terms of system efficiency,it reduces storage space by about 90% and improves query verification efficiency by approximately 77% compared to other typical works.

Keywords Blockchain,Verifiable queries,Data traceability,Distributed database

1 引言

区块链[1]作为一种分布式数据库技术,具有去中心化、不可篡改性和可追溯性等特性[2],为可信的数据管理提供了新的解决方案。近年来,区块链技术快速发展,其应用已拓展到身份管理[3-6]、数据管理[7-9]、网络安全[10-11]等领域。但是,区块链系统存在存储容量和效率的限制[12]。这是由于区块链系统中每个节点需要维护完整的数据副本,随着区块链规模的扩大,存储需求呈指数增长,从而带来了巨大的存储成本压力,并且极大地降低了数据的访问效率。

到稿日期:2025-02-05 返修日期:2025-06-15
基金项目:国家自然科学基金面上项目(62372097,62072086);国家社会科学基金重大项目(21&ZD124);中央高校基本科研业务专项资金(N2416003)
This work was supported by the General Program of National Natural Science Foundation of China(62372097,62072086),Major Program of National Social Science Foundation of China(21&ZD124) and Fundamental Research Funds for the Central Universities of Ministry of Education of China(N2416003).
通信作者:于戈(yuge@mail.neu.edu.cn)

近年来,随着对区块链技术需求的增加,在构建以区块链技术支持的复杂应用中面临了一些重要的挑战。例如,在基于时间银行的社区养老服务系统中,海量的养老服务数据不仅包括老年人发布的具体服务需求,还涵盖志愿者或专业机构服务人员为他们提供的详尽服务信息,如服务的基本描述、完成度数据等。为了确保这些数据的真实性、有效性和服务质量的精确评估,养老服务管理部门需要定期进行深度的查询与分析。然而,若将这些数据无差别地上链存储,不仅会给区块链网络带来沉重的存储负担,还可能极大地降低数据的检索效率,从而影响服务响应的及时性和管理的精准度。此外,鉴于这些数据中包含了大量涉及老年人隐私的敏感信息,必须采取严格的隐私保护措施,以维护他们的合法权益。为了妥善处理这些数据,需要综合考虑数据存储、查询效率与隐私保护等多方面的因素,寻求一种既高效又安全的解决方案。

现有的基于区块链的可验证数据管理方法通常依赖全节点和轻节点的协同工作,其中全节点通过维护多个副本(至少为 $2f+1$)来提供强一致性和数据的可验证性。然而,这种模式在大规模数据场景下对存储和计算资源的高需求,使其难以适应资源受限或对效率要求较高的应用场景。轻节点虽然减轻了存储负担,但在访问验证信息时存在性能瓶颈。

为了在现有数据库中实现轻量级的可验证功能并降低系统开销,结合区块链技术,提出了一种面向云数据库的高效验证框架。该框架通过在区块链上仅存储数据的哈希值,将完整数据存储在高可用的云数据库中,显著降低了存储需求和计算开销。此外,该轻量级设计强调对现有基础设施的兼容性,无需对数据库的存储模式进行调整,从而降低了系统的部署和集成成本。

但是,此种设计也存在一定的问题。首先,当数据库完成写入操作但区块链记录失败时,系统可能进入不一致状态,影响事务的完整性。此外,缺乏事务原子性可能引发部分提交或数据丢失的问题,从而进一步削弱系统的可靠性。为应对这个问题,采用两阶段提交协议(2PC)来管理数据库与区块链之间事务处理的一致性与原子性。其次,区块链上数据的公开性可能带来隐私泄露风险。在某些应用场景中,链上数据可能包含敏感信息,如果未加以保护,攻击者可能通过分析区块链信息来获取用户隐私数据。为应对这个问题,采用分层数据编码,保证数据本身以及保存在区块链上的数据索引均被加密,不会泄露任何用户隐私。

综上,本文提出了一种轻量级区块链可验证查询框架,用以实现高效、安全的区块链验证查询。本文的主要贡献如下:

1)提出了一种基于两阶段提交协议的用户、数据库和区块链的交互模式,设计了高效的数据同步机制,以保证区块链与数据库事务执行的原子性,并确保区块链与数据库数据的一致性;

2)提出了一种基于数据属性编码的索引树结构,以保证区块链进行准确、完整且高效的可验证查询;

3)进行了充分的理论分析和性能评价实验,证明了系统的安全性、方法的有效性和系统的高效率。

本文第 2 章介绍相关工作,第 3 章介绍系统架构,第 4 章介绍轻量级可验证查询方案,第 5 章给出安全性分析,第 6 章给出实验评价,最后总结全文。

2 相关工作

在实际应用中,区块链的不可篡改和去中心化特性使其成为验证和存储数据的理想工具。然而,与传统关系型数据库相比,区块链在数据管理方面的性能表现不佳。近年来,区块链与数据存储结合的研究取得了显著进展,许多学者提出了改进存储效率和增强系统安全性的方法。其中,链下存储成为优化区块链存储的主要方向之一。

EtherQL[13] 区块链系统,将区块链数据复制到 MongoDB,利用外部数据库管理数据。可验证查询层 VQL[14],基于区块链提供有效且可验证的查询。BigChainDB[15] 将区块链特性融入 RethinkDB,使系统兼具安全性和高效性。这些方法将链上数据同步到链外,不仅效率低,而且维护成本高。Tsang 等[16] 提出了一种新型的混合链上链下数据的存储与查询框架,将关键信息上链,同时将大容量原始数据存储于链下,从而实现数据完整性验证与高效查询。

SEBDB[17] 区块链数据库引入了关系语义,使得区块链能够支持链上和链间的关系型操作,并通过多种索引设计确保查询效率,其中使用了 B+树作为索引结构。Vchain[18] 框架降低了用户的存储和计算成本,并通过可验证的查询保证了结果的完整性。Cheng 等[19] 提出了首个无需可信第三方的可验证区块链 top-k 查询框架,基于 SMT-based ADS 与倒排 SMT 索引保障结果的完整性,实现任意区间及最新区块的高效可验证 top-k 查询。BLSQ[20] 框架通过链上存储倒排索引元数据、链下加密存储日志,并结合 MBT 索引,实现了日志的不可篡改、高效多关键字及时间范围查询和精细化访问控制。MPV[21] 方案通过设计基于 RSA 累加器的多维奇偶校验验证结构,实现了混合存储区块链中范围查询结果的细粒度认证与错误定位。Zhou 等[22] 提出的基于区块链的可验证搜索加密方案 BCVSE,实现了无需可信第三方的公正验证,并精确支持文本模糊查询。

这些方法尽管优化了区块链的查询性能,但并未从根本上改变区块链在处理大批量数据方面的困境。

FalconDB[23] 由两种类型的实体组成:保存数据库和完整区块链数据的服务器节点。服务器节点负责回答客户端的查询和更新,验证区块链的新块,并生成查询结果的证明。对于区块链来说,这种方法的负载还是过于沉重。SlimChain[24] 仅在区块链上维护账本的状态摘要,而状态数据则存储在链外的专用节点中。Shen 等[25] 提出了一种强一致性的存储架构 Cholula,其能够在离链环境下支持高效的地理复制和对拜占庭攻击的容忍。与其他离链存储方法相比,该框架优化了快速路径延迟,减少了冲突写入时的操作时间。CBCS[26] 通过使用稀疏默克尔多重证明和主从链交叉结构,实现了联盟链中账本数据的协同存储和一致性验证。这些方法尽管减轻了链上存储的负载,但对数据存储和验证方式进行了较大调整,难以直接适配现有数据库系统。

Kraft 等[27] 提出了一种针对多数据存储环境的 ACID 事务协议 Epoxy,其利用多版本并发控制(MVCC)和一个事务协调器来提供一致的事务快照,通过引入复杂的事务管理机

制或优化索引结构实现数据一致性和高效查询。然而，这些方法需要对底层存储系统进行较大改动，增加了实现难度和系统复杂性，无法直接应用于现有数据库。

此外，国内也有关于区块链存储与查询优化的研究。Jiao 等[28]提出了一种可查询且防篡改的数据库，将区块链技术应用于数据管理。Sun 等[29]也提出了一种面向联盟链的链下数据可验证查询方法，将区块链上区块内数据存储在链下数据库中，采用 Hyperledger Fabric 作为区块链平台，使用 Reids 作为链下数据库存储区块数据，结合 MPT 树实现链下数据的查询可验证。但是，它们都需要对数据库侧进行改动，与本文的需求有所不符。

本文所提出的基于联盟链的轻量级可验证数据管理方法不需要改变原有数据库存储模式，既可以实现可验证查询和数据溯源功能，又不影响原有数据库的性能和结构，能更好地适用于各种应用场景。

3 轻量级可验证查询系统结构

系统的架构设计旨在结合区块链与传统数据库的优势，以实现高效、可验证的数据管理功能。整个系统由客户端、多个数据库和区块链组成，如图 1 所示。

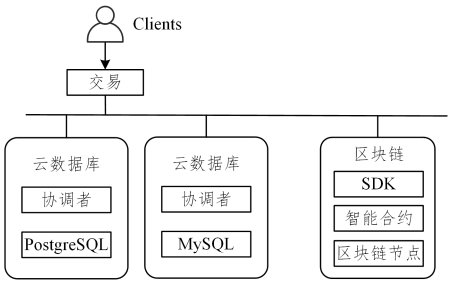

图 1 系统架构

Fig. 1 System architecture

客户端负责发起事务请求，并通过系统提供的接口完成数据查询或更新操作。客户端的操作将被传递给系统中的协同组件，以确保数据的一致性与完整性。

系统支持多个类型的数据库节点（如 MySQL 和 PostgreSQL），这些节点存储了完整的用户数据，并能够高效地处理查询和更新操作。每个数据库节点都配备了一个协调者模块，用于接收客户端请求并与其他组件交互。协调者的主要职责包括以下两点。

1）处理来自客户端的读写操作请求；

2）将操作的结果摘要（如哈希值）传递给区块链模块，以保证数据的可验证性。

区块链作为系统中的可信中介，记录了每次数据操作的验证信息。为了减轻区块链的存储负担，系统采用了轻量级存储设计，仅在区块链上记录数据的哈希值而非完整数据。区块链记录的内容包括事务的哈希摘要、时间戳以及其他相关元信息，用于后续的独立验证。

区块链模块自上而下可以抽象为 SDK 层、智能合约层和区块链节点层。SDK 层向协调器暴露统一的 API，用于对事务进行封装、签名并发送，同时监听区块事件并将提交结果回传；智能合约层承载核心业务逻辑，负责校验事务参数，将哈希摘要写入链上状态并维护索引树根哈希，为后续查询提供可验证依据；最底层的区块链节点层（Peer 节点）执行合约、背书结果并把区块追加到账本与状态数据库，从而保证各联盟成员对账本的一致视图和数据的不可篡改性。

为了应对数据库服务提供商可能面临的安全性问题，本框架设计了一套系统化的安全模式，以确保数据的一致性、完整性及可验证性。

威胁模型：在本文的威胁模型中，数据库服务提供商（如云数据库）被视为提供可信计算和存储服务的平台。然而，数据库中存储的内容可能受到远程用户的非法修改，甚至因硬件故障出现数据丢失。因此，数据库的存储内容被认为是不可信的。具体威胁包括以下两点。

1）远程篡改：恶意用户可能通过非法访问或攻击手段修改数据库中的数据内容。

2）数据丢失：由于单机存储的特性，数据库内容在遭受故意删除或意外丢失后可能无法恢复。

4 轻量级可验证查询处理方法

4.1 事务的原子性

确保区块链与数据库数据的一致性，是系统实现的核心挑战。这需要确保用户提交的数据在数据库和区块链中都得到有效处理，并且数据在两个系统中的状态保持一致。为了解决这个问题，采用了两阶段提交协议，以确保事务的原子性和一致性。协议流程如图 2 所示。

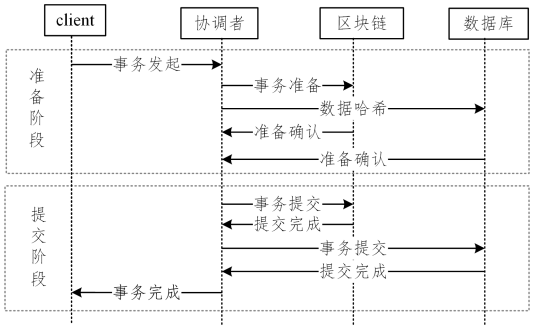

图 2 两阶段提交流程

Fig. 2 Process of 2PC

第一阶段 准备阶段

1）事务发起：客户端通过协调器向相关的数据库节点发起事务请求，包含数据的操作类型（如查询、插入、更新或删除）及内容。

2）事务准备：各数据库节点在接收到请求后，会先执行操作，但不会立即提交结果。相反，这些节点将生成一个包含关键操作数据的哈希摘要（此哈希摘要包括操作类型、受影响的数据记录的关键字段值等，但不包括可能导致不一致的时间戳或其他元数据），并将其发送给协调者。

3）一致性验证：协调器将汇总所有数据库节点的哈希值，协调者在确保所有数据库提交的哈希值相同时，将这个哈希值作为全局数据摘要，并将该摘要提交到区块链中。此时，区块链只暂时记录事务摘要，但不标记为已提交。

4）准备确认：如果所有数据库节点均返回成功，协调器会向各节点发送准备提交的指令；否则，发送中止指令。

第二阶段 提交阶段

1)事务提交:在收到协调器的提交指令后,各数据库节点会正式将事务结果写入持久化存储,并返回提交成功的确认消息;同时,各节点将本地日志状态更新为"已提交"。

2)区块链记录更新:协调器在确认所有数据库节点均提交成功后,将事务状态更新为"已完成",并将对应的哈希摘要在区块链上标记为最终状态。这一步确保了事务的全局可验证性。

3)事务完成:客户端收到协调器的最终事务状态,完成数据操作。如果提交失败,协调器会通知各数据库节点执行回滚操作,以确保数据的一致性。

通过上述过程,系统确保了数据在数据库和区块链中的一致性,并且保证了数据的完整性和安全性。

如果在准备阶段或提交阶段出现错误,数据库或区块链都可以发起回滚请求。在收到回滚请求后,数据库服务器和区块链节点都会将之前的操作回滚到初始状态,确保系统在错误发生时能够有效地进行恢复。

而当用户成为了拜占庭节点后,数据库和区块链由于得不到用户的回应,会视为处理失败,对事务进行回滚。

通过以上两阶段提交协议,可以确保用户提交的数据同时存储在数据库和区块链中,并且保证了数据的一致性。无论在何种情况下,系统都能够有效地处理数据,并且保持系统的稳定性和可靠性。

4.2 并发控制

为了在并发环境中维护全局一致性并提高系统的效率,本文采用了时间戳机制来实现高效的并发控制。

事务集合:$T=\{T_1,T_2,\cdots,T_n\}$。数据项集合:$D=\{x_1,x_2,\cdots,x_m\}$。用 $r_i[x]$ 表示事务 T_i 对数据项 x 的读操作,$w_i[x]$ 表示写操作。

当用户将事务提交到协调器时,协调器为每个事务 T_i 分配一个唯一的、单调递增的时间戳 TS,如式(1)所示:

$$TS:T\rightarrow\mathbb{N},i<j\Rightarrow TS(T_i)<TS(T_j) \tag{1}$$

事务的时间戳在事务开始时确定,并在事务的整个生命周期中保持不变。这个时间戳表示事务的逻辑开始时间。在两阶段提交过程中,除了数据内容与哈希值,这个唯一时间戳也会在协调器和数据库之间传递。

对每个数据项 $x\in D$,维护两类时间戳:

$RTS(x)=\max\{TS(T_i)\mid T_i$已对$x$执行过读$\}$

$WTS(x)=\max\{TS(T_i)\mid T_i$已对$x$执行过写$\}$

在准备阶段结束,事务准备提交时,协调器通过比较数据版本的时间戳来确定是否可以安全地读取或写入数据。

当事务 T_i 请求执行 $r_i[x]$ 时,检查 $TS(T_i)<WTS(x)$ 是否成立,若成立,则说明已有"更晚"事务写过 x,当前读将违反时间戳顺序,中止 T_i;否则,允许读取,并更新。

$$RTS(x)\leftarrow\max(RTX(x),TS(T_i)) \tag{2}$$

当事务 T_i 请求执行 $w_i[x]$ 时,检查 $TS(T_i)<WTS(x)$ 或 $TS(T_i)<WTS(x)$ 是否成立,若任一成立,则说明要写的 x 已被"更晚"事务读或写过,当前写将破坏可串行性,中止 T_i;否则,允许写入,并更新。

$$WTS(x)\leftarrow TS(T_i) \tag{3}$$

时间戳机制通过确保事务按照时间戳的顺序进行提交,保证了全局一致性,避免了不一致的数据读取和脏写。

使用时间戳机制,减少了对锁的需求。传统的锁机制在处理高并发事务时可能会导致大量的等待和死锁。时间戳机制允许基于时间戳排序的事务并行执行,显著减少了对锁的需求,从而减少了等待时间并提高了事务吞吐量。

4.3 数据可验证查询

本文提出了一种先进的查询编码和验证方法。首先,数据用户将查询范围转换为一组特定的立方体编码 CQ,并生成对应的陷门集,以此确保查询数据范围的安全性和隐私性。这些陷门进一步组合成陷门矩阵 $\boldsymbol{M}(Q)$,用于树状索引结构中的快速数据匹配。这种设计使得查询数据的快速定位成为可能,从而提高了查询效率。树状索引参考 4.3.3 小节。

为了在区块链上实现数据的可验证查询,必须全面考虑多个关键因素,如数据的隐私保护,查询结果的正确性,以及结果的完整性。下面将详细介绍为达到这些目标而设计的方案。该方案构建了一个轻量级的区块链可验证查询框架,确保即使在开放的区块链环境中,用户的数据也能得到严格的保护,同时用户还可以对存储在区块链上的数据进行准确且完整的验证。算法 1 对查询算法进行了详细的描述。该算法在不考虑查询条件的情况下的时间复杂度为 $O(\log n)$。

算法 1 可验证查询算法

```
输入:查询条件 Q
输出:查询结果 R,验证对象 VO
codes← EncodeQuery(Q)
trapdoor←GenerateTrapdoor(codes)
for each node intree_index:
    if BloomCheck(trapdoor):
        if node.is_leaf:
            if VerifyMerkleTree(node):
                R←R∪ {node.data}
                v←CreateProof(node.data)
                VO←VO ∪ {v}
        end
    else:
        break;
rerurn 〈R,VO〉
```

在查询过程中,利用布隆过滤器技术进一步加速数据匹配。通过顶部到底部的搜索,可以快速检查每个节点的布隆过滤器,以确定是否包含查询范围内的数据。如果某节点的布隆过滤器与陷门集中的哈希值完全匹配,那么该节点可能包含符合查询条件的数据。这种自顶向下的搜索策略大大减少了不必要的数据访问,进一步提高了查询效率。

找到符合条件的节点后,通过默克尔树提供的验证机制来确保数据的完整性和正确性。默克尔树的设计使得即便在分布式存储环境中,也能通过验证从叶节点到根节点的哈希链来确保数据未被篡改。这种高效的查询编码和数据验证方法,不仅提升了大规模分布式系统中的数据查询和处理速度,也有效保护了数据的安全性,降低了数据泄露和篡改的风险。

该查询方案目前支持单属性查询和范围查询,同时也为未来支持多属性范围查找提供了扩展性,从而为更复杂的

数据检索需求提供了强大的支持。

4.3.1 数据存储

为了在区块链上节约存储空间并保护数据隐私，将数据的哈希值而非原始数据存储到区块链中。哈希函数具有不可逆性，能够有效保护数据内容不被直接访问，从而保证了数据的安全性，并大幅减少了数据泄露的风险。仅存储数据的哈希值，既保障了数据隐私的需求，又维护了数据的完整性和可验证性，使得在需要验证数据时，可以通过比对哈希值来确认数据的真实性和完整性。

4.3.2 数据属性编码

为了使存储在区块链上的数据可以被有效查询，必须对数据进行适当的索引。然而，直接索引可能会暴露敏感数据信息，因此本文采用一种数据属性的编码加密方法来实现既保密又高效的数据索引。

属性范围定义：对于每种数据属性，首先定义其可能的取值范围。例如，将年龄属性的取值范围定义在[0,120]之间。

分层分段编码：将上述定义的范围分为若干一级子片段，再将每个一级子片段细分为二级子片段，依此类推，直至子片段的划分达到所需的精确度。每个子片段按照其层级和数值范围进行编码。

对于树中第 l 层的某节点，其覆盖的子区间记为 $[i,j]$。定义编码函数，如式(4)所示：

$$C_{i,j,l}=H(i \parallel j \parallel l) \tag{4}$$

其中，H 为单向哈希函数，$\parallel$ 表示拼接。

对于一个第 l 层的子片段 x，其范围为 $[i,j]$，通过单向哈希函数计算 $(i+j+l)$ 的哈希值，得到该子片段的唯一编码 C_x。如图 3 所示，数据 D 所处的子片段为 $\{g_1,g_2,g_3\}$。

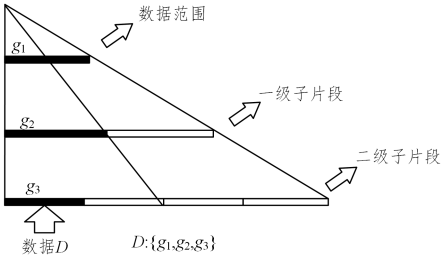

图 3 数据分段示例

Fig. 3 Example of data segmentation

数据编码生成：每条数据记录根据其属性值，被映射到多个这样的子片段中。因此，一条数据记录 D_i 将对应多个层级的子片段，从而形成一组编码 $C_i=\{c_1,c_2,\cdots,c_k\}$，其中 k 是子片段层次的总数。

对于某条数据记录值 $v\in[a,b]$，其对应的编码集合为：

$$C(v)=\{C_{i,j,l} \mid v\in[i,j],0\leqslant l\leqslant k\} \tag{5}$$

从根到叶，所有包含 v 的区间节点的编码都要加入集合，用于后续查询匹配。

查询编码生成：为执行查询，需要生成一组能够覆盖整个查询范围的编码。这是通过从顶层的总段开始，逐级确定与查询范围有交集的子片段来实现的。如果一个子片段完全在查询范围内，则将其编码加入查询编码集；如果只是部分交集，则需要进一步探索其子片段。该过程一直持续到叶节点，最终形成了一个包含所有相关子片段编码的集合。

这种自上而下的数据编码生成策略，不仅使查询过程高效，而且通过精确控制编码的生成，大大减小了误报的可能性。即使在出现误报的情况下，数据用户也可以通过简单地检查不在查询范围内的数据记录来消除误报。算法 2 描述了编码算法的实现过程。该算法构建树的时间复杂度为 $O(n^2)$，查找的时间复杂度为 $O(n)$，其中 n 为树的高度。

上述方法在不牺牲数据隐私的前提下，实现了区块链上数据的高效、安全查询，确保了数据的可验证性与一致性。

算法 2 数据编码算法

输入：数据范围下限 min，数据范围上限 max，编码层数 k，用户数据 data

输出：码结果 C

```
Function buildTree(k,min,max,level)
    if level==k:
        return newTreeNode(min,max,k,level)
    mid←(min+max) / 2
    node← new TreeNode(min,max,k,level)
    node.left ← buildTree(k,min,mid,level 1)
    node.right ← buildTree(k,mid+1,max,level+1)
    return node
Function findDataCode(root,data,result)
    if root=null:
        return
    if data≥ root.start and data ≤ root.end:
        addroot.code to result
    if root.left and data <=root.left.end:
        findDataCode(root.left,data,result)
    if root.right and data >=root.right.start:
        findDataCode(root.right,data,result)
```

4.3.3 索引树构建

本文提出了一种基于数据编码的索引树构造方法，该方法包括构建树、编码树和添加验证信息这 3 个主要步骤，旨在实现数据的高效索引和验证。整个过程强调了数据隐私保护和查询验证的重要性。

本文构建的索引树为块间索引，可以提升定位查询数据所在区块的效率。索引树构建如图 4 所示。

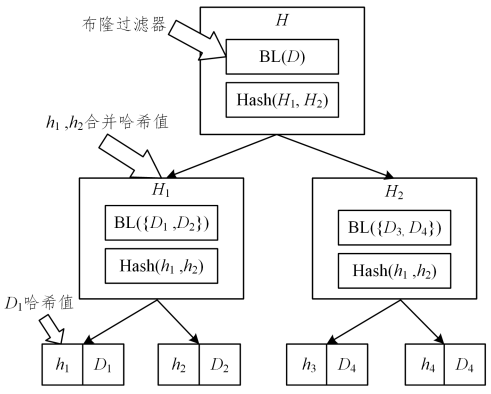

图 4 索引树示意图

Fig. 4 Schematic diagram of index tree

首先，需要构造一个根节点，该根节点包含所有数据记录的编码集合 $D=\{D_1,D_2,\cdots,D_n\}$。接着，将集合 D 随机均匀

地划分为两个子集 D_l 和 D_r,其分别作为根节点的左、右子节点。此过程递归执行,直至每个叶子节点仅包含单一数据记录。在叶子节点中,索引相应的加密数据记录,保证了数据的隐私性和安全性。

其次,为了优化索引结构并提升查询效率,对每个节点的数据不直接存储原始编码,而是采用布隆过滤器。布隆过滤器通过对数据编码进行压缩,不仅显著减少了索引的存储空间,还能快速处理查询请求。具体来说,对于节点 v,构建一个布隆过滤器 $BL(v)$,其中存储了该节点所有数据编码的集合 $D(v)$。

最后,这棵索引树需要具备数据验证的能力。因此本文提出了利用默克尔树结构来增强数据的完整性和验证能力。

默克尔树是一种二叉树,其中每个叶节点包含数据块(如文件、交易记录等)的加密哈希值。非叶节点则包含其子节点的哈希值的合并哈希。持续进行这种从底部到顶部的哈希过程,直到树的顶部形成一个单一的哈希值,称为“根哈希”或“默克尔根”。

每个叶子节点存储对应数据编码的哈希值,而内部节点则存储其子节点哈希值的组合哈希。这种自底向上的哈希链构建方式确保了从任何叶节点到根节点路径上的数据的完整性,从而支持了数据验证的无误性。

默克尔树的引入,使得在不泄露具体内容的情况下,验证数据的完整性和正确性成为可能。通过验证根节点的哈希值,可以确保数据结构中的任何单个元素或子集合未被篡改。这种结构特别适合于分布式系统中的数据验证,如区块链技术,其中任何对数据的修改都需要重新计算从受影响节点到根节点的哈希路径。

这种结构化的索引和验证方法,不仅提高了数据处理的效率,还增强了系统对数据安全和隐私的保护能力。

本文提出的轻量级可验证数据管理方法,能够解决区块链与数据库数据的一致性问题,并提供高效的安全保障。通过两阶段提交协议,框架确保了数据在数据库和区块链中的一致性。该协议分为准备阶段和提交阶段,若出现错误,系统可通过回滚机制恢复状态。在数据可验证查询方面,框架采用了查询编码与验证技术,包括生成特定的数据属性编码和陷门集,以保护数据隐私,并通过布隆过滤器加速数据匹配。默克尔树结构则用于验证数据的完整性,确保数据在分布式存储环境中的准确性和安全性。这些措施有效提升了系统的稳定性、可靠性和查询效率。

5 安全性分析

在设计本文的轻量级可验证数据管理框架时,考虑了一系列安全威胁,特别是来自不可信区块链全节点和数据库的潜在风险。系统的安全框架是在严格的威胁模型下构建的,以确保能够有效防御这些威胁并维护数据的完整性和隐私。

数据的可验证性:本方案中数据会经过两轮验证,以确保数据的正确性和完整性。

1)当用户在数据库中执行操作时,协调器模块会对操作涉及的数据计算一份不可逆的一致性哈希,并将其存储于区块链中。用户需要验证时,将数据库中的数据进行哈希,再与区块链中的数据进行对比验证,确保数据正确无误。

2)当用户根据查询条件从区块链中查询到数据哈希之后,协调器利用默克尔索引树为每条命中记录构造验证对象,收集从记录到所在叶节点到根节点的路径上所有兄弟节点的哈希以组成成员证明,若遇到与查询陷门不相交的子树,则返回该节点哈希作为剪枝证据。用户最终算出根哈希并与链上存根对比,即可保证查询范围外不存在遗漏或冗余数据,实现正确性与完整性的双重保障。

数据库的安全性:在本文构建的威胁模型中,数据库服务提供商(如云数据库平台)被假定为提供可信的计算与存储服务。尽管如此,存储于数据库中的数据可能遭遇远程用户的未授权修改,或由于硬件故障而发生数据丢失的事件。基于这些潜在风险,数据库内的存储内容应被视为不可信赖的资源。这种假设促使系统设计必须能够在存储环境不可信的情况下,保障数据的完整性和安全性。

1)两阶段提交协议:本文引入两阶段提交协议,要求数据库和区块链系统之间的每一项数据操作都必须经过双方的同步确认。此机制确保了数据更改需要在两个系统之间达成一致,从而防止数据库在执行事务时的恶意操作。两阶段提交协议不仅提供了数据一致性保证,还能防止数据库管理员或恶意攻击者单方面对数据进行篡改或删除。通过实现双重确认,系统有效地降低了数据丢失或业务流程混乱的风险,提高了系统的可靠性。

2)数据一致性审核:系统允许用户定期对数据库和区块链中的数据进行一致性校验。此措施旨在检测数据库和区块链间的数据不一致性,从而揭示可能的恶意操作。一致性审核机制提供了额外的数据完整性检查点,有助于及时发现并纠正数据的异常情况。这不仅提升了系统的可信度,还为数据管理提供了一个有效的监控工具,从而降低了潜在的数据安全风险。

区块链节点的安全性:尽管区块链以其去中心化和数据不可篡改的特性被广泛认为是安全的,但在某些情况下,如51%攻击节点共谋,区块链仍然可能面临作恶的风险。这些风险可能导致区块链记录的数据被恶意修改或替换。针对这些风险,本文采取了对应的应对措施。

1)默克尔树验证:通过为每个数据块构建默克尔树,并将根哈希存储在区块链上,使得即使在节点部分不可信的情况下,用户也可以独立验证任何数据块的完整性。确认计算得到的根哈希与区块链上存储的根哈希相匹配,以验证数据的完整性和未被篡改。默克尔树验证机制提供了一种强大的数据完整性保障方法,即使在区块链节点发生部分不可信的情况下,也能确保数据未被篡改。这种机制显著提高了数据验证的可靠性,有助于防止由节点攻击或共谋造成的数据篡改。

2)加密存储:首先,对于用户数据在区块链上存储的哈希值,不具备任何有效信息。其次,在区块链上对数据所要查询的属性也进行了复杂的哈希编码。加密存储机制能有效防止区块链节点从存储的哈希值中获取用户数据的实际内容,从而保护了数据隐私。通过确保只有哈希值而非实际数据被存储,系统降低了敏感信息泄露的风险,并提升了数据的安全性。

6 实验结果与分析

实验采用 Intel(R) Xeon(R) Silver 4110 CPU @ 2.10 GHz 处理器，内存为 16 GB，源代码基于笔者团队开发的联盟区块链系统 NEUchain[30] 实现，由 C++编程语言实现。使用 Ethereum 数据集 1 与 foursquare 数据集 2 进行实验，以下简称数据集为 ETH 和 4SQ。通过在 ETH 中溯源以太币的产生与使用过程，以及在 4SQ 中追踪人员位置变化，来验证上文提出的轻量级可验证数据管理方法。

6.1 存储效率

表 1 列出了使用 ETH 和 4SQ 两个数据集对原数据进行存储和对轻量级处理后的数据进行存储所占用的存储空间。

根据表 1 中的数据，采用轻量级存储数据的方法相对于原始数据在 ETH 和 4SQ 数据集上都显著减少了存储空间。对于 ETH 数据库，原始数据占用了 22.6 MB 的存储空间，而采用轻量级存储数据后仅占用了 2.04 MB 的存储空间，减少了约 90%。对于 4SQ 数据库，原始数据占用了 75.7 MB 的存储空间，而采用轻量级存储数据后仅占用了 11.7 MB 的存储空间，减少了约 85%。

表 1 存储空间

Table 1 Storage space

(MB)

数据集	原数据	轻量级存储数据
ETH	22.6	2.04
4SQ	75.7	11.7

通过采用轻量级存储数据的方法，本方案能够显著地减少区块链的存储空间，从而降低了存储成本并提高了存储效率。这种减少存储空间的效果对于区块链数据库尤为重要，因为区块链通常会产生大量的数据，而有效地减少存储空间可以降低维护成本并提高系统整体性能。

6.2 两阶段提交消耗时间

本节分别对准备阶段和提交阶段的消耗时间进行了实验。图 5 展示了准备阶段和提交阶段的时间消耗与交易数量之间的关系。在准备阶段，随着交易数量的增加，时间消耗几乎保持稳定，表明系统在处理准备阶段任务时具有较高的效率和稳定性。

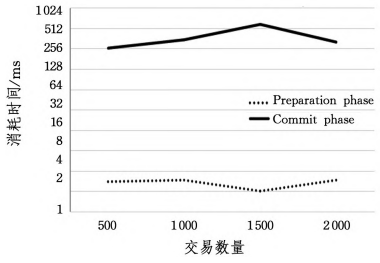

图 5 准备阶段和提交阶段时间的消耗

Fig. 5 Preparation phase and commit phase time consumption

在提交阶段，时间消耗略有增加。虽然交易数量在一定范围内波动较小，但随着交易数据的上链，提交阶段的时间消耗相对较高。这是因为提交阶段需要将数据通过区块链进行上链验证和存储，因此相比准备阶段，提交阶段的处理涉及更多的计算和数据传输，导致时间消耗略有增加。

实验结果显示，准备阶段保持低而稳定的时间消耗；而提交阶段则由于数据上链的复杂性而需要更长的处理时间，但依然能够在较短的时间内完成任务。

6.3 验证效率

本节对所提方法验证所消耗的时间进行了实验。从原始数据集中随机选取数百万条数据记录，然后对不同数据量的查询结果的验证时间进行评估。

图 6 展示了本文方法与 vchain+在面对不同数量的查询结果时的验证时间对比。从结果上看，两种方法所消耗的时间相差不多，并且都随着数据量的增大而增加。这是因为，两种方法都需要进行默克尔证明，本质上相差不多。本文方法使用布隆过滤器，并且相比 vchain+，不涉及更加复杂的查询，故验证效率更高。

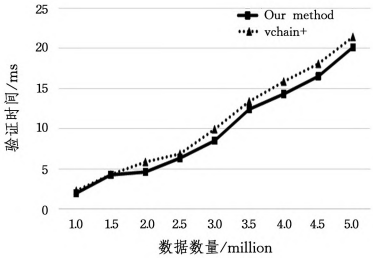

图 6 验证延迟

Fig. 6 Verify delay

6.4 查询效率

本节进行了针对范围查询的性能实验。实验的具体流程是：首先随机生成 10 个查询；然后执行这些查询，并分别计算每个查询的全节点查询延迟时间；最后对这些查询的延迟时间进行平均。实验分别对不同区块数目下的查询性能进行了测试。

图 7 和图 8 分别展示了建立索引与无索引在 4SQ 和 ETH 数据集上的性能比较结果。从结果中可以明显看出，本文方案在性能上有着较为显著的提升。以 4SQ 范围查询为例，当查询区块数量为 8 192 时，所提方案的查询时间仅为 0.048 s；相比之下，无索引的查询时间则显著延长，优化了约 71%。值得注意的是，随着区块数量的逐渐增加，本文索引方案也呈现出较为稳定的性能，不会出现过高的波动。这说明，所提方案在处理大规模数据时依然能够保持较高的性能水平，具有一定的稳定性和可靠性。

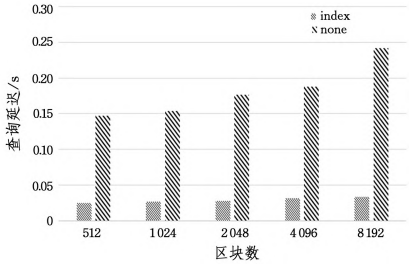

图 7 查询延迟(4SQ)

Fig. 7 Query delay(4SQ)

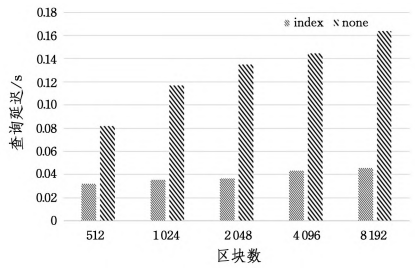

图 8 查询延迟(ETH)

Fig. 8 Query delay(ETH)

图 9 展示了本文方案与同样结合区块链技术与数据库技术的 SEBDB 系统进行对比实验的结果。实验测试了在不同区块数量的情况下,本文方案与 SEBDB 方案在可验证查询上所花费的时间。最终实验表明,本文方案相比 SEBDB 优化了约 77.04%。

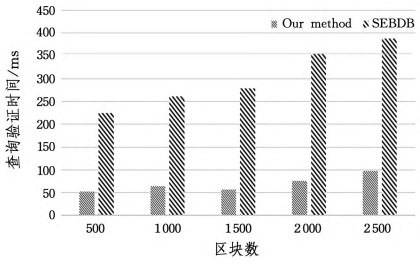

图 9 查询验证时间(4SQ)

Fig. 9 Query verification time(4SQ)

综上所述,本文索引方案在可验证查询性能上表现出明显的优势,不仅能够有效降低查询延迟,而且在处理大规模数据时仍然能够保持稳定性,为数据查询和处理提供了有效的解决方案。

结束语 本文提出并实现了一个基于区块链的轻量级可验证数据管理系统,该系统有效融合了区块链的安全特性与传统数据库的高效性能,实现了在不修改原有数据库的基础上的区块链轻量级数据可验证查询。根据一系列分析与实验,系统在数据一致性、安全性、查询效率等方面均表现出色。未来的研究将进一步探索和优化算法的效率,增强系统对更复杂场景的适应性。

参 考 文 献

[1] YU G,NIE T Z,LI X H,et al. The challenge and prospect of distributed data management techniques in blockchain systems [J]. Chinese Journal of Computers,2021,44(1):28-54.

[2] HE P,YU G,ZHANG Y F,et al. Survey on blockchain technology and tts application prospect[J]. Chinese Journal of Computers,2017,44(4):1-7,15.

[3] CHEN J,YAO S,YUAN Q,et al. Certchain:public and efficient certificate audit based on blockchain for TLS connections[C]// Proceedings of IEEE INFOCOM. 2018:2060-2068.

[4] KUBILAY M Y,KIRAZ M S,MANTAR H A. CertLedger: A new PKI model with certificate transparency based on blockchain[J]. Computers & Security,2019,85:333-352.

[5] SHEN M,LIU H,ZHU L,et al. Blockchain-assisted secure device authentication for cross-domain industrial IoT[J]. IEEE Journal on Selected Areas in Communications,2020,38(5):942-954.

[6] ADJA Y C E,HAMMI B,SERHROUCHNI A,et al. A blockchain-based certificate revocation management and status verification system[J]. Computers & Security,2021,104:102209.

[7] TRUONG N B,SUN K,LEE G M,et al. GDPR-compliant personal data management:a blockchain-based solution[J]. IEEE Transactions on Information Forensics and Security,2019,15:1746-1761.

[8] XIONG Z,ZHANG Y,LUONG N C,et al. The best of both worlds:A general architecture for data management in blockchain-enabled Internet-of-Things [J]. IEEE Network, 2020, 34(1):166-173.

[9] SHAFAGH H,BURKHALTER L,HITHNAWI A,et al. Towards blockchain-based auditable storage and sharing of IoT data[C]// Proceedings of the 2017 on Cloud Computing Security Workshop. 2017:45-50.

[10] HARI A,LAKSHMAN T V. The internet blockchain:A distributed,tamper-resistant transaction framework for the internet[C]// Proceedings of the 15th ACM Workshop on Hot Topics in Networks. 2016:204-210.

[11] XING Q,WANG B,WANG X. POSTER:BGPCoin:A trustworthy blockchain-based resource management solution for BGP security[C]// Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017:2591-2593.

[12] SHAO Q F,JIN C Q,ZHANG Z,et al. Blockchain:architecture and research progress[J]. Chinese Journal of Computers,2018,41(5):969-88.

[13] LI Y,ZHENG K,YAN Y,et al. EtherQL:A query layer for blockchain system[C]// Proceedings of DASFAA. 2017:556-567.

[14] WU H,PENG Z,GUO S,et al. VQL:Efficient and verifiable cloud query services for blockchain systems[J]. IEEE Transactions on Parallel and Distributed Systems,2021,33(6):1393-1406.

[15] MCCONAGHY T,MARQUES R,MÜLLER A,et al. Bigchaindb:a scalable blockchain database[R]. Berlin:White Paper,BigChainDB,2016:53-72.

[16] TSANG Y P,LEE C K M,ZHANG K,et al. On-chain and off-chain data management for blockchain-internet of things:a multi-agent deep reinforcement learning approach[J]. Journal of Grid Computing,2024,22(1):16.

[17] ZHU Y,ZHANG Z,JIN C,et al. SEBDB:Semantics empowered blockchain database[C]// Proceedings of ICDE. 2019:1820-1831.

[18] XU C,ZHANG C,XU J. Vchain:Enabling verifiable boolean range queries over blockchain databases[C]// Proceedings of SIGMOD. 2019:141-158.

[19] CHENG J,QI S,AN B,et al. Lightweight verifiable blockchain top-k queries[J]. Future Generation Computer Systems,2024,156:105-115.

[20] LI W, FENG Y, LIU N, et al. A secure and efficient log storage and query framework based on blockchain[J]. Computer Networks, 2024, 252: 110683.
[21] LIU Q, PENG Y, XU M, et al. MPV: Enabling fine-grained query authentication in hybrid-storage blockchain[J]. IEEE TKDE, 2024, 36(7): 3297-3311.
[22] ZHOU F, JIAO Z, WANG Q, et al. BCVSE: Verifiable searchable encryption scheme with blockchain supporting fuzzy query [J]. Arabian Journal for Science and Engineering, 2024, 49(3): 4401-4418.
[23] PENG Y, DU M, LI F, et al. FalconDB: Blockchain-based collaborative database[C]// Proceedings of SIGMOD. 2020: 637-652.
[24] XU C, ZHANG C, XU J, et al. SlimChain: Scaling blockchain transactions through off-chain storage and parallel processing [J]. PVLDB Endowment, 2021, 14(11): 2314-2326.
[25] SHEN D, DUMAS C, SARDINA C, et al. Cholula: Fast, fault-tolerant, and stronglyconsistent off-chain object storage[C]// Proceedings of 4th International Conference on Blockchain Computing and Applications. IEEE, 2022: 189-194.
[26] ZHOU J, WANG N, LIU A, et al. Cbcs: A scalable consortium blockchain architecture based on world state collaborative storage[J]. Electronics, 2023, 12(3): 735.
[27] KRAFT P, LI Q, ZHOU X, et al. Epoxy: ACID transactions across diverse data stores [J]. PVLDB Endowment, 2023, 16(11): 2742-2754.
[28] JIAO T, SHEN D, NIE T. BlockchainDB: Querable and immutable database[J]. Journal of Software, 2019, 30(9): 2671-2685.
[29] SUN Y M, FAN H B, PENG M Y, et al. A method of off-chain data verifiable query for consortium blockchain [J]. Modern Electronics Technique, 2023, 46(19): 70-74.
[30] PENG Z, ZHANG Y, XU Q, et al. NeuChain: a fast permissioned blockchain system with deterministic ordering [J]. PVLDB Endowment, 2022, 15(11): 2585-2598.

WU Moxun, born in 2000, postgraduate, is a student member of CCF(No. N5671G). His main research interests include query processing and blockchain technology.

YU Ge, born in 1962, professor, Ph. D supervisor, is a member of CCF (No. 05408F). His main research interests include distri-buted system and big data management.

(责任编辑:李亚辉)