计算机研究与发展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202330272
60(11): 2455−2468, 2023

# 基于学习索引的图式区块链高效可验证查询机制

常 健 林立成 李彬弘 肖 江 金 海
(华中科技大学计算机科学与技术学院 武汉 430074)
(大数据技术与系统国家地方联合工程研究中心(华中科技大学) 武汉 430074)
(服务计算技术与系统教育部重点实验室(华中科技大学)武汉 430074)
(集群与网格计算湖北省重点实验室(华中科技大学)武汉 430074)
(j_chang@hust.edu.cn)

# Efficient and Verifiable Query Mechanism of DAG Blockchain Based on Learned Index

Chang Jian, Lin Licheng, Li Binhong, Xiao Jiang, and Jin Hai
(*School of Computer Science and Technology*, *Huazhong University of Science and Technology*, *Wuhan* 430074)
(*National Engineering Research Center for Big Data Technology and System* (*Huazhong University of Science and Technology*), *Wuhan* 430074)
(*Key Laboratory of Services Computing Technology and System* (*Huazhong University of Science and Technology*), *Ministry of Education*, *Wuhan* 430074)
(*Key Laboratory of Cluster and Grid Computing* (*Huazhong University of Science and Technology*), *Wuhan* 430074)

**Abstract** Blockchain system has been broadly applied and studied in the last few years with query being essential to facilitate a wide range of blockchain application scenarios such as data provenance in the logistics chain. As blockchain data continues to grow, the efficiency and verifiability of data query have become hot issues in blockchain research. The high concurrent blocks of the DAG-based blockchain make it difficult for queries to traverse sequentially like the traditional chain structure. Breadth-first or depth-first query strategies can be adopted according to the DAG structure, but this method has the problems of low efficiency and poor verification. We propose an efficient and verifiable query mechanism, called Lever, for DAG-based blockchains based on the learned index. The mechanism aims to improve the efficiency and verifiability of blockchain query by introducing learned index technology with the data distribution characteristic to optimize the indexing process in the blockchain. Learned index is a novel indexing technology that reduces the storage space and query time of indexes by learning the distribution of data. In this paper, learned index is applied to the mapping between epoch height and timestamp of the DAG-based blockchain, greatly improving query speed and efficiency. At the same time, in order to speed up the data filtering of multiple blocks in an epoch, a Bloom filter is added to the block header, and an aggregated Bloom filter is produced by miner for each epoch. In addition, to guarantee the authentication (i.e., correctness and completeness) of query results, the mechanism introduces a sorted Merkle tree to perform proof with the Merkle tree branch by combining the

**收稿日期**: 2023-04-03; **修回日期**: 2023-06-12
**基金项目**: 国家重点研发计划项目 (2021YFB2700700); 湖北省重点研发计划项目 (2021BEA164); 国家自然科学基金项目 (62072197); 广东省重点研发计划项目 (2020B0101090005)
This work was supported by the National Key Research and Development Program of China (2021YFB2700700), the Key Research and Development Program of Hubei Province (2021BEA164), the National Natural Science Foundation of China (62072197), and the Key Research and Development Program of Guangdong Province (2020B0101090005).
**通信作者**: 肖江(jiangxiao@hust.edu.cn)

filtering and sorting processes, effectively reducing the size of the verification object (*VO*), thereby enhancing the verifiability of blockchain query. Experimental results show that Lever can effectively improve the efficiency and verifiability of DAG-based blockchain query, and Lever improves query performance by up to 10 times and reduces *VO* size overhead by 90% compared with the basic query mechanism of Conflux.

**Key words** DAG blockchain; verifiable query; learned index; aggregated Bloom filter; sorted Merkle tree

**摘 要** 区块链技术近年来受到了广泛关注,并应用于各个领域,数据查询是其在应用过程的一个重要技术,如物流链中的数据溯源等.随着区块链系统中交易数据量的持续增长,支持高并发事务处理的图式区块链成为区块链技术的研究热点.图式区块链的高并发区块使得数据查询难以像传统链式结构依次遍历,可以根据图式结构采用广度优先或深度优先遍历策略,但这种查询方式存在效率低、验证难等问题.针对图式区块链数据查询的效率和可验证性问题,提出了一种基于学习索引的高效可验证的图式区块链查询机制 Lever.该机制通过引入学习索引技术对图式区块链中时序数据分布特征进行学习以实现对索引过程的优化,旨在提高图式区块链查询的效率和可验证性.学习索引是通过学习数据分布来减少索引存储空间和查询时间的新型索引技术,将学习索引应用于图式区块链的纪元高度与时间戳的映射关系中,通过函数运算的方式定位查询数据,提高查询速度和效率.同时,为了加快纪元内多个区块数据的过滤速度,在每个区块头部添加布隆过滤器,并为每个纪元生成一个聚合布隆过滤器,从而提高纪元内的数据遍历速度.此外,为保证查询结果的正确性和完整性,该机制结合布隆过滤器和排序默克尔树生成可验证对象,通过部分默克尔树分支实现对布隆过滤器假阳性的不存在证明,有效减小验证对象的规模,从而提高图式区块链查询过程的数据传输效率.实验结果表明,Lever 能有效提高基于 DAG 的图式区块链查询效率和可验证性,与 Conflux 的基本查询机制相比,该机制的查询性能最高提升了 10 倍,可验证对象大小开销可以降低 90%.

**关键词** 图式区块链;可验证查询;学习索引;聚合布隆过滤器;排序默克尔树

**中图法分类号** TP311

区块链技术具有分布式、去中心化和防篡改等特性,已被广泛应用于数字金融、供应链等各个领域.然而,由于传统链式区块链的数据存储结构和共识机制等限制[1],现有的区块链系统普遍存在交易处理确认延迟高、吞吐率低和可扩展性差等局限性,限制了其应用范围.为了解决这些问题并满足实际应用需求,研究学者提出了基于有向无环图(directed acyclic graph, DAG)的图式区块链方案,将传统链式结构扩展成图式结构,如 IOTA [2], Byteball[3], Conflux[4] 等.这些图式区块链解决方案也带来新的挑战,由于交易的并发度高,数据查询无法像传统链式结构依次遍历.根据图式区块链结构可以采取广度优先遍历策略或深度优先遍历策略,这些查询方式存在效率低、可验证性差等问题.本文提出了一种基于学习索引的图式区块链高效可验证查询机制,该机制通过引入学习索引结构和基于图式区块链时序数据特征的查询优化策略,实现了高效的图式区块链数据查询和验证.同时,该机制能够有效应对索引结构和验证开销大等问题,为图式区块链技术的应用发展提供了一种新的查询方案的技术支持.

近年来,为提高吞吐量、降低交易时延,图式区块链应运而生,其结构与传统的链式区块链相比也有较大变化.图式区块链结构扩展了传统链式区块链的数据结构和共识机制,它采用基于图数据结构 DAG 的分布式账本技术,不仅能够保证事务处理的高并发,同时其独特的拓扑排序机制能够实现交易数据保持全局有序.图式区块链采用与传统链式结构相同的 P2P 网络实现整个网络节点的连接,所有节点对等且保持同步,保证系统的安全和可靠性.每个全节点都具有保存维护和验证全网数据区块、转发和广播交易信息、支持轻节点数据查询等功能.IOTA 以每笔交易作为 DAG 的顶点[2],这种方式不需要挖矿节点对交易数据验证打包,用户发起交易不需要产生交易费,交易之间可以并行处理从而提高了整个网络的吞吐量.在整个图式区块链的执行过程中,新交易对它之前产生的 2 笔或以上交易进行引用,从而完成交易数据验证. Conflux 以区块作为 DAG 顶点形成树图结构[4],矿工向图式区块链系统中并行产生新的交易数据时,全节点将在本地保存该交易并通过 P2P 网络广播给相邻的节点.矿工可以并行打

包区块，这些区块引用多个当前最新的区块，从而使得各区块之间并发产生的同时保持全局有序.

然而，图式区块链这种基于 DAG 的数据结构，给查询带来了 2 方面的挑战. 首先在查询效率方面，不同于传统的链式区块链，图式区块链的交易并发度高，交易数据规模更为庞大，区块间引用关系多样. 2 个相邻的纪元(epoch)之间可能包含着数个并发区块，数据结构更为复杂，使得遍历区块链数据的查询方式效率低下. 此外，图式区块链依然采用 LevelDB[5] 数据存储方式. 这种结构主要是针对写密集型的应用场景，在提升写效率的同时降低了读数据的效率[6]，难以满足实际应用场景的需求，图式区块链亟需一种高效索引机制来加速查询. 另一方面则是查询结果的可验证性，与传统中心化数据库管理方式的不同之处在于，区块链系统采用分布式全账本形式[7]，在不可信环境中数据是分布式存储在多个全节点上. 随着区块链系统中产生的历史数据不断累积[8]，客户端对历史数据查询的需求也将增加，这使得系统需要支持对可能存在恶意节点的查询结果进行验证. 因此，在实现高效查询的同时，保证图式区块链数据查询结果的可验证性成为另一个重要方面.

高效的索引机制设计是加快数据查询的重要手段之一. 通过使用索引，服务提供者可以快速定位所需数据，而无需扫描整个数据集. 现有的区块链查询工作大多聚焦于传统的树结构索引(如 B+树索引[9])，其主要是基于数据指针结构，索引结构被提前载入内存当中，这种缓存索引机制仅适用于小规模数据库并能够提升查询效率. 但是，区块链数据量只增不减，且成指数级规模递增. 当规模达到 TB 级甚至 PB 级时，树结构索引所占用的空间也随之递增，此时索引结构将不能全部载入内存，只能不断通过 I/O 访问磁盘，导致查询性能降低. 总体而言，传统的树结构索引在大规模数据情况下存在 2 方面主要问题：1)存储空间占用大. 例如，B+树索引的空间复杂度为 $O(n)$，数据量只增不减的情况下，索引结构所占用的存储空间不可忽视. 2)查询速度变慢. 例如，树结构索引每次遍历都需要从树根节点依次访问整棵树的深度直到叶子节点，路径上不相关的间接节点也都需要被访问，其时间复杂度为 $O(\log n)$，这使得在大规模数据上的查询性能变得较差.

针对传统树结构索引存在的这些问题，一种新的数据索引结构——学习索引[10]，近年来被研究者提出，它是一种结合机器学习和索引机制的查询优化技术[11]. 本文观察图式区块链数据的时序分布特征，根据时间戳与纪元间的关系，采用适合的学习模型自动学习出最优的索引函数，并将其参数作为索引存储在列表中. 当进行数据查询时，节点可以通过模型计算快速定位查询数据所在的大概位置. 相比传统的树结构索引，学习索引的查询效率更高、扩展性更好，更适用于区块链系统. 本文利用图式区块链数据的分布规律，将机器学习训练方法应用于图式区块链数据索引的构建过程中，其目的是通过参数列表代替传统树结构索引，降低区块链系统中索引所占的空间大小，同时利用函数运算的方式代替对树结构索引的依次遍历，提升索引的查询性能.

本文的主要贡献包括 3 个方面：

1)针对图式区块链的数据分布特征，提出了一种基于学习索引的高效查询机制 Lever，加快图式区块链数据查询.

2)为了提升图式区块链纪元内区块的遍历速度，设计聚合布隆过滤器，该机制能有效过滤不存在的查询数据，提升查询效率；针对布隆过滤器存在假阳性的问题，提出排序默克尔树结构，该机制能够保证查询结果的可验证性并降低可验证对象的大小.

3)实验结果表明，与 Conflux 的基本查询机制相比，Lever 的查询性能最高可以提升 10 倍，可验证对象大小最多能够降低 90%.

接下来，本文将分析现有的区块链查询相关技术和工作，并总结现有工作存在的局限性. 然后介绍图式区块链的基本概念以及本文工作将涉及的相关技术，紧接着针对图式区块链的数据分布特征提出一种基于学习索引的查询优化方法，阐述图式区块链索引构建的实现细节和性能优化策略.

## 1 相关工作

随着区块链技术的广泛应用，用户对区块链上数据查询的功能和效率提出越来越高的要求[12]，特别是在区块链金融、电子商务、供应链等在线分析应用场景中. 然而原生的区块链系统存在查询功能单一、查询效率低和查询可信验证难等问题. 当前区块链查询相关研究主要聚焦在链式区块链上的高效查询、可验证查询和功能性查询.

1)高效查询. 贾大宇等人[13] 提出了一种区块链高效查询方法 ElasticQM，利用 B-M 树实现高效的数据查询. 在查询层增加查询超节点和校验节点，用户可以缓存查询结果，再次查询相同数据时可以加快

数据访问速度, 提高查询效率. Ruan 等人[14]提出了一个细粒度、安全、高效的区块链数据追溯系统 Lineagchain, 支持智能合约的账户数据查询和历史数据追溯. 在智能合约执行期间获取溯源信息, 并有效地存储在默克尔树中. 此外, Lineagchain 提供了一种新的跳表索引结构, 支持快速高效的溯源查询处理. Zhang 等人[15]提出了 GEM$^2$-Tree 数据结构来减少查询过程中产生的 gas 开销. 蔡磊等人[16]对分布在多个区块中的数据进行有效处理, 通过建立物化视图, 利用字典树加快多物化视图维护进程, 以区块为单位提高查询性能. 针对 Hyperledger Fabric 平台的时间查询问题, Gupta 等人[17]提出 TQF 框架改进 Fabric 上关于时间数据查询的性能, 将数据的时间范围按照不同的时间间隔进行划分, 建立细粒度时间片数据索引, 提高 Fabric 上时间查询的性能. 总结发现, 这些数据结构优化并未有效利用区块链系统具有的时序特征, 对于查询效率的提高有限.

2)可验证查询. 为了在轻节点上实现可验证查询, 降低用户的存储和计算成本, Xu 等人[18]提出了一种在不可信环境下的可验证查询处理框架 vChain. 轻节点客户端需要接收和验证全节点返回的查询结果和累加器验证对象, 其中累加器的密钥需要占用大量的网络和计算开销. 为了实现完整性验证, Dai 等人[19]提出一个轻量级的可验证查询方法 LVQ, 降低节点存储需求和网络开销, 实现轻量化数据验证, 减少轻节点的数据存储. 通过将布隆过滤器(Bloom filter, BF)的哈希值存储在区块的头部, LVQ 引入了一种新的 BMT 结构用于轻量级查询, 该结构通过合并多个连续的布隆过滤器来降低查询结果的通信代价. 针对部分数据存储在链上、原始数据存储在链下的混合存储架构, Zhang 等人[20]研究了基于混合存储区块链中的可验证范围查询. 针对外包数据的完整性验证问题, Hao 等人[21]提高了数据完整性验证的安全性和效率, 而不需要借助第三方审计. 除此之外, 另外一种方法通过可信执行环境实现轻节点上的可验证查询, Shao 等人[22]提出了一种基于可信硬件 Intel SGX 的可验证查询认证方式. 轻节点无需接收或验证全节点返回的查询结果和加密证明, 减少了网络传输和本地计算的开销, 并通过基于冷热数据的分层存储方案来缓解可信内存空间有限的问题. 在可信飞地(Enclave)和非可信内存中分层组织数据, Enclave 中只缓存热数据, 冷数据存于非可信内存中. 这种方法依赖于可信硬件, 但是 Enclave 的可信内存有限仅 128 MB, 其性能和处理能力无法实现大规模应用. 目前这些验证机制主要是针对串行区块的链式结构, 而无法直接应用于含有并行区块的图式结构中.

3)功能性查询. 对于功能性查询, 由于区块链底层基于 Key-Value 键值对的存储方式在一定程度上导致了查询功能单一的问题, 并且随着账本数据规模增大, 进一步导致查询速度降低. EtherQL[23]和 VQL[24]采用基于以太坊设计的数据查询层, 以提取存储在底层区块链系统中的交易, 为区块链系统提供高效且可验证的数据查询服务, 通过间接查询分布式数据库而不是 P2P 网络来获得交易和区块. 查询层维护了一个专门的分布式数据库, 可以提供包括范围查询和 TOP-$K$ 查询在内的分析区块链数据的高效查询方式. Zhu 等人[25]提出并实现了新的区块链数据库 SEBDB, 设计了类似数据库 SQL 语句的查询语句和索引结构, 查询效率进一步提高. SEBDB 是首个区块链数据库平台, 同时兼顾可用性和可扩展性. 首先, 系统将事务表示为具有多个属性的元组存储在预定义表中. 其次, SEBDB 使用类 SQL 语言作为通用接口, 将关系数据语义添加到区块链平台, 以支持丰富的数据查询功能. 以上的功能性查询框架, 通过引入数据库实现链下多功能查询, 难以支持在链上的多功能查询.

## 2 预备知识

本节对系统设计中涉及的关键技术进行初步的介绍, 包括图式区块链、学习索引和布隆过滤器.

### 2.1 图式区块链

图式区块链采用 DAG 作为底层存储模型的数据结构, 支持并发处理交易和打包区块, 与传统链式区块链相比具有更高的吞吐量性能. 图式区块链具有实时性强和确认速度快的特点, 因此更适用于物联网、即时通讯、小额结算等领域[26].

图式区块链沿用了链式区块链的 P2P 网络结构来组织全网节点, 这使得网络中每个节点都是平等且相互连接. 由于区块链去中心化的特性, 可能存在恶意节点篡改查询结果的情况, 这要求图式区块链查询必须具备可验证性. 在另一方面, 图式区块链结构更为复杂, 典型的 DAG 图式区块链模型将图中的每个节点定义为一笔交易, 如 IOTA [2]. 与链式区块链相比, 其优点在于交易处理并行度更高, 从而具有更低的处理延迟. 然而, 这种方案将新发布的交易任意地添加到现有的节点上, 导致 DAG 拓扑向不可预测

的方向增长. 为了保证交易数据有序, 一些工作将区块作为 DAG 中的每个节点, 扩展了链式区块链的共识算法(如 Conflux[4] 和 OHIE[27]), 允许并发处理多个区块以提高系统吞吐量. 所有并发区块被组织成一个固定的 DAG 结构, 从而实现有向拓扑增长, 并通过区块排序算法来确定数据的完整顺序. Conflux 采用一条主链来指导 DAG 拓扑的生长方向, OHIE 将多个单链实例组织成一个平行链结构, 这两类基于 DAG 的区块链因其高安全性、良好的可扩展性和支持智能合约的能力而成为当前图式区块链的主流. 但这种支持高并发区块的图式结构相较于链式结构, 其数据查询变得复杂且低效. 本文针对具有主链和引用关系的树图结构 Conflux 建立高效索引机制, 提升图式区块链数据查询的效率, 同时保证查询结果的可验证性.

### 2.2 学习索引

学习索引利用机器学习技术学习数据分布[9], 并通过函数运算, 即通过参数列表采用函数的方式代替传统指针的方式进行数据定位, 从而提高数据查找速度并减小索引占用的存储开销. Kraska 等人[28]提出了学习索引(learned index)的概念, 他们根据数据的分布规律采用机器学习的方式获取数据之间的映射关系, 并提出了递归模型索引(recursive-model indexes, RMI)解决学习索引的精度问题. RMI 将数据集分为更小的子集, 并在每个子集上训练模型, 从而提高查找精度. 上层学习模型的输出用于选择下层模型, 最后一层模型将直接预测键值在数据集中的位置, 由于模型精度的问题, 在 RMI 输出预测的位置后, 还需要在数组中进行搜索. 为了减少最后的搜索时间, 最底层模型在训练时会保存一个误差阈值, 定位后只需在误差范围内进行搜索.

索引结构(如 B+Tree[9])在数据查询技术中扮演着重要角色, 但是这种基于树结构的索引机制没有考虑被索引数据分布的信息, 每条数据被视为独立的个体. 学习索引通过机器学习模型将数据分布所构建的映射关系应用于索引中, 利用模型参数值代替传统的数据指针结构, 从而降低索引存储开销, 同时利用函数运算的方式快速定位数据并提升查询性能. 传统的 B+树索引使用简单的叶子指针指向原始数据, 数据之间缺乏联系, 仅使用单一的有序分布来映射数据, 其时间复杂度和空间复杂度都比学习索引开销大. 学习索引采用学习模型计算的方式替代传统树形索引的遍历方式, 给定输入 *key* 值返回对应数据的位置, 可以大幅度地降低索引的空间代价并提高查询性能.

### 2.3 布隆过滤器

布隆过滤器[29]通过对数据集合中的元素进行哈希映射来判断一个数据集合中是否存在某个特定数据元素, 它是目前针对“存在查询”类型最经典的数据结构. 在区块数据结构中, 布隆过滤器可以作为区块头的一部分对区块内的交易进行过滤, 从而加快对块内数据的遍历. 在传统数据库, 也可以使用它来判断某个键值是否存在于特定表格中, 以此减少对磁盘的访问.

布隆过滤器由位数组哈希函数构成, 具有快速过滤数据、低占用空间等优点[30]. 布隆过滤器在插入新元素时, 使用 Hash 函数运算添加到位数组对应的值. 在查询数据时, 再次通过 Hash 函数运算查询元素, 如果计算得到的值不在位数组中, 则可以判断该数据元素一定不存在于原始数据集合当中. 值得注意的是, 由于哈希函数在一定概率下存在哈希碰撞冲突, 当计算出某查询元素哈希存在于该位数组中时, 并不能判断该元素一定存在于原始数据集合中. 这种现象被称为布隆过滤器的假阳性, 即它只能过滤不存在于集合中的数据元素, 对于存在于集合中的元素存在一定的误判. 布隆过滤器的假阳性误判率与其位数组的长度具有负相关关系, 位数组越长误判率越低. 但是在大数据集合上, 需要很高的空间成本才能获得较低的假阳性, 这也在一定程度上限制了布隆过滤器在大规模数据集上的使用效率.

## 3 系统概述

本节介绍一种基于学习索引的图式区块链高效可验证查询机制 Lever, 并讨论其系统结构和工作流程.

### 3.1 系统架构

图式区块链系统中所有的区块数据保存在系统的全节点上, 账本数据由网络中各全节点维护. 客户端(轻节点)由于计算和存储能力有限, 仅保存部分的区块头数据进行验证, 对存储空间的需求大大降低. 另一方面, 由于客户端本地未保存区块链的账本数据, 它需要向全节点发起查询请求, 依赖于全节点返回正确的查询结果. 在存在恶意节点的不可信环境中, 客户端需要对查询结果的正确性和完整性进行验证. 为保证查询结果的可验证性, 全节点在返回结果的同时需要包含可验证对象数据.

系统架构如图 1 所示, 包括客户端、学习索引、

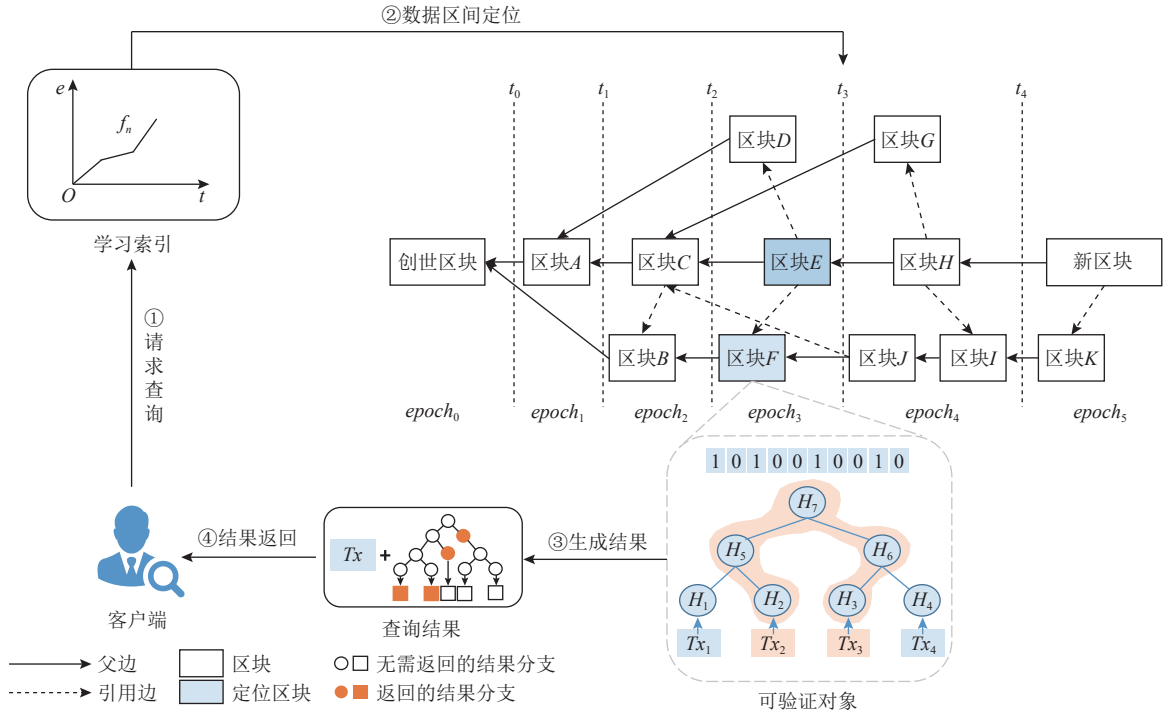



Fig. 1 The achitecture of Lever

图 1 Lever 系统架构

区块数据、可验证对象和查询结果 5 个部分. 具体来说, 客户端对需要查询的交易数据发送查询请求, 并接收全节点返回的查询结果数据. 同时, 客户端本地同步保存最新区块头数据, 通过本地计算验证查询结果. 学习索引由矿工在打包区块产生新的纪元时构建, 索引机制采用动态分段函数学习纪元高度与时间戳之间的映射关系. 区块数据由提供查询服务的全节点同步保存, 系统中的主区块(例如区块 $A$, $C$, $E$ 等)由共识节点产生, 根据主区块和并发区块之间的引用关系, 系统中区块被划分为依次递增的纪元, 每个纪元内包含不同数量的区块数据. 可验证对象是由全节点根据查询结果生成的验证数据, 主要包含 2 种情形: 数据存在和数据不存在. 数据存在则根据查询结果生成存在性证明, 即交易数据与其默克尔树路径; 数据不存在则需返回不存在证明, 即布隆过滤器对应的位数组, 或布隆过滤器出现假阳性时对应的排序默克尔树分支. 查询结果由查询请求对应的交易数据和可验证对象构成, 全节点根据请求数据完成查询后, 生成查询结果返回给客户端.

本文针对图式区块链的时序数据特征, 设计了基于学习索引的高效查询机制 Lever, 加快图式区块链数据查询. 首先当客户端向全节点发送查询请求时, 全节点通过构建的学习索引快速定位查询数据所在的纪元区间. 其次, 为了提升图式区块链纪元内区块的遍历速度, 通过设计聚合布隆过滤器, 快速过滤不存在查询数据的区块, 提升查询效率. 最后, 针对布隆过滤器可能存在的假阳性, 采用排序默克尔树结构, 客户端在接收查询结果数据以外, 会额外收到全节点返回的默克尔树分支作为可验证对象数据, 该机制能够保证查询结果的高效性并降低可验证对象的大小. 客户端利用本地同步保存的区块头数据对全节点返回的结果数据进行验证, 从而保证查询结果的可验证性.

### 3.2 系统工作流程

Lever 利用学习索引取代传统树结构索引, 以函数的方式存储函数参数信息, 避免对每条数据采用指针的方式降低索引占用的存储空间, 并通过函数运算实现对纪元间数据的快速定位. 同时, 系统根据定位后的区间对纪元内的区块进行遍历. 每个纪元内采用聚合布隆过滤器对区块内的交易进行哈希映射, 并据此快速过滤不存在的查询数据. 区块内通过排序默克尔树对布隆过滤器可能存在的假阳性做进一步判断, 存在则返回交易数据对应的默克尔树路径, 不存在则返回相邻交易的排序默克尔树分支作为不存在证明.

系统工作流程如图 1 所示, 当客户端向全节点查询某地址在一段时间内的所有交易时, ①客户端向全节点发送包含交易地址和时间戳的查询请求; ②

全节点首先获取查询请求中的时间戳，并通过所构建的学习索引快速定位查询数据所在的纪元区间；③全节点通过聚合布隆过滤器对纪元内的多个区块进行过滤，如果不存在则直接遍历时间范围内的其他纪元，如果存在则需要分别对每个区块头的布隆过滤器进行检验，对于可能存在假阳性的区块通过默克尔树进一步遍历，并生成对应的可验证对象和查询结果数据；④全节点向客户端返回查询结果，包括查询请求对应的结果数据和查询结果对应的可验证对象数据. 客户端在收到全节点返回的结果后，通过区块头和本地计算对查询结果进行验证.

## 4 面向图式区块结构的可验证学习索引设计

在本节中，我们将介绍 Lever 的详细设计，包括基于纪元间的时序高度学习索引构建过程、基于纪元内的多区块聚合布隆过滤器、基于区块内交易的可验证排序默克尔树和 Lever 高效数据查询过程. 本部分将从这 4 个方面具体阐述各模块内容.

### 4.1 基于纪元间的时序高度学习索引

本文采用机器学习的方法实现图式区块链数据之间的映射关系，如图 2 所示. 纪元间的学习索引通过学习数据集的时序分布特征来建立，相比于基于 B+树的索引机制，能够实现更高的查询效率和更低的存储开销.

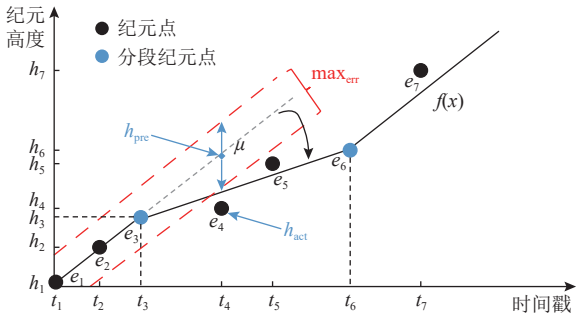



Fig. 2 Learned index between epochs

图 2 纪元间的学习索引

不同于传统的树结构索引方法需要将所有的数据指针都存储在索引结构中，本文所构建的学习索引只需将数据的一些统计特征和函数参数存储在列表中. 学习索引通过训练后的模型参数列表实现数据定位，可以实现内存的最小化占用和更高效的数据查找速度. 节点进行查询时，利用存储的参数值计算查询条件来预测目标数据的位置，从而避免对整个索引结构进行遍历的开销，降低开销. 通过使用学习索引，节点可以实现更快的查询速度和更低的存储开销，从而提高在大规模图式区块链数据场景中的查询性能.

在图式区块链中纪元的定位仅依靠其高度值，无法直接根据时间属性获取某段时间内的纪元数据，但通过观察发现时间戳与纪元高度存在映射关系. 因此本文在构建索引时利用纪元的时序特征，并采用机器学习的思想构建模型，建模出时间戳到纪元高度的映射关系. 在模型中，我们采用分段线性回归的方法来拟合映射关系. 为了控制预测值与实际值之间的误差范围，Lever 在动态回归过程中引入了误差阈值 $\mu$，以确保异常值偏离正常分布时也可以被索引，从而保证模型的准确度.

具体而言，模型在训练过程中通过上、下误差边界控制回归函数的分段数，上边界为预测值$h_{pre}$加上误差阈值$\mu$，下边界为预测值$h_{pre}$减去误差阈值$\mu$. 如图 2 所示，点$e_1$和$e_2$是构成起始函数的 2 个基本点，点$e_3$在这 2 个边界内，则当前回归模型不需要更新. 由于点$e_4$在误差边界之外，即实际值$h_{act}(e_4)$不在预测范围$(h_{pre}\pm\mu)$之内，此时，点$e_3$作为新分段回归函数的起点，点$e_4$将成为新模型中的点进行训练. 时间区间$[t_1, t_3]$为已训练好的模型，模型中数据的预测值与实际值偏差控制在误差阈值$\mu$以内. 对于偏离误差范围的点$e_4$，在时间区间$[t_3, t_4]$构建新的模型. 点$e_5$和$e_6$在新模型的误差范围以内，模型保持不变，直到点$e_7$成为新的偏离点生成新的回归模型.

同时，由于区块链不可篡改的特性，历史数据的分布不可改变，完成训练的学习索引模型不会因为新产生的区块数据而改变. 因此，学习索引模型一旦被构建并保存便不再需要进行重新训练，降低了模型训练的计算成本.

### 4.2 基于纪元内的多区块聚合布隆过滤器

通过学习索引快速定位到纪元后，为了避免依次遍历区块内的每条交易数据，本文采用一种高效的查询过滤机制，即在区块头引入布隆过滤器. 我们采用布隆过滤器进行区块内交易数据集合的元素检查，首先将区块中所包含的交易数据构成原始数据集，然后通过哈希函数对数据集中的每个元素进行哈希运算，将得到的值映射到隆过滤器的位数组中，最后比对位数组中的比特值进行元素检查.

纪元内的每个区块都在区块头保存相应位数组，遍历区块内数据前，先通过位数组进行过滤，成功过滤的区块无需遍历块内交易数据. 利用布隆过滤器提高查询过程中块内数据的遍历速度，这主要得益

于其时间和空间的高效优势,每一个比特位即可实现对一条交易数据的映射,通过一次哈希运算即可实现对块内所有交易数据的过滤.为了进一步加快过滤纪元内不含目标交易数据的区块,本文将多个区块中的布隆过滤器进行聚合,如图 3 所示.

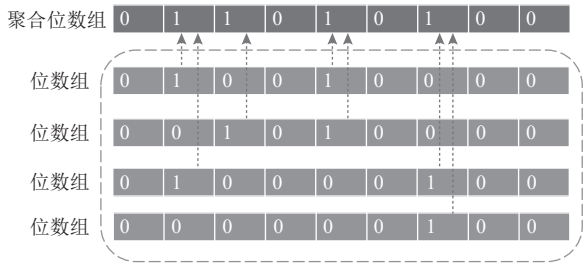



Fig. 3 Intra-epoch aggregated Bloom filter

图 3 纪元内聚合布隆过滤器

聚合布隆过滤器将多个区块的布隆过滤器位数组通过或运算聚合在一起,以实现更好的去重效果和查询过滤性能.与单个区块布隆过滤器相比,聚合布隆过滤器可以更好地处理交易数据集合较大、数据分布不均等问题.聚合布隆过滤器的具体实现和工作步骤为:

1)将纪元按区块分成多个部分,并为每个区块创建一个独立的布隆过滤器;

2)将多个布隆过滤器通过或运算进行合并聚合;

3)当需要查询某个元素是否存在于数据集合中时,将该元素传入聚合布隆过滤器中查询;

4)如果聚合布隆过滤器返回该元素不存在于纪元数据集合中,则认为该元素一定不存在于该纪元内的所有区块数据集合中;

5)如果聚合布隆过滤器返回该元素存在于数据集合中,则认为该元素可能存在于数据集合中,因此需要对区块中的每一个独立布隆过滤器进一步判断.

聚合布隆过滤器的优势在于,数据不存在的情况下可以快速过滤纪元内的多个区块,提高去重和查询的准确率和效率.在查询过程中,客户端向全节点发送查询请求信息,全节点根据请求信息通过哈希函数进行运算,将得到的值与布隆过滤器的位数组进行对比.如果为 0 则表明查询数据一定不存在于该数据集合中,此时位数组也可以作为不存在证明构建可验证对象.但当运算结果比对为 1 时,并不能确定查询数据一定存在于该数据集中,这是因为哈希函数存在碰撞冲突,布隆过滤器此时的结果可能误报,即假阳性.

布隆过滤器只能确定某个元素一定不存在于特定集合中,但由于哈希冲突可能导致判断存在的元素并不一定存在于该集合中.其中,包含的元素个数是影响布隆过滤器中误报率的重要因素之一,通过增加位数组的长度能够有效降低误报率,但同时会增大布隆过滤占用的存储空间.为了克服布隆过滤器存在的误报率,接下来本文通过设计排序默克尔树解决假阳性问题,保证查询结果的正确性和完整性.

### 4.3 基于区块内的可验证排序默克尔树

客户端可以通过 2 种基本的方式实现对查询结果的验证:1) 全节点在完成查询操作后,直接将包含查询结果的所有区块直接返回给客户端.该方案只能保证查询结果的正确性,客户端通过对比区块内的全部交易数据获得正确的查询结果.但是该方案需要额外返回大量的无关数据,并且无法保证查询结果的完整性,即全节点可能遗漏包含查询数据的区块,而客户端只能验证已收到的区块数据,无法验证是否有数据丢失.2) 客户端在存储能力范围内尽量保存更多的账本数据,此时客户端将由轻节点逐渐变成全节点,当同步完整的账本数据后,可以实现在本地直接查询.这种方式直接采用空间换时间,客户端的存储开销与轻节点的初衷相悖.

本文设计的图式区块链可验证机制与上述 2 种基本方案不同,其目标主要有 2 点:1) 全节点在查询过程中返回尽量少的数据.由于返回的数据主要包含结果数据和验证数据 2 方面,结果数据的大小与客户端的查询请求相关,验证数据的大小与验证机制相关.此目标主要围绕降低验证数据大小,使得全节点在生成可验证对象时产生较小的证明数据量.2) 客户端本身无需存储大量的账本数据.验证机制应避免对客户端造成较大的额外存储开销,客户端仅保存区块头数据,通过验证计算与所同步的区块头哈希值进行对比,保证查询结果的可验证性.此目标主要围绕利用客户端的计算性能降低额外存储的数据,从而保证客户端在存储方面的轻量化.

图式区块链系统采用了与比特币系统设计中相同的最基本默克尔树结构.如图 4 所示,在一个区块内矿工打包了 4 笔区块链交易,它们的交易地址元素($Tx\,eacdf$, $Tx\,acdbg$, $Tx\,dfeag$, $Tx\,bdcfe$)用于构建默克尔树,基本的默克尔树直接根据当前的交易顺序进行构建,叶子节点元素之间的关系随机.当查询 $Tx\,caedf$ 布隆过滤器出现假阳性时,全节点返回的不存在证明需要包含完整默克尔树的数据.这种方式将导致验证对象占用的存储空间较大,造成结果数据的网络传输和客户端的验证开销增大.

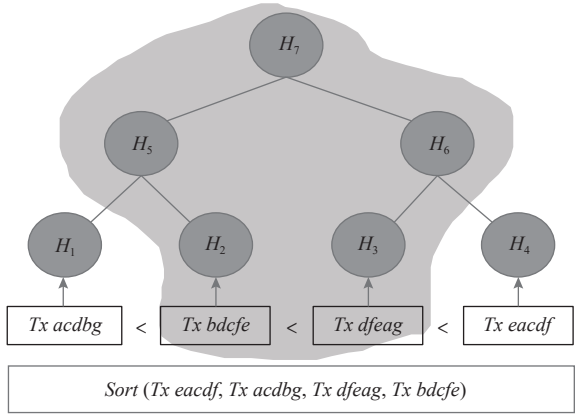



Fig. 4 Intra-block sorted Merkle tree

图 4 区块内排序默克尔树

本文采用排序默克尔树降低验证对象大小，首先将交易数据集合中的元素通过排序函数$Sort(\cdot)$排序为$(Txacdbg < Txbdcfe < Txdfeag < Txeacdf)$，然后基于排序后的元素构建默克尔树，得到如图 4 所示的排序默克尔树结构. 由于叶子节点之间的元素有序，当出现不存在的元素$Txcaedf$时，可以通过对比相邻元素$Txbdcfe$和$Txdfeag$的大小. 因此，针对布隆过滤器出现假阳性，不存在证明可以利用排序默克尔树的相邻叶子节点分支，可验证对象仅需返回相邻叶子节点和对应路径，客户端即可判断数据是否存在于分支之间.

### 4.4 Lever 的高效数据查询过程

为了便于说明，本文首先考虑一个特定的时间戳，并应用于具有学习索引的点查询. 然后，将其扩展到时间范围条件，以说明如何高效地处理时间范围查询请求. Lever 维护学习索引函数的每个分段函数参数，包括起点、斜率和截距. 由于时间戳具有递增属性，因此本文采用变体二分查找算法快速定位查询值所在的分段. 查找元素所在指定段的时间复杂度为$O(\mathrm{lb}\, n)$，其中$n$为分段参数列表的长度. 一旦找到了具有相应时间戳的分段，我们就可以通过计算该段中的函数来快速定位给定时间戳对应的纪元高度. 在创建分段线性函数时，键的实际位置与回归函数运算的位置保持在一个范围内（即误差阈值$\mu$）. 根据分段的斜率和截距参数，给定时间戳$t$的预测位置计算方式为：

$$h_{\mathrm{pre}}(t) = t \times N_{slope} + N_{intercept}, \tag{1}$$

其中，$N_{slope}$和$N_{intercept}$分别为分段函数对应的斜率和截距.

通过动态分段线性函数构造索引，可以将元素的真实位置限制在误差范围内. 因此，在计算得到预测位置后，采用顺序遍历的方法对误差范围内的纪元进行局部检索. 给定时间戳$t$和误差阈值$\mu$的实际位置范围为：

$$h_{\mathrm{act}}(t) \in [h_{\mathrm{pre}}(t) - \mu, h_{\mathrm{pre}}(t) + \mu]. \tag{2}$$

时间范围查询是点查询的一种特殊扩展情况，具有时间范围附加条件，它需要检查查询的数据结果是否在给定的时间范围内，如算法 1 所示. 查询请求为$Q = (addr, \langle t_1, t_2\rangle)$，表示对地址$addr$在时间段$t_1$~$t_2$相关交易的查询. 首先，全节点找到$t_1$与$t_2$所对应的学习索引分片$(seg_1, seg_2)$，并读取相应分片的斜率$N_{slope}$与截距$N_{intercept}$. 根据斜率与截距，全节点通过函数运算得到包含$t_1$和$t_2$的纪元范围$(pre_epoch_1 - \mu, pre_epoch_1 + \mu)$. 对于范围内的所有纪元，全节点执行 5 个操作：1）使用聚合布隆过滤器检测纪元内是否存在地址$addr$的相关交易；2）若存在相关交易，则搜索纪元内所有时间戳在$\langle t_1, t_2\rangle$之间的区块，检查区块内是否存在地址$addr$的相关交易；3）若区块内包含$addr$相关交易$Tx$，则将$Tx$加入结果集，否则就加入$Tx$的前序与后继交易；4）将相应排序默克尔树分支加入验证对象集合；5）返回结果集$TxSet$与验证对象集$VO$. 因此，与点查询不同，时间范围查询的条件对查询的总时间和结果大小有较大的影响. 这 2 种查询类型的主要区别在于，时间范围查询需要判断给定时间范围的 2 个端点. 由于分段函数满足给定的误差阈值，因此在分段中查找键的开销可以得到控制. 更准确地说，在误差边界范围内查询数据的时间复杂度为$O(1+2\mu)$. 在连续时间范围内，分段线性函数指向的键要么连续地分布在同一段函数中，要么存在于相邻的分段函数中. 当节点进行时间范围查询时，Lever 首先在误差范围内直接从时间范围的起始点位置开始扫描，然后遍历相邻分段函数中，直到时间范围的结束点.

**算法 1.** 时间范围可验证查询算法.

输入：$Q = (addr, \langle t_1, t_2\rangle)$;

输出：$R = (TxSet, VO)$.

① $(seg_1, seg_2) \leftarrow VBS(t_1, t_2)$;

② $pre\_epoch_i \leftarrow seg_i.slope \times seg_i.intercept$ $(i=1, 2)$;

③ for $epoch$ in $(pre\_epoch_1 - \mu$ to $pre\_epoch_2 + \mu)$ do

④ if $epoch.agg\_bf(addr)$ == true then

⑤ for $blk$ in $epoch.block\_id$ do

⑥ if $blk.timstamp$ in $[t_1, t_2]$ && $blk.bf(addr)$ == true then

⑦ if $Tx$ exists then

⑧ $TxSet.append(Tx)$;

```
⑨                else
⑩                   TxSet.append (Tx.prior, Tx.next);
⑪                end if
⑫                VO.append (SMT.subtree);
⑬             end if
⑭          end for
⑮       end if
⑯ end for
⑰ return R = (TxSet, VO).
```

## 5 可验证性分析

为实现在不可信的区块链环境中进行可验证查询, 客户端需要对返回的查询结果进行验证. 全节点除了返回查询请求对应的查询结果数据以外, 还需要额外返回与结果相关的证明数据, 即可验证对象. 客户端根据查询结果和可验证对象数据进行计算, 并与本地保存的区块头数据进行对比, 从而实现对查询结果的验证. 对查询结果的验证目标主要包括 2 个方面: 正确性和完整性.

### 5.1 正确性分析

查询结果的正确性, 即验证全节点返回的结果数据是在区块链账本中未被篡改且真实存在的. 当前的 Conflux 系统实现中矿工在进行区块打包时, 将区块中的交易采用默克尔树的形式进行组织, 并将默克尔树根保存在区块头. 客户端根据全节点返回的交易数据进行该默克尔哈希运算, 直到得出默克尔树根. 如果计算得到的值与本地同步的区块头数据值相同, 则可以证明返回的结果数据是真实存在且未被篡改, 否则查询结果不正确. 因此, 全节点可以通过返回交易默克尔树直接作为查询结果的正确性证明, 客户端基于该证明, 通过本地计算即可对查询结果的真实性进行验证.

### 5.2 完整性分析

查询结果的完整性, 即验证全节点返回的结果数据没有遗漏. Lever 采用聚合布隆过滤器首先判断纪元内是否包含查询数据, 如果聚合布隆过滤器返回值为 0, 则该数据一定不存在于该纪元内, 此时全节点返回聚合布隆过滤的位数组作为不存在证明. 当聚合布隆过滤器返回值为 1 时, 全节点对纪元内各区块的区块头布隆过滤器进行遍历, 查找可能存在交易数据的区块. 如果遍历区块该交易数据真实存在, 则返回对应交易数据; 如果遍历区块后该交易数据并不存在, 说明布隆过滤器产生了误报, 全节点需要将该数据对应排序默克尔树的 2 个相邻边界叶子节点和默克尔树路径分支作为不存在证明返回给客户端. 客户端在验证查询结果时, 首先利用布隆过滤器的位数组验证纪元内不存在所查询的交易数据; 其次, 如果出现假阳性, 客户端利用排序默克尔树分支和相邻叶子节点进行对比, 即可验证查询数据的不存在性, 从而判断查询结果未被遗漏.

## 6 实验评估

### 6.1 实验环境

本文实验使用的数据集是基于 Conflux 系统中的数据结构所生成, 它包含 2 048 epoch, 8 192 个区块, 交易被重新组织成〈*epoch*, *block*, *address*, *amount*, *timestamp*〉作为 Lever 的输入数据. 实验采用 Intel Xeon Platinum 8369B 服务器 16 核 32 GB 内存, CPU 3.5 GHz, 运行 CentOS 7.6 操作系统. Lever 系统基于 Rust 语言编写, 搭建了一个用于创建图式区块链系统和索引结构的开源框架. 实验从索引构建开销、查询性能和误差阈值对系统的影响这 3 个方面进行评估. 通过 2 个基准测试来评估 Lever 的索引构建开销: 1)索引构建所需的 CPU 时间, 包括纪元间的学习索引、纪元内的布隆过滤器和排序默克尔树; 2)不同纪元高度下索引结构占用的存储开销. 在系统参数影响评估实验中, 本文在不同纪元高度下对索引构建开销和查询性能进行了评估, 并选择了查询延迟、误报率和验证开销 3 个评估指标.

### 6.2 索引构建开销

实验对矿工节点在打包数据时构建区块的总时间和构建学习索引所需的时间进行了比较, 其中学习索引的误差阈值 $\mu$ 设置为 5, 参数设置原因可以通过 6.3 节误差阈值对系统查询的影响分析可知. 从图 5

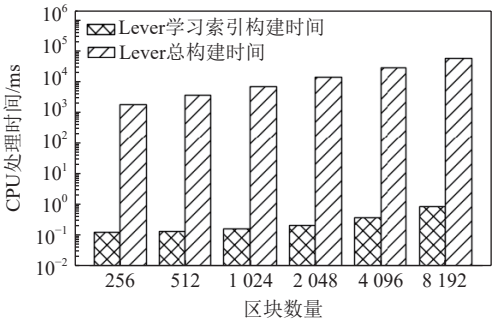



Fig. 5 The overhead of index building

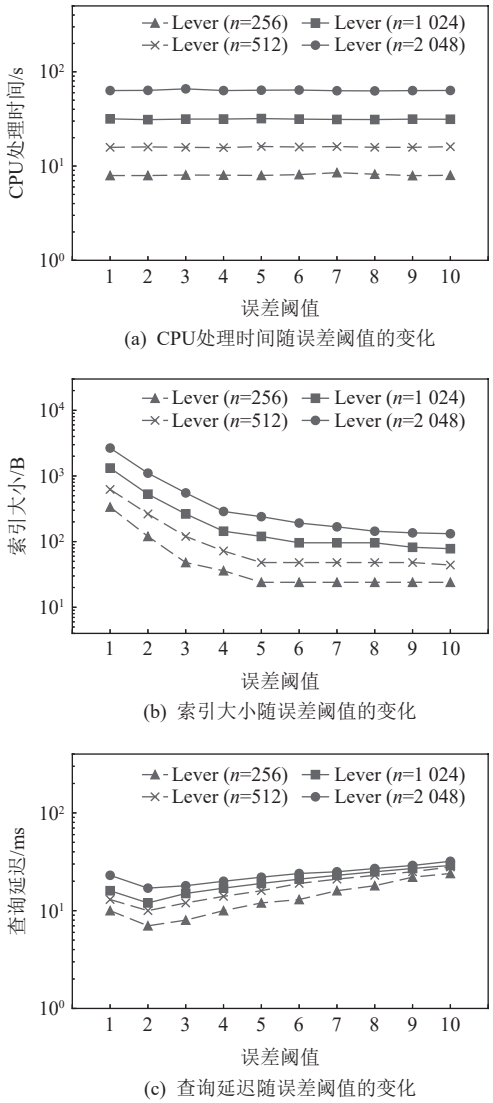
图 5 索引构建开销

可以看出，随着生成区块数量的增加，两者的构建时间都有所增长. 但总的区块构建时间是学习索引构建时间的 $10^4$ 倍，这是因为在构建学习索引时，本文采用动态更新的方式，仅在新增数据与学习索引模型预估的误差在误差阈值外时更新模型，其占用的CPU 时间开销并不大. 值得一提的是，对于一段特定区间的学习索引模型，其存储的参数为起始时间戳、斜率和对应的截距，这种分段参数方式相比于数据指针方式降低了索引的存储开销.

### 6.3 误差阈值对系统查询的影响

实验评估了不同误差阈值对 CPU 处理时间、索引大小和查询延迟的影响. 本文针对纪元高度进行了可扩展性实验，采用 256~2 048 个不同数量的纪元高度数据集. 图 6(a)和图 6(b)显示了 CPU 处理时间和索引大小的性能，误差阈值从 1 递增到 10. 实验结果表明，随着误差阈值的增加，CPU 处理时间几乎保持不变，而索引大小随着误差阈值的增加逐渐减小，其原因在于较大的误差阈值会使得分段函数的数量降低，存储的参数减少，但这种情况下预测值与实际值之间的误差范围也随之增大，模型的精确度会降低. 随后，实验评估了误差阈值对查询延迟的影响. 图 6(c)显示了不同区块高度的查询延迟，当误差阈值小于 4 时，查询延迟略有振荡，然后随着误差阈值的增加而增加，其原因在于较大的误差阈值在数据定位时需要对更多的纪元进行遍历. 在 CPU 处理时间基本保持不变的条件下，综合考虑查询延迟和索引大小 2 个因素，当误差阈值为 5 时，查询延迟增幅并不高且索引占用的空间降幅较大，因此，实验中选取误差阈值为 5 作为系统参数.

图 7 显示了 Lever 在区块数量为 256~8 192 且误差阈值为 5 时的查询性能. 实验将图式区块链 Conflux 上的基本查询方法作为实验基准（DAG Baseline），并采用传统的B+Tree 结构建立纪元间索引作为对比，在此基础上对本文提出的 Lever 框架进行消融实验，主要包括不含学习索引的 Lever（Lever w/o Index）以及不含有聚合布隆过滤器的 Lever（Lever w/o AGG-BF）. 实验结果表明，随着区块数量的增多，Lever 和 Lever w/o AGG-BF 查询延迟基本保持不变，维持在 10~20 ms 之间，其查询效率是 Lever w/o Index 的 10 倍左右. B+Tree 和 B+Tree w AGG-BF 的查询延迟相较于 DAG Baseline 平均降低 60%，然而它们的查询延迟是 Lever 的 5 倍以上，其原因在于 B+Tree 结构是基于二分查找策略，相比于线性依次遍历有较大提升，但对比于函数运算其查询延迟依然较大. DAG Baseline 和 Lever w/o Index 的查询延迟与区块数量基本呈线性关系，当区块数量达到 8 192 时，两者的查询延迟都高于 $10^3$ ms，其原因主要是随



(a) CPU处理时间随误差阈值的变化

(b) 索引大小随误差阈值的变化

(c) 查询延迟随误差阈值的变化

Fig. 6 Impact of different error bounds on system performance

图 6 不同误差阈值对系统性能的影响

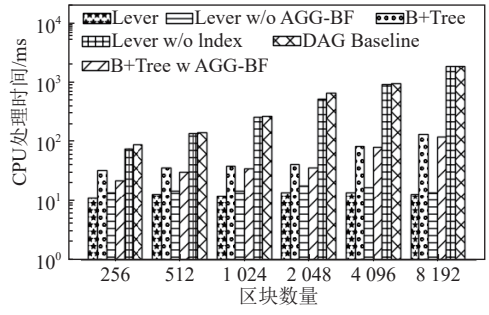



Fig. 7 Query latency with different number of blocks

图 7 不同区块数量下的查询延迟

着区块数量的增加,这 2 类查询类型都需要依次访问每个纪元并遍历整个区块数据集.

### 6.4 系统参数影响

布隆过滤器的长度会在 2 个方面对系统产生影响: 1)布隆过滤器的假阳性概率; 2)布隆过滤器占用的存储大小. 本文针对不同布隆过滤器位数组长度对这 2 个因子进行评估,以选择较为合适的布隆过滤器位数组长度作为系统参数. 如图 8 所示,实验选取大小约为 6 000 笔交易的区块,布隆过滤器比特位长度在 [1 000, 500 000] 范围内对假阳性概率和布隆过滤器占用的存储空间大小进行对比.

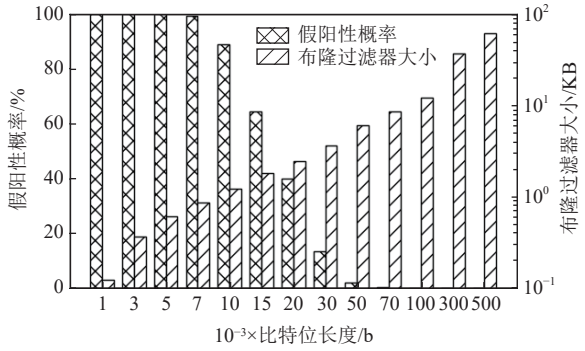



Fig. 8 False positive of Bloom filter at different bit lengths

图 8 不同位长度下布隆过滤器的假阳性

从实验中可以总结出 2 个方面的关键点:1)对于假阳性概率,如图 8 所示,布隆过滤器长度超过 10 000 b 时,假阳性概率迅速降低,直到布隆过滤器的长度大于 50 000 b 时,假阳性概率趋于 0.01. 假阳性概率高会使得在查询过程中,将一些不包含记录的区块作为目标块返回给轻节点,这会增加全节点返回的可验证对象大小和对应的通讯开销. 2)对于布隆过滤器占用存储空间大小,从图 8 可以直观地发现,布隆过滤器的比特位长度与其占用存储空间大小呈正相关. 实验结果表明,随着布隆过滤器比特位的增加,布隆过滤器假阳性概率降低,被误判的区块数量减少,查询效率提高. 理论上,实验可以取任何大于 50 000 b 的参数值作为布隆过滤器的比特位. 但在实际情况下,需要考虑空间开销和对应的假阳性概率,因此在实验中,本文将布隆过滤器的比特位设置为 50 000 b 作为系统参数.

实验针对布隆过滤器在假阳性条件下,分别对使用排序默克尔树和不使用排序默克尔树返回的可验证对象数据集大小进行了评估,如图 9 所示.

在查询过程中,当关键字存在于区块中时,两者

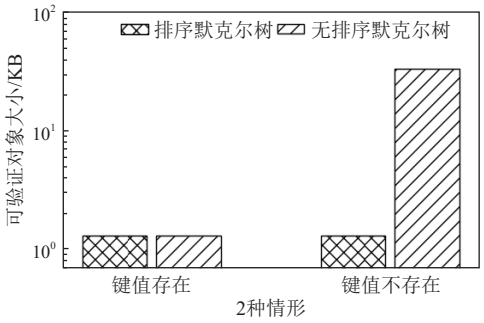



Fig. 9 The size of the verifiable object in two cases

图 9 2 种情形下的可验证对象大小

的存在性证明一致,因此其可验证对象大小相同. 当关键字不存在于区块中时,无排序默克尔树返回的可验证对象数据集大小约是排序默克尔树返回的可验证对象集大小的 25 倍. 其原因在于,当布隆过滤器出现假阳性时,无排序默克尔树需要返回区块内所有的数据集作为可验证对象,而排序默克尔树能够返回区块内的边界交易和默克尔树分支作为可验证对象.

## 7 结 论

本文提出了一种基于学习索引的高效可验证的图式区块链查询机制 Lever,该机制通过引入学习索引技术对图式区块链中时序数据分布特征进行学习来实现对索引过程的优化. 针对图式区块链数据查询的效率和可验证性问题,将学习索引应用于图式区块链的纪元高度与时间戳的映射关系中,通过函数运算的方式定位查询数据,提高图式区块链的查询速度和效率. 同时,为了加快纪元内多个区块数据的过滤速度,在每个区块头部添加布隆过滤器,并为每个纪元生成一个聚合布隆过滤器,从而提高纪元内的数据遍历速度. 此外,为保证查询结果的正确性和完整性,该机制结合布隆过滤器和排序默克尔树生成可验证对象,通过部分默克尔树分支实现对布隆过滤器假阳性的不存在证明,有效减小验证对象的规模,从而保证图式区块链查询可验证性并提高验证效率. 实验结果表明,Lever 能有效提高基于 DAG 的图式区块链查询效率和可验证性,与 Conflux 的基本查询机制相比,Lever 的查询延迟可以降低至 1/10,不存在证明的可验证对象大小开销可以大幅度降低.

未来工作主要将从 2 个方面展开: 1)关于查询类型的扩展,如何将学习索引扩展到其他属性值,支持更多类型的查询. 2)如何在保证查询结果可验证性

的前提下，将验证机制进一步优化.

**作者贡献声明**：常健设计了研究方案并撰写论文；林立成提出了算法思路和完成实验；李彬弘负责方案讨论、论文修改；肖江提出指导意见并修改论文；金海提出指导意见及讨论定稿.

## 参 考 文 献

[1] Nakamoto S. Bitcoin: A peer-to-peer electronic cash system [EB/OL]. [2019-09-27].https://bitcoin.org/bitcoin.pdf

[2] Popov S. The tangle [EB/OL]. [2020-02-17]. http://www.descryptions.com/Iota.pdf

[3] Churyumov, A. Byteball: A decentralized system for storage and transfer of value [EB/OL]. [2021-07-02].https://byteball.org/Byteball.pdf

[4] Li Chenxing, Li Peilun, Zhou Dong, et al. A decentralized blockchain with high throughput and fast confirmation [C] //Proc of the USENIX Annual Technical Conf (ATC). Berkeley, CA: USENIX Association, 2020: 515−528

[5] Xu Rui, Liu Zihao, Hu Huiqi, et al. An efficient secondary index for spatial data based on LevelDB [C] //Proc of the 25th Int Conf on Database Systems for Advanced Applications (DASFAA). Berlin: Springer, 2020: 750−754

[6] Yu Ge, Nie Tiezheng, Li Xiaohua, et al. The challenge and prospect of distributed data management techniques in blockchain systems[J]. Chinese Journal of Computers, 2021, 44(1): 28−53 (in Chinese)
(于戈, 聂铁铮, 李晓华, 等. 区块链系统中的分布式数据管理技术——挑战与展望[J]. 计算机学报, 2021, 44(1): 28−53)

[7] Chen Huashan, Pendleton M, Njilla L, et al. A survey on ethereum systems security: Vulnerabilities, attacks, and defenses[J]. ACM Computing Surveys, 2020, 53(3): 1−43

[8] Jin Hai, Xiao Jiang. Towards trustworthy blockchain systems in the era of “Internet of value”: Development, challenges, and future trends[J]. Science China Information Sciences, 2022, 65: 1−11

[9] Wang Qing, Lu Youyou, Shu Jiwu. Sherman: A write-optimized distributed B+ Tree index on disaggregated memory [C] //Proc of the 2022 Special Interest Group on Management of Data (SIGMOD). New York: ACM, 2022: 1033−1048

[10] Zhang Zhou, Jin Peiquan, Xie Xike. Learned indexes: Current situations and research prospects[J]. Journal of Software, 2021, 32(4): 1129−1150 (in Chinese)
(张洲, 金培权, 谢希科. 学习索引：现状与研究展望[J]. 软件学报, 2021, 32(4): 1129−1150)

[11] Chang Jian, Li Binhong, Xiao Jiang, et al. Anole: A lightweight and verifiable learned-based index for time range query on blockchain systems [C] //Proc of the 28th Int Conf on Database Systems for Advanced Applications (DASFAA). Berlin: Springer, 2023: 519−534

[12] Zhou Wei, Wang Chao, Xu Jian, et al. Privacy-Preserving and decentralized federated learning model based on the blockchain[J]. Journal of Computer Research and Development, 2022, 59(11): 2423−2436 (in Chinese)
(周炜, 王超, 徐剑, 等. 基于区块链的隐私保护去中心化联邦学习模型[J]. 计算机研究与发展, 2022, 59(11): 2423−2436)

[13] Jia Dayu, Xin Junchang, Wang Zhiqiong, et al. ElasticQM: A query model for storage capacity scalable blockchain system[J]. Journal of Software, 2019, 30(9): 2655−2670 (in Chinese)
(贾大宇, 信俊昌, 王之琼, 等. 存储容量可扩展区块链系统的高效查询模型[J]. 软件学报, 2019, 30(9): 2655−2670)

[14] Ruan Pingcheng, Chen Gang, Dinh T T A, et al. Fine-grained, secure and efficient data provenance on blockchain systems[J]. Proceedings of the VLDB Endowment, 2019, 12(9): 975−988

[15] Zhang Ce, Xu Cheng, Xu Jianliang, et al. GEM$^2$-Tree: A gas-efficient structure for authenticated range queries in blockchain [C] //Proc of the 35th Int Conf on Data Engineering (ICDE). Piscataway, NJ: IEEE, 2019: 842−853

[16] Cai Lei, Zhu Yanchao, Guo Qingxing, et al. Efficient materialized view maintenance and trusted query for blockchain[J]. Journal of Software, 2020, 31(3): 680−694 (in Chinese)
(蔡磊, 朱燕超, 郭庆兴, 等. 面向区块链的高效物化视图维护和可信查询[J]. 软件学报, 2020, 31(3): 680−694)

[17] Gupta H, Hans S, Aggarwal K, et al. Efficiently processing temporal queries on hyperledger fabric [C] //Proc of the 34th Int Conf on Data Engineering (ICDE). Piscataway, NJ: IEEE, 2018: 1489−1494

[18] Xu Cheng, Zhang Ce, Xu Jianliang. vChain: Enabling verifiable Boolean range queries over blockchain databases [C] //Proc of the 2019 Special Interest Group on Management of Data (SIGMOD). New York: ACM, 2019: 141−158

[19] Dai Xiaohai, Xiao Jiang, Yang Wenhui, et al. LVQ: A lightweight verifiable query approach for transaction history in bitcoin [C] //Proc of the 40th Int Conf on Distributed Computing Systems (ICDCS). Piscataway, NJ: IEEE, 2020: 1020−1030

[20] Zhang Ce, Xu Cheng, Wang Haixin, et al. Authenticated keyword search in scalable hybrid-storage blockchains [C] //Proc of the 37th Int Conf on Data Engineering (ICDE). Piscataway, NJ: IEEE, 2021: 996−1007

[21] Hao Kun, Xin Junchang, Wang Zhiqiong, et al. Outsourced data integrity verification based on blockchain in untrusted environment[J]. World Wide Web, 2020, 23(4): 2215−2238

[22] Shao Qifeng, Pang Shuaifeng, Zhang Zhao, et al. Authenticated range query using SGX for blockchain light clients [C] //Proc of the 25th Int Conf on Database Systems for Advanced Applications (DASFAA). Berlin: Springer, 2020: 306−321

[23] Li Yang, Zheng Kai, Yan Ying, et al. EtherQL: A query layer for blockchain system [C] //Proc of the 22nd Int Conf on Database Systems for Advanced Applications (DASFAA). Berlin: Springer, 2017: 556−567

[24] Peng Zhe, Wu Haotian, Xiao Bin, et al. VQL: Providing query efficiency and data authenticity in blockchain systems [C] //Proc of the 35th Int Conf on Data Engineering Workshops (ICDEW).

Piscataway, NJ: IEEE, 2019: 1−6

[25] Zhu Yanchao, Zhang Zhao, Jin Cheqing, et al. SEBDB: Semantics empowered blockchain database [C] //Proc of the 35th Int Conf on Data Engineering (ICDE). Piscataway, NJ: IEEE, 2019: 1820−1831

[26] Zhang Changgui, Zhang Yanfeng, Li Xiaohua, et al. Survey of new blockchain techniques: DAG based blockchain and sharding based blockchai[J]. Computer Science, 2020, 47(10): 2423−2436 (in Chinese)
(张长贵, 张岩峰, 李晓华, 等. 区块链新技术综述: 图型区块链和分区型区块链[J]. 计算机科学, 2020, 47(10): 2423−2436)

[27] Yu Haifeng, Nikoli´c I, Hou Ruomu, et al. OHIE: Blockchain scaling made simple [C] //Proc of the 2020 IEEE Symp on Security and Privacy (S&P). Piscataway, NJ: IEEE, 2020: 90−105

[28] Kraska T, Beutel A, Chi E H, et al. The case for learned index structures [C] //Proc of the 2018 Special Interest Group on Management of Data (SIGMOD). New York: ACM, 2018: 489−504

[29] Xu Ke, Li Yanbiao, Xie Gaogang, et al. Efficient name lookup using hybrid counting Bloom filters[J]. Journal of Computer Research and Development, 2023, 60(5): 1136−1150 (in Chinese)
(许可, 李彦彪, 谢高岗, 等. 基于混合计数布隆过滤器的高效数据名查找方法[J]. 计算机研究与发展, 2023, 60(5): 1136−1150)

[30] Athanassoulis M, Ailamaki A. BF-tree: Approximate tree indexing[J]. Proceedings of the VLDB Endowment, 2014, 7(14): 1881−1892

**Chang Jian**, born in 1992. PhD candidate. His main research interests include blockchain, machine learning, and distributed systems.
**常 健**, 1992 年生. 博士研究生. 主要研究方向为区块链、机器学习和分布式系统.

**Lin Licheng**, born in 2000. Master candidate. His main research interests include blockchain, machine learning, and distributed systems.
**林立成**, 2000 年生. 硕士研究生. 主要研究方向为区块链、机器学习和分布式系统.

**Li Binhong**, born in 1998. Master candidate. His main research interests include blockchain, machine learning, and distributed systems.
**李彬弘**, 1998 年生. 硕士研究生. 主要研究方向为区块链、机器学习和分布式系统.

**Xiao Jiang**, born in 1988. PhD, professor, PhD supervisor. Senior member of CCF. Her main research interests include blockchain and distributed systems.
**肖 江**, 1988 年生. 博士, 教授, 博士生导师. CCF 高级会员. 主要研究领域为区块链、分布式系统.

**Jin Hai**, born in 1965. PhD, professor, PhD supervisor. Vice chairman of CCF, fellow of CCF. His main research interest includes distributed systems.
**金 海**, 1965 年生. 博士, 教授, 博士生导师. CCF 副理事长, CCF 会士. 主要研究领域为分布式系统.